

Design Data Translators Reference

**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. 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

1

<u>Overview of the Translators</u>	9
<u>Licensing Requirements</u>	10
<u>Understanding the Translators</u>	11
<u>Running the Translators</u>	12

2

<u>Design Translation Using XStream Translator</u>	15
<u>Overview</u>	17
<u>Running the XStream Translator</u>	17
<u>Design Translation During XStream In</u>	21
<u>XStream In Form</u>	24
<u>XStream In More Options Form</u>	38
<u>XStream In Editor Forms</u>	76
<u>XStream In Options Names in GUI and Template File</u>	89
<u>Design Translation During XStream Out</u>	97
<u>XStream Out Form</u>	99
<u>XStream Out More Options Form</u>	113
<u>XStream Out Editor Forms</u>	153
<u>XStream Out Option Names in GUI and Template File</u>	168
<u>Fast Stream Out</u>	180
<u>Layer Mapping in XStream</u>	181
<u>Automatic Layer Mapping</u>	183
<u>Using Map Files During Translation</u>	186
<u>Preserving Connectivity Information</u>	220
<u>Sample Log File</u>	222
<u>Sample Summary File</u>	223
<u>Sample Template File</u>	225
<u>Performance Guidelines for XStream</u>	228
<u>Guidelines for XStream Out</u>	231

Design Data Translators Reference

<u>Chip Finishing Flow</u>	233

3

<u>Design Translation Using XOasis Translator</u>	237

<u>Overview</u>	239

<u>Running the XOasis Translator</u>	241

<u>Design Translation During XOasis In</u>	245

<u>XOasis In Form</u>	248

<u>XOasis In More Options Form</u>	262

<u>XOasis In Editor Forms</u>	299

<u>XOasis In Option Names in GUI and Template File</u>	311

<u>OASIS to OpenAccess Mapping</u>	317

<u>Mapping OASIS Records to OpenAccess Objects</u>	318

<u>Design Translation During XOasis Out</u>	321

<u>XOasis Out Form</u>	323

<u>XOasis Out More Options Form</u>	336

<u>XOasis Out Editor Forms</u>	372

<u>XOasis Out Names in GUI and Template File</u>	384

<u>Layer Mapping in XOasis</u>	399

<u>Automatic Layer Mapping in XOasis</u>	401

<u>Using Map Files During Translation</u>	403

<u>Sample Log File</u>	437

<u>Sample Summary File</u>	438

<u>Sample Template File</u>	440

<u>Performance Guidelines for XOasis</u>	444

<u>Guidelines for XOasis Out</u>	446

<u>Chip Finishing Flow</u>	447

4

<u>Netlist Import Using Spice In</u>	449

<u>Introduction to Spice In</u>	450

<u>Licensing in Spice In</u>	452

<u>Affects of Control Commands during Import</u>	455

<u>Using Spice In</u>	458

Design Data Translators Reference

<u>Running Spice In from GUI</u>	459
<u>Running Spice In from Command Line</u>	488
<u>Inherited Connections Supported</u>	490
<u>Creating a Spice In Parameter File</u>	497
<u>Mapping Parameter File Options with the Spice In GUI Options</u>	499
<u>Creating a Device-Map File</u>	502
<u>The propMatch Construct</u>	505
<u>The propMap Construct</u>	507
<u>Sub-Circuit Level Parameters</u>	512
<u>Handling Terminal Order for Sub-Circuits</u>	514
<u>Creation of Terminal Order</u>	515

5

<u>Design Translation Using CDL Out</u>	517
<u>Overview</u>	518
<u>Using CDL Out</u>	518
<u>Preparing a Template File</u>	541
<u>Preparing the si.env File</u>	544
<u>How CDL Out Translates Data</u>	548
<u>CDL Out Formats</u>	559
<u>CDL Out Output Files</u>	562
<u>How to Customize CDL Netlist</u>	563

6

<u>Design Translation Using LEF/DEF Translator</u>	565
<u>Overview</u>	567
<u>Methods of Design Translation</u>	568
<u>Running DEF In and DEF Out from Command Line</u>	580
<u>Using the oaMapFile option for different OpenAccess Database Formats</u>	589
<u>Colorless Layers and Colorless LPPs (Virtuoso Advanced Node for Layout Only)</u>	593

Design Data Translators Reference

Running LEF In and LEF Out from GUI	596
Running DEF In and DEF Out from GUI	611
Preparing a Template File	627
Setting the Environment Variables in LEF In and LEF Out	628
Setting the Environment Variables in DEF In and DEF Out	629
Viewing Native OpenAccess Libraries in the Virtuoso Studio Design Environment	630

7

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

Introduction to Allegro Translator	633
Using the Allegro Translator	634
Methods Used to Perform Package Design Translation	635
Import Allegro File Form	635
Export Allegro File Form	644
Design Object Mapping	646
Translating Parametrized Tline Cells	648
Data Model Mapping	649
Layer Mapping	652
Troubleshooting Allegro File Translation Issues	653

A

Stream Format

Overview	656
About GDSII	656
Stream File Description	657
Differences in Stream Format Versions	675
Stream Syntax	675
Example of a Stream Format File	677
How to read, write and print Stream dates to avoid Y2K problems	686

B

<u>Migrating from CDL In to Spice In</u>	689
<u>Spice In Advantages over CDL In</u>	689
<u>Differences Between Spice In and CDL In</u>	690

C

<u>Mapping LEF to the Technology File</u>	691
<u>LEF Data Map</u>	691

D

<u>List of XStream Messages</u>	731

E

<u>List of XOasis Messages</u>	783

F

<u>Design Data Translators Environment Variables</u>	829
<u>List of Design Data Translators Environment Variables</u>	831
<u>XST_AUTO_LOAD_TEMPLATE_NAME</u>	834
<u>XST_IN</u>	836
<u>XST_OUT</u>	837
<u>xstAlwaysLookupMapFilesInTech</u>	838
<u>xstEnableGUIHistory</u>	840
<u>xstHideMapAllColorToLocked</u>	841
<u>xstIgnoreCoveredViaShapes</u>	842
<u>xstInOverwriteLibPopup</u>	843
<u>xstOutAutoPopulateIfEmptyLCV</u>	844
<u>xstShowCompletionMsgBox</u>	845

Design Data Translators Reference

<u>strmFileExtension</u>	846
<u>convertPin</u>	847
<u>layerMap</u>	848
<u>logFile</u>	849
<u>maxVertices</u>	850
<u>objectMap</u>	851
<u>summaryFile</u>	852
<u>enableColoring</u>	853
<u>strmFilenameSuffix</u>	854
<u>strmFilenamePrefix</u>	855
<u>XStream Shell Environment Variables</u>	856
<u>XST_OUT_CC_CHOP</u>	858
<u>xoasEnableGUIHistory</u>	859
<u>XOASIS_AUTO_LOAD_TEMPLATE_NAME</u>	860
<u>XOASIS_IN</u>	862
<u>XOASIS_OUT</u>	864
<u>xoasAlwaysLookupMapFilesInTech</u>	866
<u>xoasOutAutoPopulatelfEmptyLCV</u>	867
<u>xoasInOverwriteLibPopup</u>	868
<u>xoasShowCompletionMsgBox</u>	869
<u>oasisFileExtension</u>	870
<u>enableColoring</u>	871
<u>oasisFilenamePrefix</u>	872
<u>oasisFilenameSuffix</u>	873
<u>XOasis Shell Environment Variables</u>	874
<u>XOASIS_IN_VM_NT</u>	875

Overview of the Translators

The design data translators convert designs from OpenAccess database to different formats and conversely. These translators fall into three categories: general, netlist, and physical translators.

This user guide is aimed at CAD developers and designers who are using third party tools which are not OA compliant and require an intermediate format for transferring the design data.

This manual has been written with an assumption that you are familiar with the development and design of integrated circuits. It has also been assumed that you are familiar with the Stream/GDSII format, the CDL format, and the LEF/DEF format. This manual contains reference information about the following Cadence® products:

- XStream In, which translates designs from GDSII Stream format into the OpenAccess database.
- XOasis Out, which translates designs from OpenAccess database into the GDSII Stream format.
- LEF In, which translates a LEF file to an OpenAccess technology library and macro information in the OpenAccess database.
- LEF Out, which translates an OpenAccess technology library and macro information to a LEF file.
- DEF In, which reads a DEF file and creates an OpenAccess cellview.
- DEF Out, which writes a DEF file from an OpenAccess cellview.
- Spice In, which translates designs from HSpice, Spectre, Spice, and CDL formats into the OpenAccess database.
- CDL Out, which translates designs from OpenAccess database into the Circuit Description Language format.

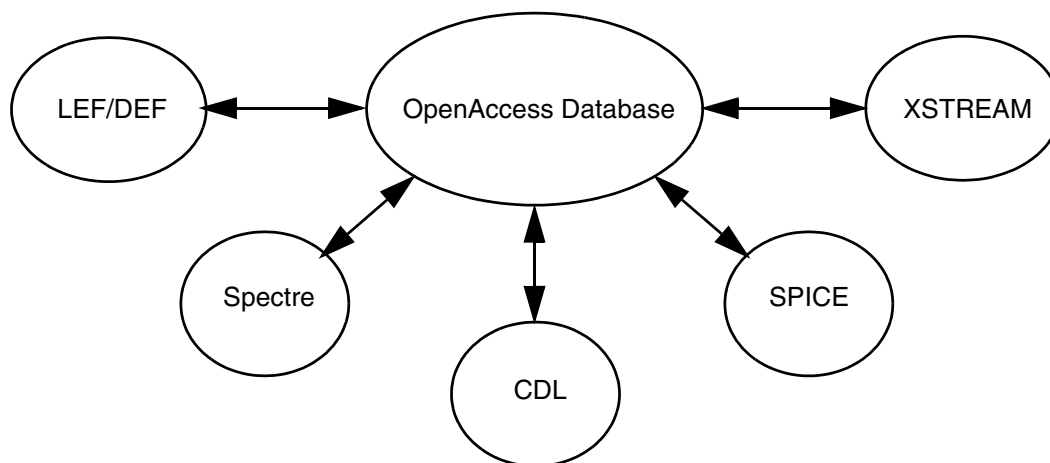
The Cadence® Virtuoso® Studio Design Environment is the foundation on which most Cadence tools are built. The Virtuoso Studio Design Environment on OpenAccess is an open

Design Data Translators Reference

Overview of the Translators

system. You can enter your own design data with industry-standard data formats and use the Virtuoso Studio Design Environment tools to complete your job.

The design data translators let you translate a design in one format to a Cadence database or to take a Cadence database and translate it into another format. Cadence offers a variety of translators to use in integrating your work with the OpenAccess database.



Licensing Requirements

This section describes an overview of how licensing is implemented in physical translators, XStream In, XStream Out, XOasis In, XOasis Out, LEF In, LEF Out, DEF In, and DEF Out translators.

In case, a license is not available for a translator and you select *File – Import – LEF/DEF/Stream* option from CIW, a form will be displayed without a license. You can specify the values in the form and click the OK button. The translator in this case would perform a validation check for license and try to checkout the license. You will not be able to do the translation if the license is unavailable.

The 'Virtuoso Framework' (111) license is required to run all these physical design translators.

The 'Virtuoso Advance Node Framework' (95011) license is required to run all these physical design translators.

These translators work only with the Design Framework II license “111” and with no other product license. Therefore, even if Layout XL or a higher tier license is available, you will not be able to run these translators if license “111” is not available. Also, to be able to run some advanced SKILL APIs that used in `libInit.il` or Pcell SKILL code, you will require functionality based licenses.

This chapter contains the following sections:

- [Understanding the Translators](#)
- [Running the Translators](#)

Understanding the Translators

The design data translators fall into three categories: general, netlist, and physical translators.

The general translators translate EDIF and Verilog files to and from OpenAccess database. The netlist translators translate netlist designs to and from OpenAccess database. The physical translators translate physical (layout) designs to and from OpenAccess database.

General Translators

The general translators are:

- EDIF 200 Writer, which translates designs from OpenAccess database to EDIF 200 format. For information about EDIF 200 Writer, see the [*Virtuoso EDIF 200 Reader and Writer User Guide*](#).
- Verilog In, which imports a Verilog HDL description to a OpenAccess database. For information about Verilog In, see the [*Verilog In for Design Framework II User Guide and Reference*](#).

Netlist Translators

The netlist translators are:

- Spice In, which translates SPICE and Circuit Definition Language (CDL) files to OpenAccess database. For information about Spice In, see [*Chapter 4, “Netlist Import Using Spice In”*](#).
- CDL Out, which translates a OpenAccess schematic design to a hierarchical netlist in Circuit Definition Language (CDL) format. For information about CDL Out, see [*Chapter 5, “Design Translation Using CDL Out.”*](#).

Physical Translators

The physical translators are:

Design Data Translators Reference

Overview of the Translators

- Stream In, which translates Graphics Design Station (GDSII) Stream mask data from Stream format to the OpenAccess database. For information about Stream In, see [Chapter 2, “Design Translation Using XStream Translator”](#).
- Stream Out, which translates files from OpenAccess database to Graphics Design Station II (GDSII) Stream mask data format. For information about Stream Out, see [Chapter 2, “Design Translation Using XStream Translator”](#).
- LEF In, which translates a LEF file to an OpenAccess technology library and macro information in the OpenAccess database. For more information on LEF In, see [Chapter 6, “Design Translation Using LEF/DEF Translator,”](#)
- LEF Out, which translates an OpenAccess technology library and macro information to a LEF file. For more information on LEF Out, see [Chapter 6, “Design Translation Using LEF/DEF Translator,”](#)
- DEF In, which reads a DEF file and creates an OpenAccess cellview. For more information on DEF In, see [Chapter 6, “Design Translation Using LEF/DEF Translator,”](#)
- DEF Out, which writes a DEF file from an OpenAccess cellview. For more information on DEF Out, see [Chapter 6, “Design Translation Using LEF/DEF Translator,”](#)

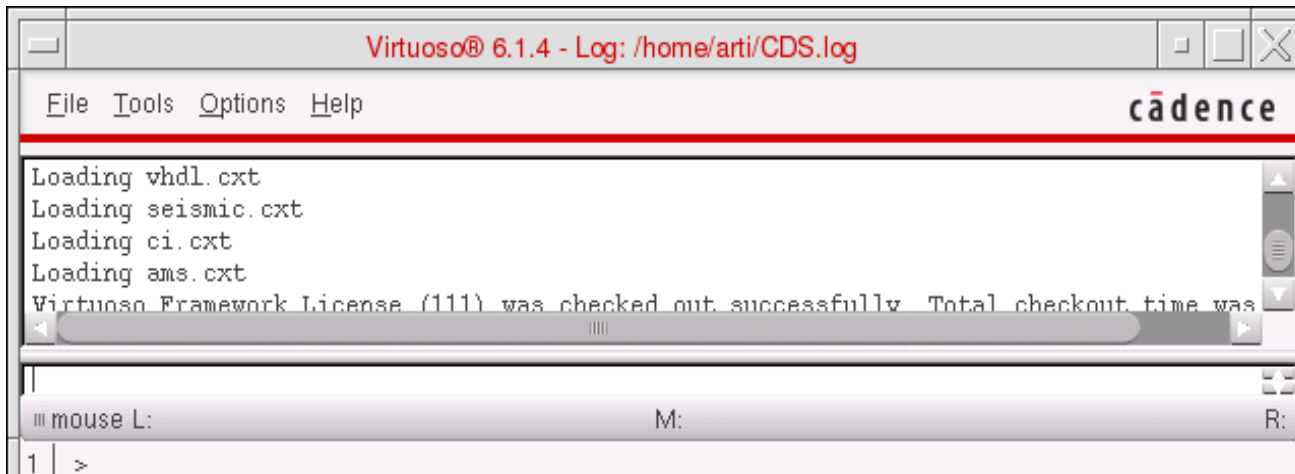
Running the Translators

You can run all the design data translators in interactive mode by using the user interface forms present in the translators. You can access these forms by invoking the appropriate menu options available in the CIW. You can also run most of the translators in batch mode by entering a command on the command line.

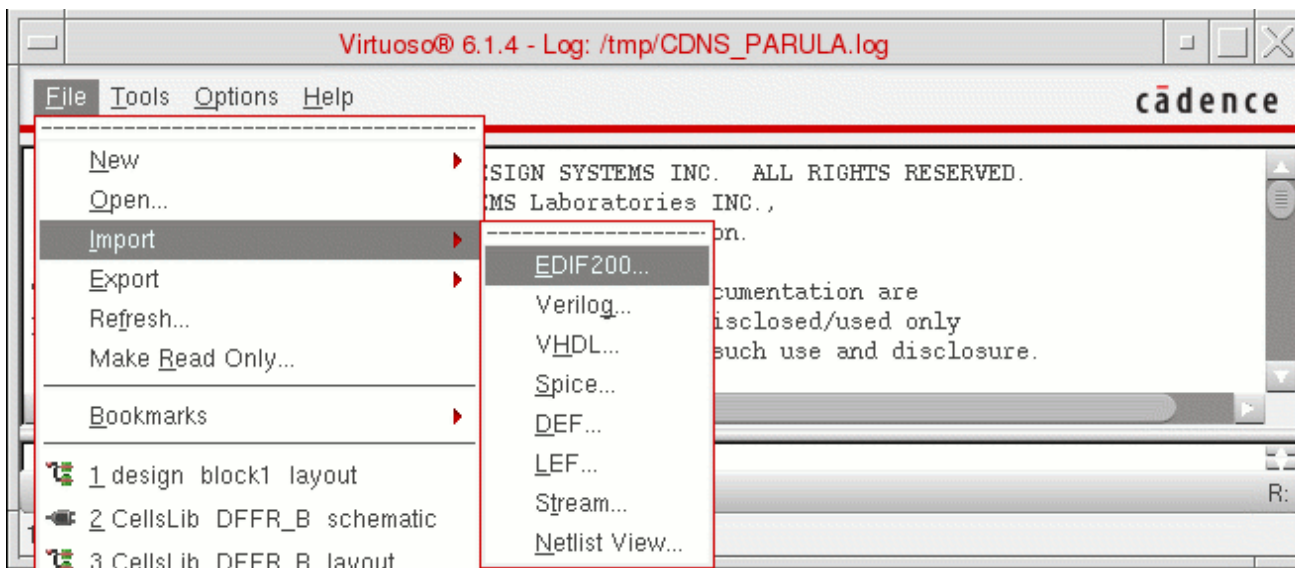
Design Data Translators Reference

Overview of the Translators

The CIW, shown below, displays a log of events when you run any of the Cadence applications, including one of the design data translators.



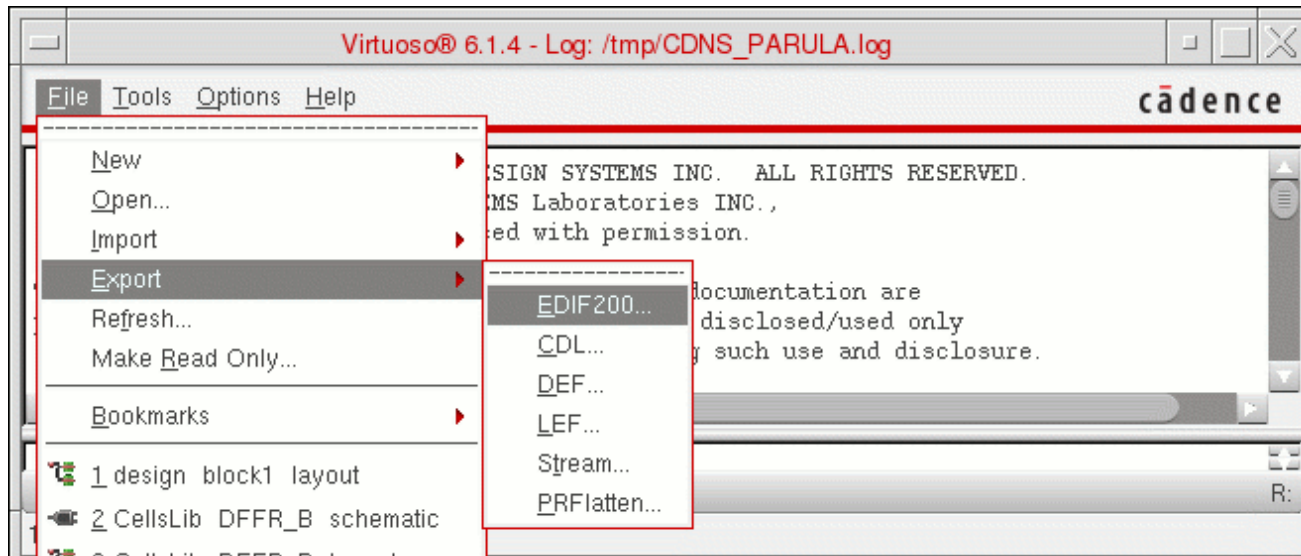
When you select *File* from the CIW banner, a pull-down menu appears. From this menu, you can choose the *Import* or *Export* menu option for using a specific translator. The following figure shows a list of translators that can be used for importing design files.



Design Data Translators Reference

Overview of the Translators

If you select *Export* from the *File* pull-down menu, the export translators appear on a pull-down menu.



Design Translation Using XStream Translator

This chapter contains the following sections:

- [Overview](#)
- [Running the XStream Translator](#)
- [Design Translation During XStream In](#)
 - ❑ [XStream In Use Model](#)
 - ❑ [Multithreading in XStream In](#)
 - ❑ [XStream In Form](#)
 - ❑ [XStream In More Options Form](#)
 - ❑ [XStream In Editor Forms](#)
 - ❑ [XStream In Options Names in GUI and Template File](#)
 - ❑ [Seeding the XStream In GUI](#)
 - ❑ [Handling Non-Manhattan Angle Rotation and Magnification](#)
 - ❑ [Predicting Memory Requirements](#)
- [Design Translation During XStream Out](#)
 - ❑ [XStream Out Use Model](#)
 - ❑ [XStream Out Form](#)
 - ❑ [XStream Out More Options Form](#)
 - ❑ [XStream Out Editor Forms](#)
 - ❑ [XStream Out Option Names in GUI and Template File](#)
 - ❑ [Seeding the XStream Out GUI](#)

Design Data Translators Reference

Design Translation Using XStream Translator

- ☐ [Variant Generation during XStream Out](#)
- ☐ [Hierarchical Coloring during XStream Out](#)
- ☐ [GZip Support](#)
- [Fast Stream Out](#)
- [Layer Mapping in XStream](#)
- [Automatic Layer Mapping](#)
- [Using Map Files During Translation](#)
- [Preserving Connectivity Information](#)
- [Sample Log File](#)
- [Sample Summary File](#)
- [Sample Template File](#)
- [Performance Guidelines for XStream](#)
 - ☐ [Guidelines for XStream In](#)
 - ☐ [Guidelines for XStream Out](#)
- [Chip Finishing Flow](#)

Overview

The Stream format stores mask generation data for designing integrated circuits. It represents a layout of different design layers the way they finally appear on a chip. The Stream format is widely used in the industry for:

- Archiving design data in interchangeable format.
- Exchanging intellectual property with other vendors.
- Exchanging data between various tools to complete the design cycle.
- Transferring data in a compact form between various design groups.

To translate a design in the Stream format to the OpenAccess database and conversely, you can use the XStream translator. XStream consists of two modules, XStream In and XStream Out. XStream In translates designs in the Stream format to the OpenAccess database. XStream Out translates designs from the OpenAccess database to the Stream format.

This chapter provides information about various methods of running the XStream translator. In this chapter, you read about various XStream In options, XStream Out options, and mapping files. In addition, you will also view the sample log file, sample summary file, and the sample template file. In this chapter, you will also read about the methods of preserving OpenAccess objects during a cycle of XStream Out and XStream In.

Running the XStream Translator

You can translate design files by running the XStream translator either from the Textual User Interface (TUI) or from the Graphical User Interface (GUI). In this section, you will learn about the methods of running the XStream translator by using TUI and GUI. Additionally, you will also read about the interactive mode, the method of invoking help, and setting the environment variables.

Note: XStream does not report less than half width extension in the log file.

Running XStream In and XStream Out Using TUI

You can run XStream In and XStream Out in the batch mode by using the `strmin` and `strmout` commands. Use any of the two methods described below to run these commands.

- Specify the options as command-line arguments

```
$strmin -option1 value1 -option2 value2
$strmout -option1 value1 -option2 value2
```

Design Data Translators Reference

Design Translation Using XStream Translator

where,

<i>option1</i>	First option
<i>value1</i>	Value of the first option
<i>option2</i>	Second option
<i>value2</i>	Value of the second option

Example 2-1 Running XStream In by specifying the `library` and `strmFile` options as command-line arguments.

```
$strmin -library lib1 -strmFile in.gds
```

- Write the options and their values in a text file and then specify the text file as an argument to `strmin` and `strmout`.

```
$strmin -templateFile textfileName  
$strmout -templateFile textfileName
```

where,

<i>-templateFile</i>	Name of the template file option
<i>textfileName</i>	Name of the template file containing option values

To view the template file, see [Sample Template File](#).

Example 2-2 Running XStream In by specifying the `strmTechGen` and `strmFile` options as command-line arguments.

```
$strmin -strmTechGen <techFileName> -strmFile <StreamFileName>
```

Note: If both `-strmTechGen` and `-library` options are specified, then the `-library` option will take precedence and `-strmTechGen` option will be ignored.

Output

After you specify the `-strmTechGen` option, the following files are generated:

- ASCII technology file is generated with the `layerDefinitions()` section. The `layerDefinitions()` section have four sub-sections: `<techLayers>`, `<techPurposes>`, `<techhLayerPurposePriorities>`, and `<techDisplays>`.
- The `<techFileName>.layermap` file is generated with `oaLayer` and `oaPurpose` to Stream layer and datatype mapping. This file contains the layer map entries in the form

Design Data Translators Reference

Design Translation Using XStream Translator

of quadruplets:

```
<oaLayer oaPurpose streamLayer dataType>.
```

Example 2-3 Running XStream Out by writing options and their values in the text file `template.txt`.

```
$strmout -templateFile template.txt
```

The contents of text file appear in the format shown below:

```
option1      value1
option2      value2
```

For boolean options, do not mention any value. When you specify a boolean option, its value is accepted as `true`. Otherwise, its value is `false`.

You can combine the above two methods by specifying some options on the command line and other options in the template file. This method is a combination of the above two methods.

If you specify the same option in the template file and the command line, the option specified in the command line takes precedence.

```
$strmin -option1 value1 -option2 value2 -templateFile textfileName
$strmout -option1 value1 -option2 value2 -templateFile textfileName
```

where,

<i>option1</i>	First option
<i>value1</i>	Value of the first option
<i>option2</i>	Second option
<i>value2</i>	Value of the second option
<i>templateFile</i>	Name of the template file
<i>textfileName</i>	Name of the text file

Example 2-4 Running XStream In by specifying `library`, `strmFile`, and `templateFile` as command-line arguments.

```
$strmin -library lib1 -strmFile in.gds -templateFile template.txt
```

In the above example, the `topCell` and `libName` options are taken from the command line and all other options are taken from `template.txt`.

Important

You must specify a library name and an output file name during XStream Out. If you do not specify these names, an error message is generated. Similarly, during XStream In, you must specify the input Stream file name and a library name, otherwise, a error message is generated.

Using Help

If you want help using XStream In and XStream Out, you can use the `-help` or `-h` options. These commands provide help on all the options that you can specify with the `strmin` and `strmout` commands. Help is displayed even when you type `strmin` or `strmout` without specifying options.

Example 2-5 Using help during XStream In and XStream Out

```
strmin -h
strmout -h
```

Running XStream In and XStream Out from the GUI

You can run XStream In by choosing the *File–Import–Stream* command from the CIW. Similarly, you can run XStream Out by choosing the *File–Export–Stream* command from the CIW.

Note: `LD_LIBRARY_PATH` is now automatically set when you invoke XStream from the command line or from GUI.

Preserving GUI History in XStream

The `-guihistory` command lets you use the last preserved GUI history at command line. These values can be overwritten at command line, if needed.

GUI history is saved only when the SKILL environment variable `xstEnableGUIHistory` is set to `t`. By default, the environment variable is set to `nil`, which means that the command is disabled.

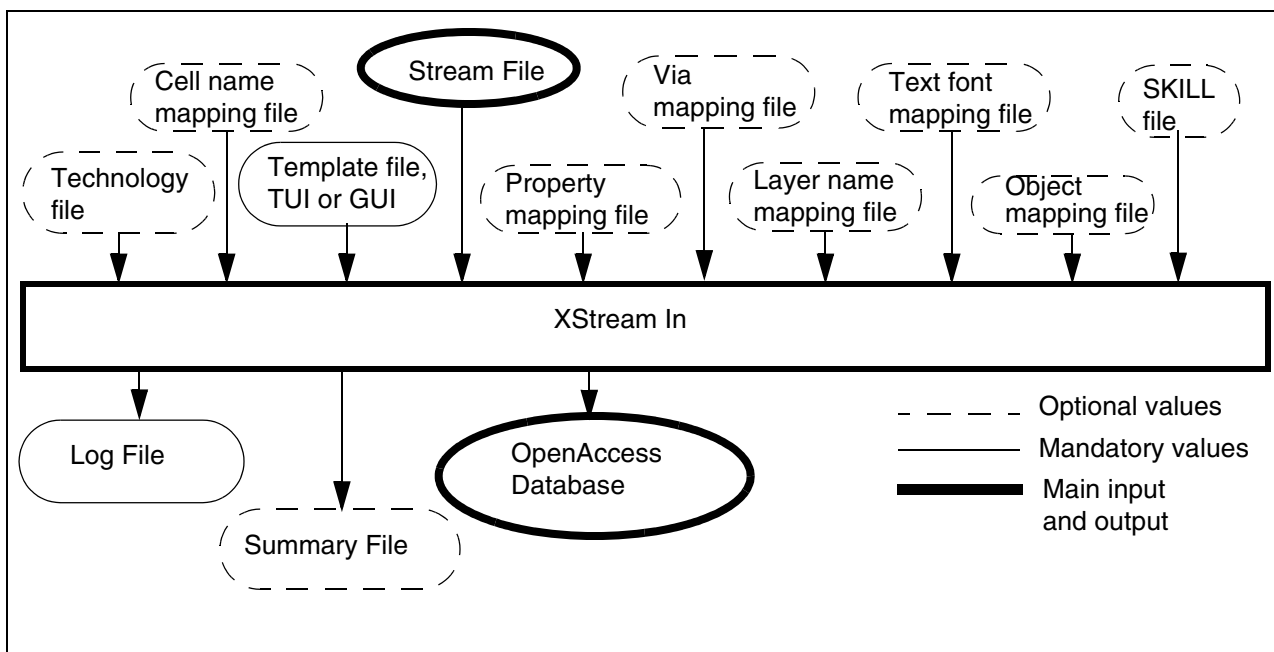
GUI history values are used across Virtuoso sessions. Therefore, when `xstEnableGUIHistory` is enabled, the XStream form also reads these values.

These values are also given the highest precedence. Therefore, if both pre-seeding and GUI history values are available, the XStream form will have the values from the GUI history.

Design Translation During XStream In

XStream In Use Model

XStream In converts design data in the Stream format to designs in the OpenAccess database. The inputs to the XStream In process are XStream In options, optional mapping files, user-defined SKILL files, and technology file information. After the design has been streamed in, the log file, the summary file, and the OpenAccess library are generated.



The following table provides generic information about the files used during XStream In.

File Type	Description
Stream file	A Stream-formatted file that is to be translated to OpenAccess.
Template file	A file that contains the option settings you want to use during XStream In. You can load the option settings from the template file into the <i>XStream In</i> form.
	In the XStream In template file, '\$' is considered as a special character because it is a UNIX environment variable. However, if you want '\$' to be treated as a normal character, you can escape it by using '\' before it.

Design Data Translators Reference

Design Translation Using XStream Translator

File Type	Description
Technology file	A file used to compile the OpenAccess technology file for the OpenAccess database during XStream In.
SKILL file	An optional file used for mapping cell names, text fonts, property names, and layer names.
OpenAccess database	The output database
Log file	A file used for recording the translation process. It also keeps a log of all the messages.
Summary file	A file that contains a summary of the translated design.

For details on the map files, see [Design Translation During XStream In](#).

Multithreading in XStream In

XStream In uses multithreading to improve performance during translation. XStream evaluates the machine configuration and then applies the best value to be used for multithreading based on available CPU cores. The maximum number of threads are capped at 2.

Multithreading works differently when:

- [Stream In to Virtual Memory Mode is Enabled](#)
- [Stream In to Virtual Memory Mode is Disabled or XStream is Run from Command Line](#)

Stream In to Virtual Memory Mode is Enabled

When [Import to Virtual Memory](#) is enabled, translation occurs in virtual memory. In this mode, by default XStream uses one thread for translation.

You can also manually set number of cores to be used for multithreading with the [XST_IN_VM_NT](#) environment variable. For example, the following sets number of cores to be used for multithreading in VM mode to 4 .

```
setenv XST_IN_VM_NT 4
```

If you set the value as 0 or 1, it implies no cores will be used and multithreading will not take

Design Data Translators Reference

Design Translation Using XStream Translator

place.

Stream In to Virtual Memory Mode is Disabled or XStream is Run from Command Line

When *Import to Virtual Memory* is disabled or XStream is run from command line, a maximum of two CPU cores are used for translation.

Design Data Translators Reference

Design Translation Using XStream Translator

XStream In Form

You can import the Stream file from the GUI by choosing the *File–Import–Stream* option from the CIW. This method will enable you to display the XStream In form.

Note: The form displays values from the previous XStream In session, if the changes made were applied before exiting XStream In.

This section describes the various XStream In options on the XStream In form. For a list of equivalent template file options, refer to the [XStream In Options Names in GUI and Template File](#) section.

The screenshot shows the 'XStream In' dialog box. A red box highlights the 'Stream File' and 'Library' fields, both marked '(required)'. A red arrow points to a dropdown arrow icon next to these fields, labeled 'Mandatory Options'. Below this, the 'Top Cell' and 'View' (set to 'layout') fields are visible. The 'Template File' field has a file icon. A checkbox for 'Import to Virtual Memory' is present. A list of expandable sections includes 'Technology', 'Generate Technology Information', 'Layer Map', 'Object Map', and 'Log File'. A red arrow points to the 'Layer Map' section, labeled 'Expand to view additional options'. A second red box highlights the expanded 'Layer Map' section, which contains a 'File Name' field, a 'Translate Unmapped LPP' checkbox, and a 'Use Automatic Layer Mapping' checkbox. At the bottom are buttons for 'Translate', 'Apply', 'Cancel', 'Reset All Fields', 'More Options', and 'Help'.

The main XStream In form comprises mandatory options that are essential to run the translation. The form also comprises a few other most commonly used options, such as Layer Map and Object Map.

The following sections on the main form can be expanded to view additional options:

Design Data Translators Reference

Design Translation Using XStream Translator

- [Technology](#)
- [Generate Technology Information](#)
- [Layer Map](#)
- [Object Map](#)
- [Log File](#)

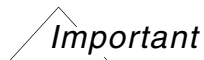
Discussed below are the XStream In options available on the main form.

Stream File

Use the *Stream File* field to specify the name of the Stream file that is to be translated to OpenAccess. Alternatively, click the *File Browser* button to select the Stream file. This field supports file names that are relative to the run directory or are absolute.

You can either specify the absolute or relative path. If you specify the absolute path, the `.gds` file will be searched at the specified location. However, in case of relative path, the `.gds` file will be searched in the run directory. By default, the run directory is the current directory.

Command-line option: `strmFile`



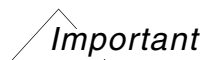
You must select a Stream file before running XStream In.

Library

Use the *Library* field to specify the target library where data is to be translated. The library name should be a logical name and follow the naming conventions supported by OpenAccess. You can also select an existing library as the target library from the drop-down list.

When the library does not exist, a new library is created in the run directory. If the target library already exists, data is translated to the library.

Command-line option: `library`



You must select a select a destination library before running XStream In.

Top Cell

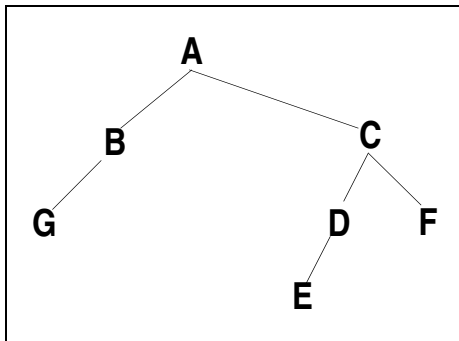
Use the *Top Cell* field to specify the name of the cell at the top of the hierarchy that you want to translate to OpenAccess. The Stream format supports hierarchical designs. So, the *Top Cell* option determines the node from where the design is to be translated.

Default value: ""

Command-line option: `topCell`

Example 2-6 Translating cells by using the Top Cell option

Consider the hierarchical structure below where A, B, C, D, E, F, and G are cell names. You want to translate these cells using the *Top Level Cell* field.

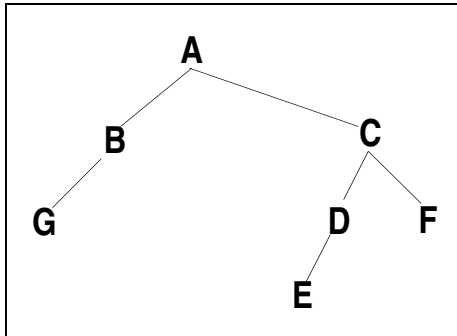


When C is specified as the top cell in the *Top Cell* field, cells C, D, E, and F are translated.

View

Use the *View* field to specify view names that are created in the target library during translation. If you specify a cellview name in the cell name map file or SKILL file, then the view name specified in these files takes precedence over the name specified in the XStream In form.

Example 2-7 Translating cells C, D, E, and F into the layout1 view



In the above figure, if C is specified as the top cell and `layout1` as the view name, then the structures C, D, E, and F are translated with `layout1` as the view name.

Default value: `layout`

Command-line option: `view`

Template File

Use the *Template File* field to specify an existing template file. You can also click the *File Browser* button to select the template file. When this file is loaded, the XStream In form is updated with the option settings specified in it. If the correct template file is loaded, then all the GUI fields in the form are reset.

Click the *Save* button to save the current option settings to the template file.

Command-line option: `templateFile`

Import to Virtual Memory

Select the *Import to Virtual Memory* check box to read a GDSII/Stream file and create an OpenAccess design in the virtual memory. You can also specify the number of CPU cores that should be used for multithreading during translation. For more information, see [Multithreading in XStream In](#).

Note: You should use this option in a new library and not in an existing library. This means that instead of using the option in a pre-created library, you should allow XStream In create the new library.

Additionally, [Disable Info Message IDs](#) and [Disable Warning Message IDs](#) fields are populated with *, when the *Import to Virtual Memory* check box is selected, to indicate that

Design Data Translators Reference

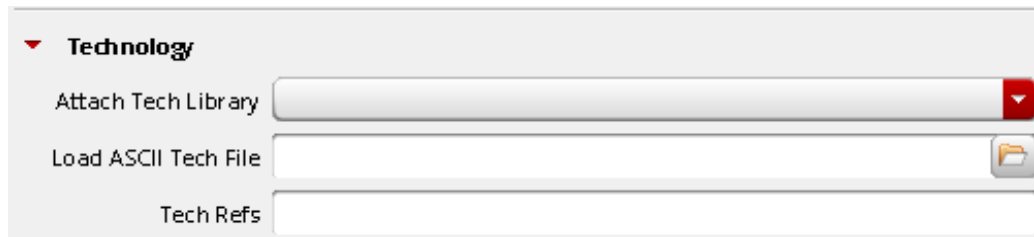
Design Translation Using XStream Translator

all the information (`INFO`) and warning (`WARNING`) messages are disabled for CIW and the Stream In log file. If you manually change the value of these fields, or if you remove * from them, then if *Import to Virtual Memory* check box is selected, information and warning messages, if there are any, are not displayed in the CIW. They are logged in the log file.

For more information, refer to the [Chip Finishing Flow](#) section.

Technology

Expand *Technology* to specify a technology library, file, database that you would like to use during translation.



Attach Tech Library

Use the *Attach Tech Library* drop-down list box to attach an existing technology file to the target library. The *Attach Tech Library* drop-down list box lists all the technology libraries specified in the `cds.lib` file.

If you select this option, XStream In attaches the technology file from the library specified in the *Attach Tech Library* field instead of creating a new one. This option takes precedence over the *Load ASCII Tech File* option.

If the target library already contains a technology file, then the *Attach Tech Library* option is ignored. An attached technology file is treated as read-only by XStream In and its contents remain unaltered during XStream In.

It is essential to have a layer map file while using the *Attach Tech Library* option.

Default value: ""

Default value: `attachTechFileofLib`

Load ASCII Tech File

Use the *Load ASCII Tech File* field to create or merge a technology file in the target library. Alternatively, click the *File Browser* button to select the technology file. The technology file is created when the ASCII technology file is compiled during XStream In. If the technology file already exists in the target library, then the technology file that has been created by compiling the ASCII technology file, is merged with the existing one.

The ASCII technology file is searched with respect to the run directory if the complete path of the file is not specified.

When a target library is attached to another technology library, XStream In ignores this option and generates a warning message. If the target library is not attached to any other technology library, a technology file is created or merged with the technology file created from the compilation of the ASCII technology file.

If the absolute path to the current directory exists, then after you click the *File Browser* button, the File Browser dialog box will be open the current directory. Otherwise, the *File Browser* dialog box will be open in the run directory.

Command-line option: `loadTechFile`

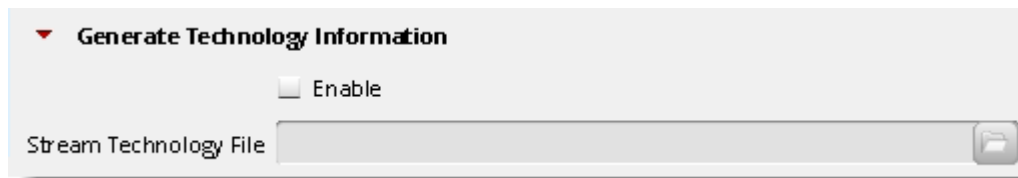
Tech Refs

Use the *Tech Refs* field to specify the technology databases that will be referenced by the technology database created by XStream In. You can specify multiple technology databases in this field. If there are conflicts in the technology graph, that is, technology database created by XStream In and the referenced technology databases that you specify in this field have conflicts, the tool stops and generates an error to specify that XStreamIn has detected a conflict in the technology database graph.

Command-line option: `techRefs`

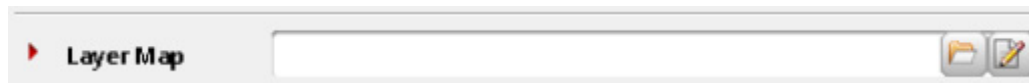
Generate Technology Information

Expand *Generate Technology Information* and select the *Enable* check box to specify the target technology file where data needs to be translated in the *Stream Technology File* field.



After you enable the *Generate Technology Information* option, all other fields except the *Stream Technology File* field are disabled.

Layer Map

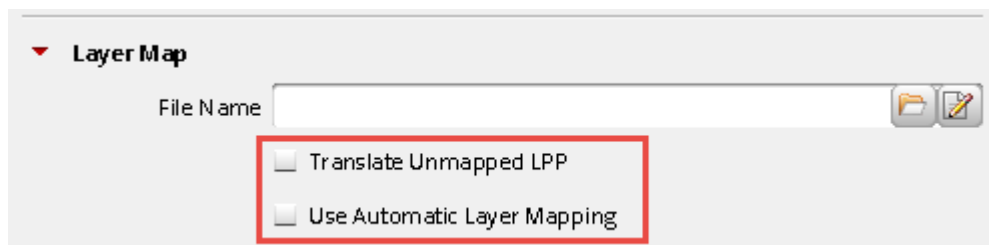


Use the *Layer Map* field to specify an existing layer map file. You can also click the *File Browser* button to select the layer map file. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `layerMap`

Alternatively, to create new layer mapping records or load a new layer map file, leave the Layer Map field blank, and click the *Editor* button to display the XStream In Layer Map File Editor Form.

Expand *Layer Map* to view additional options.



Translate Unmapped LPPs

Use the *Translate Unmapped LPP* option to map unmapped layers. This option works only when a different user-defined layer mapping, layer map file or SKILL function, has been specified. When you enable *Translate Unmapped LPP*, first, the user-specified layer mapping is applied, and then, automatic layer mapping is performed to translate unmapped layers. For more information about translating unmapped layers in Stream In, see [Automatic Translation of Unmapped Layers in XStream In](#).

Default value: Deselected

Command-line option: `translateUnmappedLPP`

Use Automatic Layer Mapping

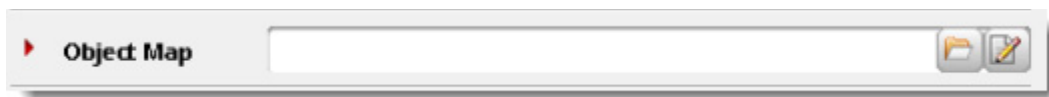
Use the *Use Automatic Layer Mapping* option to map layers automatically. This option works only if no layer mapping has been specified. Otherwise, this option is ignored and layers are mapped with the specified mapping.

For more information about automatic layer mapping, see [Automatic Layer Mapping in XStream In](#).

Default value: Deselected

Object Map

Object mapping file maps the object type and sub type to the Stream layer-datatype pair. The object mapping file determines which OpenAccess object should be translated as a shape on which Stream layer-datatype.



Use the *Object Map* field to specify an existing object map file. You can also click the *File Browser* button to select the object map file. Next, click the *Editor* button to view the specified file in editor.

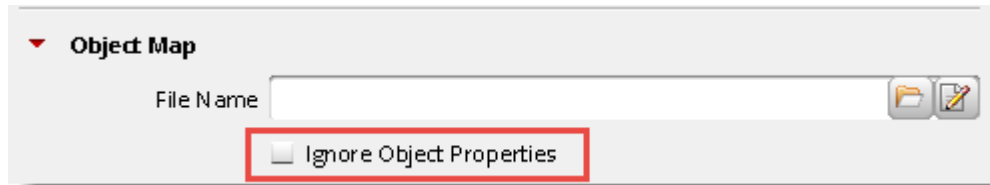
Command-line option: `objectMap`

Alternatively, to create new object mapping records or load a new object map file, leave the Object Map field blank, click the *Editor* button to display the [XStream In Object Map File Editor Form](#).

Design Data Translators Reference

Design Translation Using XStream Translator

Expand *Object Map* to view additional options.



Ignore Object Properties

Use the *Ignore Object Properties* option to ignore property information stored at property attribute 126.

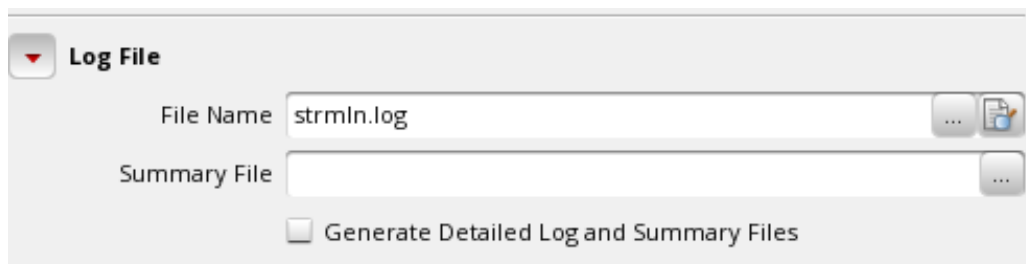
Property information (on PROPATTR 126) is written by using an object-map during XStreamOut to translate objects as shapes in GDSII. XStreamOut, by default would write special properties to these shapes which are used by XStreamIn to automatically recreate objects from these shapes. Ignoring these properties means that these objects, which are shapes with special properties in GDSII, will now be translated as shapes in OpenAccess.

Default value: Deselected

Command-line option: `ignoreObjectProp`

Log File

Expand Log File to specify log and summary files to be used during XStream In translation.



Use the *File Name* field to specify name of a log file that will be used to record the translation process steps and the messages generated by XStream. After the translation is completed, the log file is displayed in a separate read-only text window. Alternatively, click the Browse button to select the log file.

The path specified for the log file should be a valid file path. It can be absolute or relative to the run directory. For more information, see [Sample Log File](#).

Design Data Translators Reference

Design Translation Using XStream Translator

Default file name: strmOut.log

Command-line option: logFile

Summary File

Use the *Summary File* field to specify a file that will provide a summary of the translated design. If you do not specify a file name in the *Summary File* option before translation, the summary file will not be generated.

The path specified for the summary file can be absolute or relative to the run directory. To view the sample summary file, see [Sample Summary File](#).

Command-line option: summaryFile

Generate Detailed Log and Summary Files

Use the *Generate Detailed Log and Summary Files* option to include additional summary information in the XStream In log file and XStream In summary file.

With this option is enabled, following sections with additional information are added to the XStream In summary file:

- **Individual Cell Statistics:** Cellview summary added to this section is sorted alphabetically according to library/cell/view.

The bBox column is also added. This column lists the boundary boxes of the translated cells in the following format:

```
((lowerLeft.x lowerLeft.y) (upperRight.x upperRight.y))
```

Here, *lowerLeft* and *upperRight* are the lower-left and upper-right coordinates of the boundary box.

Individual Cell Statistics

Library/Cell/View	Instance	Array	Polygon	Rect	Path	Text	Node	bBox
blkgLib/cell/layout	1	0	0	2	0	0	0	((-22.900 -22.000) (4.100 7.800))
blkgLib/cel2/lay	0	0	0	1	0	0	0	((1.900 0.700) (6.800 4.500))
blkgLib/top/layout	2	0	0	1	0	0	0	((1.900 0.700) (6.800 4.500))

Design Data Translators Reference

Design Translation Using XStream Translator

- **Statistics of Layers:** Layer summary added to this section is sorted in the same order in which the layer-purpose pairs are defined in the technology file or the layer palette.

Statistics of Layers

Cadence Layer	Cadence Purpose	Stream Layer	Datatype	Polygon	Rect	Path	PathSeg	Text	Node
L0	drawing	0	0	0	1	0	0	0	0
L1	drawing	1	0	0	1	0	0	0	0
L1	drawing	1	0	0	1	0	0	0	0
L1	drawing	1	0	0	1	0	0	0	0

- **Top Cells:** This section lists the referenced top cells

Top Cells

Cell Name

topCell1
topCell2
topCell3
topCell4

- **Missing Cells:** This section lists the cells that are referenced but are missing from the input GDS file and the referenced library, if it is specified.

Missing Cells

Cell Name

missingcell1
missingcell2
missingcell3

Design Data Translators Reference

Design Translation Using XStream Translator

- **Dropped Layers:** This section lists the `strmLayer:strmType` pair for which no mapping is found. Shapes on these layer-purpose pairs are not translated.

Dropped Layers	
Stream Layer	Stream Datatype
81	0
82	0
83	0

- **Individual Advanced Object Statistics:** This section lists the shapes translated as Place and Route objects during XStream In. It lists the object-type, sub-object type, layer name (for layerBlockage), `strmLayer`, `strmType`, and the count of the objects translated.

Individual Advanced Object Statistics					
Object Type	Object Sub-type	Layer Name	Stream Layer	Stream Datatype	Count
AreaHalo	placement		4	3	2
Boundary	PR		0	3	1
AreaBlockage	placement		1	3	2

Default value: Deselected

Command-line option: `verbose`

Coloring Mode

Coloring Mode ☒ Enable

To enable color mapping during XStream In, select the *Enable* check box for the *Coloring Mode* option.

Default value: Selected

Command-line option: `enableColoring`

Design Data Translators Reference

Design Translation Using XStream Translator

Important

The layer map file, which is either specified through the Layer Map field or is available in the technology, is required if you enable coloring mode. Otherwise, translation will be terminated and an error message will be displayed. (See, [Coloring Mode](#))

If the layer map and object map files do not have any color entry, the `-enableColoring` option is not considered during translation.



Translate

Use the *Translate* button to translate the Stream file to OpenAccess.

Apply

Use the *Apply* button to translate the Stream file to OpenAccess. After the translation is completed, the *XStream In* form does not disappear.

Important

Clicking the *Translate* or the *Apply* button, saves the values you specify in the XStream In form and when you start a new session of XStream In, these last applied changes will be visible in the form. If you make any updates to the values but do not apply the changes by clicking the *Translate* or *Apply* button, these new updates will not be visible the next time you open the XStream In form. The form will continue displaying values from the last applied changes.

Progress Bar



The *Progress bar* is displayed, when you click the *Translate* or *Apply* button. It moves left and right to indicate that the translation is in progress stops when the translation is completed. You can also click the *Stop* button to stop the process.

Design Data Translators Reference

Design Translation Using XStream Translator

Cancel

Use the *Cancel* button to exit the *XStream In* form.

Reset All Fields

Use the *Reset All Fields* button to set the GUI fields to their default values, which are displayed when the tool is invoked.

More Options

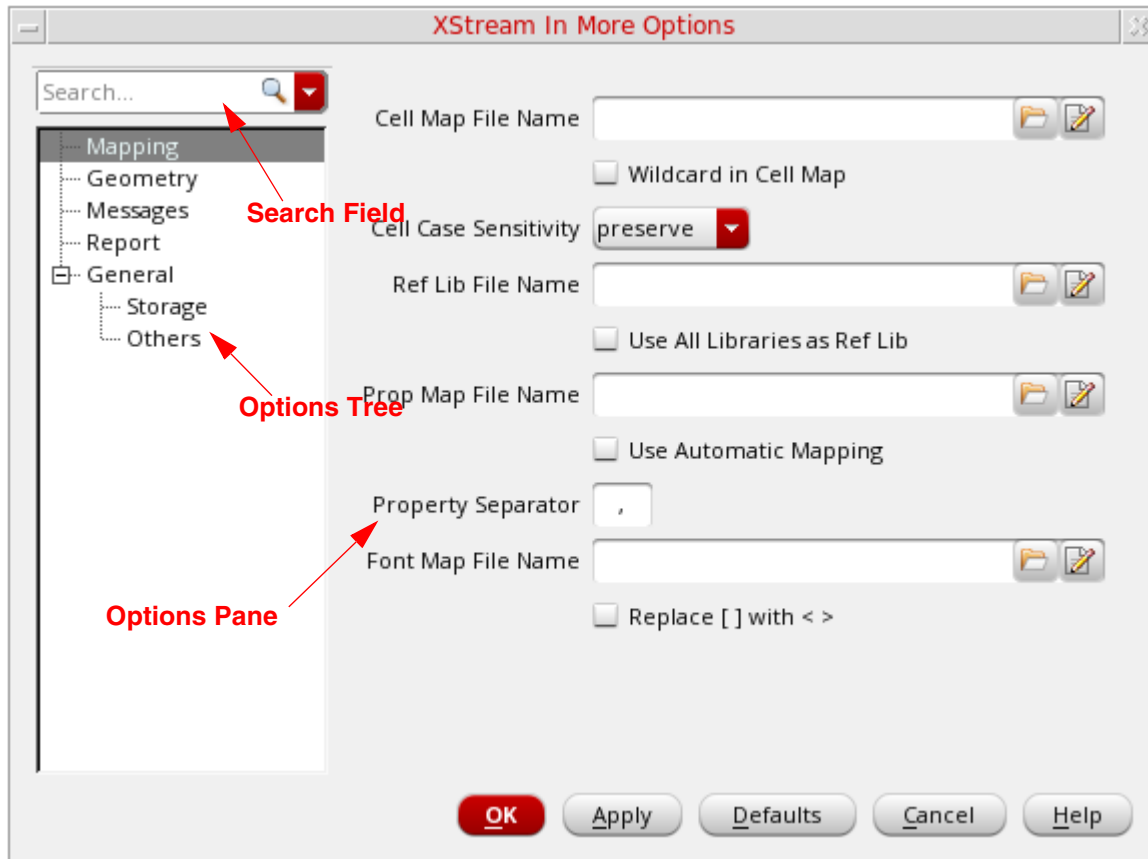
Use the *More Options* button to display the [XStream In More Options Form](#) which features additional options that you can set before starting XStream In translation.

Help

Use the Help button to invoke online help.

XStream In More Options Form

This section discusses various options available on the XStream In More Options form.

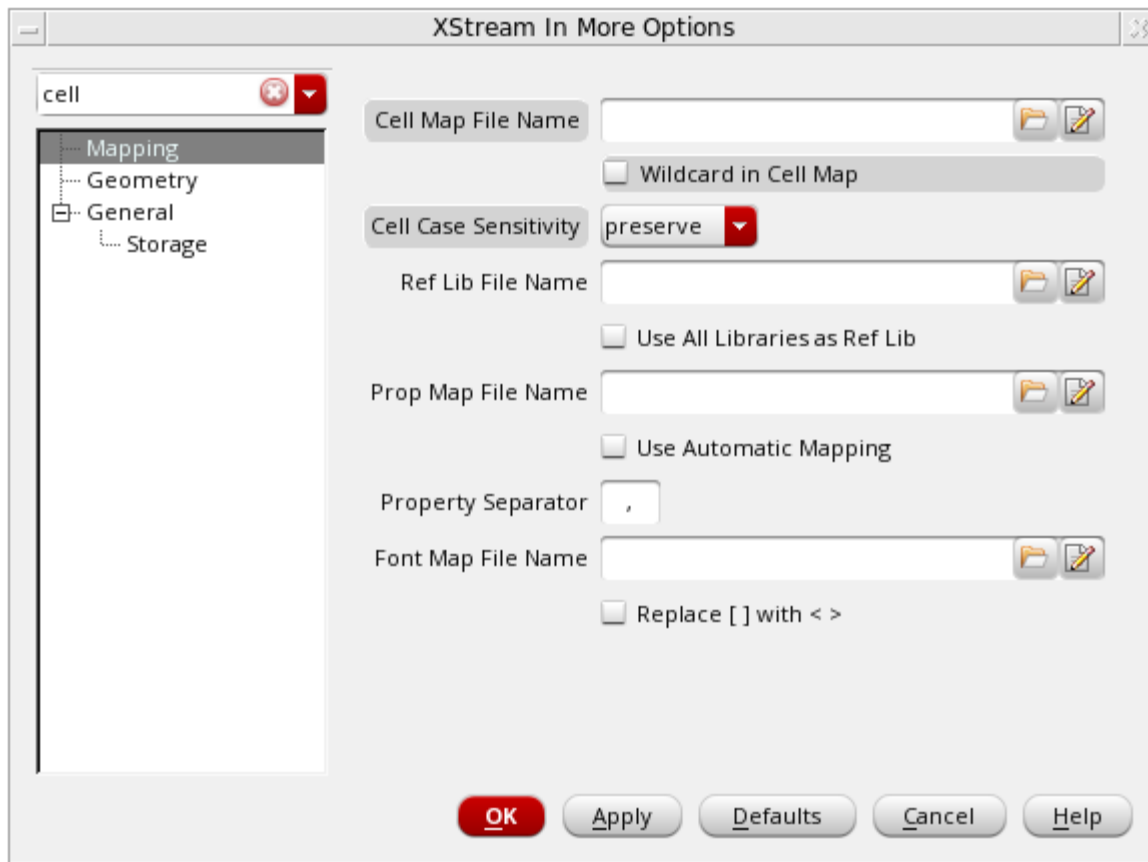


Search Field

Use the Search field to quickly look for options you want to access or the options related to the value you type in this field. For example, if you type `cell`, only the options related to cell will be visible in the options tree. The related options will also be highlighted in the options pane.

Design Data Translators Reference

Design Translation Using XStream Translator



Options Tree

The Options tree enables you to access the following option panes where you can perform additional settings.

- [Mapping Options](#)
- [Geometry Options](#)
- [Messages Options](#)
- [Report Options](#)
- [General: Storage Options](#)
- [General: Others Options](#)

Options Panes

Options panes comprise different options related to the option currently selected in the tree.

OK

Click *OK* to save your current settings for the More Options form and exit.

Apply

Click *Apply* to save your settings on the More Options form. When you click *Apply*, the form is not closed.

Defaults

Click *Defaults* to set default values for all options in the More Options form.

Cancel

Click *Cancel* to close the More Options form without saving any changes.

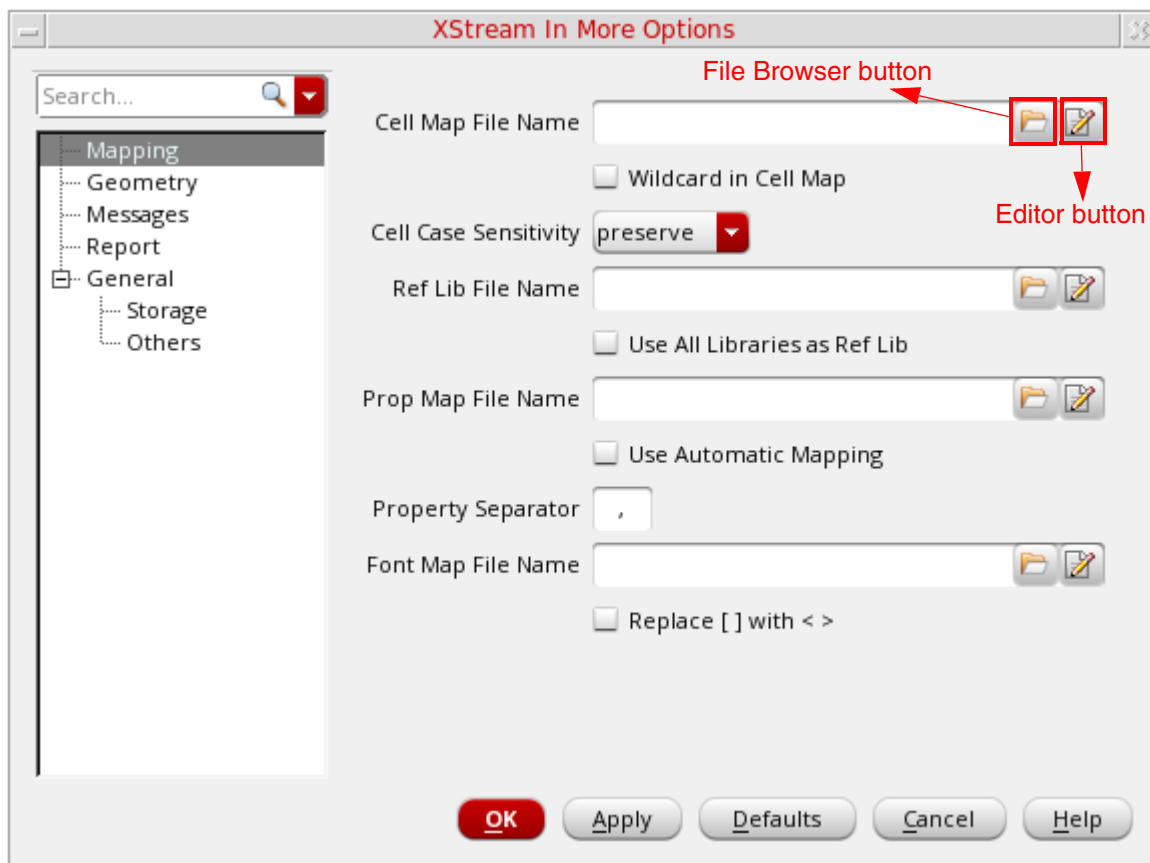
Design Data Translators Reference

Design Translation Using XStream Translator

This section covers the option panes that you can access from the Options tree in the XStream In More Options form.

Mapping Options

Use the *Mapping* options pane to specify additional mapping files to be used during XStream In translation.



This section describes options on the *Mapping* options pane.

Cell Map File Name

Use the *Cell Map File Name* field to specify an existing cell map file. You can also click the *File Browser* button to select the cell map file. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `cellMap`

Design Data Translators Reference

Design Translation Using XStream Translator

To create new cell mapping records or load a new cell map file, leave the *Cell Map File* field blank, and click the *Editor* button to display the XStream In Cell Map File Editor Form.

Wildcard in Cell Map

Use the *Wildcard in Cell Map* option to use wild cards while streaming in. The option supports the characters '*' and '?' as wild cards and '\' as the escape character. If you don't use this option, wild card will be considered as a special character during translation.

Consider the following example:

```
*   INV_*           *           INV\*_*_ver1
```

Here, during XStream In translation, the created cellview will be INV_NAND. if the structure in the GDS is INV*_NAND_ver1. In this example, in INV*_*_ver1, the first '*' is treated as special character and the second '*' as the wild card.

If you don't use *Wildcard in Cell Map*, the default behavior will be same as a normal cell map.

Consider that the following cell map file is used with the *Wildcard in Cell map* option to add prefix and suffix to all STRUCT names, except the topcell, in the GDS during XStream In translation:

```
#####
#lib                cell                view      struct
#
XST_TARGET_LIB      topCell             layout    topCell
XST_TARGET_LIB      pre_*_suf           layout    *
#####
```

The cell map will add the prefix `pre_` and the suffix `_suf` to all STRUCT names except the topcell.

Command-line option: `wildCardInCellMap`

Cell Case Sensitivity

Use the *Cell Case Sensitivity* drop-down-list to specify whether XStream In changes the case of letters in cell names. **lower**

The *lower* option translates all cell names to lowercase. For example, the name `Obj1` becomes `obj1`.

Design Data Translators Reference

Design Translation Using XStream Translator

upper

The *upper* option translates all cell names to uppercase. For example, the name `Obj1` becomes `OBJ1`.

preserve

The *preserve* option translates cell names without changing case. For example, the name `Obj1` remains `Obj1`.

Default value: `preserve`

Command-line option: `case`

XStream In ignores this option when cell mapping is done. You can map cell names in a cell name map table or in a SKILL file. For more information, see [Cell Name Map File](#).

Ref Lib File Name

Use the *Ref Lib File Name* field to specify an existing reference library name. You can also click the *File Browser* button to select the reference library. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `refLibList`

To select a library name from the list of available libraries to be used as a reference libraries, leave the *Reference Library File Name* field blank, and click the *Editor* button to display the [XStream In Reference Library File Editor Form](#).

Use All Libraries as Ref Lib

Use the *Use All Libraries as Ref Lib* check box to select all available libraries as reference libraries.

If you invoke Stream In from the command line, you can write the list of reference libraries in a text file and provide this file as the value of the `refLibList` option. For example, consider a reference library file `refLib.list` with the following content:

```
Lib1
Lib2
Lib3
```

In this case, the `-refLibList refLib.list` command will consider all reference libraries listed in the `refLib.list` file.

Design Data Translators Reference

Design Translation Using XStream Translator

If you want to consider all the libraries defined in the `cds.lib` file as reference libraries, use the `XST_CDS_LIB` keyword (`-refLibList XST_CDS_LIB`). You can provide this keyword as a value of the `refLibList` option from the command line or by writing it in a text file containing the list of libraries.

Prop Map File Name

Use the *Prop Map File Name* field to specify an existing properties mapping file. You can also click the *File Browser* button to select the properties mapping file. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `propMap`

To create new properties mapping records or load a new properties mapping file, leave the *Property Map File Name* field blank, and click the *Editor* button to display the XStream In Property Map File Editor Form.

Font Map File Name

Use the *Font Map File Name* field to specify an existing font mapping file. You can also click the *File Browser* button to select the font mapping file. Next, click the *Editor* button to view the specified file in editor.

To create new font mapping records or load a new properties mapping file, leave the *Font Map File Name* blank, and click the *Editor* button to display the XStream In Font Map File Editor Form.

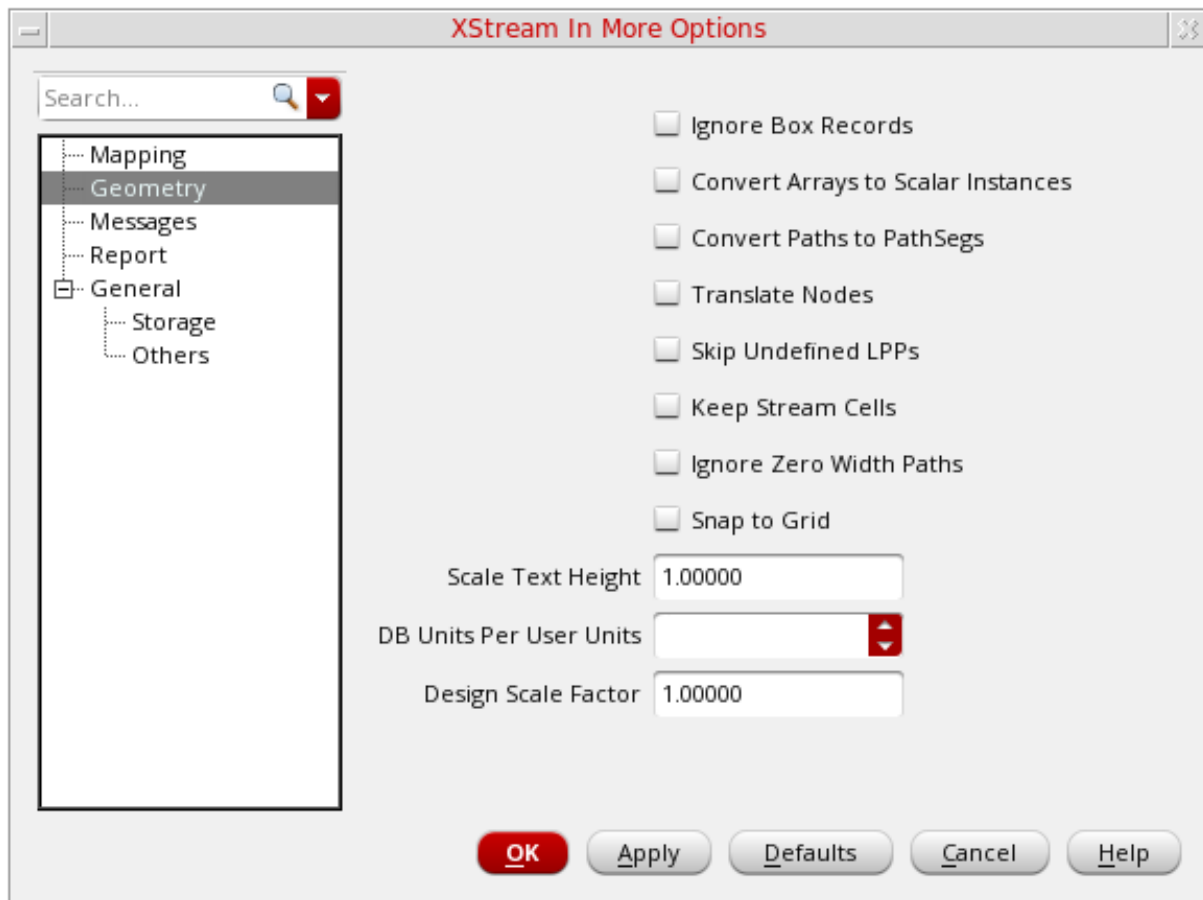
Replace [] with <>

The *Replace [] with <>* option replaces the character `[` with `<` and `]` with `>` in text records during the translation from the Stream file to a DFII library.

Command-line option: `replaceBusBitChar`

Geometry Options

This section describes options on the Geometry options pane.



Ignore Box Records

Use the *Ignore Box Records* option if you do not want to translate Stream file box records to rectangles in the OpenAccess database. If you select this option, then all the box records are dropped.

Default value: Deselected

Command-line option: `ignoreBoxes`

Convert Arrays to Scalar Instances

The *Convert Arrays to Scalar Instances* option translates Stream arrays to ROWS * COLS scalar instances in OpenAccess. If this option is not selected, each element in AREF translates to a single array in OpenAccess.

Default value: Deselected

Command-line option: `arrayInstToScalar`

Convert Paths to Pathsegs

Use the *Convert Paths to PathSegs* field to convert the 2-point orthogonal paths in the input Stream file to `pathSeg` objects in the OpenAccess database.

Default value: Deselected

All non-orthogonal paths (PATH) in the input Stream file are imported as OpenAccess path objects even when *Convert paths to pathsegs* option is used during XStream In.

Command-line option: `convertPathToPathSeg`

Translate Nodes

Use the *Translate Nodes* option to translate nodes as dots in OpenAccess. If you select this option, nodes are translated as dots, otherwise they are ignored.

Default value: Deselected

Command-line option: `translateNode`

Skip Undefined LPPs

The *Skip Undefined LPPs* option ignores the layer-purpose pairs that are not existing in the technology file. This option is selected only when you XStream In a design on an existing target library. When a user specifies the Skip Undefined LPPs option, a warning message will be displayed mentioning that the Stream layer-datatype pair is mapped to the layer-purpose, which does not exist in the technology library.

The *Skip Undefined LPPs* option is ignored when a new target library is created. This is because there is no pre-existing technology information. If the technology file of the target library is write-protected, then this option is automatically selected during XStream In.

Design Data Translators Reference

Design Translation Using XStream Translator

The *Skip Undefined LPPs* option is considered when you XStream In to a new library using an ASCII technology file. In this case, any undefined layer-purpose pairs will not be created in the OpenAccess technology for the target library.

Default value: Deselected

Command-line option: `skipUndefinedLPP`

Keep Stream Cells

The *Keep Stream Cells* option provides a way of referring to the information pertaining to a particular cell in the Stream file first instead of a reference library.

If the *Keep Stream Cells* option is not used, then the Stream file will not be searched while creating instances. If *Write mode* is set as overwrite, this option is ignored and a warning message is generated.

Default value: Deselected

Command-line option: `keepStreamCells`

Ignore Zero Width Paths

Use the *Ignore Zero Width Paths* option to ignore the translation of zero-width path. If you select this option, zero-width paths are ignored, otherwise they are translated as lines.

Default value: Deselected

Command-line option: `ignoreZeroWidthPath`

Snap to Grid

The *Snap to Grid* option adjusts the coordinates of geometric objects to coincide with the grid resolution. However, this option can shift data from the original Stream grid, resulting in a loss of precision.

Default value: Deselected

Command-line option: `snapToGrid`

Scale Text Height

Use the *Scale Text Height* field to control the height of text labels. Default value is 1. For example, you can specify the text height as 2.0 or 0.5 to double or reduce the label height to half.

Command-line option: `scaleTextHeight`

DB Units Per User Units

Use the *DB Units Per User Units* field to specify the database units per user units, which should be used during translation. This option allows you to scale the target design library as per the new `DBUPerUU`.

Note: This option is considered only if new technology information is created in the target library. This technology information contains the `DBUPerUU` value specified at the option.

Valid values: Greater than or equal to 1

Command-line option: `dbuPerUU`

Design Scale Factor

Use the *Design Scale Factor* option to scale an imported design by the factor set here. This option will accept any positive floating value. The `scale` option will not have any impact on existing options or behavior because scaling is done internally. It will work in addition to the settings in the `DBUPerUU` option.

For example, consider `DBUPerUU` is 1000 and GDS polygon size is 3000 db units (user units). If you run the command `strmin -scale 4 -dbuPerUU 2000`, the resultant polygon size will be calculated as below:

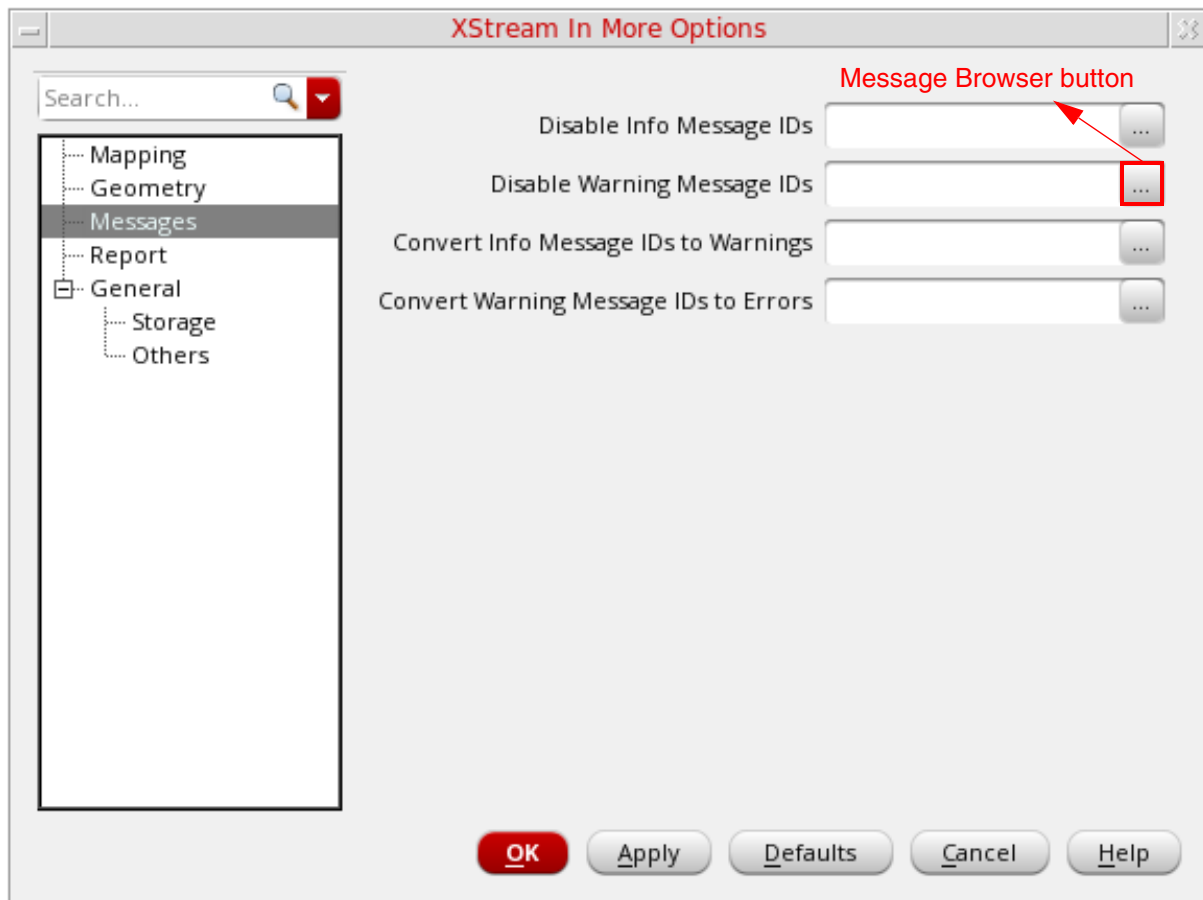
$3000 \times (2000/1000) \times 4 = 24000$ db units (user units)

Default value: 1.00

Command-line option: `scale`

Messages Options

This section describes options on the Messages options pane.



Disable Info Message IDs

Use the *Disable Info Message IDs* field to specify list of information message IDs that you do not want to print. You can also specify message IDs of dependent tools in this field.

The information message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 4374. The list of information message IDs must be separated by a comma or white space.

Command-line option: `noInfo`

Alternatively, click the *Message Browser* button to display the XStream In Info Messages form to select or specify the information messages that you do not want to print.

Disable Warning Message IDs

Use the *Disable Warning Message IDs* field to specify list of warning message IDs that you do not want to print. You can also specify message IDs of dependent tools in this field.

The warning message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 3075. The list of warning message IDs should be separated by a comma or white space.

Command-line option: `noWarn`

Alternatively, click the *Message Browser* button to display the XStream In Warning Messages form to select or specify the warning messages that you do not want to print.

Convert Info Message IDs to Warnings

Use the *Convert Info Message IDs to Warnings* field to specify list of information message IDs that you want to convert to warning messages. You can also specify message IDs of dependent tools in this field.

The information message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 4374. The list of information message IDs must be separated by a comma or white space.

Command-line option: `infoToWarn`

Alternatively, click the *Message Browser* button to display the XStream In Info Messages form to select or specify the information messages that you want to convert to warning messages.

Convert Warning Message IDs to Errors

Use the *Convert Warning Message IDs to Errors* field to specify the IDs of warning messages that you want to convert to error messages. You can also specify message IDs of dependent tools in this field.

The warning message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 3075. The list of warning message IDs should be separated by a comma or white space.

Command-line option: `WarnToErr`

Design Data Translators Reference

Design Translation Using XStream Translator

Alternatively, click the *Message Browser* button to display the XStream In Warning Messages form to select or specify the warning messages that you want to convert to error messages.

XStream In Messages Forms

This section describes the XStream In Message forms that are displayed when you click the respective Message Browser button.

- XStream In Info Messages Form to Disable Info Messages
- XStream In Warning Messages Form to Disable Warning Messages
- XStream In Warning Messages Form to Convert Warning Messages to Errors
- XStream In Warning Messages Form to Convert Warning Messages to Errors

Design Data Translators Reference

Design Translation Using XStream Translator

XStream In Info Messages Form to Disable Info Messages

This section describes options on the XStream In Info Messages form that is displayed when you click the *Message Browser* button for the Disable Info Message IDs option.

Select info messages to disable

Only XStream messages are listed below. To select message IDs of dependent tools, specify the message IDs on the Messages page or in the Other IDs field in the table below.

Id	Message
☐ Other IDs	
☐ 60	Creating a new purpose %s in the technology file.
☐ 61	Creating a new layer %s in the technology file.
☐ 62	Creating a new layer:purpose pair %s:%s in the technology file.
☐ 69	All the libraries specified in `cds.lib` will be considered as reference libraries. This is because refLi...
☐ 70	The 'Reference Library list' and the 'Keep Stream Cells' options have been used together. Any cell ...
☐ 91	Process may require virtual memory near to what it is available on the system for a process. You ...
☐ 105	Property map file contains duplicate entry for Stream attribute '%d' object type '%s' at line numb...
☐ 162	You have not used the viaMap option. If the OpenAccess design has native oaVia instances use th...
☐ 163	You have not used the viaMap option. If a viaMap file corresponding techfile or technology library...
☐ 171	The 'noObjectProp' and the 'objectMap' options have been used together. Therefore no property...

☐ Select All

OK Apply Cancel Help

Select check boxes for the XStream information messages that you want to disable.

Other IDs

Use the *Other IDs* field to specify information message IDs of other dependent tools.

Select All

Click the *Select All* check box to select all message IDs.

Cancel

Use the *Cancel* button to exit the XStream In Info Messages form without saving any changes.

XStream In Warning Messages Form to Disable Warning Messages

This section describes options on the XStream In Warning Messages form that is displayed when you click the *Message Browser* button for the Disable Warning Message IDs option.

Id	Message
☐ Other IDs	
☐ 1	Value of pinAttNum is out of range 1 127. Pins will not be preserved.
☐ 18	Performing automatic layer mapping because either the layer mapping information is missing or t...
☐ 19	Failed to create the automatic cell name map file '%s' in the run directory. Ensure that you have th...
☐ 20	Output Stream file '%s' already exists. It will be overwritten.
☐ 25	The layer map file '%s' contains an invalid record at line number '%d' and will be ignored. This is b...
☐ 26	The font map file '%s' contains an invalid record at line number '%d' and will be ignored. This is be...
☐ 27	The property map file '%s' contains an invalid record at line number '%d' and will be ignored. This ...
☐ 28	The cell map file '%s' contains an invalid record at line number '%d' and will be ignored. This is bec...
☐ 29	refLib file contains invalid record at line number %d. This record in the refLib file will be ignored.
☐ 31	Layer map file '%s' contains a duplicate entry for Stream layer:datatype '%d:%d' at line '%d'. Theref...

Select check boxes for the XStream warning messages that you want to disable.

Other IDs

Use the *Other IDs* field to specify warning message IDs of other dependent tools.

Select All

Click the *Select All* check box to select all message IDs.

Cancel

Use the *Cancel* button to exit the XStream In Warning Messages form without saving any changes.

Design Data Translators Reference

Design Translation Using XStream Translator

XStream In Info Messages Form to Convert Messages to Warnings

This section describes options on the XStream In Info Messages form that is displayed when you click the *Message Browser* button for the Convert Info Message IDs to Warnings option.

Id	Message
☐ Other IDs	
☐ 60	Creating a new purpose %s in the technology file.
☐ 61	Creating a new layer %s in the technology file.
☐ 62	Creating a new layer:purpose pair %s:%s in the technology file.
☐ 69	All the libraries specified in `cds.lib` will be considered as reference libraries. This is because refLi...
☐ 70	The 'Reference Library list' and the 'Keep Stream Cells' options have been used together. Any cell ...
☐ 91	Process may require virtual memory near to what it is available on the system for a process. You ...
☐ 105	Property map file contains duplicate entry for Stream attribute '%d' object type '%s' at line numb...
☐ 162	You have not used the viaMap option. If the OpenAccess design has native oaVia instances use th...
☐ 163	You have not used the viaMap option. If a viaMap file corresponding techfile or technology library...
☐ 171	The 'noObjectProp' and the 'objectMap' options have been used together. Therefore no property...
☐ 180	You have not used the objectMap option. The design has instance(s) of at least one of following Op...

Select check boxes of the XStream information messages listed in this form.

Other IDs

Use the *Other IDs* field to specify information message IDs of other dependent tools.

Cancel

Use the *Cancel* button to exit the XStream In Info Messages form without saving any changes.

Design Data Translators Reference

Design Translation Using XStream Translator

XStream In Warning Messages Form to Convert Warning Messages to Errors

This section describes options on the XStream In Warning Messages form that is displayed when you click the *Message Browser* button for the Convert Warning Message IDs to Errors option.

Id	Message
☐ Other IDs	
☐ 1	Value of pinAttNum is out of range 1 127. Pins will not be preserved.
☐ 18	Performing automatic layer mapping because either the layer mapping information is missing or t...
☐ 19	Failed to create the automatic cell name map file '%s' in the run directory. Ensure that you have th...
☐ 20	Output Stream file '%s' already exists. It will be overwritten.
☐ 25	The layer map file '%s' contains an invalid record at line number '%d' and will be ignored. This is b...
☐ 26	The font map file '%s' contains an invalid record at line number '%d' and will be ignored. This is be...
☐ 27	The property map file '%s' contains an invalid record at line number '%d' and will be ignored. This ...
☐ 28	The cell map file '%s' contains an invalid record at line number '%d' and will be ignored. This is bec...
☐ 29	refLib file contains invalid record at line number %d. This record in the refLib file will be ignored.
☐ 31	Layer map file '%s' contains a duplicate entry for Stream layer:datatype '%d:%d' at line '%d'. Theref...

Select check boxes of the XStream information messages listed in this form.

Other IDs

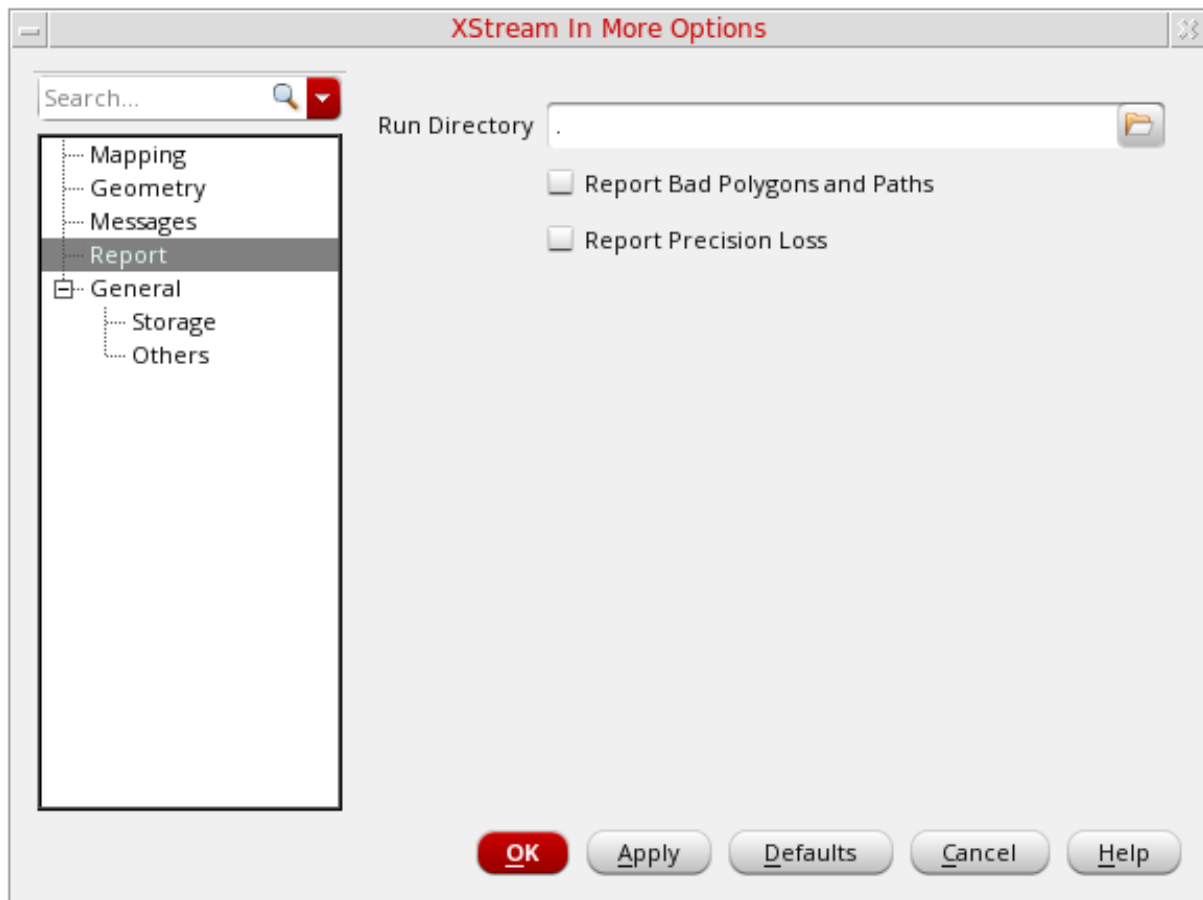
Use the *Other IDs* field to specify information message IDs of other dependent tools.

Cancel

Use the *Cancel* button to exit the XStream In Info Messages form without saving any changes.

Report Options

This section describes options on the Report options pane.



Run Directory

The *Run Directory* field accepts the default directory for all the files you specify in the *XStream In* form. The run directory will be created automatically (provided you have the appropriate permissions) if it does not exist during XStream In or XStream Out. If you want to read from or write to a file that is not in the run directory, type the absolute file path in the field.

Default value: Current working directory

Command-line option: `runDir`

Report Bad Polygons and Paths

The *Report bad polygons and path* option provides a way for reporting self-intersecting paths, coincident or collinear paths, paths that have less than two coordinates, path with an invalid begin and end extension value, an open polygon, coincident or collinear polygons, a polygon with self-intersecting points, and zero-area rectangles. However, using this option can cause a degradation in performance.

Default value: Deselected

Command-line option: `checkPolygon`

Report Precision Loss

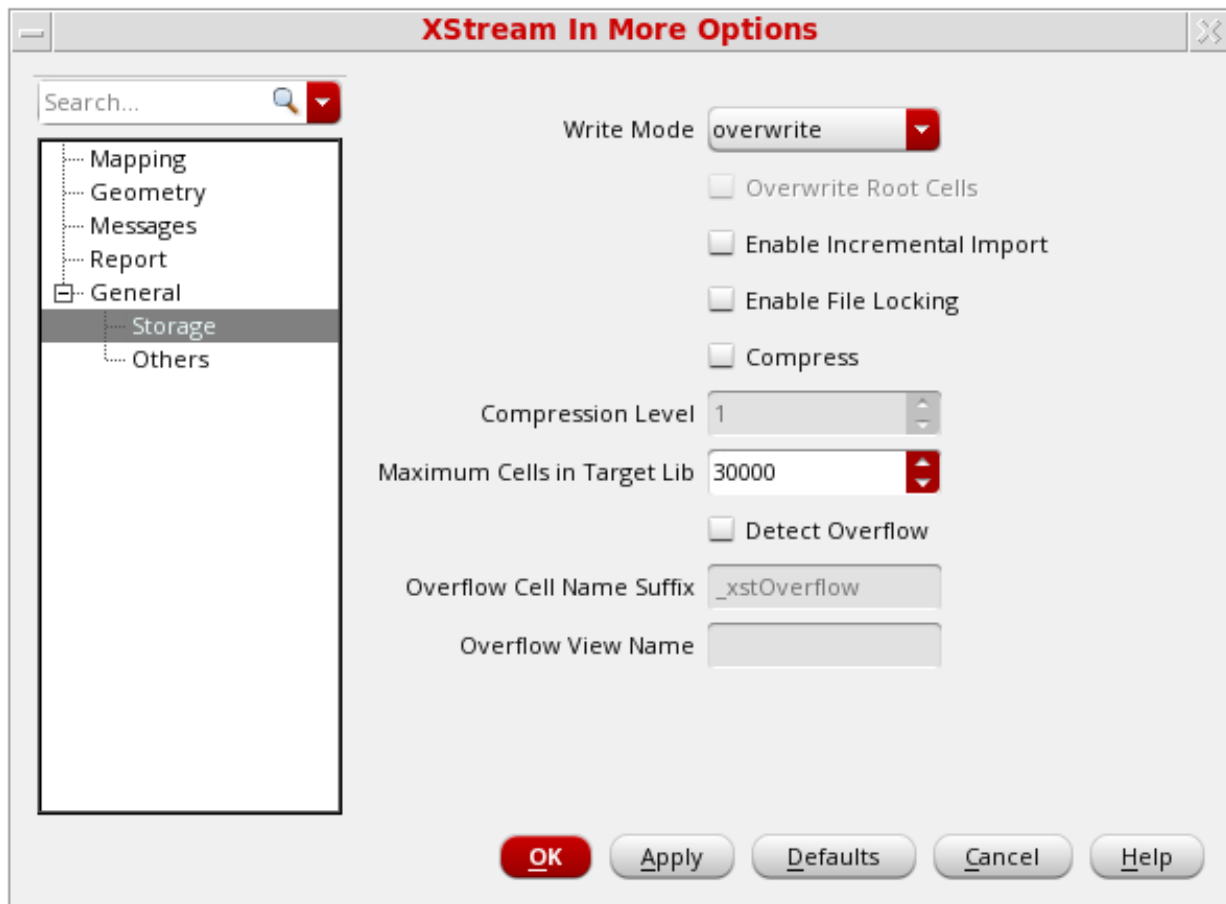
The *Report Precision Loss* option gives you an error message if there is any precision loss while scaling a design.

Default value: Deselected

Command-line option: `reportPrecisionLoss`

General: Storage Options

This section describes options on the General: Storage options pane.



Write Mode

Use the *Write Mode* field to select the write mode during the translation of the existing cells. The following options are available:

overwrite

Select the *overwrite* option to overwrite the existing cells in the target library during translation.

noOverwrite

Select the *noOverwrite* option to prevent existing cells in the target library from being overwritten during translation.

append

Select the *append* option to append data to an existing database during translation. If this option is selected, in case the GDS file contains shapes or instances that already exist in the database, they will be appended to the existing database. Therefore, the database will have duplicates of these shapes and instances.

rename

Select the *rename* option to rename the duplicate cells in the target library during translation.

Default value: overwrite

Command-line option: `writeMode`

Overwrite Root Cells

This option is available only if the *Write Mode* is set to *rename*. Using this option, all the root cells (un-instantiated cells in the GDS file) will replace the duplicate cells in the user-defined top-cell hierarchy; and the cells present in the root cell's hierarchy will be renamed.

Default value: Deselected

Command-line option: `overwriteRootCells`

Detect Overflow

Use the *Detect Overflow* option to detect overflow cellviews during translation.

OpenAccess limits the number of individual object types (like shapes, instances and properties) in a cellview. By default, the XStream In translator fails to translate a Stream file to OpenAccess if the maximum capacity of a cellview is reached for an individual object type. The *Detect Overflow* option enables XStream In to translate a Stream file even if the number of individual objects is more than the maximum capacity of a single cellview. In this scenario, overflow cellviews are created on reaching the maximum capacity of a cellview, with the remaining objects translated in the overflow cell. An instance of this overflow cell is then placed inside the main cell exceeding the maximum capacity.

Overflow cells are created with the name of the original cell suffixed with the value of the *Overflow Cell Name Suffix* option, and a view-name specified by the value of option *Overflow View Name* option.

Default value: Deselected

Design Data Translators Reference

Design Translation Using XStream Translator

Command-line option: `detectOverflow`

Overflow Cell Name Suffix

Use the *Overflow Cell Name Suffix* to specify the string to be appended to the name of the overflow cells.

Default value: `_xstOverflow`

Command-line option: `overflowCellNameSuffix`

Overflow View Name

Use the *Overflow View Name* to specify a view name for the overflow cells.

Default value: If no value is specified, value of the *View* option is used.

Command-line option: `overflowViewName`

Example 2-8 Multiple GDS files; Single StreamIn

G1. gds	G2. gds	G3. gds	G4. gds
TopCell Definition SREF A SREF B SREF C A Definition SREF C B Definition SREF C C Definition	A Definition SREF C C Definition	B Definition SREF C C Definition	C Definition

In the above example, there are multiple definitions for cells across GDS files.

If the *Write Mode* is set to *overwrite*, then the expected XStream In output is the shown below.

Example 2-9 XStream In output: Write mode is overwrite

```
TopCell1 Definition
SREF A2
SREF B3
SREF C4

A2 Definition
SREF C4

B3 Definition
SREF C4

C4 Definition
```

If the *Write Mode* is set to *noOverwrite*, then the expected XStream In output is the shown below.

Example 2-10 XStream In output: Write mode is noOverwrite

```
TopCell1 Definition
```

```
SREF A1
```

```
SREF B1
```

```
SREF C1
```

```
A1 Definition
```

```
SREF C1
```

```
C1 Definition
```

```
B1 Definition
```

```
SREF C1
```

If the *Write Mode* is set to *append*, then the expected XStream In output is the shown below.

Example 2-11 XStream In output: Write mode is append

TopCell1 ₁ Definition	
SREF A ₁₊₂	A ₁₊₂ : Cellname is 'A' and
SREF B ₁₊₃	it has shapes in cell 'A'
SREF C ₁₊₂₊₃₊₄	of G1. gds and G2. gds
A ₁₊₂ Definition	
SREF C ₁₊₂₊₃₊₄	
SREF C ₁₊₂₊₃₊₄	
B ₁₊₃ Definition	
SREF C ₁₊₂₊₃₊₄	
SREF C ₁₊₂₊₃₊₄	
C ₁₊₂₊₃₊₄ Definition	

If the *Write Mode* is set to *rename*, then the expected XStream In output is the shown below.

Design Data Translators Reference

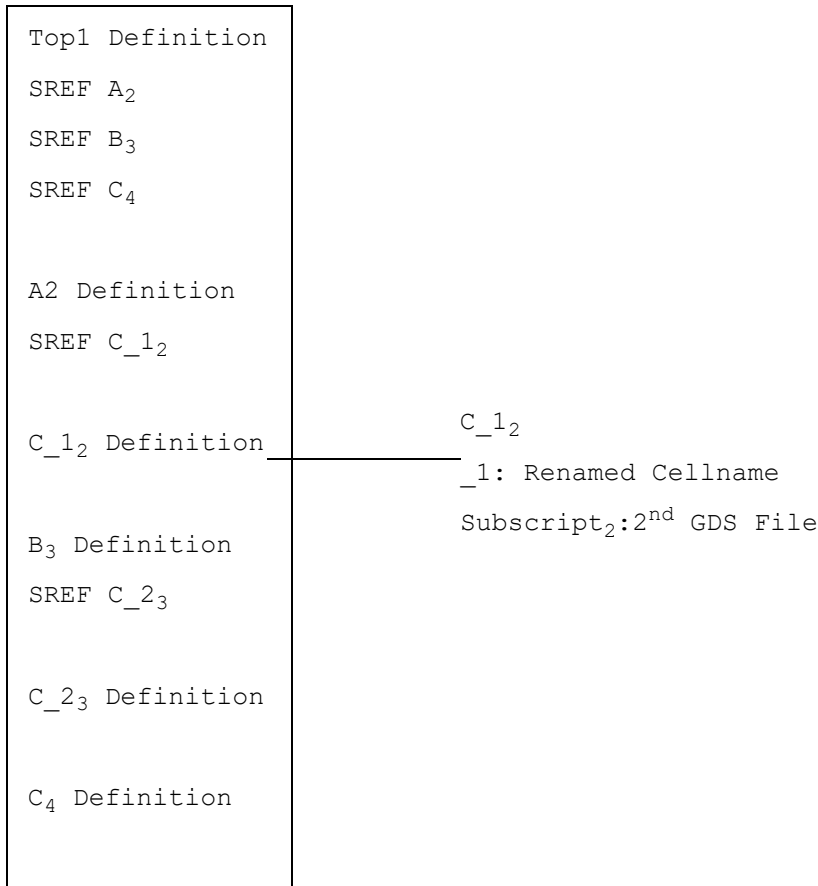
Design Translation Using XStream Translator

Example 2-12 XStream In output: Write mode is rename

TopCell ₁ Definition	
SREF A ₁	
SREF B ₁	
SREF C ₁	
A ₁ Definition	
SREF C ₁	
B ₁ Definition	
SREF C ₁	
C ₁ Definition	
A ₁₂ Definition	C ₁₂
SREF C ₁₂	₁ : Renamed Cellname
C ₁₂ Definition	Subscript ₂ :2 nd GDS File
B ₁₃ Definition	
SREF C ₂₃	
C ₂₃ Definition	
C ₃₄ Definition	

If the *Write Mode* is set to *rename* and the *Overwrite Root Cells* option is selected, then the expected XStream In output is the shown below.

Example 2-13 XStream In output: Write mode (rename) and Overwrite Root Cells



Note: If you set the `xstInOverwriteLibPopup` SKILL variable to `t` and *Write* mode to *overwrite* or *append*, then when you try to perform XStream In translation to an existing library, a message will be displayed to warn you that cells in the target library will be overwritten or appended to in this process.

Enable Incremental Import

Use the *Enable Incremental Import* option to import multiple GDS files into the same design library. The two methods to import the GDS files are:

- Specify all the GDS files in one XStream In translation.
- Specify a subset of GDS files at a time. For this, multiple XStream In translations are needed to import all the GDS files. This is called incremental import mode. During the incremental import mode, you need to select the *Enable Incremental Import* option from the first XStream In translation.

Design Data Translators Reference

Design Translation Using XStream Translator

Default value: Deselected

XStream In creates a temporary file to maintain incremental import data at the path:
`.cadence/geoTrans/.incrementalImport`

Also, a limitation for this option is that one directory can handle one incremental import. However, you can perform XStream In translation, with the *Enable Incremental Import* option deselected, in the same directory.

Command-line option: `enableIncrementalImport`

Enable File Locking

Use the *Enable File Locking* option to enable file locking during XStream In. By default, file locking will not be enabled.

If the target library exists, then this option is automatically turned off to avoid data corruption because the target library might be in use by some other process.

Default value: Deselected



If you are using the Disable file locking option, ensure that the target library is not accessed by another user simultaneously during XStream In. This option brings in chances of data corruption if a cellview is accessed by XStream In as well as by another user or any other process at the same time.

Default: `off`

Command-line option: `enableLocking`

Compress

Use the *Compress* option to enable compression during XStream In.

Default value: Deselected

Command-line option: `compress`

Compression Level

Use the *Compression Level* option to specify the level of library compression to be applied during XStream In. You can specify a value from 1 to 9 for this option. Value 1 signifies the lowest level of library compression and 9 signifies the highest level of library compression. If you do not specify any value in the *Compression Level* option, value 1 is considered as the library compression level during XStream In. This option is enabled only if the *Compress* option is selected.

Valid values: 1 to 9

Default value: 1

Command-line option: `compressLevel`

Maximum Cells in Target Library

Use the *Maximum Cells in Target Library* field to specify the maximum number of cell that you want to include in the target library. If the number of cells are more than the cells you have specified, then the other library will be created with the same name followed by “_1”. For example, if the number of cells in the target library, `trglib` exceeds the limit of the value specified in the *Maximum Cells in Target Library* field, then the new library will be created as, `trglib_1`. The number following the file name will be incremented by 1 depending on the number of libraries that are created. For example, the another library will be created as, `trglib_2`.

Valid values: 1 to 1024000

Default value: 30,000

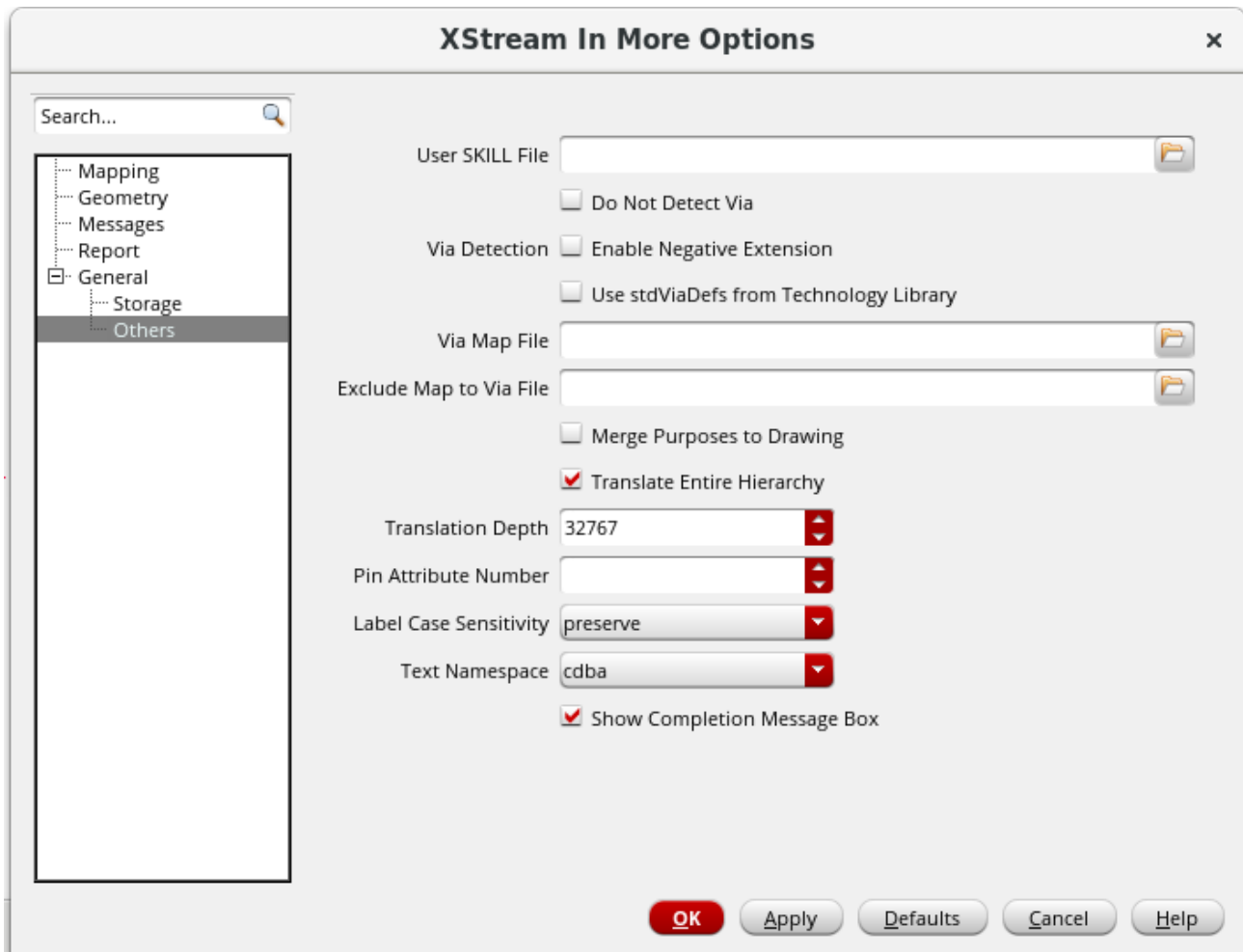
Command-line option: `maxCellsInTargetLib`

Design Data Translators Reference

Design Translation Using XStream Translator

General: Others Options

This section describes the options on the General: Others option pane.



User SKILL File

The *User SKILL File* field lets you specify a file of SKILL routines. Using this file, you can specify the mapping mechanism for cell names, layers, and fonts through the use of programming routines written in SKILL.

In order to load all the required SKILL procedures in a single file, use only one SKILL file during translation. The SKILL file is searched with respect to the run directory if the complete path of the SKILL file is not specified. For more information, see the [Chip Finishing Flow](#) video.

Note: XStream In supports only following SKILL Core, `db`, `dd`, `tech`, and `cst` SKILL functions. Application specific functions, such as `le`, `hi`, `sch`, `lx`, `via` and so on are not supported.

Command-line option: `userSkillFile`

Do Not Detect Via

Select the *Do Not Detect Via* option to disable via detection during translation.

In case, there is no via map file and the *Do Not Detect Via* option is not specified, the tool evaluates the system and if it is possible, performs via detection. This process creates standard and custom vias during XStream In.

If you do not have a via map file, and the option *Do Not Detect Via* is selected, then the translator will produce multiple OpenAccess designs to represent a via. This affects the performance of XStream In and other applications, which work on these OpenAccess designs adversely.

Via detection is a color-aware process. Therefore, when the *Coloring Mode* option is enabled, color information on vias is appropriately set during translation. (See, [Coloring Mode](#))

Note: The vias are not detected automatically for cdsGen vias.

Default value: Deselected

Command-line option: `noDetectVias`

Via Detection

Via detection is enabled by default during XStream In translation. You can specify the following additional options to be considered during translation.

■ ***Enable Negative Extension***

Use the *Enable Negative Extension* option to detect vias with the negative metal extension during via detection. Vias with the negative extension have cut shapes that are partially overlapping with the metal shapes.

This option is disabled when the *Do Not Detect Via* option is selected.

Default value: Deselected

Command-line option: `enableNegativeViaExt`

■ ***Use stdViaDefs from Technology Library***

Use the *Use stdViaDefs from Technology Library* option to enable using the standard via definitions from the technology database for via detection.

Default value: Deselected

Command-line option: `useExistingViaDefsForDetection`

Via Map File

Use the *Via Map File* field to specify a via map file. The via map file maps the Structure names from the Stream file to the corresponding viaDef names in the input technology library. For more information on the via map file, see [Via Map File](#).

Command-line option: `viaMap`

Exclude Map to Via File

Use the *Exclude Map to Via File* field to specify a file name that contains the list of STRUCTs. In XStream In, vias are detected by default. Therefore, when the *Exclude Map To Via File* option is specified, XStream In does not convert the STRUCT names as Viadefs.

The via exclusion file contains one GDS structure per line. This option supports the use of following wildcards:

Wildcard	Action
\$	Matches one unknown character.
*	Matches zero or more unknown characters.
\	Works as escape character. Use this when you want a wildcard character to be treated as a normal character.

Sample Via Exclusion File:

```
# List of structures to be excluded from via mapping
M1M2Wire      # Do not map M1M2Wire to a via
M2M3Wire*     # Do not map cells starting with M2M3Wire to a via
```

Command-line option: `excludeMapToVia`

Merge Purposes to Drawing

Use the *Merge Purposes to Drawing* option to map all Stream datatypes to the purpose `drawing` during XStream In. This option is applicable only if XStream does not find a layer map file or a SKILL file. For more information about automatic layer mapping, see [Automatic Layer Mapping](#).

Default value: Deselected

Command-line option: `mergeUndefPurposToDrawing`

Translate Entire Hierarchy

Use the *Translate Entire Hierarchy* option to translate the entire design hierarchy.

Default value: Selected

Translation Depth

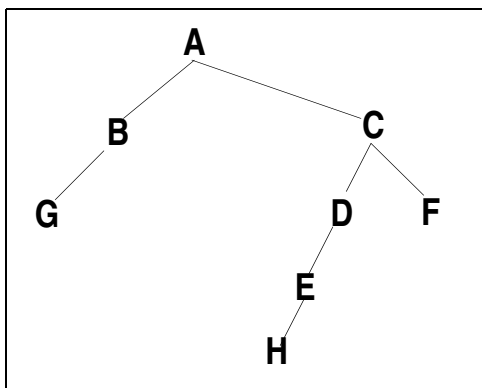
Use the *Translation Depth* option to specify the depth of the design hierarchy during XStream In. XStream In stops the translation process at the hierarchy level indicated by this option. The hierarchy depth value applies to all the top cells getting translated to OpenAccess. This option is enabled only if the *Translate Entire Hier* option is deselected.

Value values: 0 to 32767

Default value: 32767

Command-line option: `hierDepth`

Example 2-14 Translating cells C, D, and F by using the Translation Depth option



The above hierarchy shows cells A, B, C, D, E, F, G, and H placed at different levels. When cell C is selected as the top cell and the hierarchy depth is selected as 1, then the cells C, D, and F are translated.

An instance of E is created in the structure D, but its structure information is ignored. Cell H is ignored and is not translated to OpenAccess.

Pin Attribute Number

Use the *Pin Attribute Number* field to specify the Stream attribute number at which pin information is present in the input Stream file. For more information, see [Preserving Pin Connectivity Information](#).

Valid values: 0 to 127

Command-line option: `preservePinAtt`

Label Case Sensitivity

Use the *Label Case Sensitivity* drop-down list to specify if XStream In changes the casing of text labels. The following options are available:

- *lower*
The *lower* option translates the text labels to lowercase. For example, the label `Obj1` becomes `obj1`.
- *upper*
The *upper* option translates the text label to uppercase. For example, the label `Obj1` becomes `OBJ1`.
- *preserve*
The *preserve* option translates text labels without changing case. For example, the label `Obj1` remains `Obj1`.

Default value: *preserve*

You can further modify text labels by using SKILL call back `piTextMap` function.

Command-line option: `labelCase`

Text Namespace

Use the *Text Namespace* field to specify the namespace mapping for text records in the Stream file. This option enables you to specify the name space in which the TEXT records of the Stream file are in.

For consistency, any of the OpenAccess namespaces (except OS namespaces like `oaUnixNS` and `oaWinNS`) can be specified. The valid arguments for this option are `native`, `cdba`, `lef`, `def`, `verilog`, `vhdl`, `spcf`, `spice`, and `spf` is used as the default option.

For example, if a TEXT record in the Stream file has text `bus[7]` and you specify options value as `verilog`, the text object created in OpenAccess design will be `bus<7>`.

Command-line option: `strmTextNS`

Show Completion Message Box

The *Show Completion Message Box* check box enables you to display the *Stream in translation complete* message box with the number of errors and warnings after the translation is completed. The state of this check box depends on the value of the `xstShowCompletionMsgBox` SKILL variable, which is `t` by default.

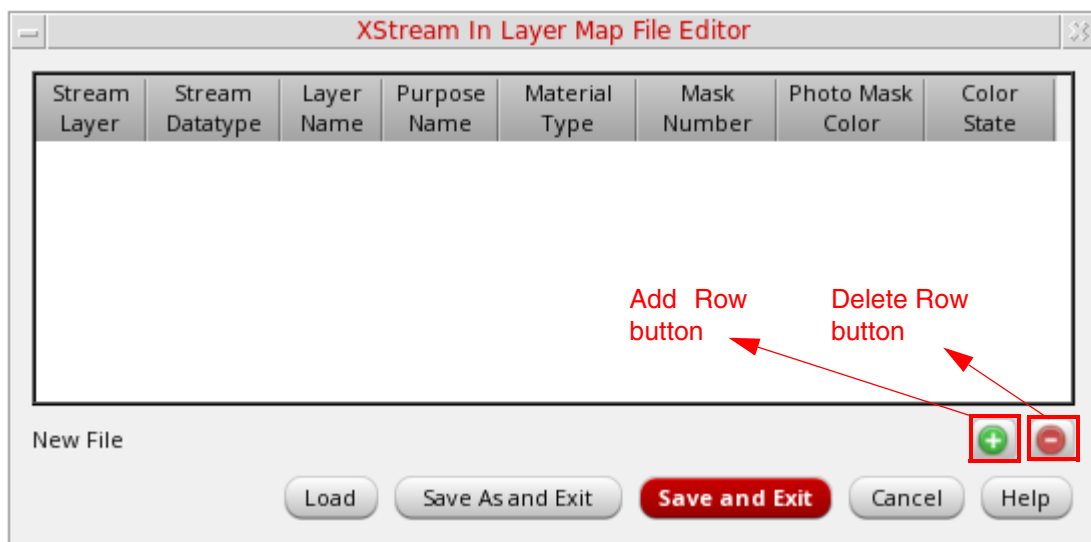
XStream In Editor Forms

This section discusses the following XStream In Editor Forms:

- [XStream In Layer Map File Editor Form](#)
- [XStream In Object Map File Editor Form](#)
- [XStream In Cell Map File Editor Form](#)
- [XStream In Reference Library File Editor Form](#)
- [XStream In Property Map File Editor Form](#)
- [XStream In Font Map File Editor Form](#)

XStream In Layer Map File Editor Form

This section describes options on the XStream In Layer Map File Editor form that is displayed when you click the *Editor* button for the [Layer Map](#) option.



Add Row

Click the *Add Row* button to create a new layer mapping record.

For more information on columns in a layer mapping record, see [Layer Map File](#).

Stream Layer

Use the *Stream Layer* field to specify the Stream layer name that you want to map to OpenAccess layer.

Stream Datatype

Use the *Stream Datatype* field to specify the Stream datatype that you want to translate to OpenAccess purpose.

Layer Name

Use the *Layer Name* field to specify the OpenAccess layer name to which you want to map the Stream layer.

Purpose Name

Use the *Purpose Name* field to specify the OpenAccess purpose name to which you want to map the Stream type.

Material Type

Use the *Material Type* field to specify the OpenAccess material type to which you want to map the Stream type.

Mask Number

Use the *Mask Number* field to specify the OpenAccess mask number to which you want to map the Stream type.

Photo Mask Color

Use the *Photo Mask Color* field to specify the photo mask color of the layer.

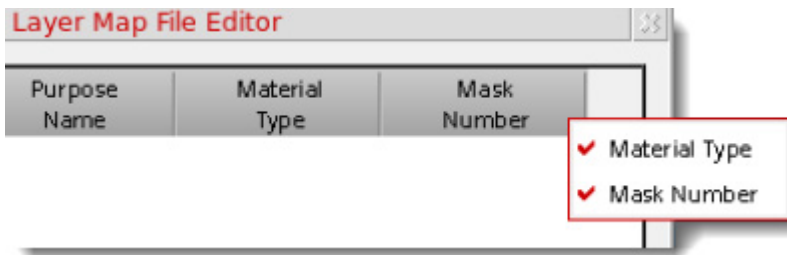
Color State

Use the *Color State* field to specify the color state of the layer.

The *Photo Mask Color* and *Color State* fields are visible only if the *Coloring Mode* option is enabled (See, [Coloring Mode](#)).

If you want to populate and select the values corresponding to the fields mentioned above, then you need to enter the valid GDS/Stream file in the *Stream File* field. In case you want to add or modify the values in any of the above mentioned fields, you need to double-click in the cells.

Show/Hide Columns



To hide either or both *Material Type* and *Mask Number* columns, right-click on the column heading row. Next, clear the check box of the column you do not want to display.

Delete Row

Click the *Delete Row* button to delete an existing layer mapping record.

Load

Use the *Load* button to load a new layer map file. You can specify multiple layer name map files by using the Load file option iteratively. A layer map file maps Stream layer number and Stream data type pairs to OpenAccess layer name and purpose name pairs. If you specify an invalid file name, a warning message is generated and the file name is ignored.

The path specified for the layer map file can be absolute or relative to the run directory. For more information, see [Layer Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the mapping information in a new file and exit the layer map file editor.

Save and Exit

Use the *Save and Exit* button to save the mapping information in the current file and exit the layer map file editor.

Cancel

Use the *Cancel* button to exit XStream In Layer Map File Editor form without saving any changes.

XStream In Object Map File Editor Form

This section describes options on the XStream In Object Map File Editor form that is displayed when you click the *Editor* button for the Object Map option.

Object Name	Sub Object Name	Layer Name	Stream Layer	Stream Type	Photo Mask Color

New File

Load Save As and Exit **Save and Exit** Cancel Help

Add Row button Delete Row button

Add Row

Click the *Add Row* button to create a new object mapping record.

For more information on columns in an object mapping record, see Object Map File.

Object Name

Use the *Object Name* field to select the name of the object to be mapped.

Sub Object Name

Use the *Sub Object Name* field to select the name of the sub object to be mapped.

Layer Name

The *Layer Name* field is displayed only when the Object Name is specified as layerBlockage. However, *N/A* will be displayed in the *Layer Name* field in case of any other Object Name.

Stream Layer

Use the *Stream Layer* field to specify the Stream layer number for the OpenAccess object. By default, the Stream layer number is specified as 0.

Stream Type

Use the *Stream Type* field to specify the Stream datatype for the OpenAccess object. By default, the Stream datatype is specified as 0.

Photo Mask Color

Use the *Photo Mask Color* field to specify the photo mask color of the layer.

The *Photo Mask Color* field is visible only if the *Coloring Mode* option is enabled. (See, [Coloring Mode](#).)

For more information, see [Object Map File](#)



A one-time warning will be displayed if the layer/object map file has color entries, but the Coloring Mode is not enabled. In addition, these entries will not be visible in the layer map file and object file editors. Also, these entries will be ignored during translation. However if you select Coloring Mode is enabled, then these entries are visible. (See, [Coloring Mode](#))

Delete Row

Click the *Delete Row* button to delete an existing object mapping record.

Load

Use the *Load* button to load the object map file. An object map file is used for preserving OpenAccess objects. If you specify an invalid file name, a warning message is generated and the file name is ignored.

The path specified for the object map file can be absolute or relative to the run directory. For more information, see [Object Map File](#).

Design Data Translators Reference

Design Translation Using XStream Translator

Save As and Exit

Use the *Save As and Exit* button to save the object mapping information in a new file and exit the object map file editor.

Save and Exit

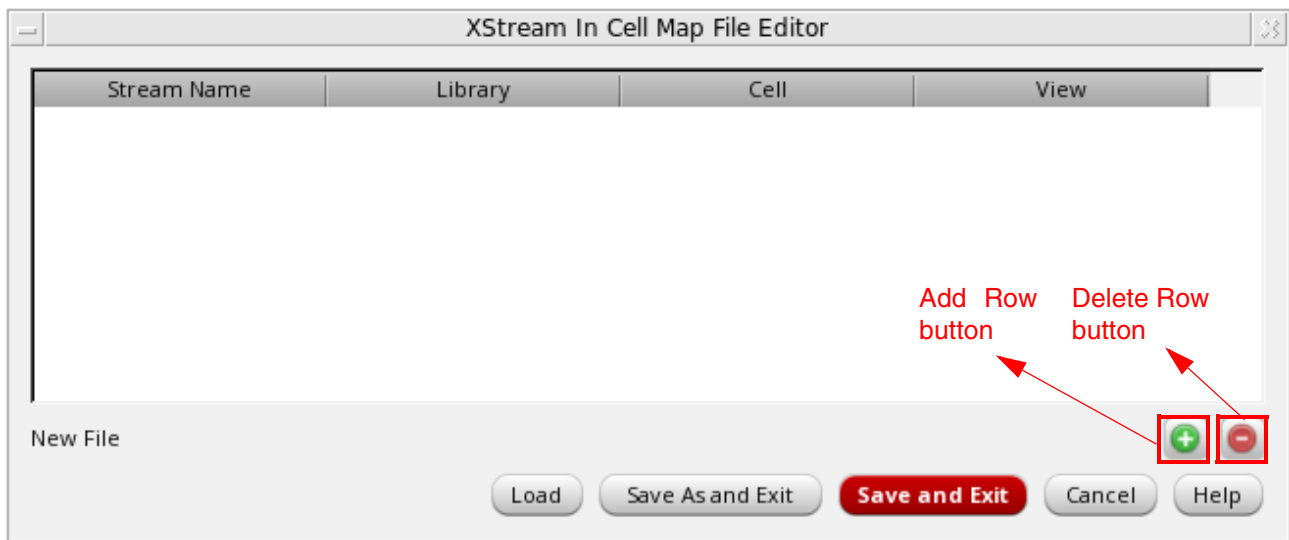
Use the *Save and Exit* button to save the object mapping information in the current file and exit the object map file editor.

Cancel

Use the *Cancel* button to exit the XStream In Layer Map File Editor form without saving any changes.

XStream In Cell Map File Editor Form

This section describes options on the XStream In Cell Map File Editor form that is displayed when you click the *Editor* button for the Cell Map File Name option.



Add Row

Click the *Add Row* button to create a new cell mapping record.

For more information on columns in a cell mapping record, see Cell Name Map File.

Stream Name

Use the *Stream Name* field to specify a name for the Stream file that you want to translate to OpenAccess.

Library

Use the *Library* option to select the library that you want to translate to OpenAccess. The *Library* drop-down list box lists all the available libraries defined in `cds.lib` as well as `XST_TARGET_LIB`.

Cell

Use the *Cell* field to specify the cell name that you want to translate to OpenAccess.

View

Use the *View* field to specify the view name of the specified cell that you want to translate to OpenAccess.

In case you want to add or modify the values in any of the above mentioned fields then you need to double-click in the cells.

Delete Row

Click the *Delete Row* button to delete an existing cell mapping record.

Load

Use the *Load* button to load a new cell name map file. A cell name map file contains a table that maps structure names in the Stream file to cellview names in the OpenAccess database.

The path specified for the cell name map file can be absolute or relative to the run directory. For more information, see [Cell Name Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the object mapping information in a new file and exit the cell map file editor.

Save and Exit

Use the *Save and Exit* button to save the object mapping information in the current file and exit the cell map file editor.

Design Data Translators Reference

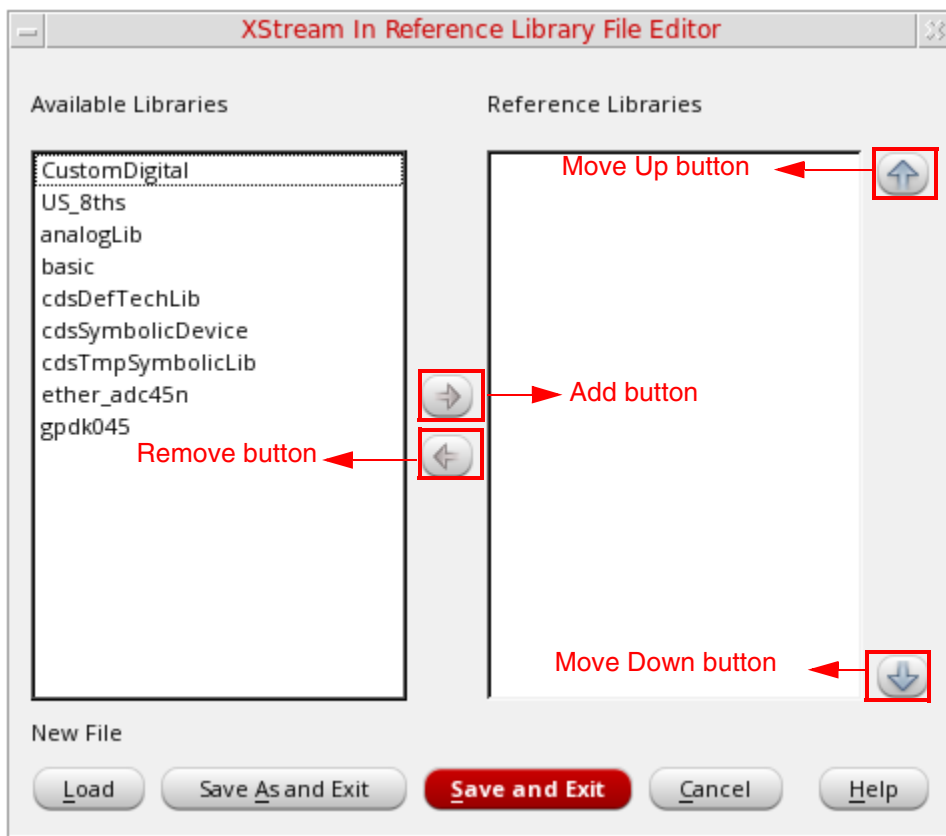
Design Translation Using XStream Translator

Cancel

Use the *Cancel* button to exit the XStream In Cell Map File Editor form without saving any changes.

XStream In Reference Library File Editor Form

This section describes options on the XStream In Ref Lib File Editor form that is displayed when you click the *Editor* button for the Ref Lib File Name option.



Note: If you want to specify the target library name as reference library then you need to use the `XST_TARGET_LIB` variable in the *Available libraries* field.

Add

Use the *Add* button to add libraries to the *Reference Libraries* field.

Design Data Translators Reference

Design Translation Using XStream Translator

The *Reference Libraries* field is used to specify logical library names that you want to consider as reference libraries. The libraries specified in this field should be present in the `cds.lib` file. Otherwise, they are not considered as reference libraries.

During XStream In, when you search for master cellviews of instances, reference libraries are searched in the same order as specified in the *Reference Libraries* field. If any of the specified libraries do not exist in the `cds.lib` file, an error message is displayed.

Remove

Use the *Remove* button to remove libraries from the *Reference Libraries* field.

Move Up

Use the *Move Up* button to move up selected reference libraries in the Reference Libraries list on the right. This helps you arrange reference libraries in order of preference.

Move Down

Use the *Move Down arrow* button to move down selected reference libraries in the Reference Libraries list on the right. This helps you arrange reference libraries in order of preference.

Load

Use the *Load* button to load the text file containing the list of reference libraries. When you specify the text file, the path and the text file name is displayed in the blank field above the *Load* button.

Save As and Exit

Use the *Save As and Exit* button to save the reference library information in a new file and exit the reference library map file editor.

Save and Exit

Use the *Save and Exit* button to save the reference library information in the current file and exit the reference library map file editor.

Design Data Translators Reference

Design Translation Using XStream Translator

Cancel

Use the *Cancel* button to exit the XStream In Reference Library File Editor form without saving any changes.

XStream In Property Map File Editor Form

This section describes options on the XStream In Prop Map File Editor form that is displayed when you click the *Editor* button for the Prop Map File Name option.

The screenshot shows the 'XStream In Property Map File Editor' window. It features a table with columns for 'Property ID', 'Object Type', and 'Property Name'. Below the table, there are 'Add Row' and 'Delete Row' buttons, which are highlighted with red boxes and labeled with red arrows. The 'Add Row' button is a green square with a white plus sign, and the 'Delete Row' button is a red square with a white minus sign. Below the table, there is a 'New File' section with an 'Option Line' containing a checkbox for 'Property Value Only' and a 'Property Separator' field with a comma. At the bottom, there are buttons for 'Load', 'Save As and Exit', 'Save and Exit' (highlighted in red), 'Cancel', and 'Help'.

Add Row

Click *Add Row* button to create a new property mapping record.

For more information on columns in a property mapping record, see Property Map File.

Property ID

Use the *Property ID* field to specify an ID for the Stream file.

Object Type

Use the *Object Type* option to select an object that you want to translate to OpenAccess.

Property Name

Use the *Property Name* field to specify the property name that you want to map to OpenAccess.

In case you want to add or modify the values in any of the above mentioned fields then you need to double-click in the cells.

Delete Row

Click the *Delete Row* button to delete an existing property mapping record.

Property Value Only

Use the *Property Value Only* format specifier to import the property value of a Stream attribute from the GDS file as the value of the specified propertyName of an objectType.

Default value: Deselected

Property Separator

Use the *Property Separator* format specifier to specify a property separator. If you specify a property separator in the property map file, XStream In ignores the property separator defined in the form.

Default value: The default property separator is a comma ‘,’.

Load

Use the *Load* button to load the property map file. A property map file is used for preserving properties associated with OpenAccess objects. If you specify an invalid file name, a warning message is generated and the file name is ignored.

The path specified for the property map file can be absolute or relative to the run directory. For more information, see [Property Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the property map information in a new file and exit the property map file editor.

Save and Exit

Use the *Save and Exit* button to save the property map information in the current file and exit the property map file editor.

Cancel

Use the *Cancel* button to exit the XStream In Prop Map File Editor form without saving any changes.

Property Separator

Use the *Property Separator* option to specify a property separator.

Default value: The default property separator is a comma ‘,’.

XStream In Font Map File Editor Form

This section describes options on the XStream In Font Map File Editor form that is displayed when you click the *Editor* button for the Font Map File Name option.

The screenshot shows a dialog box titled "XStream In Font Map File Editor". It contains a section titled "Font vs Stream Value Mapping" with four drop-down menus. The first menu (0) is set to "Fixed", the second (1) to "EuroStyle", the third (2) to "Gothic", and the fourth (3) to "Math". At the bottom of the dialog, there is a "New File" label and five buttons: "Load", "Save As and Exit", "Save and Exit" (which is highlighted with a red border), "Cancel", and "Help".

Font vs Stream Value Mapping

Use the 0, 1, 2, and 3 drop-down lists to create direct mapping for selected fonts.

Load

Use the *Load* button to load a new text font map file for font mapping. You use a text font map file to map Stream text fonts to OpenAccess text font names. If you specify an invalid file name, a warning message is generated and the file name is ignored.

The path specified for the text font map file can be absolute or relative to the run directory. For more information, see [Text Font Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the font mapping information in a new file and exit the font map file editor.

Save and Exit

Use the *Save and Exit* button to save the font mapping information in the current file and exit the font map file editor.

Cancel

Use the *Cancel* button to exit the XStream In Font Map File Editor form without saving any changes.

XStream In Options Names in GUI and Template File

The following table provides a list of all the XStream In GUI options and their corresponding names in the template file. To view the sample template file, see [XStream In Template File](#).

Name in Template File	XStream In GUI Options
arrayInstToScalar	<u>Convert Arrays to Scalar Instances</u>
attachTechFileofLib	<u>Attach Tech Library</u>
case	<u>Cell Case Sensitivity</u>
cellMap	Cell Map File - <u>Load</u>
checkPolygon	<u>Report Bad Polygons and Paths</u>
convertPathToPathSeg	<u>Pin Attribute NumberConvert Paths to Pathsegs</u>
dbuPerUU	<u>DB Units Per User Units</u>
detectOverflow	<u><i>Detect Overflow</i></u>
noDetectViasA	<u>Do Not Detect Via</u>
enableLocking	<u>Enable File Locking</u>
enableIncrementalImport	<u>Enable Incremental Import</u>
enableNegativeViaExt	<u>Enable Negative Extension</u>
excludeMapToVia	<u>Exclude Map to Via File</u>
fontMap	Fonts Map File- <u>Load</u>
hierDepth	<u>Translation Depth</u>
ignoreBoxes	<u>Ignore Box Records</u>
ignoreZeroWidthPath	<u>Ignore Zero Width Paths</u>
infoToWarn	<u>Convert Info Message IDs to Warnings</u>
keepStreamCells	<u>Keep Stream Cells</u>
labelCase	<u>Label Case Sensitivity</u> (preserve, upper, lower)
layerMap	Layer Map File - <u>Load</u>
library	<u>Library</u>
loadTechFile	<u>Load ASCII Tech File</u>
logFile	<u>Log File</u>

Design Data Translators Reference

Design Translation Using XStream Translator

Name in Template File	XStream In GUI Options
maxCellsInTargetLib	<u>Maximum Cells in Target Library</u>
mergeUndefPurposToDrawing	<u>Merge Purposes to Drawing</u>
noInfo	<u>Disable Info Message IDs</u>
noOverwriteCell	<u>noOverwrite</u>
noWarn	<u>Disable Warning Message IDs</u>
objectMap	Object Map File - <u>Load</u>
overflowCellNameSuffix	<u>Overflow Cell Name Suffix</u>
overflowViewName	<u>Overflow View Name</u>
overwriteRootCells	<u>Overwrite Root Cells</u>
pinAttNum	<u>Pin Attribute Number</u>
propMap	Property Map File - <u>Load</u>
propSeparator	<u>Property Separator</u>
reflibList	<u>Ref Lib File Name</u>
replaceBusBitChar	<u>Replace [] with <></u>
runDir	<u>Run Directory</u>
scale	<u>Design Scale Factor</u>
scaleTextHeight	<u>Scale Text Height</u>
showCompletionMsgBox	<u>Show Completion Message Box</u>
skipUndefinedLPP	<u>Skip Undefined LPPs</u>
snapToGrid	<u>Snap to Grid</u>
strmFile	<u>Stream File</u>
strmTextNS	<u>Text Namespace</u>
summaryFile	<u>Summary File</u>
techRefs	<u>Tech Refs</u>
topCell	<u>Translate</u>
translateNode	<u>Translate Nodes</u>
translateUnmappedLPPs	<u>Translate Unmapped LPPs</u>

Design Data Translators Reference

Design Translation Using XStream Translator

Name in Template File	XStream In GUI Options
userSkillFile	<u>User SKILL File</u>
useExistingViaDefsForDetection	<u>Use stdViaDefs from Technology Library</u>
viaMap	<u>Via Map File</u>
view	<u>View</u>
virtualMemory	<u>Import to Virtual Memory</u>
warnToErr	<u>XStream In Warning Messages Form to Convert Warning Messages to Errors</u>
wildCardInCellMap	<u>XStream In Reference Library File Editor Form</u>
writeMode	<u>Write Mode</u>

You can access and input values in XStream GUI using SKILL commands. Using SKILL to populate values in *XStream In* form, you need to enter the `xstInSetField` command in CIW. To access values from *XStream In* form, you need to enter the `xstInGetField` command in CIW.

All the options listed above are valid for `xstInSetField`, `xstInGetField`, template file, and `streamInKeys`. All the options except `virtualMemory` and `showCompletionMsgBox` are valid for command line executable or batch process.

Note: You can include any UNIX environment variable in template file options. For example, if you use `$TECH_DIR`, `$CDS_WORKAREA`, `$USER` variables in a template file and the variables are correctly resolved by XStream.

Seeding the XStream In GUI

You can use the following methods to set default values for a user:

■ Using the XST_IN environment variable

You can assign the XStream In template file as a value of the `XST_IN` shell environment variable.

For example:

```
envSetVal( "xstream" "XST_IN" 'string "demo.txt" )
setenv XST_IN "demo.txt":
```

Design Data Translators Reference

Design Translation Using XStream Translator

Sample XStream In option template file:

```
=====
pinAttNum          "100"
summaryFile        "xStreamIn.sum"
replaceBusBitChar
=====
```

■ Using the streamInKeys SKILL list

You can define a set of option values to load into the XStream In form in the streamInKeys SKILL list. For this, define the pre-seeding options in the .cdsinit file because the system automatically loads this file when you start Virtuoso.

The following is a sample streamInKeys SKILL list.

```
;*****
; Copyright (C) 2001 - 2018
; Cadence Design Systems Inc. All Rights Reserved.
;*****
;
;*****
;
; filename : streamIn.il
;
;*****
streamInKeys=list(nil
'hierDepth          "32767"          ;-(0-32767) - Hierarchy Depth Limit
'maxCellsInTargetLib "30000"         ;-(Maximum Cells in Target Library
'strmTextNS         "cdba"           ;-(Text NameSpace
'arrayInstToScalar  nil              ;-(Convert AREF to Scalar Instances
'attachTechFileOfLib ""              ;-(Attach Techfile of Library
'case               "preserve"       ;-(preserve | upper | lower - Case Sensitivity
'cellMap            ""               ;-(Cell Map File Name
'checkPolygon       t                ;-(Report Bad Polygons
'convertPathToPathSeg nil            ;-(Convert Two Point Path(s) to PathSeg(s)
'compress           ""               ;-(Allow libraries to be compressed
'compressLevel      "1"              ;-(Defines the compression level to use (default: 1)
'detectOverflow     "false"          ;-(Enable overflow cellview creation on reaching maximum limit
of shapes in a cellview
'enableLocking      nil              ;-(Enable File Locking
'enableNegativeViaExt nil            ;-(Enable Negative Extension for Via Detection
'excludeMapToVia     ""              ;-(Via exclusion Data file
'fontMap            ""               ;-(Font Map File Name
'ignoreBoxes        nil              ;-(Ignore BOX records
'ignoreObjectProp    nil              ;-(Ignore Object Properties
'ignoreZeroWidthPath nil            ;-(Ignore Zero Width Path(s)
'infoToWarn         ""               ;-(Treat the Specified Info Messgaes Ids as the Warnings
'keepStreamCells     nil              ;-(Keep Stream File Cells
'labelCase          "preserve"       ;-(preserve | upper | lower - Label Case Sensitivity
'layerMap           ""               ;-(Layer Map File Name
'library            "strmLib"        ;-(Library Name
'loadTechFile       ""               ;-(ASCII Technology File Name
'logFile            "strmIn.log"      ;-(Log File Name
'mergeUndefPurposToDrawing nil        ;-(Merges Undefined Purposes to "drawing" Purpose
'noDetectVias       nil              ;-(Do Not Detect possible viaDefs from Strem File and create vias
'noInfo            ""               ;-(Do not Print the Specified Info Messages Ids
'noWarn            ""               ;-(Do not Print the Specified Warning Messages Ids
```

Design Data Translators Reference

Design Translation Using XStream Translator

```

'objectMap                ""                ; - Object Mapping File Name
'overflowCellNameSuffix   "_xstOverflow"     ; - Cell name suffix of overflow cellviews
'overflowViewName         "layout"          ; - View name of overflow cellviews
'overwriteRootCells       nil               ; - overwrite top Cells in case write mode is rename
'pinAttNum                "127"             ; - (0-127) - Stream Attribute Number To Preserve Pin
'dbuPerUU                 "1000"            ; - DB Units Per User Units
'reportPrecisionLoss      nil               ; - Report Precision Loss
'propMap                  ""                ; - Property Map File Name
'propSeparator            ", "              ; - Property Separator Character
'refLibList                ""               ; - Reference Library List File
'replaceBusBitChar        nil               ; - Replace "[" With "<"
'runDir                   "."               ; - Run Directory
'scale                    "1.00000"         ; - Design Scaling Factor
'scaleTextHeight          "1.00000"         ; - Text Height Scaling Value (+ve value only)
'enableIncrementalImport  nil               ; - Enable incremental import of multiple Stream Files
'showCompletionMsgBox     t                 ; - Show the Completion Message Box
'skipUndefinedLPP         nil               ; - Skip Creating Undefined Layer-Purpose Pair in Technology
'snapToGrid               t                 ; - Snap XY to Grid Resolution
'strmFile                 "test.gds"        ; - Stream File Name
'strmTechGen              ""                ; - Generated the LPP information of specified stream file
'summaryFile              ""                ; - Summary File Name
'techRefs                 ""                ; - Ordered List of Technology Libraries
'topCell                  ""                ; - Top Cell(s) to Translate
'translateUnmappedLPP     nil               ; - Allow undefined LPP to be translated by incrementally
mapping LPP using layerMap file, SKILL hook piLayerMap, and automatic layer mapping
'translateNode            nil               ; - Convert Node(s) to Dot(s)
'userSkillFile            ""                ; - Users' Skill File Name
'useExistingViaDefsForDetection nil         ; - Enable the usage of standard viaDefs from technology database
for via detection
'verbose                  nil               ; - Generate detailed information in the log and summary files
'viaMap                   ""                ; - Via Map File Name
'view                     "layout"          ; - View Name(s) to Translate
'virtualMemory            nil               ; - Do Stream In into Virtual Memory
'warnToErr                ""                ; - Treat the Specified Warning Messgaes Ids as the Errors
'wildCardInCellMap        nil               ; - Wild Card in Cell Map
'writeMode                "overwrite"       ; - write mode of existing cells
)

```

It is possible to combine both the pre-seeding methods but Cadence recommends to avoid using both methods at the same time. If you combine both the methods, `streamInKeys` SKILL list takes precedence over template file.

To prepare a template file for XStream Out, see [Seeding the XStream Out GUI](#).

Handling Non-Manhattan Angle Rotation and Magnification

When the source GDS file contains SREF/AREF with either non-Manhattan angle rotation or non-default magnification, XStream In translates and generates variant of the original master cell. The instance in the translated OpenAccess design refers to the variant, instead of the original master cell.

Each variant has the same content as the original cell, except that the content is either rotated or magnified, or both. If original cell contains instances or vias, variants are generated

Design Data Translators Reference

Design Translation Using XStream Translator

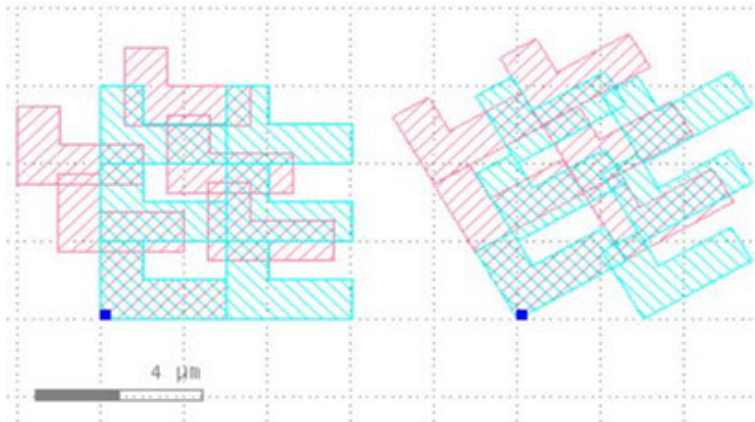
recursively for their masters. The cell name of the variant is different from the original master cell name. The variant name follows the below convention:

```
variantName = originalCellName[_m<mag>][_a<angle>][_v<version>]
```

Here, the *mag* and *angle* are floating-point values written in the `printf` format of `.15g`. The suffix, *version* is appended to the name if a cellview with the same name already exists in the library. In case of such a conflict, versions will be generated regardless of the value of `-writeMode` option.

AREF with non-Manhattan angle rotation is translated as a set of scalar instances. These instances refer to the rotated (and maybe further scaled) variant. The array lattice vectors are not affected by the referenced cell rotation and magnification and are used to calculate the individual instance displacements as is.

In the following example, the AREFs colored red are rotated by 30 degree vectors of the lattice, and the AREFs in green are not rotated. You can see that the parameter `ANGLE` of the figure on the left is 0 degrees and the parameter `ANGLE` of the rotated image on the right is 30 degrees.



When the option *Do Not Detect Via* is not selected, REFs that either point to any-angle or magnified master, or both, are not detected as vias.

Translation of Single Point Path

During XStream In, translation of single-point path to OpenAccess database depends on the end type of the path.

- Single-point paths with end type as round are converted to circle.
- Single-point paths that do not have end type as round are converted to square.

Predicting Memory Requirements

XStream now provides a memory prediction module using which memory requirements can be predicted for a GDS file up to a range of +/-10%. The range might vary if the Stream file contains repeated strings or large number of properties. During XStream In, the memory prediction module runs by default.

The memory prediction module also collects system information for the total virtual memory installed on the system. It compares different process resource limits on the system with the memory requirements and displays error, warning, or informational messages, if required.

If XStream In runs in the non-interactive mode, then these messages appear on the terminal as well as the log file. XStream In does not stop while running in this mode. If XStream In runs in the interactive mode, then these messages are displayed on the terminal and the log file. In addition, you are provided the choice of stopping the translation.

The following table provides a list of messages generated by the memory prediction module and the conditions in which these messages appear:

Message Type	Condition	Message Text
Error	If predicted peak memory falls within 5% of process virtual memory limit.	Process may require more virtual memory than available on the system for a process. You can increase this limit using <code>limit</code> command. Available Memory:YYY MB Predicted Memory:ZZZ MB
Informational	If predicted peak memory falls within 25% of process virtual memory limit.	Process may require more virtual memory than available on the system. You can increase this limit using <code>limit</code> command. Available Memory:YYY MB Predicted Memory:ZZZ MB
Error	If peak memory requirement exceeds 32-bit limits. Solaris 4GB, HP 4GB, Linux 3GB (2.4 Kernel supports more)	Process might require more virtual memory than 32-bit system can handle. You may need to run 64-bit version of the product. Available Memory:YYY MB Predicted Memory:ZZZ MB

Design Data Translators Reference

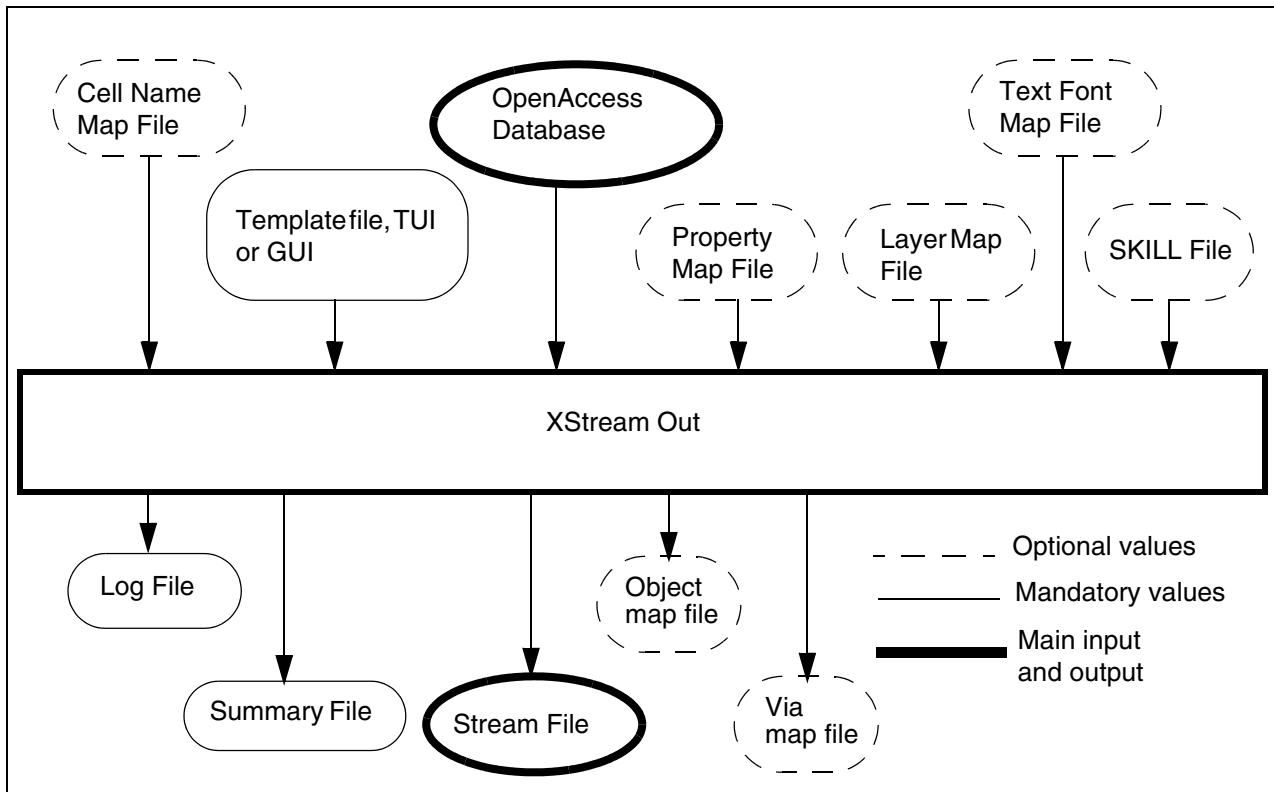
Design Translation Using XStream Translator

Message Type	Condition	Message Text
Error	If predicted peak memory is within 5% range of total SWAP and RAM on the system.	Process might require more SWAP and RAM than available on the system. Available Memory:YYY MB Predicted Memory:ZZZ MB
Informational	If predicted peak memory is within 25% range of total SWAP and RAM on the system.	Process might require more SWAP and RAM than available on the system. Available Memory:YYY MB Predicted Memory:ZZZ MB
Error	If there is not enough DATA segment available.	Memory resize failed for YYY bytes.
Error	If there is not enough of shared memory space.	OS Error Code '12' "Not enough space"

Design Translation During XStream Out

XStream Out Use Model

XStream Out translates designs from the OpenAccess database to the Stream format. The inputs to the XStream Out process are the OpenAccess database, XStream Out options, the optional mapping files, and the user-defined SKILL files. Using these inputs, XStream Out generates Stream file, log file, and an optional summary file. The XStream Out use model is shown below.



The following table provides generic information about input and output files during XStream Out.

File Type	Description
Stream File	The output Stream file.

Design Data Translators Reference

Design Translation Using XStream Translator

File Type	Description
Template File	A file that contains the parameter settings you want to use to run XStream Out. You can load the parameter settings in the template file into the <i>Virtuoso(R)XStream Out</i> form. In the XStream Out template file, '\$' is considered as a special character because it is a UNIX environment variable. However, if you want '\$' to be treated as a normal character, you can escape it by using '\' before it.
SKILL File	A file used to map cell names, fonts, and layer names.
OpenAccess database	The database you want to translate to a Stream file.
Log file	A file used for recording the translation process. It also keeps a log of all the messages.
Summary File	A file that has a summary of the translated design.

For details on map files, see [Design Translation During XStream In](#).

Once you have selected the option to do a strmout, click the *OK* button. The XStream Out form will be displayed.

After you enable the *Generate Technology Information* option, all other fields except the *Stream Technology File* field are disabled.

Multithreading in XStream Out

XStream Out uses multithreading and multiprocessing to improve performance during translation. XStream Out evaluates the machine configuration and then applies the best value to be used for multithreading based on available CPU cores. The maximum number of threads are capped at 4.

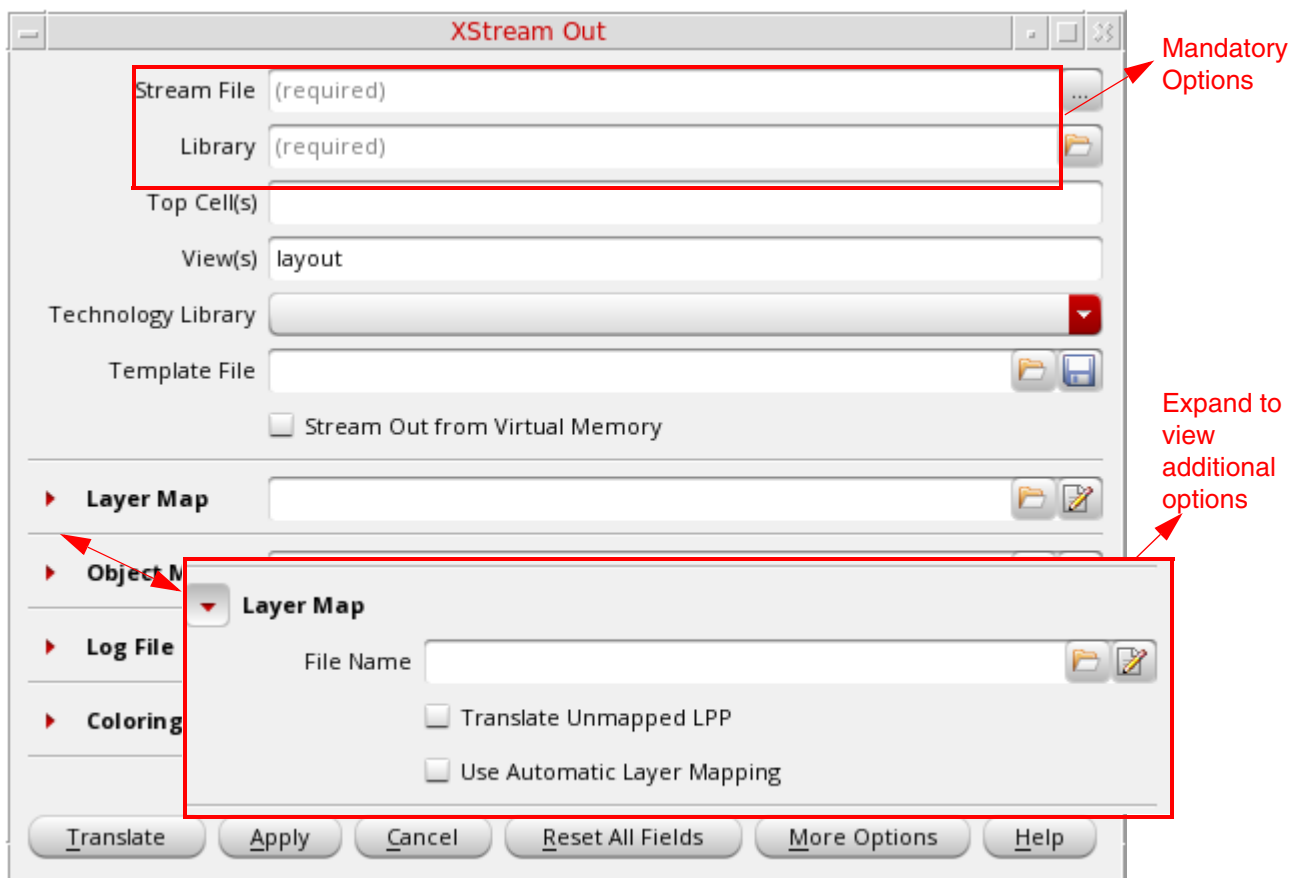
XStream Out Form

You can export the Stream file from the GUI by choosing the *File–Export–Stream* option from the CIW. This method will enable you to display the *XStream Out* form. If a cellview is open when you run this command, the *Stream File*, *Library*, *TopCell(s)*, and *View(s)* fields on this form are automatically filled.

However, if these fields already have some information, you might want to prevent automatic filling even if a cellview is open. For this, set the value of environment variable `xstOutAutoPopulateIfEmptyLCV` to `t`. When you do this, the *Stream File*, *Library*, *TopCell(s)*, and *View(s)* fields on the XStream Out main form will be filled automatically, only when they are empty.

Note: The form displays values from previous XStream Out session, if the changes made were applied before exiting XStream Out.

For a list of equivalent template file options, see [XStream Out Option Names in GUI and Template File](#).



Design Data Translators Reference

Design Translation Using XStream Translator

The main XStream Out form comprises mandatory options that are essential to run the translation. The form also features a few other most commonly used options, such as Layer Map and Object Map.

Note: By default, fields such as *Stream File*, *Library* and *TopCell(s)* in the XStream Out form, consider '\$' as a special character because it is a UNIX environment variable. However, if you want '\$' to be treated as a normal character, you can escape it by using '\' before it.

The following sections on the main form can be expanded to view additional options:

- [Layer Map](#)
- [Object Map](#)
- [Log File](#)

Discussed below are XStream Out options available on the main form.

Stream File

This is the output Stream file. Use the *Stream File* field to specify the name of the output Stream file. Alternatively, *File Browser* button to select the Stream file. The path specified for the output file can be absolute or relative to the run directory. However, if the user has not specified the Stream file name then it will be automatically generated from the top cell. This is a mandatory field.

Command-line option: `strmFile`



You must select a library before Streaming out the data.

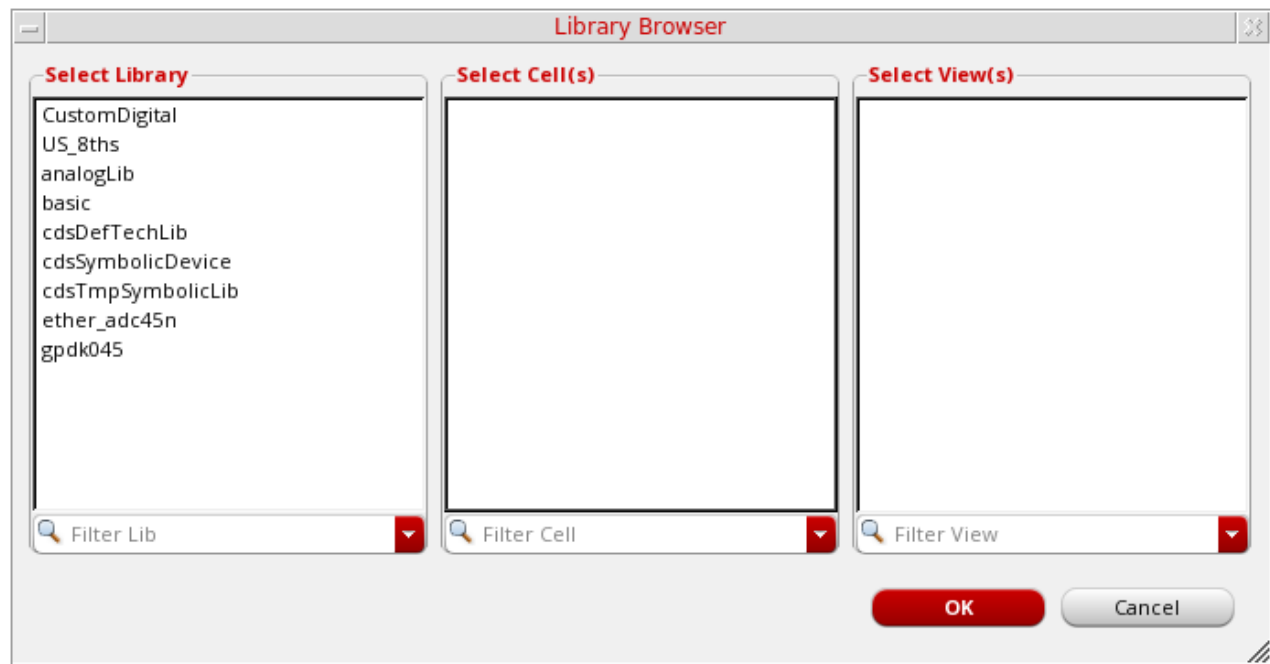
Library

Use the *Library* option to specify a library from which a design is to be translated to the Stream format. The library name should be a logical name supported in the `cds.lib` file. The library name should also follow the naming conventions supported by OpenAccess. This is a mandatory field. You can also click the *Open* button to display the Library Browser form. This form does not display hidden cells.

Command-line option: `library`

Design Data Translators Reference

Design Translation Using XStream Translator



Note: To hide cells, in Library Manager, right-click on a cell name and then select *Hide Cell*.

Note: You can also specify the `-cdslib` option to access the `cds.lib` file from any other directory. For example, to access the `cds.lib` file from any other location, use the syntax:
`virtuoso -cdslib <cds.lib file path>`

Important

You must select a library before Streaming out the data.

Top Cell(s)

Use the *Top Cell(s)* field to specify the name of the cell at the top of the hierarchy that you want to translate to the Stream file.

By default, all cells in the specified library are translated.

Command-line option: `topCell`

Note: A Pcell supermaster can also be specified in the top level cell. It will be written in the Stream file during XStream Out.

Design Data Translators Reference

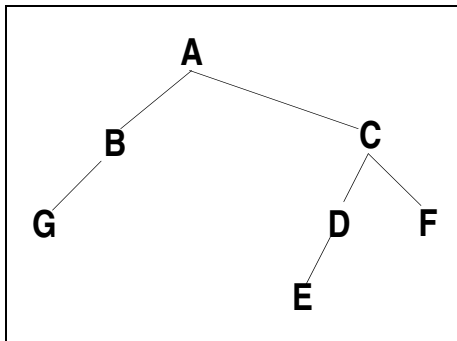
Design Translation Using XStream Translator

After you enter the top level cell in the *Top Cell* field, the Stream file will reflect the filename as `<Top Cell>.<file extension>`. The default file extension name for the output file is `.gds`. You can change the default file extension name from `.gds` to any other string by using the `xstOutExt` SKILL variable.

Note: If you have manually changed the value of *Stream File*, then changing value of the *Top Cell* field will not update the Stream file name automatically. This is because manual editing of Stream file name is given the highest priority. To automatically update the Stream file name, first empty the *Stream File* field and then update the *Top Cell* field.

Example 2-15 Translating cells by using the Top Cell option

Consider the hierarchical structure below where A, B, C, D, E, F, and G are cell names. You want to translate these cells using the *Top Level Cell(s)* field.



When C is specified as the top cell in the *Top Level Cell(s)* field, cells C, D, E, and F are translated. For instance, if you specify the top cell name as B and C, cells B, G, C, D, E, and F are translated. Below is the tabular representation on how the cells will be translated.

Top Level Cell field	Cells translated
B C	B G C D E F
C	C D E F
B D	B G D E

Note: If the Top Level Cell field is left blank, then all the cells will be translated.

Important

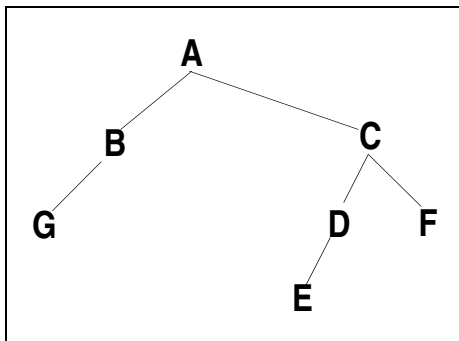
Using command line, you need to give the cell names in quotes. For example "B G".

View(s)

Use the *View(s)* field to specify view names that are created in the target library during translation. If you specify a cellview name in the cell name map file or SKILL file, then the view name specified in these files takes precedence over the name specified in the XStream In form.

Command-line option: `view`

Example 2-16 Translating cells C, D, E, and F into the layout1 view



In the above figure, if C is specified as the top cell and `layout1` as the view name, then the structures C, D, E, and F are translated with `layout1` as the view name.

Default value: `layout`

Technology Library

Use the *Technology Library* option for getting information about layers and purposes used in the design. By default, the technology file of the source library is used to get information about the layers and purposes used in the design.

Command-line option: `techLib`

Template File

Use the *Template File* field to specify an existing template file. You can also click the *File Browser* button to select the template file. When this file is loaded, the XStream Out form is updated with the option settings specified in it.

Click the *Save* button to save the current option settings to the template file.

Command-line option: `templateFile`

Stream Out from Virtual Memory

Select the *Stream Out from Virtual Memory* check box to export OpenAccess designs from virtual memory to a GDSII/Stream file on disk.

Note: If the *Stream Out from Virtual Memory* check box is selected, information and warning messages are not displayed in the CIW. They are logged in the log file.

Layer Map

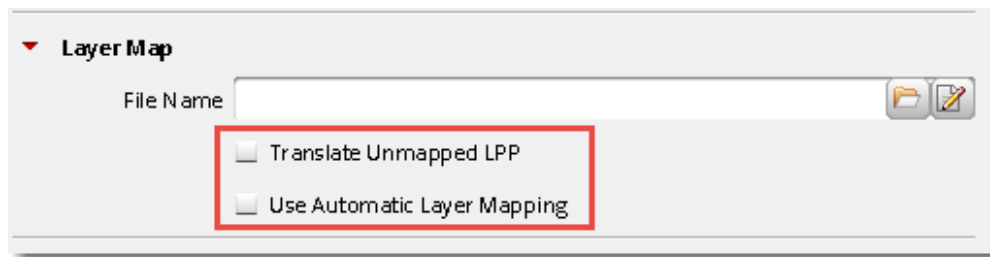


Use the *Layer Map* field to specify an existing layer map file. You can also click the *File Browser* button to select the layer map file. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `layerMap`

To create new layer mapping records or load a new layer map file, leave the *Layer Map* field blank, and click the *Editor* button to display the XStream Out Layer Map File Editor Form.

Expand *Layer Map* to view additional options.



Translate Unmapped LPPs

Use the *Translate Unmapped LPP* option to map unmapped layers. This option works only when a different user-defined layer mapping, layer map file or SKILL function, has been specified. When you enable *Translate Unmapped LPP*, first the user-specified layer mapping is applied, and then, automatic layer mapping is performed to translate unmapped layers. For more information about translating unmapped layers in Stream Out, see Automatic Layer Mapping in XStream Out.

Design Data Translators Reference

Design Translation Using XStream Translator

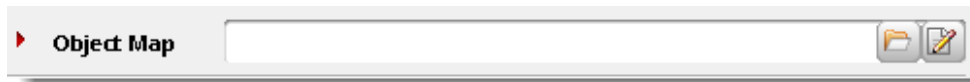
Default value: Deselected

Command-line option: `translateUnmappedLPP`

Use Automatic Layer Mapping

Use the *Use Automatic Layer Mapping* option to map layers automatically. This option works only if no layer mapping has been specified. Otherwise, this option is ignored and layers are mapped with the specified mapping. For more information about automatic layer mapping, see [Automatic Layer Mapping in XStream Out](#).

Object Map

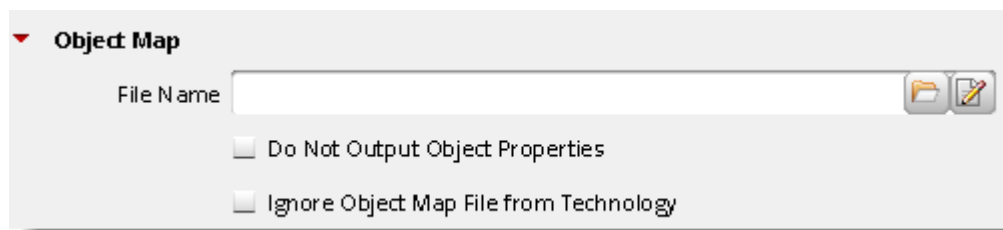


Use the *File Name* field to specify an existing object map file. You can also click the *File Browser* button to select the object map file. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `objectMap`

To create new object mapping records or load a new object map file, leave the *Object Map* field blank, and click the *Editor* button to display the [XStream Out Object Map File Editor Form](#).

Expand *Object Map* to view additional options.



Do Not Output Object Properties

Use the *Do Not Output Object Properties* option to translate OpenAccess objects as shapes, but not write the property information to the property attribute 126. If you select the *Do Not Output Properties* option, then although the shapes and OpenAccess objects are translated, no property data is written to the Stream file for OpenAccess objects.

Design Data Translators Reference

Design Translation Using XStream Translator

Default value: Deselected

Command-line option: `noObjectProp`

Ignore Object Map File from Technology

Use the *Ignore Map File from Technology* to ignore PDK-installed object map file during XStream Out.

Default value: Deselected

Command-line option: `ignoreObjectMapFromTech`

Log File

Expand Log File to specify log and summary files to be used during XStream Out translation.

Use the *File Name* field to specify name of a log file that will be used to record the translation process steps and the messages generated by XStream. After the translation is completed, the log file is displayed in a separate read-only text window. Alternatively, click the Browse button to select the log file.

The path specified for the log file should be a valid file path. It can be absolute or relative to the run directory. For more information, see [Sample Log File](#).

Default file name: `strmOut.log`

Command-line option: `logFile`

Summary File

Use the *Summary File* field to specify a file that will provide a summary of the translated design. If you do not specify a file name in the *Summary File* option before translation, the summary file will not be generated.

Design Data Translators Reference

Design Translation Using XStream Translator

The path specified for the summary file can be absolute or relative to the run directory. To view the sample summary file, see [Sample Summary File](#).

Command-line option: `summaryFile`

Generate Detailed Log and Summary Files

Use the *Generate Detailed Log and Summary Files* option to include additional summary information in the XStream Out log file and XStream Out summary file.

With this option is enabled, following sections with additional information are added to the XStream Out summary file:

- **Individual Cell Statistics:** Individual cell statistics for *Basic Objects* and *Advance Objects* is alphabetically sorted order based on Library/Cell/View.
 - **Bounding box for the cells:** *Individual Cell Statistics for basic objects* prints additional column titled *bBox*. This columns lists the boundary box of the translated cell in the following format:

`((lowerLeft.x lowerLeft.y) (upperRight.x upperRight.y))`

Design Data Translators Reference

Design Translation Using XStream Translator

Here, *lowerLeft* and *upperRight* are the lower-left and upper-right coordinates of the boundary box.

Individual Cell Statistics - Basic Objects

Library/Cell/View	Instance	Array	Polygon	Rect	Path	PathSeg	Text	TextDisplay	Line	Dot	Arc	Donut
Ellipse	bBox											
lib/cell11/layout	0	0	0	1	0	0	0	0	0	0	0	0
((-14.2450 -6.1550) (9.7400 6.1550))												
lib/cell12/layout	1	0	0	3	0	0	0	0	0	0	0	0
((-8.3200 -6.0750) (24.6650 10.6650))												
lib/topcl/layout	1	0	0	3	0	0	0	0	0	0	0	0
((-8.3200 -6.0750) (24.6650 10.6650))												

Individual Cell Statistics - Advanced Objects

Library/Cell/View	PRBdy	OtherBdy	AreaBlkg	LayerBlkg	AreaHalo	Row	Marker	CustVia	StdVia	CdsGenVia
lib/cell11/layout	0	0	0	0	0	0	0	0	0	0
lib/cell12/layout	1	1	0	3	0	0	1	0	0	0
lib/topcl/layout	1	1	0	3	0	0	1	0	0	0

- **Statistics of Layers:** Layer summary added to this section is sorted in the same order in which layer-purpose pairs are defined in the technology file or the layer pellet.

Statistics of Layers

Cadence Layer	Cadence Purpose	Stream Layer	Stream Datatype	Polygon	Rect	Path	Text	TextDisplay	Line	Dot	Arc	Donut	Ellipse	Path
Metal4	drawing	31	0	0	1	0	0	0	0	0	0	0	0	0
Metal7	drawing	38	0	0	1	0	0	0	0	0	0	0	0	0
VPNP2dum	drawing	60	0	0	2	0	0	0	0	0	0	0	0	0

- **Missing Cells:** This section lists the cells that are referenced but are missing in the OpenAccess database.

Missing Cells

Library/Cell/View

LIB/missingcell11/layout
LIB/missingcell12/layout

Design Data Translators Reference

Design Translation Using XStream Translator

- **Dropped Layers:** This section lists the layer-purpose pairs for which no mapping is found. Shapes on these layer-purpose pairs are not translated.

Dropped Layers

Cadence Layer	Cadence Purpose
Oxide	drawing
Poly	drawing
Pimp	boundary

- **Individual Advanced Object Statistics:** This section lists the Place and Route objects translated during XStream Out. It lists the object-type, sub-object type, layer name (for layerBlockage), strmLayer, strmType, and the count of the objects translated.

Individual Advanced Object Statistics

Object Type	Object Sub-type	Layer Name	Stream Layer	Stream Datatype	Count
LayerBlockage	routing	Metal1	0	0	3
Boundary	snap		20	2	1
Marker	annotation		100	0	1
Boundary	PR		20	0	1

Default value: Deselected

Command-line option: `verbose`

Coloring Mode

Expand *Coloring Mode* to specify coloring options to be used during XStream Out translation.

▼ **Coloring Mode** ☒ Enable

Enable

Select the *Enable* check box to enable color mapping during XStream Out.

Default value: Selected

Command-line option: `enableColoring`

Important

The layer map file, which is either specified through the Layer Map field or is available in the technology, is required if you enable *Coloring Mode*. Otherwise, translation is terminated and an error message is displayed. Technology library can be specified using following XStream In options:

- ☐ Attach Tech Library (`attachTechFileOfLib`)
- ☐ Load ASCII Tech File (`loadTechFile`)
- ☐ Tech Refs (`techRef`)

Technology can be either attached to the existing target library or the target library has its own technology.

The Via Cut Aref Threshold option is not supported if coloring mode is enabled. Therefore, the Via Cut Aref Threshold option will be ignored and a warning message will be displayed.

In case Merge Connected PathSegs check box is selected and coloring mode is enabled, the connected pathSegs on the colorable layers will not be merged and each pathSeg will be written as separate PATH record in GDS. The Merge Connected PathSegs option will not be considered for the pathSegs on colorable layers.

In case of the Max Vertices option, the color information of the original polygon or path will be used and all the chopped polygons will be mapped using the same mapping entry.

If the layer map and object map files do not have any color entry, the `-enableColoring` option is not considered during translation.



Video

For a video on how to lock colors and how the locked colors propagate to the streamed out GDSII file, you can see [Color Locking and Color Aware Xstream](#) on Cadence Online Support.

Map All Colors to Locked

This option is visible when `xstHideMapAllColorToLocked` is set to `nil`.

Design Data Translators Reference

Design Translation Using XStream Translator



XStream Out outputs shapes to GDS using the mapping in the layer map file so that the resulting GDS preserves the colors and color lock state for each shape. However, when you choose *Map All Colors to Locked*, there is no change to the mapping for shapes with locked colors and gray (no color) shapes. Shapes that are colored but unlocked, including lower level hierarchical shapes, will map to the locked state for their color, instead of the unlocked state.

Before XStream	After Xstream Out (Default)	After XStream Out with Map All Colors to Locked
Mask1Color, unlocked	Mask1Color, unlocked	Mask1Color, locked
Mask2Color, unlocked	Mask2Color, unlocked	Mask2Color, locked
Mask1Color, locked	Mask1Color, locked	Mask1Color, locked
Mask2Color, locked	Mask2Color, locked	Mask2Color, locked
Gray (no color)	Gray (no color)	Gray (no color)

In case of locked standard via layers, when *Map All Colors to Locked* is enabled, the corresponding shapes are mapped according to the locked color layer map entry in the [Layer Map File](#). Otherwise, if the color state for the standard via layers is unlocked, the color information is ignored and the corresponding shapes are mapped according to the gray layer map entry in the map file.

Command-line option: `mapAllColorToLocked`

Important

You can use the SKILL environment variable `xstHideMapAllColorToLocked` to hide this option by default. Add the following command in `.cdsenv`:

```
xstream xstHideMapAllColorToLocked boolean nil
```

Translate

Use the *Translate* button to translate OpenAccess files to the Stream files. This option, if used without loading the hierarchy, enables faster translation.

Design Data Translators Reference

Design Translation Using XStream Translator

If you have opened cellviews that are not yet saved, the XStream Out GUI displays a warning message to save all the opened cellview before translating

Apply

Use the *Apply* button to translate OpenAccess files to the Stream files. After the translation is completed, the *XStream Out* form does not disappear.

Cancel

Use the *Cancel* button to exit the *XStream Out* form.

Reset All Fields

Use the *Reset All Fields* button to set the GUI fields to their default values, which are displayed when the tool is invoked.

More Options

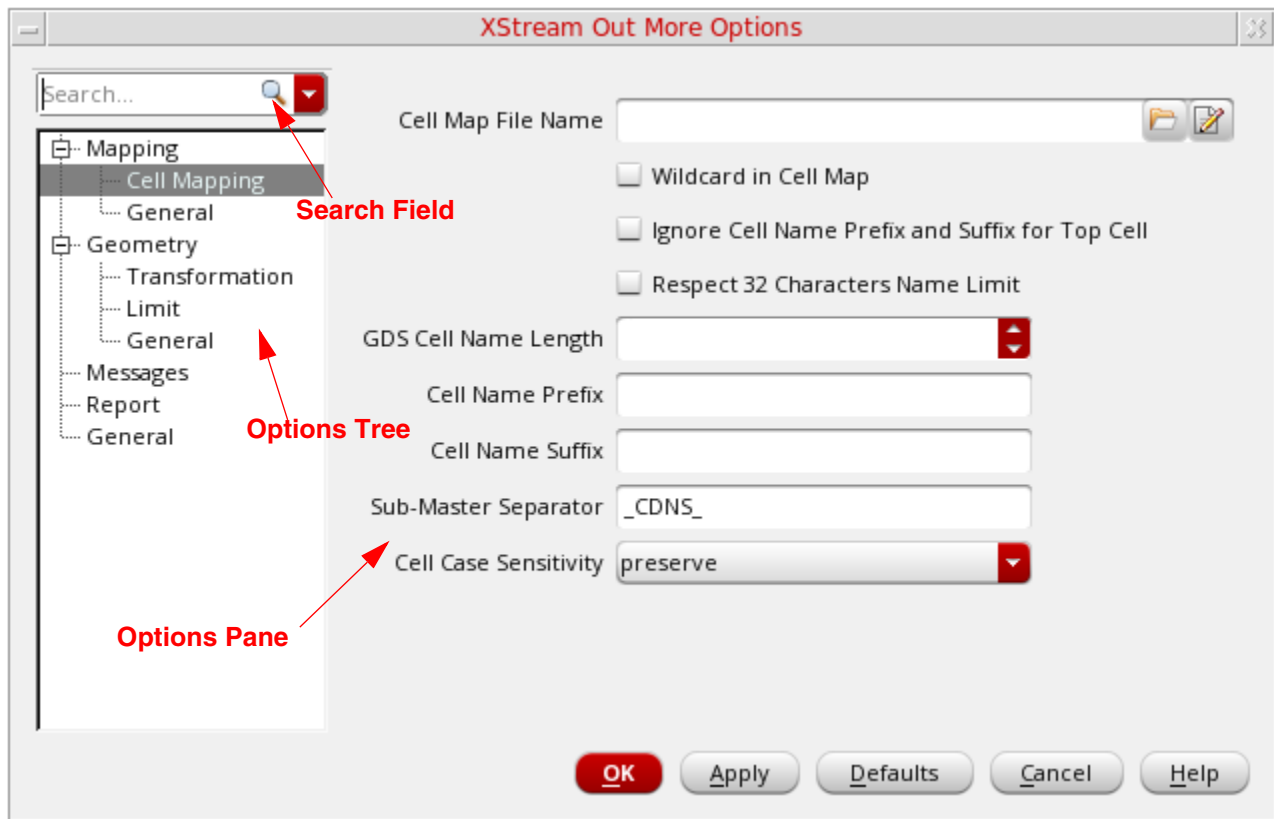
Use the *More Options* button to display the XStream Out More Options form that comprises additional options that you can set before starting XStream Out translation.

Help

Use the Help button to invoke online help.

XStream Out More Options Form

This section describes options of the XStream Out More Options form.



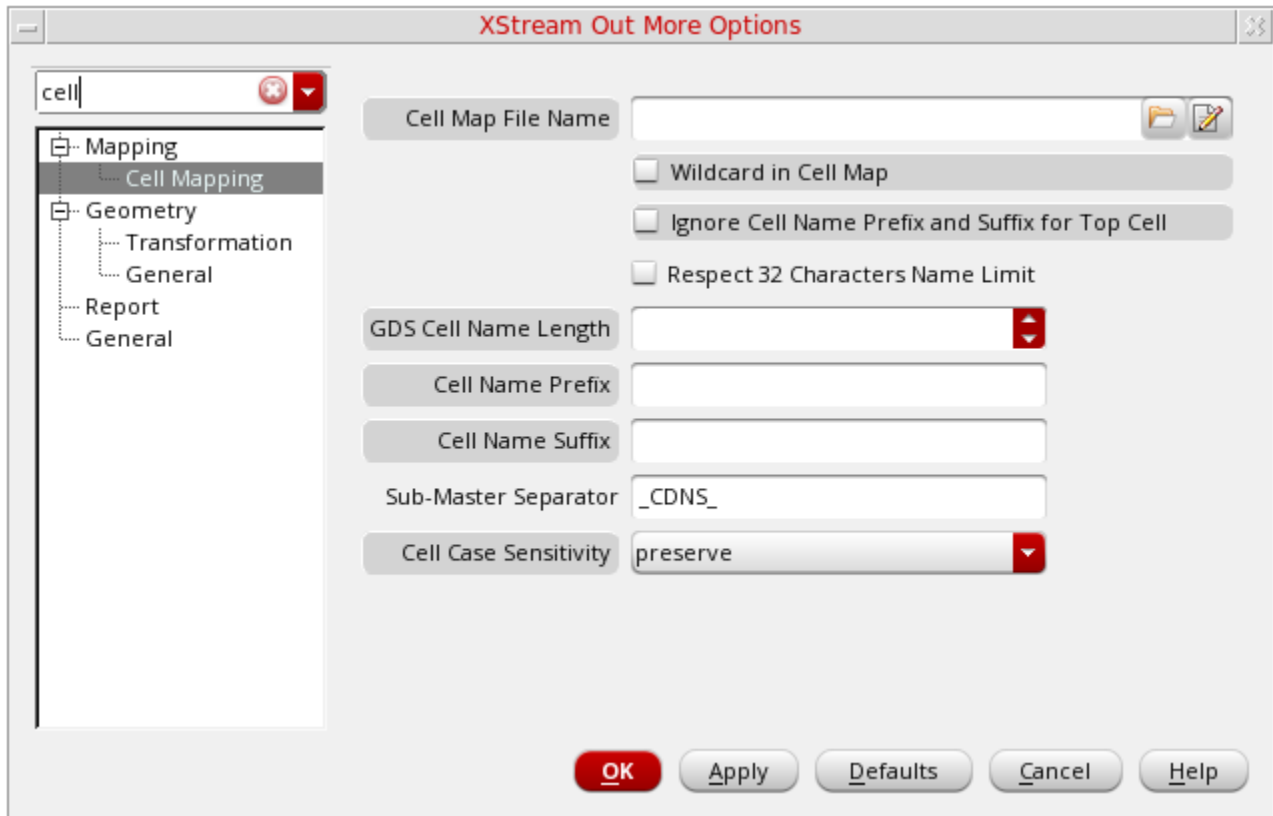
Search Field

Use the Search field to quickly look for options you want to access or the options related to the value you type in this field. For example, if you type `cell`, only the options related to cell

Design Data Translators Reference

Design Translation Using XStream Translator

will be visible in the options tree. The related options will also be highlighted in the options pane.



Options Tree

The Options tree enables you to access the following option panes where you can perform additional settings.

- Mapping Options
 - ☐ Mapping: Cell Mapping Options
 - ☐ Mapping: General Options
- Geometry Options
 - ☐ Geometry: Transformation Options
 - ☐ Geometry: Limit Options
 - ☐ Geometry: General Options

Design Data Translators Reference

Design Translation Using XStream Translator

- [Messages Options](#)
- [Report Options](#)
- [General Options](#)

Options Panes

Options panes comprise different options related to the option currently selected in the tree.

OK

Click *OK* to save your current settings for the More Options form and exit.

Apply

Click *Apply* to save your settings on the More Options form. When you click Apply, the form is not closed.

Defaults

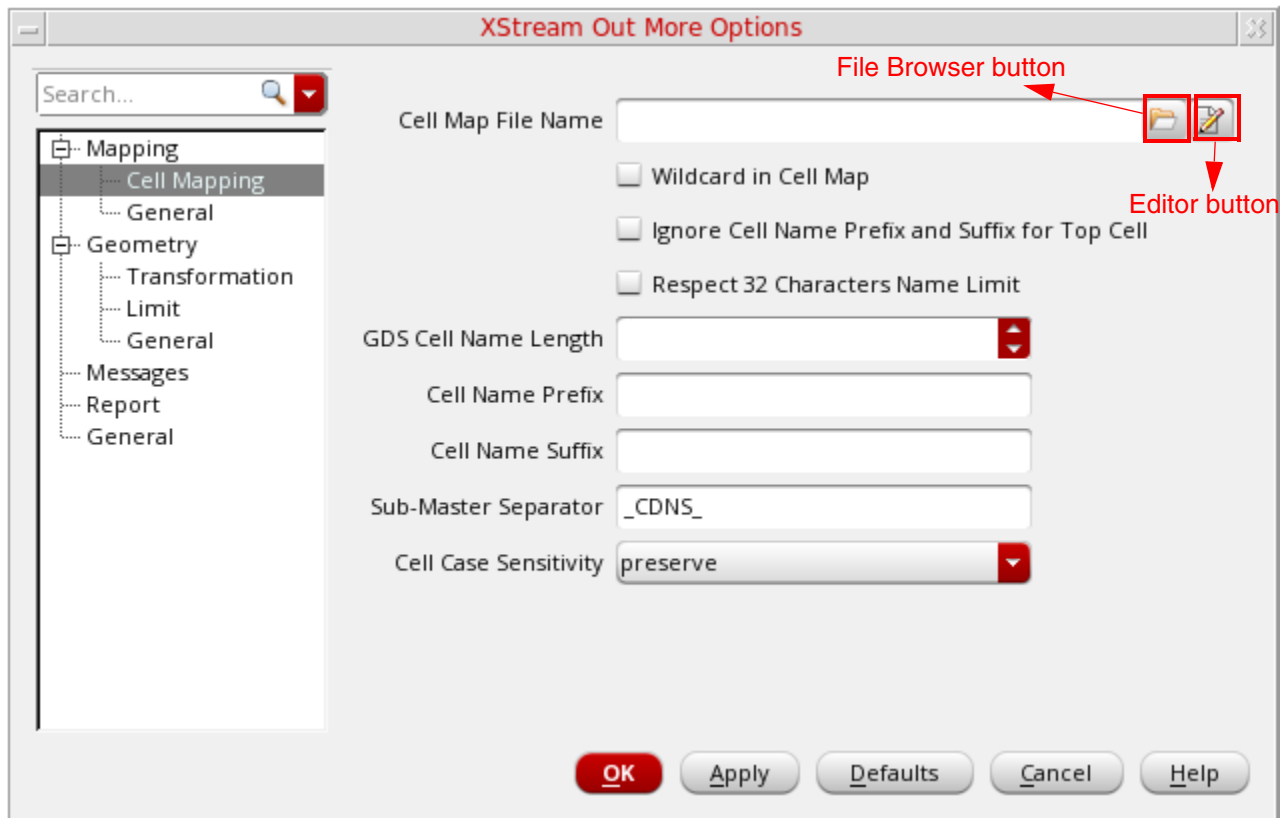
Click *Defaults* to set default values for all options in the More Options form.

Cancel

Click *Cancel* to close the More Options form without saving any changes.

Mapping: Cell Mapping Options

This section describes options on the Mapping: Cell Mapping options pane.



Cell Map File Name

Use the *Cell Map File Name* field to specify an existing cell map file. You can also click the *File Browser* button to select the cell map file. Next, click the *Editor* button to view the specified file in editor.

When *Cell Map File Name* and `poParamCellNameMap` both are specified, `poParamCellNameMap` takes precedence. However, when both *Cell Map File Name* and `poCellNameMap` are specified then *Cell Map File Name* is used.

Command-line option: `cellMap`

To create new cell mapping records or load a new cell map file, leave the *Cell Map* field blank, and click the *Editor* button to display the XStream Out Cell Map File Editor Form.

Wildcard in Cell Map

Use the *Wild Card in Cell Map* option to use wild card while streaming out. The option supports the characters '*' and '?' as wild cards and '/' as the escape character. If you don't use this option, wild card will be considered as a special character during translation.

Consider the following example:

```
*   INV_*           *           INV\*_*_ver1
```

Here, when you stream out, the created structure will be INV*_NAND_ver1 if the design cellName is INV_NAND. In this example, in INV*_*_ver1, the first '*' is treated as special character and the second '*' as the wild card.

Note: If you don't use *Wild Card in Cell Map*, the default behavior will be same as a normal cell map.

Command-line option: wildCardInCellMap

Ignore Cell Name Prefix and Suffix for Top Cell

Use the *Ignore Cell Name Prefix and Suffix for Top Cell* if you want to ignore the Cell Name Prefix and Cell Name Suffix for top level cell(s).

Default value: Deselected

Command-line option: ignoreTopCellPrefixSuffix

Respect 32 Characters Name Limit

Use the *Respect 32 Characters Name Limit* option if you want to set the GDSII character limit of 32 for cells and library names. If a cellview name is greater than 32, it is modified to create unique name within 32 characters. A record is created for such a cellview in the cell name map file.

In addition, if you have specified a cell map file, which has the structure name whose size is greater than 32 characters, then it is modified to create a unique name within 32 characters.

Default value: Deselected

Command-line option: respectGDSIINameLimit

GDS Cell Name Length

Use *GDS Cell Name Length* to set cell name length. Minimum value that you can set using this option is 10.

Command-line option: `gdsCellNameLength`

Cell Name Prefix

This option enables you to add prefix in all the cell names in OpenAccess. For example, if a prefix, *_pi* is added the cellname, *cell11* then the cellname in OpenAccess would be *_picell11*. All options, such as *Case Sensitivity* or *Respect 32 Characters Name Limit* will be applied on the modified name.

Command-line option: `cellNamePrefix`

Cell Name Suffix

This option enables you to add suffix in all the cell names in OpenAccess. For example, if a suffix, *_si* is added the cellname, *cell11* then the cellname in OpenAccess would be *cell11_si*. All options, such as *Case Sensitivity* or *Respect 32 Characters Name Limit* will be applied on the modified name.

If cell name mapping is provided either through a cellMap file or through a SKILL function, *poCellNameMap()*, then the prefix and suffix will be added to names that are already mapped.

Command-line option: `cellNameSuffix`

Sub-Master Separator

Use the *Sub-Master Separator* to change the default value of the separator added to Pcells during translation. The default separator is *_CDNS_*. The following naming convention is used:

`pCell/via master name + subMasterSeparator + timestamp + number`

Here,

- **subMasterSeparator** is configured using the `subMasterSeparator` option
- **timestamp** is an 11 digit number, where the first nine digits are the number of seconds elapsed since January 1, 1970 and the last 2 digits are micro seconds

- **number** is an incrementing number.

This naming conversion does not apply when `poParamCellNameMap` is used.

Command-line option: `subMasterSeparator`

Cell Case Sensitivity

The *Cell Case Sensitivity* option specifies whether XStream Out changes the case of letters in cell names. XStream Out uses this option when you do not supply cell name mapping information for the cell. You can map cell names in a cell name map table or in a SKILL file.

lower

The *lower* option translates all cell names to lowercase. For example, the name `Obj1` becomes `obj1`.

upper

The *upper* option translates all cell names to uppercase. For example, the name `Obj1` becomes `OBJ1`.

preserve

The *preserve* option translates the cell names without changing case. For example, the name `Obj1` remains `Obj1`.

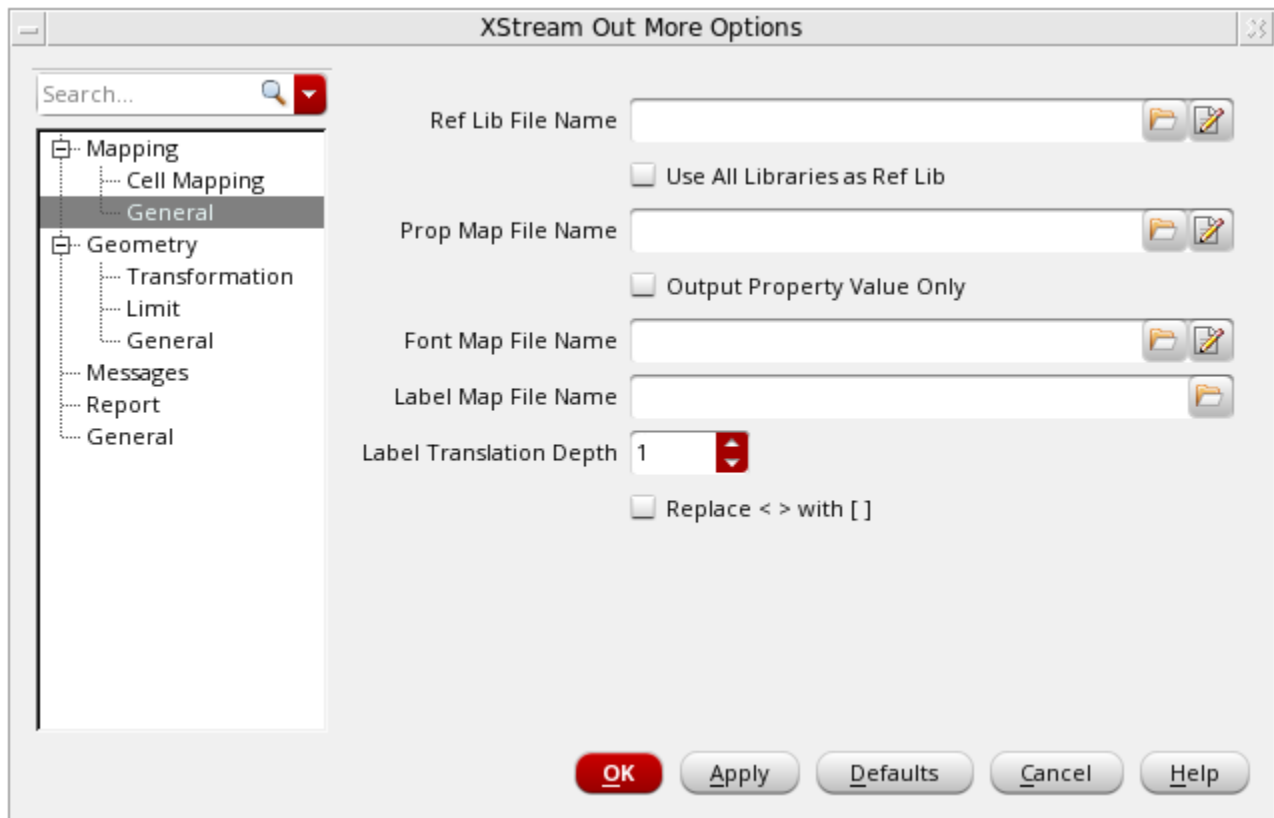
Command-line option: `case`

Design Data Translators Reference

Design Translation Using XStream Translator

Mapping: General Options

This section describes options on the Mapping: General options pane.



Ref Lib File Name

Use the *Ref Lib File Name* field to select a library name from the list of available libraries that are to be used as a reference libraries. You can also click the *File Browser* button to select the file containing list of reference libraries. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `refLibList`

To select a library name from the list of available libraries to be used as a reference libraries, leave the *Ref Lib File Name* field blank, and click the *Editor* button to display the XStream Out Reference Library File Editor Form.

Use All Libraries as Ref Lib

Use the *Use All Libraries* option to select all available libraries as reference libraries. The `refLibList` option exports only instances to cellviews from these reference libraries. It does not export any cellviews from them.

Note: To ensure that the missing or referenced cellviews are found in these libraries, you need to provide same references libraries during xStream In.

If you invoke Stream Out from the command line, you can write the list of reference libraries in a text file and provide this file as the value of the `refLibList` option. For example, consider a reference library file `refLib.list` with the following content:

```
Lib1
Lib2
Lib3
```

In this case, the `-refLibList refLib.list` command will consider all reference libraries listed in the `refLib.list` file.

If you want to consider all the libraries defined in the `cds.lib` file as reference libraries, use the `XST_CDS_LIB` keyword (`-refLibList XST_CDS_LIB`). You can provide this keyword as a value of the `refLibList` option from the command line or by writing it in a text file containing the list of libraries.

Prop Map File Name

Use the *Prop Map File Name* field to specify an existing properties mapping file. You can also click the *File Browser* button to select the properties mapping file. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `propMap`

To create new properties mapping records or load a new properties mapping file, leave the *Prop Map File Name* field blank, and click the *Editor* button to display the XStream Out Property Map File Editor Form.

Output Property Value Only

Use the *Output Property Value Only* option to export the specified *propertyName*'s value of an *objectType* as the value of the corresponding Stream attribute specified in the mapping record to the GDS file.

Command-line option: `propValueOnly`

Default value: Deselected

Font Map File Name

Use the *Font Map File Name* field to specify an existing font mapping file. You can also click the *File Browser* button to select the font mapping file. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `fontMap`

To create new font mapping records or load a new properties mapping file, leave the *Font Map File Name* field blank, and click the *Editor* button to display the XStream Out Font Map File Editor Form.

Label Map File Name

Use the *Label Map File Name* field to specify a label map file that determines the object type for which labels are created and determines the mapping of shapes on an *oaLayer* and *oaPurpose* pair to a GDS layer and GDS data type pair. The object type specified in the label map file is either a terminal or a net.

Command-line option: `labelMap`

Label Translation Depth

Use the *Label Translation Depth* field to specify the depth of the design hierarchy for which labels are generated. If you leave this field as is (default value 1), labels are generated for only the top level of the design.

Value values: 1 to 32767

Default value: 1

Command-line option: `labelDepth`

Design Data Translators Reference

Design Translation Using XStream Translator

Note: The `-labelDepth` option works only when the `-labelMap` option is specified.

Important

The term *Labels* in the *Label Translation Depth* option name refers to the `net/term/pin` name. This is different from the label created using the *Layout XL > Create > Label* command.

Replace <> with []

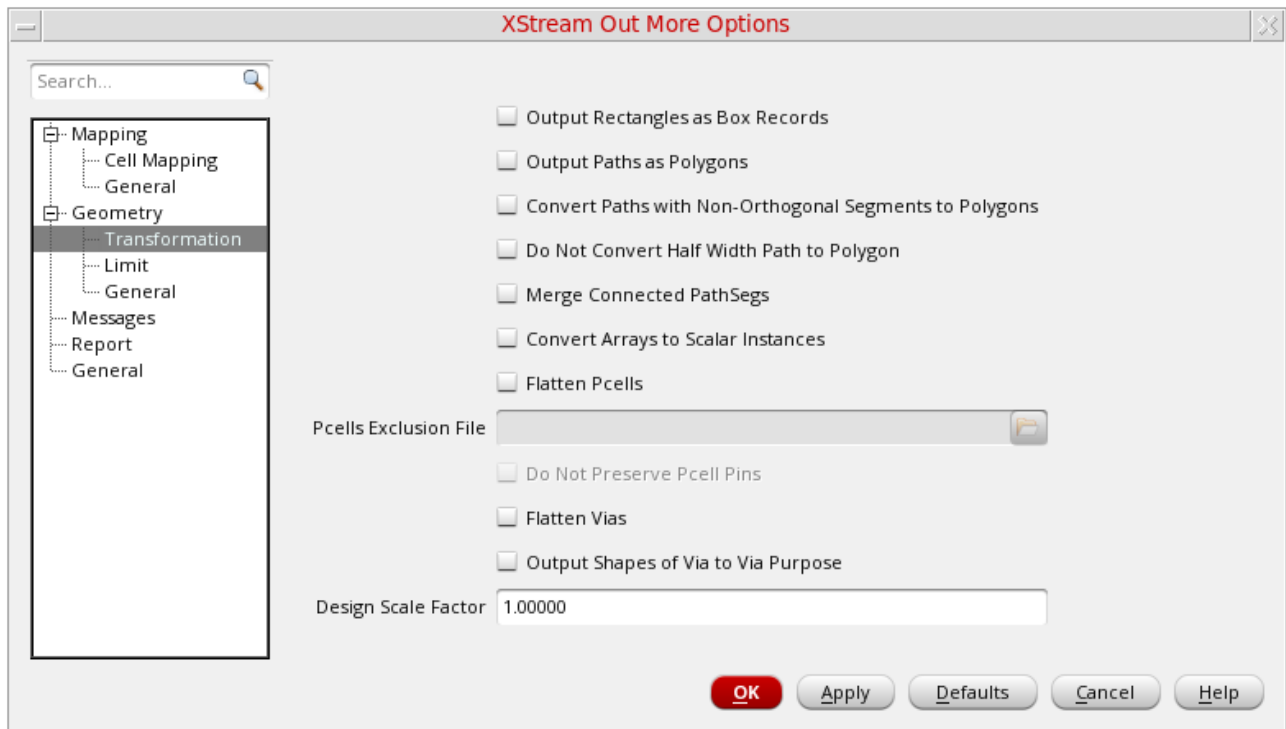
The *Replace <> with []* option replaces the character `<` with `[` and `>` with `]` in text records during the translation from the DFII library to a Stream file.

Default: `off`

Command-line option: `replaceBusBitChar`

Geometry: Transformation Options

This section describes options on the Geometry: Transformation options pane.



Output Rectangles as Box Records

Select the *Output rectangles as box records* checkbox to translate a rectangle into a BOX record. A BOX record is an alternative way of defining a rectangular shape. It consists of a coordinate representing the center of the rectangular shape and values specifying the height and width of the shape.

If you do not select this option, rectangles are translated as polygons. This option is required for backward compatibility with some old party GDSII readers and writers. Using this option, you can also preserve rectangles.

Default value: Deselected

Command-line option: `rectToBox`

Output Paths as Polygons

Select the *Output Paths as Polygons* check box to translate all paths in OpenAccess as `BOUNDARY` records in the Stream format.

Default value: Deselected

Command-line option: `pathToPolygon`

Note: However, there are the following instances where path is converted as polygon even if the *Output Paths as Polygons* option is not selected:

- Path is less than 1/2 width
- Path with acute angle
- Path with rounded extension



If the option *Do Not Convert Half Width Path to Polygon* is selected then the half width path is not converted to boundary.

Convert Paths with Non-Orthogonal Segments to Polygons

Select the *Convert Paths with Non-Orthogonal Segments to Polygons* check box to export paths with non-orthogonal segments as `BOUNDARY` records. When this check box is not selected, all paths with non-orthogonal segments are translated as `PATH` records.

Default Value: Deselected

Command-line option: `diagonalPathToPolygon`

Do Not Convert Half Width Path to Polygon

Select the *Do Not Convert Half Width Path to Polygon* check box to translate half width path as a path and not as a polygon. When this check box is not selected, all half width paths are translated as `BOUNDARY` records.

Default Value: Deselected

Command-line option: `noConvertHalfWidthPath`

Merge Connected PathSegs

Select the *Merge Connected PathSegs* check box to assemble all the connected `oaPathSegs` of the same LPP into a single PATH record during XStream Out. This reduces the size of the GDSII file.

Command-line option: `mergePathSegsToPath`

Convert Arrays to Scalar Instances

Select the *Convert Arrays to Scalar instances* check box to translate an array as a Stream array (AREF) or break an array into individual instances (SREF). When you select this option, an array in the OpenAccess database is translated as multiple SREFs in the Stream file.

Note: The *Convert Arrays to Scalar instances* option works when the *Coloring Mode* option is enabled.

For performance issues related to this option, see [Guidelines for XStream Out](#).

Default value: Deselected

Command-line option: `arrayInstToScalar`

Flatten Pcells

Select the *Flatten Pcells* check box if you want to flatten the Pcell instances in the designs.

Default value: Deselected

Command-line option: `flattenPcells`

Pcells Exclusion File

Use the *Pcells Exclusion File* option to specify the name of the exclusion file that contains a list of Pcells to be excluded during Pcell flattening. The *Pcells Exclusion File* option is available only if you select the *Flatten Pcells* option.

Example of a Pcell Exclusion file:

```
;Specify either library, library and cell, or library, cell, and view
(libraryB)
(libraryA "pcellA")
(libraryA "pcellB")
```

Design Data Translators Reference

Design Translation Using XStream Translator

```
("LibraryA" "pcellC" "layout1") ("LibraryC")
```

Default value: Not specified

Command-line option: `flattenPcellsExclusionFile`

Do Not Preserve Pcells Pins

Select this check box if you do not want to retain the connectivity information of pins in flattened Pcell instances. The *Do Not Preserve Pcell* pins option is available only if you select the *Flatten Pcells* option.

Command-line option: `doNotPreservePcellPins`

Flatten Vias

Select the *Flatten Vias* check box if you want to flatten all standard, custom, and cdsGen vias in the design.

Default value: Deselected

Command-line option: `flattenVias`

Output Shapes of Vias to Via Purpose

Select the *Output Shapes of Vias to Via Purpose* check box to export the shapes of vias according to the purpose that was defined for the vias.

If this option is used along with the *Flatten Vias* option then via shapes are flattened to the via purpose.

Command-line option: `outputViaShapesToViaPurp`

Design Scale Factor

Use the *Design Scale Factor* option to scale an exported design by the factor set here. This option accepts any positive floating value. The `scale` option will not have any impact on the existing options or behavior because scaling is done internally. It will work in addition to the settings in the `DBUPerUU` option.

Design Data Translators Reference

Design Translation Using XStream Translator

For example, consider `DBUPerUU` is 1000 and GDS polygon size is 3000 db units (user units). If you run the command `strmin -scale 4 -dbuPerUU 2000`, the resultant polygon size will be calculated as below:

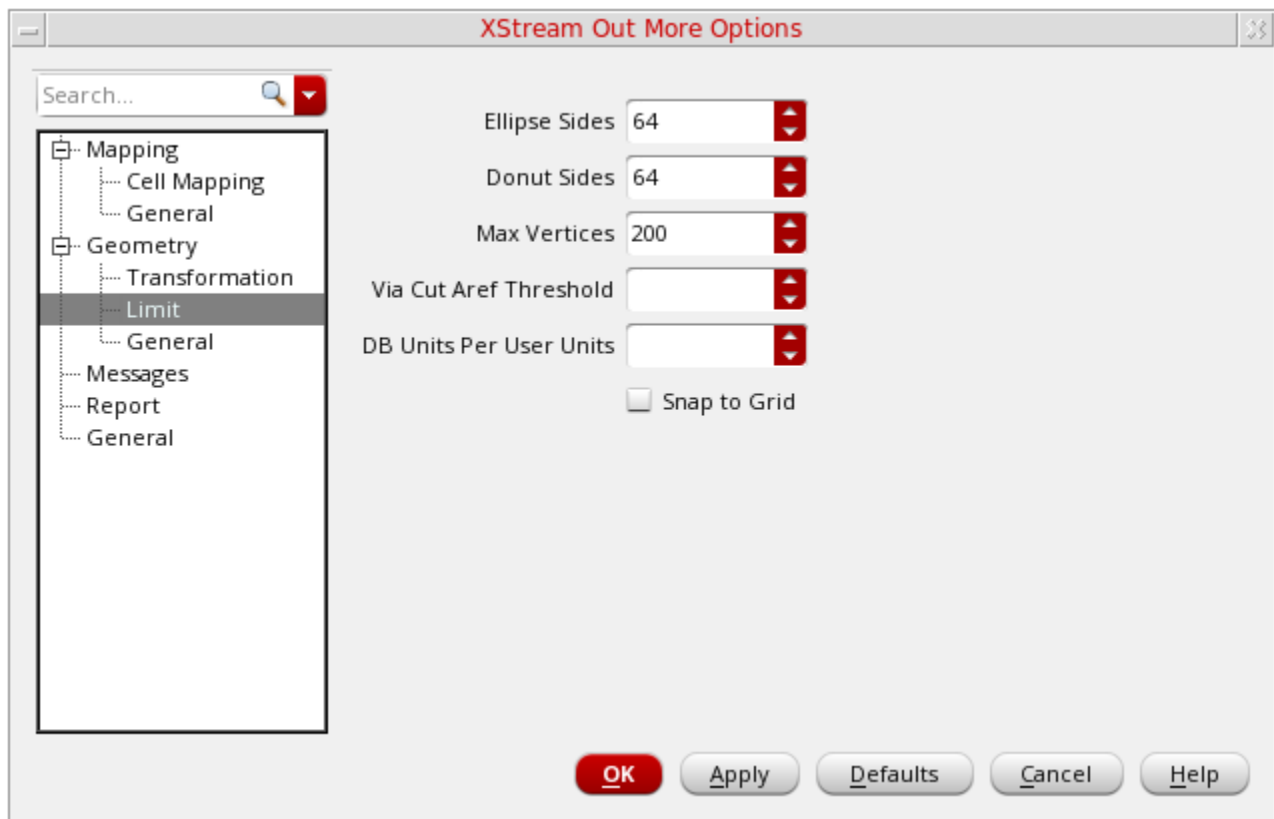
$$3000 \times (2000/1000) \times 4 = 24000 \text{ db units (user units)}$$

Default value: 1.00

Command-line option: `scale`

Geometry: Limit Options

This section describes options on the Geometry: Limit options pane.



Ellipse Sides

Specifies the number of sides for the boundary of ellipse.

Valid values: Any value between 4 and 131072 (inclusive of both values)

Default value: 64

Command-line option: `ellipseNumSides`

Donut Sides

Specifies the number of sides for the boundary of donut objects.

Valid values: Any value between 4 and 131072 (inclusive of both values)

Default value: 64

Command-line option: `donutNumSides`

Max Vertices

Use the *Max Vertices* field to specify the maximum number of coordinates for polygons and paths in Stream. *Export – Stream* divides polygons and paths in DFII that have vertices more than or equal to `maxVertices` into smaller objects.

Polygons and paths are stored differently in Design Framework II and Stream formats. The DFII database does not store both the start and end points of a polygon because they are the same point. The Stream format requires that both points be stored individually. The maximum limit imposed by the GDSII format is 200 and that by the Stream format is 8191. Therefore, the Stream/GDSII reader/writer of each EDA vendor is responsible for handling restrictions imposed by their databases with this option.

The largest DFII polygon or path *Export – Stream* can process a maximum of 8191 vertices, which translates into 8191 Stream vertices.

Valid Values: 5 to 8191

Default value: 200

Command-line option: `maxVertices`

Via Cut Aref Threshold

While translating vias, XStream Out writes BOUNDARY record for intermediate cut layer repeatedly based on the 'Row' and 'Column' values with which the device is instantiated. For example, if a via instance has M rows and N columns for the cut layer, the corresponding GDS will have MxN BOUNDARY records representing this cut layer. For large values of MxN, the size of the GDS file can grow quite large due to this reason.

Design Data Translators Reference

Design Translation Using XStream Translator

The *viaCutArefThreshold* option provide the flexibility of representing the cut layer geometry through AREF record in the GDS file. By default, XStream Out does not array the cut layer geometry. When used with non default values (+ve integers), if a via instance has non default cut layer pattern and Row * column $\geq$ viaCutArefThreshold, Streamout writes the regularly sized and spaced cut layers through AREF record in the GDS file by introducing an additional level of hierarchy and in turn reduces the GDS file size.

Use the *Via Cut Aref Threshold* field to specify the number of cut layers for the array.

Command-line option: `viaCutArefThreshold`

DB Units Per User Units

Use the *DB Units Per User Units* field to scale the design during translation, which enables XStream Out to write the `UNITS` section and the geometry in the output GDSII/Stream as per the new value. By default, the value of `DBUPerUU` in the technology information is used.

Valid values: Greater than or equal to 1

Command-line option: `dbuPerUU`

Snap to Grid

The *Snap to Grid* option adjusts the coordinates of geometric objects to coincide with the grid resolution. However, this option can shift data from the original Stream grid, resulting in a loss of precision.

Default value: Deselected

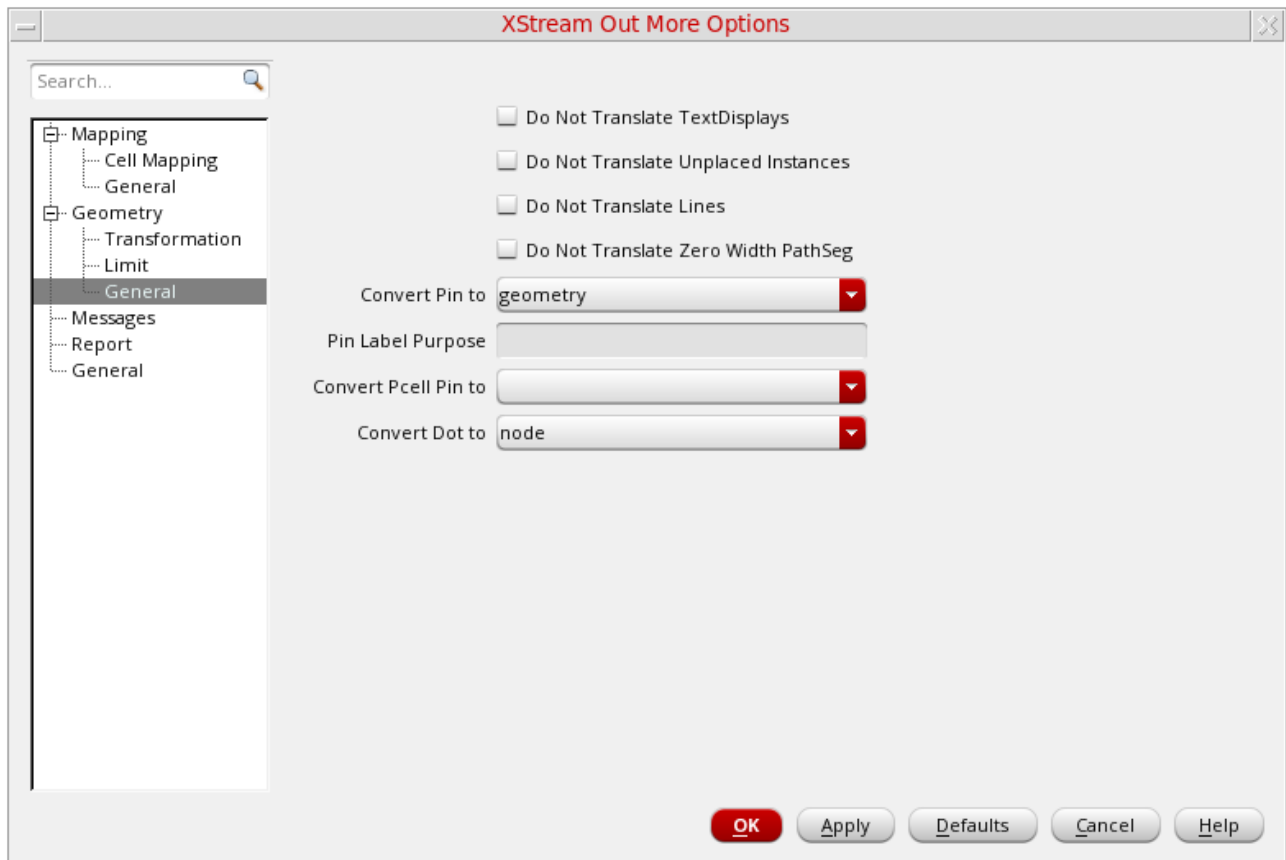
Command-line option: `snapToGrid`

Design Data Translators Reference

Design Translation Using XStream Translator

Geometry: General Options

This section describes options on the Geometry: General options pane.



Do Not Translate TextDisplays

Select the *Do Not translate text displays* option if you do not want to translate text displays into Stream files. By default, XStream Out translates the text displays into Stream files.

Default value: Deselected

Command-line option: `noOutputTextDisplays`

Do Not Translate Unplaced Instances

Unplaced instances are not visible in the Layout Editor. Therefore, if needed, select the *Do Not Translate Unplaced Instances* option to prevent these instances from getting translated during XStream Out.

Default value: Deselected

Command-line option: `noOutputUnplacedInst`

Do Not Translate Lines

Select the *Do Not Translate Lines* option if you want to ignore the lines during translation. By default, lines are translated as zero-width paths.

Default value: Deselected

Command-line option: `ignoreLines`

Ignore Zero Width Paths

Use the *Ignore Zero Width Paths* option to ignore the translation of zero-width path. If you select this option, zero-width paths are ignored, otherwise they are translated as lines.

Default value: Deselected

Command-line option: `ignoreZeroWidthPathSeg`

Convert Pin to

Use the *Convert Pin to* option to control the translation of the pins present in an OpenAccess design to Stream format. During XStream Out, pins can either be translated as geometries or text, or be ignored. Use the options described in the table below to specify whether pins are translated as labels or instances, or are ignored by the translator.

Command-line option: `convertPin`

Note: The following table also summarizes the output when a label map file is specified simultaneously with a Convert Pin to option and when it is not.

Design Data Translators Reference

Design Translation Using XStream Translator

Table 2-1 Convert Pin to Options

Option	Label Map File	Output
geometry	No	■ Pins are translated as the corresponding shape or instance.
	Yes	■ Geometry as well as text is translated.
text	No	■ Pins are translated as labels. ■ The label string can consist of a terminal, net, or pin name. ■ The net name is associated with the shape. ■ If there is no net name, terminal name is considered. In case terminal name is not specified, the pin name is considered.
	Yes	■ The corresponding entry in the label map file takes precedence over the Convert Pin to option. ■ Only text labels are translated. ■ A warning message informing that the mapping will be based on the label map file displays, and the translation continues to run.
geometryAndText	No	■ Pins are translated into shapes or instances and text labels.
	Yes	■ The label map file takes precedence over the Convert Pin to option ■ Geometry as well as text is translated ■ A warning message informing that the mapping will be based on the label map file displays, and the translation continues to run.

Design Data Translators Reference

Design Translation Using XStream Translator

Option	Label Map File	Output
ignore	No	■ Pins are not translated.
	Yes	■ If <i>Convert Pcell Pin to</i> option is not specified or if the <code>ignore</code> value is specified, then the label map file is ignored. ■ No record is translated. ■ A warning message informing that the label map option will be ignored displays, and XStream Out continues to run.

Pin Label Purpose

Use the *Pin Label Purpose* option to export pin labels corresponding to the specified purpose. Pin labels are exported according to the layer mapping for the layer on which the pin is created and the specified purpose.

This option works only when Convert Pin to is set to `geometryAndText` or `text`.

Example

Consider the following:

- There are two pins in a cellview with one pin each on `M1:drawing` and `M2:drawing`.
- A layer-map file contains the following entries:


```
M1 drawing      1 1
M2 drawing      2 1
M1 pin          2 10
```
- *Convert Pin to* option is specified as `geometryAndText`.

In this case, during XStream Out translation:

- If *Pin Label Purpose* is not selected:
 - ☐ Pin shapes are exported on `1:1` and `2:1`
 - ☐ Pin labels are exported as `1:1` and `2:1`
- If *Pin Label Purpose* is specified as `pin`:

Design Data Translators Reference

Design Translation Using XStream Translator

- ❑ Pin shapes are exported on 1:1 and 2:1
- ❑ A pin label for pin on M1:drawing is exported as per the mapping for M1:pin which is. 2:10
- ❑ Pin labels for pin on M2:drawing are exported as per the mapping for M2:drawing which is 2:1. This is because there is no mapping for M2:pin.

Convert Pcell Pin to

Use the *Convert Pcell Pin to* option to control the translation of the pins present in the Pcell submaster to Stream format. This option overrides the value of `convertPin` for the Pcell submaster. However, if the `convertPcellPin` option is not specified, value of `convertPin` is considered.

During XStream Out, pins can either be translated as geometries or text, or be ignored. For information on specifying whether the pins in the Pcell submaster are translated as labels or instances, or are ignored by the translator, see the [Convert Pin to Options](#) table.

Command-line option: `convertPcellPin`

Convert Dot To

Use the *Convert dot to* option if you want to convert dots to nodes or polygons.

node

The *Node* option converts dots to nodes.

polygon

The *Polygon* option converts dots to polygons.

ignore

The *Ignore* option ignores the dots during translation.

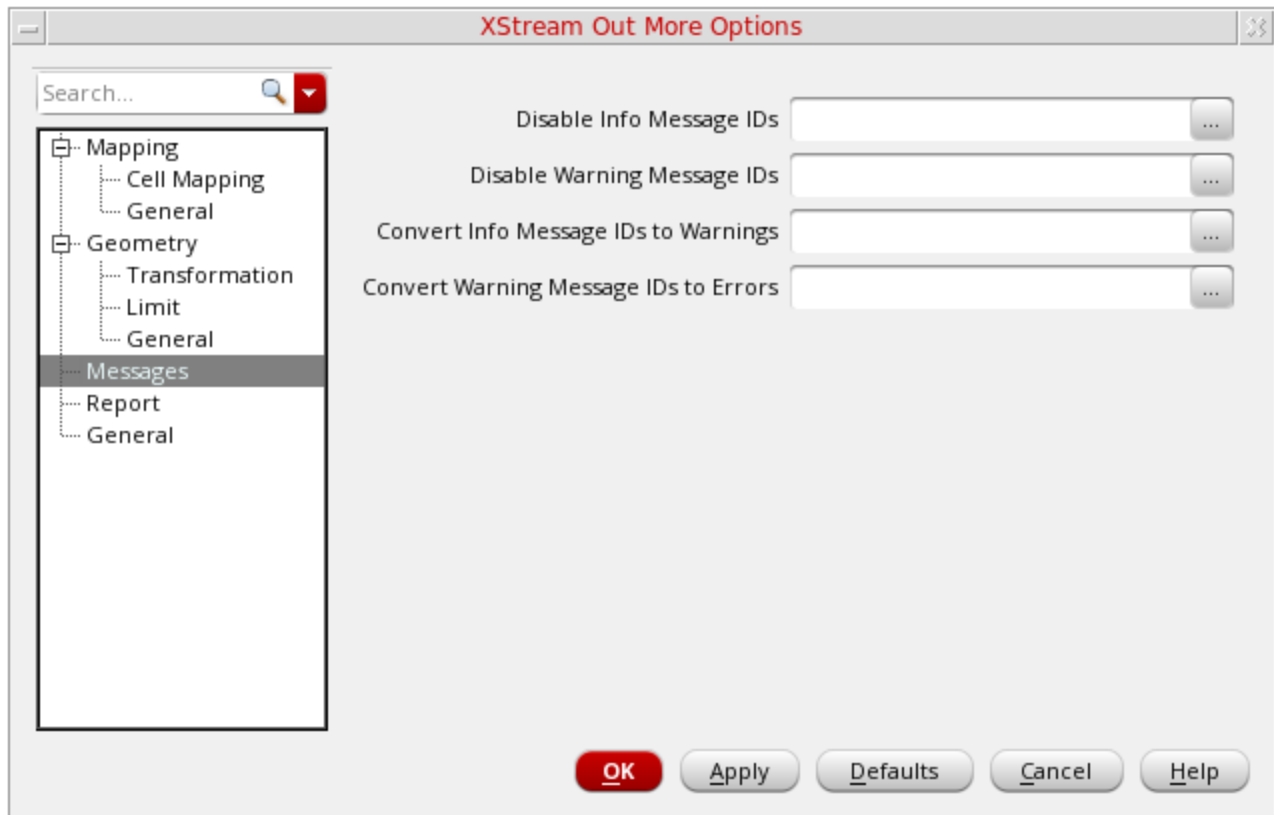
Command-line option: `convertDot`

Design Data Translators Reference

Design Translation Using XStream Translator

Messages Options

Use the Messages options pane to specify the information and warning message IDs that you want to disable or convert.



Disable Info Message IDs

Use the *Disable Info Message IDs* field to specify list of information message IDs that you do not want to print. You can also specify message IDs of dependent tools in this field.

The information message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 4374. The list of information message IDs should be separated by a white space.

Command-line option: `noInfo`

Alternatively, click the *Message Browser* button to display the XStream Out Info Messages form to select or specify the XStream Out information messages that you want to disable.

Disable Warning Message IDs

Use the *Disable Warning Message IDs field* to specify list of warning message IDs, separated by comma that you do not want to print. You can also specify message IDs of dependent tools in this field.

The warning message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 3075. The list of warning message IDs should be separated by a comma or white space.

Command-line option: `noWarn`

Alternatively, click the *Message Browser* button to display the XStream Out Warning Messages form to select or specify the warning messages that you want to disable.

Convert Info Message IDs to Warnings

Use the *Convert Info Message IDs to Warnings field* to specify list of information message IDs that you want to convert to warning messages. You can also specify message IDs of dependent tools in this field.

The information message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 4374. The list of information message IDs should be separated by a comma or white space.

Command-line option: `infoToWarn`

Alternatively, click the *Message Browser* button to display the XStream Out Info Messages form to select or specify the information messages that you want to convert to warning messages.

Convert Warning Message IDs to Errors

Use the *Convert Warning Message IDs to Errors field* to specify the IDs of warning messages that you want to convert to error messages. You can also specify message IDs of dependent tools in this field.

The warning message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 3075. The list of warning message IDs must be separated by a comma or white space.

Command-line option: `WarnToErr`

Design Data Translators Reference

Design Translation Using XStream Translator

Alternatively, click the *Message Browser* button to display the XStream Out Warning Messages form to select or specify the warning messages that you want to convert to error messages.

XStream Out Messages Forms

This sections describes the XStream In Message forms that are displayed when you click the respective Message Browser button.

XStream Out Info Messages Form to Disable Info Messages

This section describes options on the XStream Out Info Messages form that is displayed when you click the *Message Browser* button for the Disable Info Message IDs option.

Id	Message
☐ Other IDs	
☐ 60	Creating a new purpose %s in the technology file.
☐ 61	Creating a new layer %s in the technology file.
☐ 62	Creating a new layer:purpose pair %s:%s in the technology file.
☐ 69	All the libraries specified in `cds.lib` will be considered as reference libraries. This is because refLi...
☐ 70	The 'Reference Library list' and the 'Keep Stream Cells' options have been used together. Any cell ...
☐ 91	Process may require virtual memory near to what it is available on the system for a process. You ...
☐ 105	Property map file contains duplicate entry for Stream attribute '%d' object type '%s' at line numb...
☐ 162	You have not used the viaMap option. If the OpenAccess design has native oaVia instances use th...
☐ 163	You have not used the viaMap option. If a viaMap file corresponding techfile or technology library...
☐ 171	The 'noObjectProp' and the 'objectMap' options have been used together. Therefore no property...

☐ Select All

Select check boxes of the XStream information messages listed in this form that you want to disable.

Other IDs

Use the *Other IDs* field to specify information message IDs of other dependent tools.

Design Data Translators Reference

Design Translation Using XStream Translator

Select All

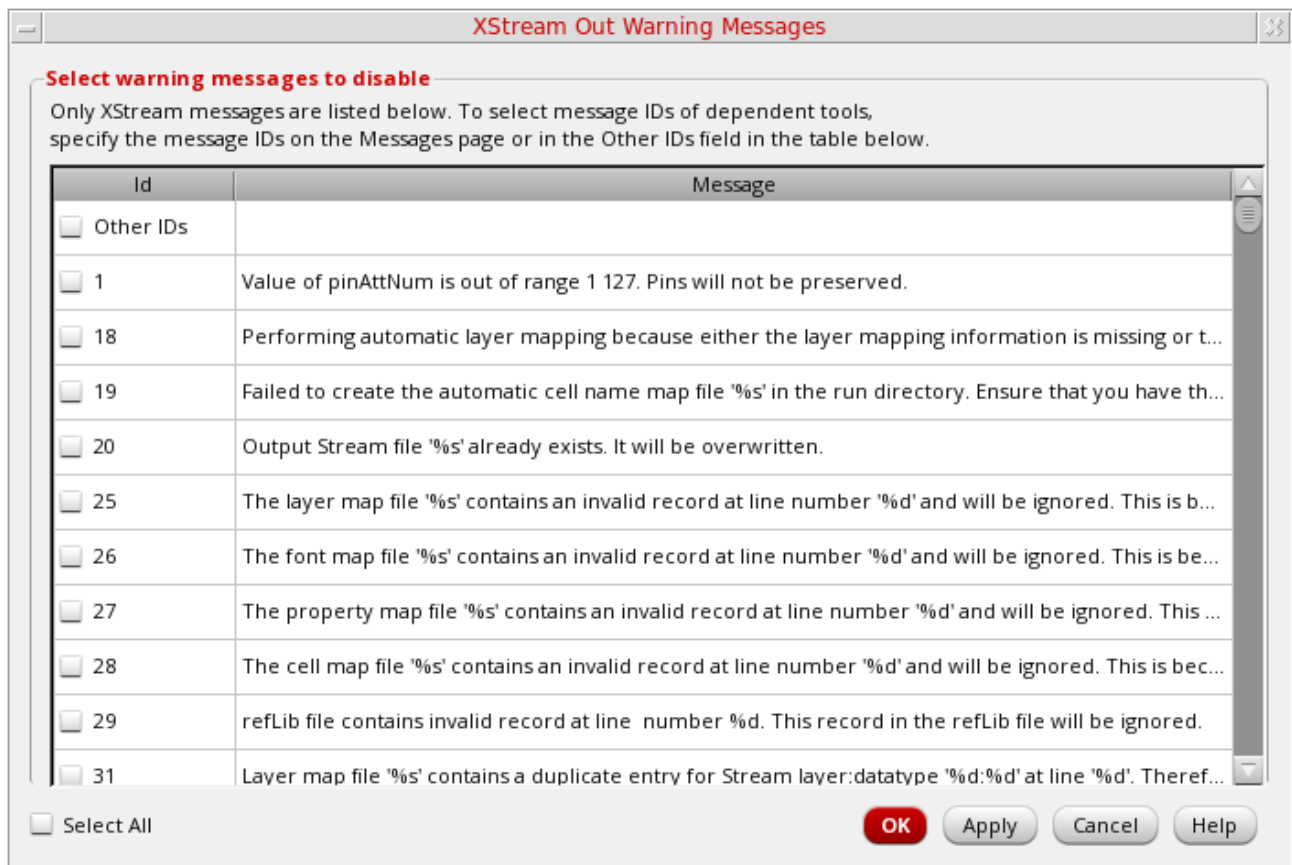
Click the *Select All* check box to select all message IDs.

Cancel

Use the *Cancel* button to exit the XStream Out Info Messages form without saving any changes.

XStream Out Warning Messages Form to Disable Warning Messages

This section describes options on the XStream Out Warning Messages form that is displayed when you click the *Message Browser* button for the Disable Warning Message IDs option.



Id	Message
☐ Other IDs	
☐ 1	Value of pinAttNum is out of range 1 127. Pins will not be preserved.
☐ 18	Performing automatic layer mapping because either the layer mapping information is missing or t...
☐ 19	Failed to create the automatic cell name map file '%s' in the run directory. Ensure that you have th...
☐ 20	Output Stream file '%s' already exists. It will be overwritten.
☐ 25	The layer map file '%s' contains an invalid record at line number '%d' and will be ignored. This is b...
☐ 26	The font map file '%s' contains an invalid record at line number '%d' and will be ignored. This is be...
☐ 27	The property map file '%s' contains an invalid record at line number '%d' and will be ignored. This ...
☐ 28	The cell map file '%s' contains an invalid record at line number '%d' and will be ignored. This is bec...
☐ 29	refLib file contains invalid record at line number %d. This record in the refLib file will be ignored.
☐ 31	Layer map file '%s' contains a duplicate entry for Stream layer:datatype '%d:%d' at line '%d'. Theref...

Select check boxes of the XStream warning messages listed in this form that you want to disable.

Design Data Translators Reference

Design Translation Using XStream Translator

Other IDs

Use the *Other IDs* field to specify warning message IDs of other dependent tools.

Select All

Click the *Select All* check box to select all message IDs.

Cancel

Use the *Cancel* button to exit the XStream Out Warning Messages form without saving any changes.

XStream Out Info Messages Form - Convert Information Messages to Warnings

This section describes options on the XStream Out Info Messages form that is displayed when you click the *Message Browser* button for the Convert Info Message IDs to Warnings option.

Select info messages to convert to warnings

Only XStream messages are listed below. To select message IDs of dependent tools, specify the message IDs on the Messages page or in the Other IDs field in the table below.

Id	Message
☐ Other IDs	
☐ 60	Creating a new purpose %s in the technology file.
☐ 61	Creating a new layer %s in the technology file.
☐ 62	Creating a new layer:purpose pair %s:%s in the technology file.
☐ 69	All the libraries specified in `cds.lib` will be considered as reference libraries. This is because refLi...
☐ 70	The 'Reference Library list' and the 'Keep Stream Cells' options have been used together. Any cell ...
☐ 91	Process may require virtual memory near to what it is available on the system for a process. You ...
☐ 105	Property map file contains duplicate entry for Stream attribute '%d' object type '%s' at line numb...
☐ 162	You have not used the viaMap option. If the OpenAccess design has native oaVia instances use th...
☐ 163	You have not used the viaMap option. If a viaMap file corresponding techfile or technology library...
☐ 171	The 'noObjectProp' and the 'objectMap' options have been used together. Therefore no property...

OK Apply Cancel Help

Design Data Translators Reference

Design Translation Using XStream Translator

Select check boxes of the XStream information messages listed in this form to convert to warnings.

Other IDs

Use the *Other IDs* field to specify information message IDs of other dependent tools.

Cancel

Use the *Cancel* button to exit the XStream Out Info Messages form without saving any changes.

XStream Out Warning Messages Form to Convert Messages to Errors

This section describes options on the XStream Out Warning Messages form that is displayed when you click the *Message Browser* button for the Convert Warning Message IDs to Errors option.

Id	Message
☐ Other IDs	
☐ 60	Creating a new purpose %s in the technology file.
☐ 61	Creating a new layer %s in the technology file.
☐ 62	Creating a new layer:purpose pair %s:%s in the technology file.
☐ 69	All the libraries specified in `cds.lib` will be considered as reference libraries. This is because refLi...
☐ 70	The 'Reference Library list' and the 'Keep Stream Cells' options have been used together. Any cell ...
☐ 91	Process may require virtual memory near to what it is available on the system for a process. You ...
☐ 105	Property map file contains duplicate entry for Stream attribute '%d' object type '%s' at line numb...
☐ 162	You have not used the viaMap option. If the OpenAccess design has native oaVia instances use th...
☐ 163	You have not used the viaMap option. If a viaMap file corresponding techfile or technology library...
☐ 171	The 'noObjectProp' and the 'objectMap' options have been used together. Therefore no property...

Design Data Translators Reference

Design Translation Using XStream Translator

Select check boxes of the XStream warning messages listed in this form that you want to convert to errors.

Other IDs

Use the *Other IDs* field to specify information message IDs of other dependent tools.

Cancel

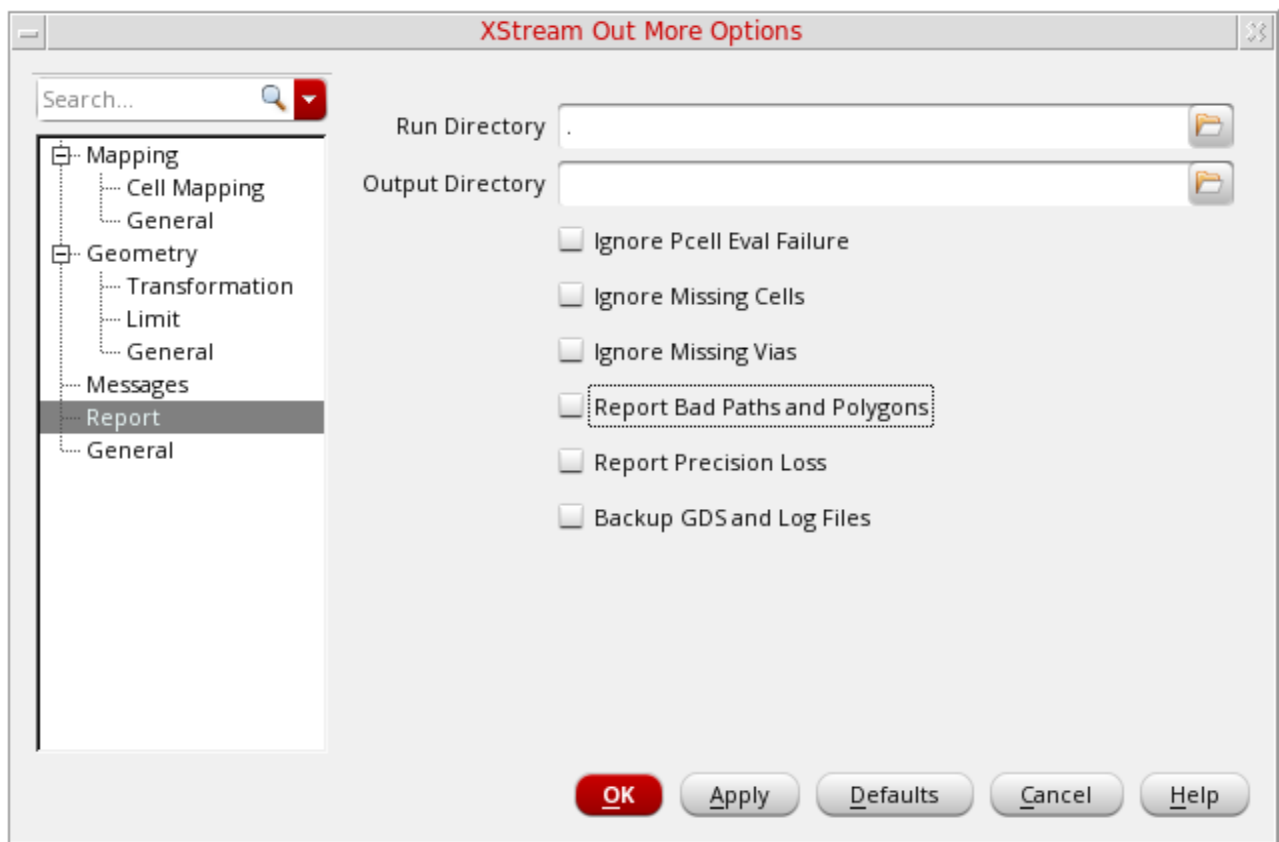
Use the *Cancel* button to exit the XStream Out Info Messages form without saving any changes.

Design Data Translators Reference

Design Translation Using XStream Translator

Report Options

This section describes options on the Report options pane.



Run Directory

The *Run Directory* field accepts the default directory for all the files you specify in the *XStream Out* form. If you want to read from or write to a file that is not present in the run directory, type the full path in the field for the file.

Use the directory browser to select the path of the run directory.

Default value: Current working directory

Command-line option: `runDir`

Output Directory

The *Output Directory* field enables you to specify the path of the output directory. The purpose of this field to output only the .gds file.

Command-line option: `outputDir`

Ignore Pcell Evaluation Failure

Select the *Ignore Pcell Evaluation Failure* option if you do not want XStream to halt upon any Pcell evaluation failure. If you do not select this option, XStream will halt upon any Pcell evaluation failure and an error message will be displayed.

Default value: `nil`

Command-line option: `ignorePcellEvalFail`

Ignore Missing Cells

Select the *Ignore Missing Cells* option to ignore missing cells during translation and continue translation with a warning message displayed in the CIW.

Default: Deselected

Command-line option: `ignoreMissingCells`

Ignore Missing Vias

Select the *Ignore Missing Vias* option to ignore missing via definitions during translation and continue translation with a warning message displayed in the CIW.

Default: Deselected

Command-line option: `ignoreMissingVias`

Report Bad Polygons and Paths

The *Report Bad polygons and paths* option provides a way for reporting self intersecting paths, coincident or collinear paths, paths that have less than two coordinates, path with an invalid begin and end extension value, an open polygon, coincident or collinear polygons, and

a polygon with self-intersecting points. However, using this option can cause a degradation in performance.

Note: A begin or end path with negative extension is truncated to zero if the length of the new path segment is greater than or equal to the half-path width. A warning is added to the log file.

Default value: Deselected

Command-line option: `checkPolygon`

In case of XStream Out, if an OpenAccess design has path with octagonal extensions then these paths are mapped to PATHTYPE1, which is circular path end according to the Stream/GDSII format. You can retain the geometry of these paths by using the Output Paths as Polygon option.

In case of XStream In, if a Stream/GDSII files has some PATH records with PATHTYPE1, which is circular path end according to the Stream/GDSII format, then these PATH records will be translated as paths with octagonal extensions. There could be a potential problem which should be corrected at the source of the Stream/GDSII file.

Report Precision Loss

The *Report Precision Loss* option gives you an error message if there is any precision loss while scaling a design.

Default value: Deselected

Command-line option: `reportPrecisionLoss`

Backup GDS and Log Files

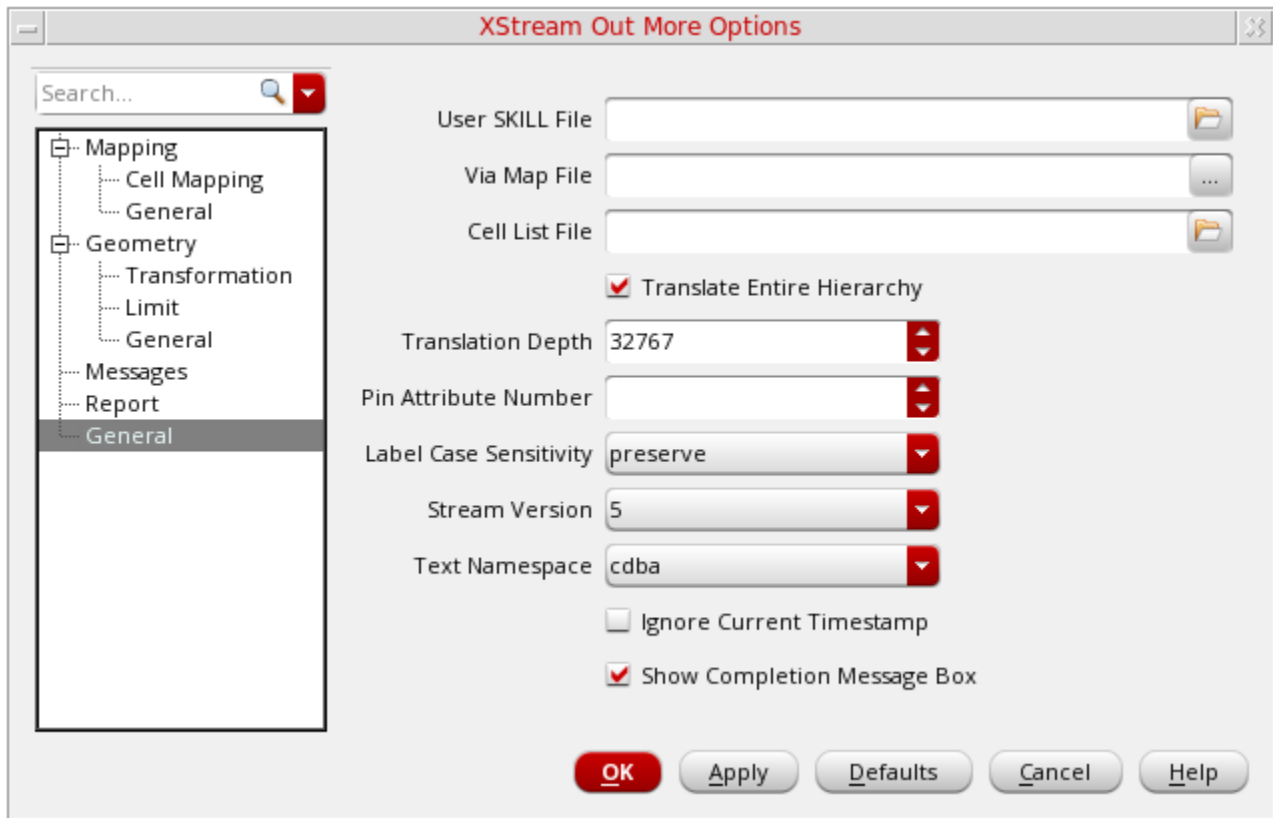
Select the *Backup GDS and Log Files* option if you want to create a backup of the GDS and log files.

Command-line option: `backGdsLogFiles`

Design Data Translators Reference

Design Translation Using XStream Translator

General Options



This section describes options on the General options pane.

User Skill File

The *User Skill File* field lets you specify a file that consists of user-defined SKILL routines. Using this file, you can specify mapping mechanism for cell names, layers, font through the use of programming routines written in SKILL.

In order to load all the required SKILL procedures in a single file, use only one SKILL file during translation. The SKILL file is searched with respect to the run directory if complete path of the SKILL file is not specified. For more information, see [Chip Finishing Flow](#) video.

Note: XStream Out supports only those SKILL functions that are essential to evaluate a Pcell. These are SKILL Core, db, dd, tech, cst, abe, vfo (vfo.cxt), and pas (pdkutils.cxt) SKILL functions. Application specific functions such as le, hi, sch, lx, and via are not supported.

Design Data Translators Reference

Design Translation Using XStream Translator

Command-line option: `userSkillFile`

Via Map File

Use the *Via Map File* field to specify a via map file. The via map file maps the Structure names from the Stream file to the corresponding viaDef names in the input technology library. This via map file can be used to recreate the original vias during subsequent Stream Ins. For more information on the via map file, see [Via Map File](#).

Command-line option: `viaMapFile`

Cell List File

The *Cell List File* field enables you to specify a file containing a list of cell names. The cell names in a cell list file is displayed in the following format:

```
<libName> <cellName> <viewName> <output GDSII file name>
```

For example, the Cell List File can be specified as:

```
<Lib1> <cell1> <view1> <outGDS1>
```

```
<*> <cell2> <*> <*>
```

Command-line option: `cellListFile`

Note: Asterisk “*” can be specified for all the fields, except `<cellName>`. However, if you only specify `<cellName>`, then also XStream Out will consider this entry as a valid entry.

Translate Entire Hierarchy

Use the *Translate Entire Hierarchy* option to translate the entire design hierarchy.

Default value: Selected

Translation Depth

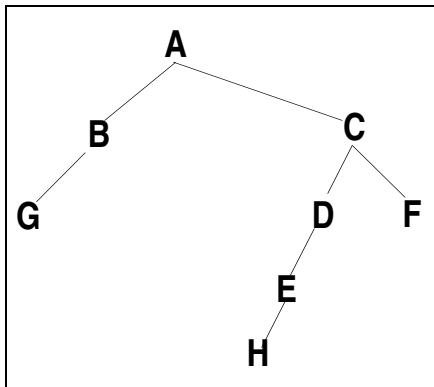
Use the *Translation Depth* field to control the depth of the design hierarchy during XStream Out. XStream Out stops the translation process at the hierarchy level indicated by this option. The hierarchy depth value applies to all the top cells getting translated. This option is enabled only if the *Translate Entire Hier* option is deselected.

Value values: 0 to 32767

Default value: 32767

Command-line option: `hierDepth`

Example 2-17 Translating cells by using the Translation Depth option



The above hierarchy shows structures A, B, C, D, E, F, G, and H placed at different levels. When structure C is selected as the top cell name and the hierarchy depth is selected as 1, the structures C, D, and F are translated.

Instance of E will first be translated as SREF inside structure D and then be translated to the Stream file, but its structure information is ignored. Structure H is ignored and is not translated.

By default, the entire hierarchy is translated.

Pin Attribute Number

Use the *Pin Attribute Number* field to preserve pin connectivity information. For more information, see [Preserving Pin Connectivity Information](#).

Valid values: 1 to 127

Command-line option: `pinAttNum`

Label Case Sensitivity

Use the *Label Case Sensitivity* option to specify if XStream Out changes the casing of letters in the text labels. The following options are available:

Design Data Translators Reference

Design Translation Using XStream Translator

- *lower* - The *lower* option translates the text labels to lowercase. For example, the label Obj1 becomes obj1.
- *upper* - The *upper* option translates the text label to uppercase. For example, the label Obj1 becomes OBJ1.
- *preserve* - The *preserve* option translates text labels without changing case. For example, the label Obj1 remains Obj1.

Default value: `preserve`

Command-line option: `labelCase`

Stream Version

Use the *Stream Version* field to specify the Stream version to be used for XStream Out. The three options are *Three*, *Five*, and *Six*. By default, rectangles are written as boundary records. If you need XStream Out to write the BOX records, use the [Output Rectangles as Box Records](#). If you use Stream version 3, all rectangles are written as boundary records because Stream version 3 does not support BOX records.

In addition, Stream version 3 does not support variable extensions for paths. For example, consider the following Stream output when the *Stream Version* is set to *Five*.

```
PATHLAYER-0 DATATYPE-0 PATHTYPE 4 WIDTH 1000 BGNEXTN=2000  
ENDEXTN=3000 XY=(0,0) (5000,0) ENDEL
```

For the same path, XStream gives the following Stream output if the *Stream Version* is set to *Three*.

```
PATHLAYER-0 DATATYPE-0 PATHTYPE 0 WIDTH 1000 XY=(-2000,0) (8000,0)  
ENDEL
```

Default value: 5

Command-line option: `strmVersion`

For more information on different Stream versions, see [Differences in Stream Format Versions](#).

Design Data Translators Reference

Design Translation Using XStream Translator

Text Namespace

Use the *Text Namespace* field to specify the namespace mapping for pin labels or *textDisplays* associated with the pins. This option enables you to specify the name space in which the TEXT records of the Stream file will be.

For consistency, any of the OpenAccess namespaces (except OS namespaces like `oaUnixNS` and `oaWinNS`) can be specified. The valid arguments for this option are `native`, `cdba`, `lef`, `def`, `verilog`, `vhdl`, `spef`, `spice`, and `spf`. The default option is `cdba`.

Command-line option: `strmTextNS`

For example, if a pin label in the input design is `bus<7>` and you specify verilog namespace, it will be translated in the Stream file as `bus[7]`.

Table 2-2 Examples of Name Mapping

Type	Native	VHDL	Verilog	CDBA	LEF/DEF	SPEF	SPF	SPICE
lowercase or case sensitive	bigchip	bigchip BIGCHIP	bigchip	bigchip	bigchip	bigchip	bigchip	bigchip BIGCHIP
case sensitive mixed case	BigChip	\BigChip\	BigChip	BigChi p	BigChip	BigChip	BigChi p	%Big%C hip
keyword	and	\and\	\and_	and	and	and	and	and
unneeded escape	ESC_tra sh	\trash\	ESC_tr ash	ESC_t rash	ESC_tr ash	ESC_tra sh	ESC_tr ash	ESC_tras h
embedded space	foo bar	\foo bar\	\foo#20 bar_	foo#20 bar	foo%20 bar	foo#20b ar	foo#20 bar	foo#20ba r
device names	aux	aux	aux	aux	aux	aux	aux	aux
back-slash	\a\b	\\a\b\	\\a\b_	#5ca# 5cb	\\a\b	\\a\b	#5ca# 5cb	#5ca#5cb
forward slash	#2fa#2f b	/a/b\	/a/b_	#2fa#2 fb	/a/b	/a/b	/a/b	a/b
special characters*	a<1#3a 2>	\a<1:2>\	\a<1:2> _	a#3c1: 2#3e	a<1:2>	a<.:2>	a<1:2 >	a<1:2>
bit index	[]	()	[]	<>	[]	[]	scalar only by default	[]

Design Data Translators Reference

Design Translation Using XStream Translator

Table 2-2 Examples of Name Mapping

Type	Native	VHDL	Verilog	CDBA	LEF/ DEF	SPEF	SPF	SPICE
hierarchy	/	:	.	'	/	/	/	.

Let's consider another example, if a pin label in VHDL is `bus(7)` and you specify CDBA namespace, then it will be translated in the Stream file as `bus<7>`.

Although the special characters look similar to a name with an index expression, this row shows an identifier, with special characters, that is being used as an atomic identifier. The character indicates a blank space.

Ignore Current Timestamp

Use the *Ignore Current Timestamp* option to ignore the current timestamp while exporting GDS.

When you select this option:

- Time stamp is not added as a suffix to the standard via and Pcell names. As a result, their names do not change every time you generate the GDS.
- Exported GDS is time-independent. This means that time information is not added to the exported GDS and therefore, contents of GDS do not change every time you run XStream Out.

Command-line option: `ignoreCurrentTimestamp`

Show Completion Message Box

The *Show Completion Message Box* check box enables you to display the Stream out translation complete message box with the number of errors and warnings after the translation is completed. The state of this check box depends on the value of the `xstShowCompletionMsgBox` SKILL variable, which is `t` by default.

XStream Out Editor Forms

This section discusses the following XStream Out Editor Forms:

- [XStream Out Layer Map File Editor Form](#)
- [XStream Out Object Map File Editor Form](#)
- [XStream Out Cell Map File Editor Form](#)
- [XStream Out Reference Library File Editor Form](#)
- [XStream Out Property Map File Editor Form](#)
- [XStream Out Font Map File Editor Form](#)

XStream Out Layer Map File Editor Form

This section describes options on the XStream Out Layer Map File Editor form that is displayed when you click the *Editor* button for the [Layer Map](#) option.

Layer Name	Purpose Name	Stream Layer	Stream Type	Material	Mask Number	Qualifier	Photo Mask Color	Color State

New File

Add Row button Delete Row button

Load Save As and Exit Save and Exit Cancel Help

Add Row

Click the *Add Row* button to create a new layer mapping record.

For more information on columns in a layer mapping record, see [Layer Map File](#).

Layer Name

Use the *Layer Name* field to specify a DFII layer that you want to translate to the Stream format.

The number of layers present in this option are taken from the technology file.

Purpose Name

Use the *Purpose Name* field to specify a DFII purpose that you want to translate to the Stream format.

The number of purposes present in this option are taken from the technology file.

Stream Layer

Use the *Stream Layer* field to specify the Stream layer name to which DFII layer name has to be translated.

Stream Datatype

Use the *Stream Datatype* field to specify the Stream type to which DFII purpose name has to be translated.

Material Type

Use the *Material Type* field to specify the OpenAccess material type to which you want to map the Stream type.

Mask Number

Use the *Mask Number* field to specify the OpenAccess mask number to which you want to map the Stream type.

Qualifier

Use the *Qualifier* field to specify the OpenAccess qualifier to which you want to map the Stream type.

Photo Mask Color

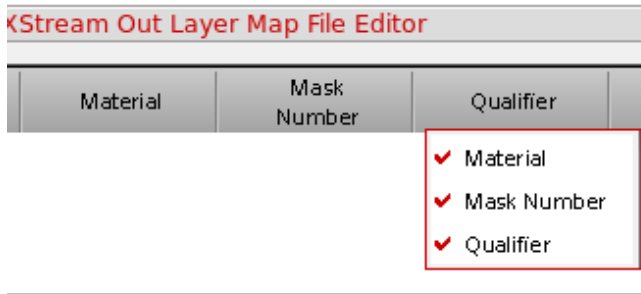
Use the *Photo Mask Color* field to specify the photo mask color of the layer.

Color State

Use the *Color State* field to specify the color state of the layer.

The *Photo Mask Color* and *Color State* fields are visible only if *Coloring Mode* is enabled. See, [Coloring Mode](#).

Show/Hide Columns



To hide either or both *Material*, *Mask Number*, and *Qualifier* columns, right-click on the column heading row. Next, clear the check box of the column you do not want to display.

Delete Row

Click the *Delete Row* button to delete an existing layer mapping record.

Load

Use the *Load* option to load a new layer map file. You can specify multiple layer name map files by using the Load file option iteratively. Using a layer map file, you can map OpenAccess layer name and purpose name pairs to Stream layer number and Stream data type pairs. If you specify an invalid file name, a warning message is generated and the file name is ignored.

The path specified for the layer map file can be absolute or relative to the run directory. For more information, see [Layer Map File](#).

Save As and Exit

Use the *Save As and Exit* button to the save the mapping information in a new file and exit the layer map file editor.

Save and Exit

Use the *Save and Exit* button to the save the mapping information in the current file and exit the layer map file editor.

Design Data Translators Reference

Design Translation Using XStream Translator

Cancel

Use the *Cancel* button to XStream Out Layer Map File Editor form without saving any changes.

Design Data Translators Reference

Design Translation Using XStream Translator

XStream Out Object Map File Editor Form

This section describes options on the XStream Out Object Map File Editor form that is displayed when you click the *Editor* button for the Object Map option.

Object Name	Sub Object Name	Layer Name	Stream Layer	Stream Type	Photo Mask Color

Add Row button Delete Row button

New File

Load Save As and Exit Save and Exit Cancel Help

Add Row

Click the *Add Row* button to create a new object mapping record.

For more information on columns in an object mapping record, see Object Map File.

Object Name

Use the *Object Name* field to select the name of the object to be mapped.

Sub Object Name

Use the *Sub Object Name* field to select the name of the sub object to be mapped.

Layer Name

The *Layer Name* field is displayed only when the Object Name is specified as layerBlockage. Use the *Layer Name* field to specify the OpenAccess layer name to which you want to map the Stream layer.

Stream Layer

Use the *Stream Layer* field to specify the Stream layer number for the OpenAccess object. By default, the Stream layer number is specified as 0.

Stream Type

Use the *Stream Type* field to specify the Stream datatype for the OpenAccess object. By default, the Stream datatype is specified as 0.

Photo Mask Color

Use the *Photo Mask Color* field to specify the photo mask color of the layer.

The *Photo Mask Color* field is visible only if the *Coloring Mode* option is enabled. See [Coloring Mode](#))

A one-time warning will be displayed if the layer/object map file has color entries, but the Coloring Mode is not enabled. In addition, these entries will not be visible on the Layers and Object tabs. Also, these entries will be ignored during translation. However if you enable the Coloring Mode, then these entries are visible.

Delete Row

Click the *Delete Row* button to delete an existing object mapping record.

Load

Use the *Load* button to load the object map file. An object map file is used for preserving OpenAccess objects. If you specify an invalid file name, a warning message is generated and the file name is ignored.

The path specified for the object map file can be absolute or relative to the run directory. For more information, see [Object Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the object mapping information in a new file and exit the object map file editor.

Save and Exit

Use the *Save and Exit* button to save the object mapping information in the current file and exit the object map file editor.

Design Data Translators Reference

Design Translation Using XStream Translator

Cancel

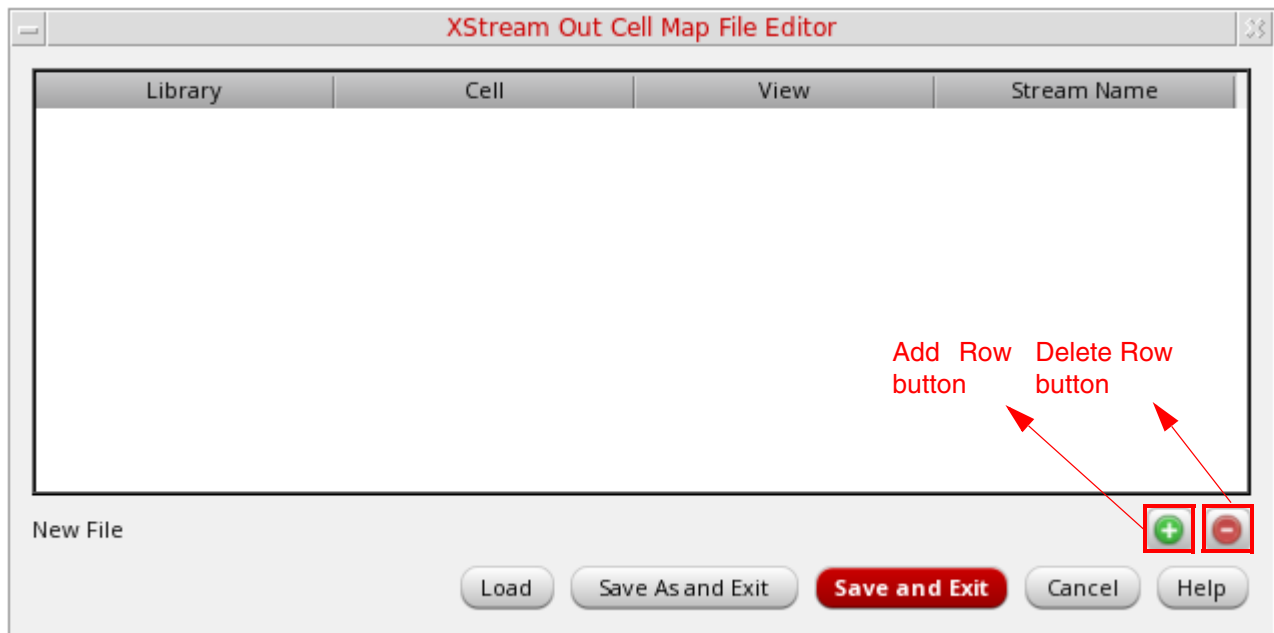
Use the *Cancel* button to exit the XStream Out Object Map File Editor form without saving any changes.

Design Data Translators Reference

Design Translation Using XStream Translator

XStream Out Cell Map File Editor Form

This section describes options on the XStream Out Cell Map File Editor form that is displayed when you click the *Editor* button for the Mapping: Cell Mapping Options option.



Add Row

Click the *Add Row* button to create a new cell mapping record.

For more information on columns in a cell mapping record, see Cell Name Map File.

Library

Use the *Library* field to select a library from which you want to translate a design to OpenAccess. All libraries present in the *Library* field are a list of reference libraries present in the `cds.lib` file.

Cell

Use the *Cell* option to select a library cell that you want to translate to the Stream format. For example, in the above figure, cell `M1_M2` has been selected from library `techLib` for translation.

Design Data Translators Reference

Design Translation Using XStream Translator

View

Use the *View* option to select a view of the specified cell that you want to translate to the Stream format. In the above figure, view `via` of cell `M1_M2` will be translated to the Stream format.

Stream Name

Use the *Stream Name* field to specify the Stream file name in which you want to translate the selected library, cell, and view.

Delete Row

Click the *Delete Row* button to delete an existing cell mapping record.

Load

Use the *Load* button to load a new cell name map file. A cell name map file contains a table that maps cellview names in the OpenAccess database to structure names in the Stream file.

The path specified for the cell name map file can be absolute or relative to the run directory. For more information, see [Cell Name Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the cell mapping information in a new file and exit the cell map file editor.

Save and Exit

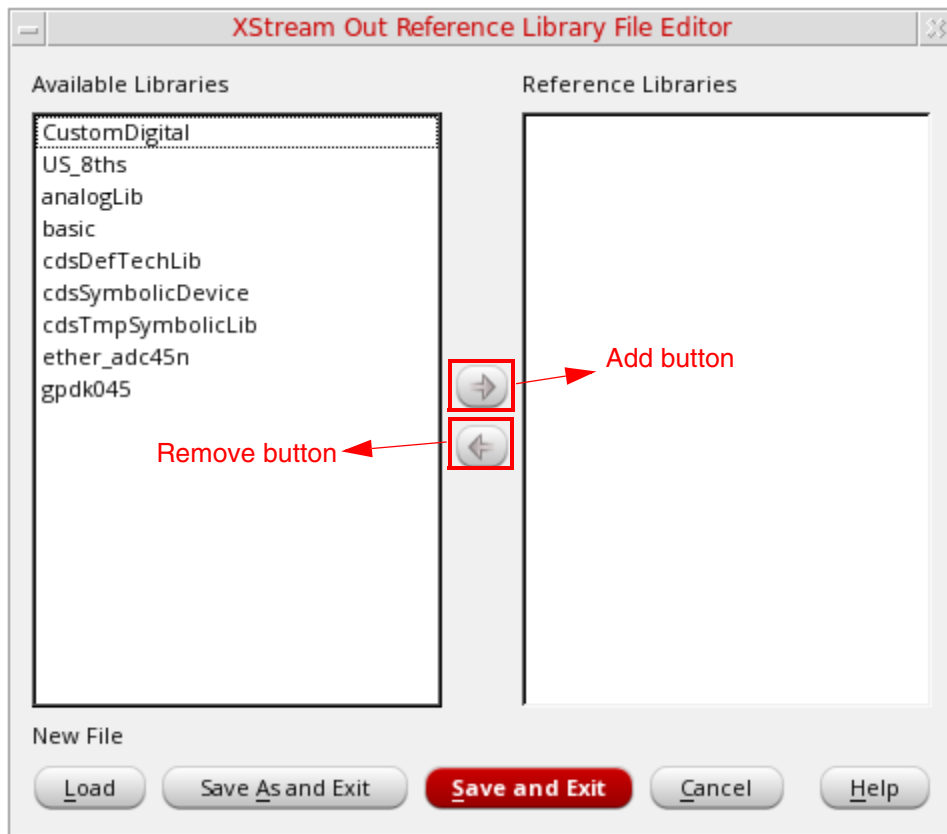
Use the *Save and Exit* button to save the cell mapping information in the current file and exit the cell map file editor.

Cancel

Use the *Cancel* button to exit the XStream Out Cell Map File Editor form without saving any changes.

XStream Out Reference Library File Editor Form

This section describes options on the XStream Out Reference Library File Editor form that is displayed when you click the *Editor* button for the Ref Lib File Name option.



Note: If you want to specify the target library name as reference library then you need to use the `XST_TARGET_LIB` variable in the *Available libraries* field.

Add

Use the *Add arrow* button to add libraries to the *Reference Libraries* field.

The *Reference Libraries* field is used to specify logical library names that you want to consider as reference libraries. The libraries specified in this field should be present in the `cds.lib` file. Otherwise, they are not considered as reference libraries.

During XStream Out, when you search for master cellviews of instances, reference libraries are searched in the same order as specified in the *Reference Libraries* field. If any of the specified libraries do not exist in the `cds.lib` file, an error message is displayed.

Remove

Use the *Remove* button to remove libraries from the *Reference Libraries* field.

Move Up

Use the *Move Up arrow* button to move up selected reference libraries in the Reference Libraries list on the right. This helps you arrange reference libraries in order of preference.

Move Down

Use the *Move Down arrow* button to move down selected reference libraries in the Reference Libraries list on the right. This helps you arrange reference libraries in order of preference.

Load

Use the *Load* button to load the text file containing the list of reference libraries. When you specify the text file, the path and the text file name is displayed in the blank field above the *Load* button.

Save As and Exit

Use the *Save As and Exit* button to save the reference library information in a new file and exit the reference library file editor.

Save and Exit

Use the *Save and Exit* button to save the reference library information in the current file and exit the reference library file editor.

Cancel

Use the *Cancel* button to exit the XStream Out Ref Lib File Editor form without saving any changes.

XStream Out Property Map File Editor Form

This section describes options on the XStream Out Prop Map File Editor form that is displayed when you click the *Editor* button for the Prop Map File Name option.

The screenshot shows the 'XStream Out Property Map File Editor' window. It features a table with columns for 'Property Name', 'Object Type', and 'Property ID'. Below the table, there are 'Add Row' and 'Delete Row' buttons, which are highlighted with red boxes and arrows. The 'Add Row' button is a green square with a white plus sign, and the 'Delete Row' button is a red square with a white minus sign. Below the table, there is a 'New File' section with an 'Option Line' containing a checkbox for 'Property Value Only' and a 'Property Separator' field with a comma. At the bottom of the window, there are five buttons: 'Load', 'Save As and Exit', 'Save and Exit' (highlighted in red), 'Cancel', and 'Help'.

Add Row

Click *Add Row* button to create a new property mapping record.

For more information on columns in a property mapping record, see Property Map File.

Property Name

Use the *Property Name* field to specify the property name that you want to map to the Stream format.

Object Type

Use the *Object Type* option to select the object that you want to translate to the Stream format. The supported object types are: Arc, Dot, Ellipse, Line, Path, Polygon, Rect, Text, ArrayInst, ScalarInst, PathSeg, Route, Marker, Row, AreaBoundary, PRBoundary, SnapBoundary, ClusterBoundary, AreaBlockage, LayerBlockage, AreaHalo, Pin, stdVia, and customVia.

Property ID

Use the *Property ID* field to specify the Stream ID on which you want to map the selected object type and the property name.

Property Value Only

Use the *Property Value Only* format specifier to export the specified *propertyName*'s value of an *objectType* as the value of the corresponding Stream attribute specified in the mapping record to the GDS file. If *Property Value Only* is specified in the property map file, XStream Out ignores the *Property Value Only* defined in the form.

Default value: Deselected

Property Separator

Use the *Property Separator* format specifier to specify a property separator.

Default value: The default property separator is a comma ‘,’.

Delete Row

Click the *Delete Row* button to delete an existing property mapping record.

Load

Use the *Load* button to load a property map file. A property map file preserves properties associated with OpenAccess objects. If you specify an invalid file name, a warning message is generated and the file name is ignored.

The path specified for the log file should be a valid file path. It can be absolute or relative to the run directory. For more information, see [Property Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the property map information in a new file and exit the property map file editor.

Save and Exit

Use the *Save and Exit* button to save the property map information in the current file and exit the property map file editor.

Cancel

Use the *Cancel* button to exit XStream Out Prop Map File Editor form without saving any changes.

XStream Out Font Map File Editor Form

This section describes options on the XStream Out Font Map File Editor form that is displayed when you click the *Editor* button for the Font Map File Name option.

Font vs Stream Value Mapping		
EuroStyle	1	▼
Gothic	1	▼
Math	1	▼
Roman	0	▼
Script	0	▼
Stick	0	▼
Fixed	0	▼
Swedish	0	▼
MilSpec	0	▼

New File

Use the drop-down lists to create direct mapping for selected fonts.

Load

Use the *Load* button to load a new text font map file for font mapping. You use a text font map file to map Stream text font numbers to OpenAccess text font names. If you specify an invalid file name, a warning message is generated and the file name is ignored.

The path specified for the text font map file can be absolute or relative to the run directory. For more information, see [Text Font Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the font mapping information in a new file and exit the font map file editor.

Save and Exit

Use the *Save and Exit* button to save the font mapping information in the current file and exit the font map file editor.

Design Data Translators Reference

Design Translation Using XStream Translator

Cancel

Use the *Cancel* button to exit XStream Out Font Map File Editor form without saving any changes.

XStream Out Option Names in GUI and Template File

The following table provides a list of all the XStream Out GUI options and their corresponding names in the template file. To view the sample template file, see .

Name in the Template File	XStream Out GUI Options
arrayInstToScalar	<u>Convert Arrays to Scalar Instances</u>
case	<u>Default value: Deselected</u>
cellListFile	<u>Cell List File</u>
cellMap	Cell Map File - <u>Load</u>
cellNamePrefix	<u>Cell Name Prefix</u>
cellNameSuffix	<u>Cell Name Suffix</u>
checkPolygon	<u>Report Bad Polygons and Paths</u>
convertDot	<u>Convert Dot To</u>
convertPcellPin	<u>Convert Pcell Pin to</u>
convertPin	<u>Convert Pin to</u>
dbuPerUU	<u>DB Units Per User Units</u>
scale	<u>Design Scale Factor</u>
doNotPreservePcellPins	<u>Do Not Preserve Pcells Pins</u>
donutNumSides	<u>Donut Sides</u>
ellipseNumSides	<u>Ellipse Sides</u>
flattenPcells	<u>Flatten Pcells</u>
flattenPcellsExclusionFile	<u>Pcells Exclusion File</u>
fontMap	Fonts Map File - <u>Load</u>
gdsCellNameLength	<u>GDS Cell Name Length</u>
hierDepth	<u>Translation Depth, Translate Entire Hierarchy</u>
ignoreLines	<u>Do Not Translate Lines</u>
ignoreMissingCells	<u>Ignore Missing Cells</u>
ignoreMissingVias	<u>Ignore Missing Vias</u>

Design Data Translators Reference

Design Translation Using XStream Translator

Name in the Template File	XStream Out GUI Options
ignoreObjectMapFromTech	<u>Ignore Object Map File from Technology</u>
ignorePcellEvalFail	<u>Ignore Pcell Evaluation Failure</u>
ignoreCurrentTimestamp	<u>Ignore Pcell Evaluation Failure</u>
ignoreTopCellPrefixSuffix	<u>Ignore Cell Name Prefix and Suffix for Top Cell</u>
infoToWarn	<u>Convert Info Message IDs to Warnings</u>
labelCase	<u>Label Case Sensitivity</u> (preserve, upper, lower)
labelDepth	<u>Label Translation Depth</u>
labelMap	<u>Use the Cancel button to exit XStream Out Font Map File Editor form without saving any changes.</u>
layerMap	Layer Map File - <u>Load</u>
library	<u>Library</u>
logFile	<u>Log File</u>
maxVertices	<u>Max Vertices</u>
mapAllColorToLocked	<u>Map All Colors to Locked</u>
noConvertHalfWidthPath	<u>Do Not Convert Half Width Path to Polygon</u>
noInfo	<u>Disable Info Message IDs</u>
noObjectProp	<u>Object Type</u>
theextDisplays	<u>Do Not Translate TextDisplays</u>
noOutputUnplacedInst	<u>Do Not Translate Unplaced Instances</u>
noWarn	<u>Disable Warning Message IDs</u>
objectMap	Object Map File - <u>Load</u>
outputDir	<u>Output Directory</u>
outputViaShapesToViaPurp	<u>Output Shapes of Vias to Via Purpose</u>
pathToPolygon	<u>Output Paths as Polygons</u>
pinAttNum	<u>Pin Attribute Number</u>
pinLabelPurp	<u>Pin Label Purpose</u>

Design Data Translators Reference

Design Translation Using XStream Translator

Name in the Template File	XStream Out GUI Options
propMap	Property Map File - <u>Load</u>
propValueOnly	<u>Property Name</u>
rectToBox	<u>Output Rectangles as Box Records</u>
refLibList	Reference Library Map File - <u>Load</u>
replaceBusBitChar	<u>Replace <> with []</u>
respectGDSIINameLimit	<u>Respect 32 Characters Name LimitGDS Cell Name Length</u>
runDir	<u>Run Directory</u>
showCompletionMsgBox	<u>Ignore Current Timestamp</u>
snapToGrid	<u>Snap to Grid</u>
strmFile	<u>Stream File</u>
strmTextNS	<u>Text Namespace</u>
strmVersion	<u>Stream Version</u>
subMasterSeparator	<u>Sub-Master Separator</u>
summaryFile	<u>Summary File</u>
techlib	<u>Technology Library</u>
topCell	<u>Top Cell(s)</u>
translateUnmappedLPP	<u>Translate Unmapped LPPs</u>
userSkillFile	<u>User Skill File</u>
viacutArefThreshold	<u>Via Cut Aref Threshold</u>
viaMap	<u>Via Map File</u>
view	<u>View(s)</u>
virtualMemory	<u>Stream Out from Virtual Memory</u>
warnToErr	<u>Convert Warning Message IDs to Errors</u>
wildCardInCellMap	<u>Wildcard in Cell Map</u>

You can also access and input values in XStream GUI using SKILL commands. To populate values in *Stream Out* form, you need to enter the xstSetField command in CIW. To access values from *Stream Out* form, you need to enter the xstGetField command in CIW.

Design Data Translators Reference

Design Translation Using XStream Translator

All the options listed above are valid for `xstSetField`, `xstGetField`, template file, and `streamOutKeys`. All the options except `virtualMemory` and `showCompletionMsgBox` are valid for command line executable or batch process.

Seeding the XStream Out GUI

You can use the following methods to set default values for a user:

■ Using XST_OUT environment variable

You can assign the XStream Out template file as a value of the `XST_OUT` shell environment variable.

For example:

```
envSetVal( "xstream" "XST_OUT" 'string "demo.txt" )
setenv XST_Out "demo.txt":
```

Sample XStream Out option template file:

```
=====
pinAttNum          "100"
summaryFile        "xStreamOut.sum"
replaceBusBitChar
=====
```

■ Using streamOutKeys SKILL list

You can define a set of option values to load into the XStream In form in the `streamOutKeys` SKILL list. For this, define the pre-seeding options in the `.cdsinit` file because the system automatically loads this file when you start Virtuoso.

The following is a sample `streamOutKeys` SKILL list:

```
;*****
; Copyright (C) 2001 - 2018                                     *
; Cadence Design Systems Inc. All Rights Reserved.             *
;*****
;
;*****
;
; filename : streamOut.il
;
;*****
streamOutKeys=list(nil
'hierDepth          "32767"          ; - (0-32767) - Hierarchy Depth Limit
'strmTextNS         "cdba"           ; - Text NameSpace
'viaCutArefThreshold "0"             ; - Write a STRUCT if the number of cuts are more the specified
'arrayInstToScalar  nil              ; - Convert Arrays to Scalar Instances
'backupGdsLogFiles  nil              ; - Backup GDSII and Log Files
'case               "preserve"       ; - preserve | upper | lower - Case Sensitivity
'cellListFile       ""               ; - The Cell List File Name
'cellMap            ""               ; - Cell Map File Name
```

Design Data Translators Reference

Design Translation Using XStream Translator

'cellNamePrefix	""	;- Prefix to be added to each STRUCT
'cellNameSuffix	""	;- Suffix to be added to each STRUCT
'checkPolygon	nil	;- Report Bad Polygons
'convertDot	"node"	;- polygon node ignore - Convert Dots to
'convertPcellPin	"geometry"	;- geometry text geometryAndText ignore - Convert Pcell Pin to
'convertPin	"geometry"	;- geometry text geometryAndText ignore - Convert Pin to
'diagonalPathToPolygon	nil	;- Convert Paths with non-orthogonal segments to polygons
'doNotPreservePcellPins	nil	;- Do not Preserve PCell Pins
'donutNumSides	"64"	;- Number of sides for the BOUNDARY of donut
'ellipseNumSides	"64"	;- Number of sides for the BOUNDAR of ellipse
'dbuPerUU	"1000"	;- DB Units Per User Units
'reportPrecisionLoss	nil	;- Report Precision Loss in Scaling
'flattenPcells	nil	;- Convert PCells to Geometry
'flattenPcellsExclusionFile	""	;- A file specifying list of Pcells to exclude from flattening
'flattenViaShapesToViaPurp	nil	;- Flatten Vias' Shapes to Via Purpose
'flattenVias	nil	;- Convert Vias to Geometry
'fontMap	""	;- Font Map File Name
'gdsCellNameLength	"0"	;- Truncate GDS Cell/STRUCT name length to specified value
'ignoreCurrentTimestamp	nil	;- Ignores current timestamp
'ignoreLines	nil	;- Ignores Lines During Translation
'ignoreMissingCells Translation	nil	;- Ignores Missing cellViews During Translation and Continue
'ignoreMissingVias and Continue Translation	nil	;- Ignores Missing standard/custom vias During Translation
'ignorePcellEvalFail	nil	;- Ignores the PCell Evaluation Failure
'ignoreTopCellPrefixSuffix	nil	;- Ignores cellName Prefix and Suffix for topCell
'ignoreZeroWidthPathSeg	nil	;- Ignores zero width PathSeg during translation
'ignoreObjectMapFromTech	nil	;- Ignores object map file from technology
'infoToWarn	""	;- Treat the Specified Info Messgaes Ids as the Warnings
'labelCase	"preserve"	;- preserve upper lower - Label Case Sensitivity
'labelDepth	"1"	;- Hierarchical depth to add labels to
'labelMap	""	;- Label Map File Name
'layerMap	""	;- Layer Map File Name
'library	"LIB"	;- Library Name
'logFile	"strmOut.log"	;- Log File Name
'maxVertices	"200"	;- Maximum Vertices in Path/Polygon
'mergePathSegsToPath	nil	;- Merge the Connected Pathsegs to Path
'noConvertHalfWidthPath	nil	;- Do not Convert The Half Width Path to Boundary
'noInfo	""	;- Do not Print the Specified Info Messages Ids
'noObjectProp	nil	;- Do not Write Special OA Object Properties
'noOutputTextDisplays	nil	;- Ignores Text Display Objects
'noOutputUnplacedInst	nil	;- Ignores the Unplaced Instances
'noWarn	""	;- Do not Print the Specified Warning Messages Ids
'objectMap	""	;- Object Map File Name
'outputDir	""	;- The Output Directory for Stream File
'pathToPolygon	nil	;- Convert Paths to Polygons
'pinAttNum	"0"	;- (0-127) - Stream Attribute Number for Preserving Pins
'propMap	""	;- Property Map File Name
'propValueOnly	nil	;- Output Property Values Only
'rectToBox	nil	;- Output Rectangle as BOX
'refLibList	""	;- Reference Library List File Name
'replaceBusBitChar	nil	;- Replace "<>" With "[]"
'respectGDSIINameLimit	nil	;- Respect GDSII limits
'runDir	""	;- Run Directory
'scale	"1.00000"	;- Design Scaling Factor
'showCompletionMsgBox	t	;- Show the Completion Message Box
'snapToGrid	nil	;- Snap XY to Grid Resolution
'strmFile	"top.gds"	;- Stream File Name
'strmVersion	"5"	;- 3 5 6 - Stream Version
'subMasterSeparator	"_CDNS_"	;- Separator to used for sub-master naming
'summaryFile	""	;- Summary File Name
'techLib	""	;- Technology Library Name

Design Data Translators Reference

Design Translation Using XStream Translator

```
'topCell                "top"                ;- Top Cell(s) to Translate
'translateUnmappedLPP    nil                  ;- Allow undefined LPP to be translated by incrementally
mapping LPP using layerMap file, SKILL hook poLayerMap, and automatic layer mapping
'userSkillFile           ""                   ;- Users' Skill File Name
'verbose                 nil                  ;- Generate detailed information in the log and summary files
'viaMap                  ""                   ;- Via Map File Name
'view                    "layout"            ;- View Name(s) to Translate
'virtualMemory           nil                  ;- Do Stream Out from Virtual Memory
'warnToErr               ""                   ;- Treat the Specified Warning Messages Ids as the Errors
'wildCardInCellMap       nil                  ;- Wild Card in Cell Map
)
```

It is possible to combine both the pre-seeding methods but Cadence recommends to avoid using both methods at the same time. If you combine both the methods, `streamOutKeys` SKILL list takes precedence over template file.

Variant Generation during XStream Out

XStream Out generates the color variant that is used to support unlocked color shapes. The effective color of the unlocked color shapes is hierarchy and context dependant. This implies that the unlocked color shapes in a cell can have different colors based on the hierarchy at the `topCell` and its context. In addition, a cell with unlocked color shapes can have many color variants based on the effective color of the unlocked shapes, and XStream Out has to write all those color variants in order to preserve the effective color.

Therefore, to have a GDS with effective color, the following options are required:

- The `-topCell` option to specify the hierarchy and context.
- The `-enableColoring` option to enable the coloring mode.
- The layer map file with the unlocked color mapping.

The array instances can be written as multiple SREF records in the GDS file whenever there is a need for variant generation. This is because both the `-enableColoring` and `-topCell` options are specified and the layer map file has unlocked mapping entries. Therefore, different color variants might need to be generated for each element of an array.

In addition, the GDS can be generated without effective color or hierarchy or context-specific color for the unlocked color shapes. This is because the `-topCell` option is not specified but the `-enableColoring` option is specified and the layer map file has unlocked mapping entries. Therefore, the hierarchy context should be defined using the `-topCell` option.

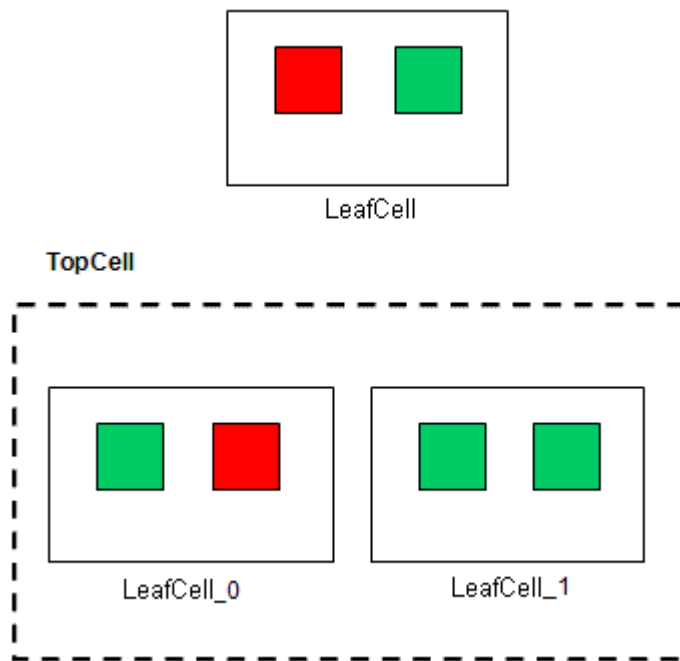
In case of the unlocked color state, XStream Out also saves the information related to the color variant. Let us consider a scenario where a cell has two rectangles with unlocked green and red colors. Now, in the top cell, the placement of this cell may cause violation of the minimum color distance constraint. Due to this, the effective color of the shapes may change

Design Data Translators Reference

Design Translation Using XStream Translator

and color variants of the same instance are generated accordingly. XStream Out will be able to save this information and write color variants as different structs in the GDS file.

Consider the following layout where the rectangle is drawn on the `Metal1:drawing` layer-purpose pair.



Consider the following entries in the layer map file:

<code>Metal1</code>	<code>drawing</code>	<code>1</code>	<code>1</code>	<code>mask1Color</code>	<code>unlocked</code>
<code>Metal1</code>	<code>drawing</code>	<code>2</code>	<code>1</code>	<code>mask1Color</code>	<code>unlocked</code>

XStream Out will write color variants of the instance, `LeafCell`, as different structs in the following GDS file:

STRNAME `LeafCell_0`

BOUNDARY LAYER=1 DATATYPE=1 XY=... ENDEL

BOUNDARY LAYER=2 DATATYPE=1 XY=... ENDEL

ENDSTR

STRNAME LeafCell_1

BOUNDARY LAYER=1 DATATYPE=1 XY=... ENDEL

BOUNDARY LAYER=1 DATATYPE=1 XY=... ELDEL

ENDSTR

STRNAME TopCell

SREF SNAME=LeafCell_0 X=... Y=... ENDEL

SREF SNAME=LeafCell_1 X=... Y=... ENDEL

ENDSTR

Hierarchical Coloring during XStream Out

In XStream Out, hierarchical mode is the mode when you are translating a fully colored hierarchical chip and are exporting colors on all the shapes. In a hierarchical design, effective colors are shifted and color variants are generated for the cells.

To specify the hierarchical coloring mode during XStream Out, specify `-enableColoring -topCell <cellname>` during translation, and include both `locked` and `unlocked` mapping specifications in the layer map file.

If the shape that is being translated belongs to the pre-colored shape categories, these shapes will be mapped according to the locked specifications in the layer map file.

Note: The two user command-line options `-enableColoring` and `-topCell`, and the layer map file determine the multi-patterning mode and behavior of translation using XStream Out.

Pre-coloring during XStream Out

In XStream Out, if you are translating a full chip starting with the top-cell instead of individual cells, you can also export coloring on a set of pre-colored shapes, and map those shapes based on the locked color specifications specified in the layer map file.

The pre-colored shapes are categorized as:

Design Data Translators Reference

Design Translation Using XStream Translator

- Locked shapes, which are the shapes with `locked` color state. These shapes are user-specified colored and locked shapes.
- Anchored shapes

When translating the coloring information on a hierarchical chip, XStream Out, uses the effective color of the shapes by default. One exception to the default behavior is when the color state of the shape is `locked`. In this case, the effective color on the shape should always be the same as the assigned color, so that the translator can use the assigned color instead.

To specify the pre-coloring mode during XStream Out, specify `-enableColoring -topCell <cellname>` during translation and include the `locked` mapping specifications in the layer map file.

AREF Record Generation Behavior in XStream — Deviation from the Stream Format

For rotated arrayed instances, the AREF records generated by XStream in the output Stream file do not conform to the Stream format. This has been happening since many years and the data generated by XStream has become an industry standard. XStream as well as the third-party Stream readers recognize it as desired and are able to read this data correctly.

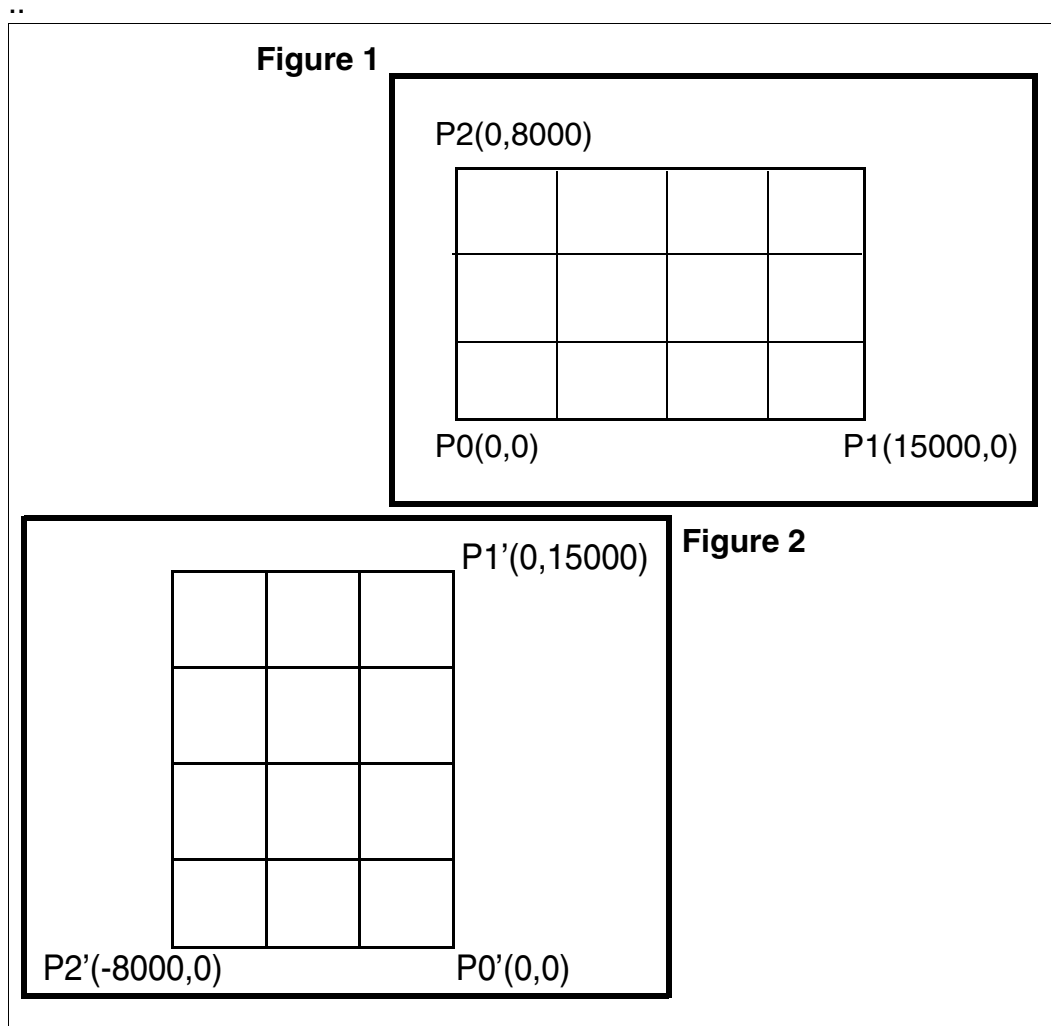
In Stream format, an AREF is represented as described below.

Name	Description
SNAME	Name of the structure
COL	Number of columns
ROW	Number of rows
ANGLE	Angle of rotation
Three XY coordinates	
XY0	Origin point
XY1	Displacement from the origin by the inter-column spacing times the number of columns.
XY2	Displacement from the origin by the inter-row spacing times the number of rows.

Design Data Translators Reference

Design Translation Using XStream Translator

Consider the following example of an arrayed instance



The arrayed instance is placed at zero angle as shown in Figure1. It is represented by the following information in the Stream format:

COL= 4
ROW= 3
ANGLE=0
XY0 = P0 (0,0)
XY1 = P1(15000,0)
XY2 = P2(0,8000)

XStream writes this information correctly in the Stream file.

Figure 2 shows the same arrayed instance rotated at an angle of 90 degrees. Notice that the points P0, P1, and P2 have moved to P0', P1', and P2'. In the Stream format, the coordinates should be represented by the following information:

```
COL= 4
ROW= 3
ANGLE=90
XY0 = P0 (0,0)
XY1 = P1(15000,0)
XY2 = P2(0,8000)
```

But XStream writes the displaced points after rotation instead of writing the original points. The following information is written by XStream:

```
COL= 4
ROW= 3
ANGLE=90
XY0 = P0' (0,0)
XY1 = P1' (0, 15000)
XY2 = P2' (-8000, 0)
```

GZip Support

With increasing complexity of designs, the size of the GDSII file is also increasing. As a result, these files take up a lot of disk space. To overcome this problem GDSII files can be kept in compressed format. But this introduces extra step of gzip/gunzip while importing and exporting using XStream In and XStream Out.

XStream In

You can import a compressed GDSII file, which is in the gzipped format. The only condition is that the input compressed GDSII file name should end with the `.gz` extension. For example, to compress `strmin.gds`, append `.gz` to the file name as follows:
`strmin.gds.gz`.



You cannot use the multithreading option while translating a compressed GDSII file.

XStream Out

You can export to a compressed GDSII file, which is in the gzipped format. The only condition is that the output compressed Stream file name should end with the `.gz` extension. For example, to compress `strmout.gds`, append `.gz` to the file name as follows:
`strmout.gds.gz`.

Note: As of now only gzipped the compressed (`.gz`) format is supported.

Fast Stream Out

You can use the *Export Stream From VM* option located in the *File* menu of Layout Editor to export your design that is currently open in the layout window. When you choose this option, translation starts immediately in Virtual Memory mode and XStream Out GUI is not launched.

This process considers the following arguments during translation from the current layout window:

- The current library (`-library`).
- The current cell (`-topCell`).
- The current view (`-view`).
- The Stream filename (`-strmFile`), which is `<topCell.gds>` by default.

Two SKILL environment variables `strmFilenamePrefix` and `strmFilenameSuffix` control the prefix and the suffix of the GDS filename, respectively. If the GDS file already exists, it will be suffixed with `_1`, `_2` during export.

- The `layerMap` and `objectMap` files, if either one or both files are present in the technology, or if the self-technology library contains these files.
- The `enableColoring` SKILL environment variable controls the value for *Coloring Mode* used during XOasis Out translation.

After the translation is complete, a message is displayed, informing you whether the translation has completed successfully or not. A detailed log is saved in the `strmOut.log` file.

For more information see, [XStream Fast Export SKILL Environment Variables](#).

Layer Mapping in XStream

Layer mapping during XStream translation can be done by using the following ways:

- By using user-defined layer map file. (see [Layer Map File](#))
- By using the user-defined [piLayerMap/poLayerMap](#) function in SKILL file
- By performing automatic layer mapping. (see [Automatic Layer Mapping](#))
- By using the `techlibname.layermap` file, if it is defined in the technology

Note: Here, the layer map file and the SKILL file are user-defined. While, `techlibname.layermap` file is system defined and is available in the technology by default. The translation process gives priority to user-defined input.

Default Precedence for Layer Mapping

By default, XStream applies only one type of layer mapping at a time. Given below is the sequence in which it gives precedence or priority to the different methods used for layer mapping:

1. If you provide a layer map file, the layer map file is used for layer mapping.
2. If you provide a user-defined SKILL file, layer mapping is done using the [piLayerMap/poLayerMap](#) function defined in the SKILL file. This type of layer mapping is also referred to as dynamic layer mapping.

Note: If you provide a layer map file as well as a SKILL file for layer mapping, the layer map file will take precedence over the SKILL file. The [piLayerMap/poLayerMap](#) SKILL function will not be considered in this case.

3. If you do not provide a layer map file or SKILL file, XStream will automatically search for the `techlibname.layermap` file in the technology library. Additionally, in case of Incremental Technology Database (ITDB) hierarchy, XStream will traverse through all the layer map files in the hierarchy. If a layer map file is present in the technology library with some other name, it will not be detected by default.
4. If both the layer map file and the SKILL file are not found, automatic layer mapping takes place.

Note: If you want to force XStream to map layers automatically and ignore the `techlibname.layermap` file, you can override the default behavior of XStream by specifying the `XST_AUTO_LM` keyword for the `-layerMap` option. When you specify the `XST_AUTO_LM` keyword:

- ❑ Even if a layer map file with the name `techlibname.layermap` exists, it will be ignored.

- ❑ XStream will use its internal layer mapping scheme.

Precedence for Layer Mapping when Translate Unmapped LPP is Selected

By default, XStream applies only one type of mapping during translation. Therefore, if you specify the layer map file or the SKILL file, the tool will refer to it, and perform the translation. In this case, any unmapped layers are ignored and not translated.

However, if you choose to translate unmapped LPPs, the tool will first look for and apply any user-defined mapping (layer file or SKILL file), based on the default precedence. Next, the tool will perform automatic layer mapping, which will then translate all unmapped layers also.

Automatic Layer Mapping

XStream uses number-based mapping to perform automatic layer mapping in XStream In and XStream Out.

Automatic Layer Mapping in XStream In

While performing automatic layer mapping, XStream In tries to use the `strmLayer/strmType` number for the `dfIILayer:dfIIPurpose` number and creates `L<strmLayer>/P<strmLayer> (layer:purpose)`, if required.

For the reserved `dfIILayer:dfIIPurpose` number, it creates `L<strmLayer>_XSTIN/P<strmType>_XSTIN dfIILayer/dfIIPurpose`.

During XStream In, if you do not provide a layer map file, then for each Stream layer or Stream datatype present in the Stream file, a new layer-purpose pair is created in the technology file. For example, if Stream layer is 2 and Stream datatype is 3, then layer L2 and purpose P3 is created in the technology file. For every new layer-purpose pair, XStream In populates the `techDisplays` section of the technology file. The display packets are defined in the default `display.drf` file are used to display the newly created layer-purpose pair. XStream supports around 75 display packets. These include: `y0 - y9`, `annotate - annotate9`, `hilite - hilite9`, `Unrouted - Unrouted9`, `y0Flt - y9Flt`, `designFlow - designFlow9`, `text - text2`, `device - device2`, `deviceAnt`, `deviceLbl`, `markerAno`, `markerInf`, `markerAck`, `markerSer`, `markerScr`, `markerCrt`, and `markerFat`.

Also, if you select the *Merge Purposes To Drawing* option then all the undefined purposes are mapped to drawing.

Automatic Translation of Unmapped Layers in XStream In

Automatic layer mapping works differently in XStream In when you use the `-translateUnmappedLPP` option. In XStream In, all shapes on an unmapped `strmLayer/strmType` combination are mapped to `L<strmLayer>_XSTIN:P<strmType>_XSTIN dfIILayer:dfIIPurpose`.

For example, if there is an unmapped `strmLayer/strmType` combination `100/100`, then all shapes on this will be mapped to `dfIILayer:dfIIPurpose L100_XSTIN:P100_XSTIN`.

Automatic Layer Mapping in XStream Out

By default, while performing automatic layer mapping, XStream Out tries to use the `dfIILayer:dfIIPurpose` number for the `strmLayer/strmType` number. For `dfIILayer:dfIIPurpose L<strmLayer>_XSTIN/P<strmType>_XSTIN` it uses `strmLayer/strmType`.

Automatic layer mapping works the same way when you use the `-translateUnmappedLPP` option to map unmapped layers during XStream Out.

For example, if there is an unmapped layer:

`M1/drawing1` with the `dfIILayer/dfIIPurpose` number as `30/1`, then, all shapes on `M1/drawing1` will be mapped to `strmLayer/strmType 30/1`

However, if there is an unmapped layer:

`L100_XSTIN:P100_XSTIN` with a random `dfIILayer/dfIIPurpose` number, then, all shapes on `L100_XSTIN/P100_XSTIN` will be mapped to `strmLayer/strmType 100/100`.

Under certain conditions, the default automatic layer mapping during XStream Out might lead to collisions. During a collision, the shapes on two or more `DFII layer/purpose` might get translated to the same `strmLayer/strmType`.

Collisions during XStream Out can happen, if either of the following conditions is true:

- The `dfIILayer` number or `dfIIPurpose` number is greater than 65535 in the design.
- The `-translateUnmappedLPP` option is used and a layer map file with conflicting entries is defined.

To prevent collisions, automatic mapping is done differently when either of the above-mentioned conditions is true.

If `dfIILayer` number or `dfIIPurpose` number is greater than 65535 in the design, and if we have:

Shape `s1` on `M1/drawing` with the `dfIILayer/dfIIPurpose` number `1/-1`

Shape `s2` on `L65536/drawing` with the `dfIILayer/dfIIPurpose` number `65536/-1`

Then, `s1` will be mapped to `1/0` and `s2` will be mapped to `5000/0`.

To prevent collision:

Design Data Translators Reference

Design Translation Using XStream Translator

- When the `dfIILayer/dfIIPurpose` number is lower than 65535, default automatic mapping in which the `dfIILayer` number is the same as the `strmLayer` is done. Here, the `dfIILayer` number for M1 is 1. Therefore, shape `s1` is mapped according to this number.
- When the `dfIILayer/dfIIPurpose` number is higher than 65535, then non-default automatic mapping that starts from 5000 is performed. Shape `s2` is mapped according to this.

If `-translateUnmappedLPP` is used, and if we have:

Shape `s1` on M1/drawing with `dfIILayer/dfIIPurpose` number 1/-1

Shape `s2` on L10/drawing with `dfIILayer/dfIIPurpose` number 10/-1

And, the layer map file has the following entry:

Metall	drawing	10	0

Then, `s1` will be mapped to 1/0 and `s2` will be mapped to 5000/0.

To prevent collision:

- When the `dfIILayer/dfIIPurpose` number is included in the layer map file, mapping is done as defined in the file. Shape `s1` is mapped according to this.
- Even when the `dfIILayer` number is lower than 65535, default automatic layer mapping is not done because the `dfIILayer` number 10 is used in the layer map file. Therefore, non-default automatic layer mapping that starts from 5000 is performed for shape `s2`.

After XStream Out, the mapping information is available in the `xStrmOut_layermap.txt` file. During automatic layer mapping, for each (layer, purpose) pair in OpenAccess design, if the layer number and purpose number are within the valid range for Stream (0 to $2^{16}-1$), they are mapped to the corresponding Stream layer and datatype. Purpose drawing is mapped to purpose 0.

Also, those layer numbers and purpose numbers that are in the range of 2^{16} to $2^{32}-1$, are mapped to the available Stream layer and purpose in the range, 0 to $2^{16}-1$.

Note: If `xStrmOut_layermap.txt` is not writable, then a new file `xStrmOut_layerMap.userName.txt` is generated. However, if `xStrmOut_layerMap.userName.txt` is also not writable, a new file `xStrmOut_layermap.userName.n.txt` is generated automatically and the mapping information is added to this file. Here, *n* stands for 1, 2, and so on.

Using Map Files During Translation

Map files control the behavior of XStream In and XStream Out. Using map files, you can map layers, cellview names, fonts, and properties in the OpenAccess database to the Stream layers, structure names, Stream fonts, and attributes in the Stream file to each other. The mapping support by using the SKILL file is yet to be implemented.

The various types of map files that are used during translation are:

- [Layer Map File](#)
- [Label Map File](#)
- [Text Font Map File](#)
- [Property Map File](#)
- [Cell Name Map File](#)
- [Via Map File](#)
- [Object Map File](#)

Layer Map File

A layer map file is an ASCII file that maps the layer numbers and data types used in the Stream file to the layer-purpose pairs in OpenAccess, and vice-versa. You can provide multiple layer map files for mapping. In addition to mapping layers for translation, you can also map layers by merging many layers into a single layer.

Layer Map File Format and Conventions

A layer map file has the following format:

```
<Layer Name> <Purpose Name> <Stream Layer Number> <Stream DataType>  
[Material Type] [Mask Number] [Qualifier] [Photomask Color] [Color  
State]
```

This file must follow the conventions described below.

- It must have the following four mandatory fields:
 - ☐ OpenAccess layer name
 - ☐ OpenAccess purpose name
 - ☐ Stream layer number

Design Data Translators Reference

Design Translation Using XStream Translator

- ☐ Stream data type
- It can also include the following optional fields:
 - ☐ Material Type
 - ☐ Mask Number
 - ☐ Qualifier
 - ☐ Photomask Color
 - ☐ Color State
- The fields must be separated by one or more spaces or tabs.
- Each mapping record must be a single line.
- The Stream layer number and the Stream data type must be in the range 0 to 32767.
- Any record starting with ; or # is considered a comment and is ignored.

Using Mandatory Fields In the Layer Map File

You can map a Design Framework II layer-purpose to more than one Stream layer and data type pair. To do this, separate the ranges with hyphens and the individual numbers and ranges with commas. The layer map file must contain all layers that occur in the data translation. *Export - Stream* converts only layers listed in the layer map file. (It does not convert layers that are not listed in the layer map file.) The following is an example of an *Export - Stream* layer mapping file:

# DFII #Layer Name	DFII Layer-Purpose	Stream Layer Number	Stream Data Type
metal1	drawing	1, 2	0
metal1	drawing	3, 4-9	0
text, diff	drawing	5	0
metal, pwell	drawing	6	0

Design Data Translators Reference

Design Translation Using XStream Translator

The first entry in the above example, is the same as the following two entries:

# DFII #Layer Name	DFII Layer-Purpose	Stream Layer Number	Stream Data Type
metal1	drawing	1	0
metal1	drawing	2	0

Similarly, you can map more than one Stream layer to one Design Framework II layer-purpose pair. The following is an example of an *Import - Stream* layer map file:

# DFII #Layer Name	DFII Layer-Purpose	Stream Layer Number	Stream Data Type
metal	drawing	13, 10, 18	1
poly	boundary	40	0-63

The first entry in the above example, is the same as the following three entries:

# DFII #Layer Name	DFII Layer-Purpose	Stream Layer Number	Stream Data Type
metal	drawing	13	1
metal	drawing	10	1
metal	drawing	18	1

Note: The valid range for layer/datatype is 0-32767 (inclusive of both values).

Design Data Translators Reference

Design Translation Using XStream Translator

Sample Layer Map File

#Open Access Layer Name	Open Access Purpose Name	Stream Layer	Stream DataType
text	drawing	5	0

Important

The same Stream layer number and data type pair might be mapped to more than one Design Framework II on OpenAccess layer-purpose pairs. In this scenario, XStream In translates all the entries for that Stream layer only to the first Design Framework II layer-purpose pair specified in the layer map file.

If you want XStream In to retain the different layer-purpose pairs mapped to the same Stream layer number and data type pair, update the layer name map file used during the XStream Out stage itself. Change either the Stream layer number or the Stream data type for all such layer-purpose pairs. Use the updated layer name file to again run XStream Out and then XStream In to obtain the desired results.

For example, if the layer map file looks like:

#OpenAccess Layer Name	OpenAccess Purpose Name	Stream Layer Number	Stream DataType
metal1	drawing	2	0
metal2	boundary	2	0
metal3	boundary	2	1

where, metal1 and metal2 have the same Stream layer number and data type. Even though metal3 has the same Stream layer number but it differs from metal1 and metal2 in the Stream data type. So, XStream In translates the Stream layer number 2 with data type 0 only to the first layer-purpose pair mapped to this Stream layer and data type, that is, metal1 drawing. XStream In translates Stream layer number 2 with data type 1 to metal3 boundary.

Using Optional Fields In the Layer Map File

A layer map file can have Material Type, Mask Number, Qualifier, Photomask Color, and Color State optional fields.

Note: Photomask Color and Color State fields are supported in advanced nodes only.

- The layer map file format with optional fields work with both the command line and GUI.

Design Data Translators Reference

Design Translation Using XStream Translator

- The translation of shapes is based on the layerMap entries.
- Material type and Mask number are considered during XStream In and ignored during XStream Out.
- Qualifier is considered only during XStream Out and is ignored during StreamIn. Valid values for the Qualifier column are Pin, floating, cutSize:<float>:<float>, ignoreLPP, and WSPRegionType. The complete mapping record is ignored during XStream In when the ignoreLPP qualifier is used.

Here, qualifier:

- ❑ pin ensures that the specified shape is a pin. The qualifier supports pins onto a different Stream layer datatype.
(see [Example for Qualifier Pin Shapes](#))
- ❑ floating checks if the specified shape is connected to any other shape. It fills onto a different Stream layer datatype. This qualifier is only valid for purpose fill and fillOPC.
- ❑ cutsize ensures that the bbox of the given shape matches the specified cutsize. The qualifier enables mapping the via cuts with different cut sizes onto different Stream layer and Stream datatype. The cut sizes (height:width) are calculated from the bounding box of the shapes on a cut layer, whose material type is cut. This qualifier is valid only for material cut, contactlessMetal, or trim.
(see [Example for Qualifier cutsize](#))
- ❑ ignoreLPP ignores the specified LPPs during mapping and does not display a missing LPP warning for them. This qualifier is not valid with any other optional field. The complete mapping record is ignored during XStream In when ignoreLPP qualifier is used
(see [Example for Qualifier IgnoreLPP](#))
- ❑ WSPRegionType enables the translation of Width Spacing Pattern (WSP) regions.

Syntax for using WSPRegionType:

```
<OA layer> <OA purpose> <LayerNumber> <DataTypeNumber>  
[<WSPRegionType>]
```

Here, WSPRegionType can be:

WSPRegionType:user, which is created by the user

WSPRegionType:system, which is created by the system

When WSP entries are added to the layer map file, there should not be any other optional field. If any optional field is added, the entire entry is ignored.

Design Data Translators Reference

Design Translation Using XStream Translator

XStream Out supports design-specific WSPDefs and multiple-WSPDefs per LPP. The translator exports the WSPDefName as a text record on the same LPP as the pattern region shape in the GDS file.

(see [Example for WSPRegionType](#))

- The `Photomask Color` and `Color State` optional fields are supported both in XStream In and XStream Out.
 - ❑ Valid values for `Photomask Color` field are `mask1Color`, `mask2Color`, `mask3Color`, `mask4Color`, `mask5Color`, `mask6Color`, `mask7Color`, and `mask8Color`
 - ❑ Valid values for the `Color State` field are `locked` and `unlocked`.
(see [Example for Color Fields](#))

Video

For a video on how to lock colors and how the locked colors propagate to the streamed out GDSII file, you can see [Color Locking and Color Aware Xstream](#) on Cadence Online Support.

Allowed Combinations for the Mandatory and Optional Fields

- ❑ `<Material>`
- ❑ `<Material> <MaskNumber>`
- ❑ `<Material> <MaskNumber> <Qualifier>`
- ❑ `<Material> <Qualifier>`
- ❑ `<Qualifier>`
- ❑ `<Material Type> <PhotoMaskColor> <ColorState>`
- ❑ `<Material Type> <MaskNumber> <PhotoMaskColor> <ColorState>`
- ❑ `<Material Type> <MaskNumber> <Qualifier> <PhotoMaskColor> <ColorState>`
- ❑ `<Material Type> <Qualifier> <PhotoMaskColor> <ColorState>`
- ❑ `<Qualifier> <PhotoMaskColor> <ColorState>`

Example for Qualifier cutsizes

The cases using different layerMap entries with the `cutsizes` qualifier are listed below:

Design Data Translators Reference

Design Translation Using XStream Translator

Let us assume a design has the following shapes on cut layer, `Via1`, and purpose, drawing:

- R1: a rect with a bounding box of `0.5x0.5`
- R2: a rect with a bounding box of `1.0x1.0`
- R3: a rect with a bounding box of `2.0x4.0`
- R4: a rect with a bounding box of `4.0x2.0`
- P1: a polygon with a bounding box of `0.5x0.5`
- P2: a polygon with a bounding box of `1.0x1.0`
- P3: a polygon with a bounding box of `1.0x0.5`
- PA1: a path with a bounding box of `0.5x0.5`
- PA2: a path with a bounding box of `1.0x1.0`
- **Case 1:**

```
Via1    drawing    2          1          cut
```

In this case, all the shapes (R1, R2, R3, R4, P1, P2, P3, PA1, and PA2) are written on `Streamlayer:datatype, 2:1`.

■ Case 2:

```
Via1    drawing    2          1          cut
```

```
Via1    drawing    2          2          cut    cutSize:1.0:1.0
```

In this case, all the shapes with bounding box `1.0x1.0` (R2, P2, and PA2) are written on `Streamlayer:datatype, 2:2`. All other shapes use default mapping, therefore, R1, R3, R4, P1, P3, and PA1 will be written on `Streamlayer:datatype, 2:1`.

■ Case 3:

```
Via1    drawing    2          2          cut    cutSize:1.0:1.0
```

In this case, all the shapes with bounding box `1.0x1.0` (R2, P2, and PA2) are written on `Streamlayer:datatype, 2:2`. All other shapes are dropped with a one-time warning message because default mapping is not provided.

■ Case 4:

```
Via1    drawing    2          3          cut    cutSize:2.0:4.0
```


In this case, all the shapes with bounding box `2.0x4.0` and `4.0x2.0` (R3 and R4) are written on `Streamlayer:datatype, 2:3`. All other shapes are dropped with a one-time warning message because default mapping is not provided.

■ Case 5:

No mapping provided for `Via1` drawing.

In this case, all the shapes are dropped with a one-time warning message because no mapping is provided.

Example for Qualifier Pin Shapes

Let's assume you have pin shapes and some regular shapes on an LPP (layer-purpose pair). Now, if you want to Xstream Out only pin shapes, other than the regular shapes, on to a different Stream layer, then you can use the `pin` qualifier. For example, if layer map is:

```
M4 drawing 8 1
```

then, all `<M4 drawing>` data should go on `<8:1>`. However, if you want the pin shapes on the same layer `<M4 drawing>` should go on Stream layer-`datatype, <7:1>`, then the following entry needs to be specified in the layer map.

```
M4 drawing 7 1 pin
```

The cases using different layerMap entries with the pin qualifier are listed below:

■ Case 1:

Let us assume you have the following entries in the layer map file:

```
M4 drawing 8 1
```

```
M4 drawing 7 1 pin
```

Then, all regular shapes will go to Stream layer, `<8:1>`. Whereas, all pin shapes will go to `<7:1>`.

■ Case 2:

Let us assume you only have the following entry in the layer map file.

```
M4 drawing 8 1
```

Then, all the shapes, including pin shapes will go to Stream layer, `<8:1>`.

■ Case 3:

Let us assume you only have the following entry in the layer map file.

```
M4 drawing 7 1 pin
```

Then, all pin shapes will go to Stream layer, <7 : 1>. All other shapes will be dropped.

Example for Qualifier IgnoreLPP

Let us assume you have the following entries in the layer map file:

```
m1 drawing 1 1
```

```
m2 drawing 0 0 ignoreLPP
```

Additionally, there are three shapes, one each on:

```
m1 drawing
```

```
m2 drawing
```

```
m3 drawing
```

Then, the shape on `m1 drawing` will go to the Stream layer: data type <1 : 1> and the shapes on the `m2 drawing` and `m3 drawing` LPPs will not be written in the GDS file. However, warning 35, which informs that objects in a specific LPP are ignored, will be displayed only for the shape on the `m3 drawing`. The warning will not be displayed for the shape on `m2 drawing` because the `ignoreLPP` qualifier is specified for this entry in the layer map file .

Example for WSPRegionType

Consider a layer map file with the following entries:

Metal1	localWSP	1	0	Pin
Metal1	localWSP	2	0	WSPRegionType:user
Metal1	localWSP	3	0	WSPRegionType:system

In this case, the first entry in the layer map file will be ignored during translation.

During Stream Out, translated regions are written to the GDS layer-datatype pair depending on their created type, user or system.

Design Data Translators Reference

Design Translation Using XStream Translator

Currently, this feature is supported only by XStream Out. During XStream In, the create type attribute of a region is not restored. As a result, all region shapes in the GDS will appear as user created after GDS import.

Example for Color Fields

Metal1	drawing	1	1		mask1Color	locked
Metal1	drawing	1	2		mask2Color	locked
Metal1	drawing	1	3	pin	mask2Color	locked
Metal1	drawing	1	4			
Via1	drawing	2	1			
Metal2	drawing	3	1	metal		
Via2	drawing	4	1	cut	4	
Metal3	fill	5	1	metal		floating
Via3	drawing	6	1	cut	6	cutSize:1.0:1.0
Metal4	drawing	7	1	pin		
Metal4	drawing	8	1			

Enhanced Layer Mapping for XStream In

The layer map functionality is enhanced to set the color information on the created shape based on the mapping entry used for `strmLayer` or `strmType`. XStream In supports color mapping for both locked and unlocked color state, so that created shapes can have either locked or unlocked color state.

In the layer map file, XStream In considers all the color mapping entries, if the `-enableColoring` option is specified. Otherwise all the color mapping entries will be ignored and the warning message will be displayed.

The Photo mask color information, which is specified in an entry of the `layerMap` file is validated against the number of photo mask color in the technology library for the particular `oaLayerName`. In case there is any inconsistency in the entry of the `layerMap` file, then the entry will be ignored and the warning message will be displayed.

In addition, XStream In uses the color information of the mapped entry to set the color information on the created shapes. Therefore, in XStream In both the locked color shapes and unlocked color shapes can be created. Also, coloring can be done on the third party GDS with the color state as either locked or unlocked. Here, the key to differentiate the color information is `strmLayer/strmType`.

Enhanced Layer Mapping for XStream Out

The layer map functionality is enhanced to map color shapes to `strmLayer` or `strmType` based on the color information of the shape. XStream Out supports color mapping for both locked and unlocked color shapes, according to the mappings specified in the layer map file.

In the layer map file, XStream Out considers all the color mapping entries, if the `enableColoring` option is specified. Otherwise all the color mapping entries will be ignored and a warning message will be displayed.

The photo mask color information, which is specified in an entry of the `layerMap` file is validated against the number of photo mask color in the technology library for the particular `oaLayerName`. In case there is any inconsistency in the entry of the `layerMap` file, then the entry will be ignored and a warning message will be displayed.

Note: If XStream Out encounters an unlocked color shape, then it will try to use non-color mapping and the one-time warning message will be displayed for it. If non-color mapping is also not available for that `oaLPP`, then a warning will be displayed and the un-mapped shapes will be ignored.

XStream Out follows the following order to map the locked color shape on a specific `strmLayer/strmType`.

1. The entry with the Photo Mask Color and Color State fields of the shape will be considered. If such color aware mapping is not available for the `oaLPP` with specific color, then a warning will be displayed on per-shape basis.
2. The entry without the Photo Mask Color and Color State fields will be considered. If non-color aware mapping is not available for that `oaLPP`, then this un-mapped shape will be ignored and a warning will be displayed on per shape basis.

The color aware mapping will use the following order to consider the *qualifier* field:

1. **Color Pin Mapping:** The mapping entry with Photomask Color, Color State, and *Pin* as Qualifier.
2. **Color cutSize Mapping:** The mapping entry with Photomask Color, Color State, and `cutSize:<float>:<float>` as Qualifier.

3. **Normal color mapping:** The mapping entry with `Photomask Color` and `Color State`.

Non-Color aware mapping will use the following order to consider the *qualifier* field:

1. **Pin Mapping:** The mapping entry with *Pin* as *Qualifier*.
2. **CutSize Mapping:** The mapping entry with `cutSize:<float>:<float>` as *Qualifier*.
3. **Normal/default mapping:** The mapping entry with only four must fields and without *Qualifier* field.

Label Map File

The label map file is an ASCII file that generates labels for the top level nets and terminals of a design. You use the `-labelMap` option to specify that labels be included during XStream Out. To XStream Out label information for nets and terminals that are present in the various levels in the hierarchy, you can use the `-labelDepth` option to specify the level in the hierarchy where label information will be streamed out.

The following table describes the two options:

Option	Description
<code>-labelDepth</code> <i>number</i>	Specifies to what depth of the design hierarchy labels are generated. If the <code>-hierDepth</code> option is also used, and the <code>-labelDepth</code> value is greater than the <code>-hierDepth</code> value, labels are generated for all levels of the hierarchy. If the <code>-labelDepth</code> option is not specified, labels are generated for only the top level of the design.
<code>-labelMap</code> <i>filename</i>	Specifies a label map file that determines the object type for which labels are created and determines the mapping of shapes on an <code>oaLayer</code> and <code>oaPurpose</code> pair to a GDS layer and GDS data type pair. The object type specified in the label map file is either a terminal or a net.

Labels are created for each top level cell, or if the `-labelDepth` option is used, labels are created for each top level cell and its children down to the specified level of hierarchy. Labels are created for all nets at the top level. If a net has a terminal and the terminal name differs from the net name, the label that is created corresponds to the terminal name. If a label map file includes more than one mapping for a combination of object, `oaLayer`, and `oaPurpose`, the first of these entries in the label map file is used and a warning is displayed.

Design Data Translators Reference

Design Translation Using XStream Translator

If you do not provide a label map file, XStream will automatically search for the `techlibname.labelmap` file in the technology library. A label map file, if present in the technology library with some other name, is not detected. In case of Incremental Technology Database (ITDB) hierarchy, XStream will traverse through all the label map files in the hierarchy.

The label map file format uses one line per mapping where the five fields are each separated from others by tabs or spaces. Lines beginning with the `#` character are comment lines and are ignored by the translator.

Following is an example of a label map file.

#	Object	oaLayer	oaPurpose	GDS Layer Number	GDS Data Type
	Term	Metal5	drawing	131	0
	Term	Metal6	drawing	132	0
	Net	Metal5	drawing	131	1
	Net	Metal6	drawing	132	1
	Term	Metal5	drawing	131	2
	Term	Metal6	drawing	132	2
	Net	Metal5	drawing	131	3
	Net	Metal6	drawing	132	3
	Term	Metal5	drawing	131	4
	Term	Metal6	drawing	132	4
	Net	Metal5	drawing	131	5
	Net	Metal6	drawing	132	5

The label map file uses five required fields, listed in the following table.

Field Name	Value Type
Object	[Net Term]
Layer	An oaLayer name.
Purpose	An oaPurpose name.
GDS Layer	A numeric value representing a GDS layer.

Field Name	Value Type
GDS Data Type	A numeric value representing a GDS data type.

Text Font Map File

The text font map file is an ASCII file that maps the Stream text font numbers to OpenAccess text fonts. You can specify the text font map file for mapping. You can also specify the text font mapping information in a SKILL file.

If you provide a text font map file, the mappings specified in the file are used for mapping fonts. If you specify a SKILL file, then layer mapping is done by using the SKILL functions. However, if you specify both a text font mapping file and a SKILL file for font mapping, the translator ignores the SKILL file.

You can create a text map file by using a text editor. A text font map file must conform to the following conventions:

- It must have two fields:
 - ☐ Stream font number
 - ☐ OpenAccess font name
- The fields must be separated by one or more spaces or tabs.
- Each mapping record should be a single line.
- Any record starting with a # is considered a comment and is ignored.

A sample text font map file is as follows:

```
#This is a sample XStream In text font file
0   Fixed
1   EuroStyle
2   Gothic
3   Math
```

Note: Legal Stream font numbers are 0, 1, 2, and 3. If you use any other number, the resulting Stream font number is unpredictable. For more information about Stream font numbers, see Appendix A [Stream File Description](#).

There is a minor difference in the font names that you use in the text font map file as compared to the font names displayed in the GUI. In the template file do not use the `oac` prefix and the `Font` suffix. For example, define the `oacFixedFont` font as `Fixed`.

Design Data Translators Reference

Design Translation Using XStream Translator

During XStream Out, if a font name in OpenAccess is not mapped using a text font map file, then the fonts are translated to the Stream file as shown below:

Stream Font Number	OpenAccess Font Name
0	Fixed
1	EuroStyle
2	Gothic
3	Math
0	All other fonts

If any font map in the text font map file is not in the proper format, a warning message is generated and the mapping for that font is ignored.

Property Map File

A property map file is an ASCII file that can be used to map object types and property names from the OpenAccess database to the Stream file. You can create a property map file using any text editor.

In a property map file,

- Each mapping record must be a single line.
- Fields in the mapping record must be separated with one or more spaces or tabs.
- The same attribute number can be used to store the value of multiple number of properties.
- Multiple properties separated by a property separator can be specified on the same attribute number.
- Records starting with # are considered comments and are ignored.
- Multiple properties separated by a comma can be specified in the same mapping record.
- There is one option record and several mapping records.
 - The syntax for an option record is:

`[-f<Format_Specifier>]`

where, the *Format_Specifier* lets you specify the format in which properties and their values are written to the Stream file. It can have the following values:

Design Data Translators Reference

Design Translation Using XStream Translator

- `-fv`: Format the output to show the property values only. The property value is written as a value of the Stream attribute specified in the mapping record. This shows “inputOutput” in the Stream file.
- `-fp` (default setting): Format the output to show the property’s name-value pair. This shows “accessDir=inputOutput” in the Stream file.
- A mapping record in the property map file consists of three fields: attribute number, object type, and property name.

where,

<i>attributeNumber</i>	Specifies the Stream attribute number that is used to store the value of the property. This can be any number other than 126 and 127 which are reserved attribute numbers.
<i>objectType</i>	Specifies the object type for which a particular property is to be translated. Valid values of OpenAccess object types are Arc, Donut, Dot, Ellipse, Line, Path, Polygon, Rect, Text, ArrayInst, ScalarInst, PathSeg, Route, Marker, Row, AreaBoundary, PRBoundary, SnapBoundary, ClusterBoundary, AreaBlockage, AreaHalo, LayerBlockage, LayerHalo, and Pin.
<i>propertyName</i>	Specifies the property name to be preserved. Valid values include any ASCII string.

In OpenAccess, database objects, such as instances and shapes, can have certain properties attached to them. By default, these properties are not translated during XStream Out. A property map file provides a mechanism to translate and preserve these properties during XStream Out and XStream In. It is also used to map object types and property names from the OpenAccess database to the Stream attribute numbers in the Stream file to each other.

During XStream Out, sufficient information is written to the Stream file for preserving the properties if you translate these properties as property name-value pairs. Therefore, while streaming in the same information, you must not specify a property map file. A property map file is required to preserve properties when you translate only the value of a property during XStream Out. If the same attribute number is associated with different user property names of the same object, then properties get separated in the output Stream file.

Sample Property Map File

```
-fp -s ';'
12      rect          signame
13      path          prop1, prop2
14      scalarInst    instName
```

In the sample property map file shown above, *prop1* and *prop2* are associated with the same attribute number 13. During XStream Out, these properties get separated by a property separator in the output Stream file.

Property map file supports certain keywords which are not OpenAccess properties. These keywords are described below.

- `instName`

You can translate and preserve instance names during XStream Out and XStream In. This can be done by specifying `instName` as a property for the object type `scalarInst` in the property map file.

- `netName`

The `netName` keyword is used to preserve information pertaining to net names. For more information, see [Preserving Net Names](#). Ideally, property ID should be same for all type of objects. This is because GDS supports fewer objects and different property IDs can cause issue in preserving the `netName`.

Cell Name Map File

A cell name map file is an ASCII file that stores mapping information between the OpenAccess cellview names and Stream cell names. A cell name map file can be created by using any text editor. You can also specify cell name mapping information in a SKILL file. In a cell name map file:

- Each mapping record must be a single line.
- Fields must be separated with one or more spaces or tabs.
- Any line starting with `#` is considered a comment and is ignored. Blank lines are also ignored.
- Each record must contain the following four fields:
 - ❑ Library name
 - ❑ OpenAccess cell name
 - ❑ OpenAccess view name

❑ Stream structure name

If any record in the cell name map file is not in the proper format, the record is ignored and a warning message is generated. You can map cell names by providing a cell name map file, a user-defined SKILL file, or by using the *Case* option. The *Case* option is used when the cell name map file or the user-defined SKILL file is not used for cell mapping.

In addition to changing the cellview names by using a cell name map file or a SKILL file, cellview names also change automatically. The situations in which cellview names are automatically changed are described below.

- When the OpenAccess design consists of two cellviews having same names but present in different libraries. Similarly, two cellviews having same cell names but different view names. In such a case, cellview names are automatically changed because the Stream format does not support two structures having same names.

Example 2-18 When two libraries have same cell names

Consider a situation where a library `lib1` has a cell `cell1` and a view `view1`. Similarly, another library `lib2` has a cell `cell1` and a view `view1`. During XStream Out, `cell1` is automatically mapped and assigned a name because the Stream format does not support two cells having same names.

Example 2-19 When the library names and cell names are same but view names are different

Consider a situation where a library `lib1` has a cell `cell1` and a views `view1` and `view2`. During XStream Out, `cell1` is automatically mapped and assigned a new name because the Stream format does not support two cells with the same name.

- When an OpenAccess cell name consists of a character that is not supported in the Stream format. Valid characters supported by the Stream format are A-Z, a-z, 0-9, _, ?, and \$. An invalid character is replaced by underscore in the Stream format. When the *Respect GDSII Name Limit* option is `on` and a cellview name in OpenAccess is more than 32 characters.

Design Data Translators Reference

Design Translation Using XStream Translator

Sample Cell Name Map File

#Library Name	OpenAccess Cell Name	OpenAccess View Name	Stream Structure Name
reflib1	cell1	view1	STCELL1
*	cell2	view2	STCELL2
XST_TARGET_LIB	cell3	view3	STCELL3
XST_TARGET_LIB	cell4	view4	STCELL4

During XStream Out, the cell mapping information of the cells that are defined in the source library or any other library, and have been automatically mapped, is recorded in the `xStrmOut_cellMap.txt` file. All the cells that are present in the source library and are automatically mapped have library name as `XST_TARGET_LIB`. The `XST_TARGET_LIB` will be replaced with the library name specified using the `-lib` option. The `*` keyword denotes that this mapping is applied to the specified cellview present in any reference library.

Note: `XST_TARGET_LIB` keyword is applicable only for XStream In.

Note : In case `xStrmOut_cellMap.txt` is not writable, a new file (`xStrmOut_cellMap.userName.n.txt`) is generated automatically and the mapping information is added to this file. The name of the each new file changes incrementally. For example, `xStrmOut_cellMap.userName.1.txt`, `xStrmOut_cellMap.userName.2.txt`, and so on.

Via Map File

A via map file is a SKILL file that stores via mapping information during the XStream Out - XStream In round trip. A via represents a physical connection between two adjacent shapes that are on different layers. By default, during XStream Out vias are translated as SREFs in the Stream file and recreated as instances during subsequent Stream Ins.

To preserve and recreate vias in OpenAccess, XStream requires the `viaDef` names from the technology file and the via parameters. This information can be stored in the via map file during XStream Out. Via map file allows you to preserve via information during XStream Out - XStream In round trips.

During XStream Out if you use the `-viaMap` option then XStream creates a Stream file and a via map file. The via map file maps the Structure names from the Stream file to the corresponding `viaDef` names in the input technology library. This via map file is then used to recreate the original vias during subsequent Stream Ins.

Important

The viaMap file is meant to preserve the via mapping information on a XStream Out-XStream In round-trip only.

Also, when using Stream files from third-party vendors you may want some Structures to be treated as vias during XStream In. For this, you can create viaDefs in the technology library and manually create a via map file. You can create a via map file using any text editor.

Note: Processing vias is much faster as compared to processing instances. You may have designs containing millions of via-like structures. Therefore, using a via map file during XStream In ensures that the layout editing and processing tools work faster for designs created by XStream In.

In a via map file:

- Each mapping record must be a SKILL list.
- Any line starting with # is considered a comment and is ignored. Blank lines are also ignored.
- Each record must contain the following three fields:
 - sub master name
 - viaDef name
 - list of non-default parameter values
- If any record in the via map file is not in the proper format the record is ignored and a warning message is generated.
- For preserving vias ensure that the original technology file in which viaDefs are defined exists. If you do not specify the technology library during XStream In and the target library does not have a technology file associated with it, then the `-viaMap` option is not available during XStream In.
- Before starting the translation the via map file is loaded and for each entry in the via map file the corresponding viaDefs are searched in the technology file. If a viaDef does not exist in the technology file, then the record in the via map file is ignored and a warning message issued.
- In addition if you instantiate a via with the same parameter values multiple times in a design, then only one entry is created in the via map file during XStream Out. If the same sub master names are specified multiple times in the via map file, only the first record is considered during XStream In; all subsequent records are ignored and warning messages issued.

Design Data Translators Reference

Design Translation Using XStream Translator

You can specify the name of the via map file from the command line by using the `-viaMap` option during XStream Out, as shown below.

```
strmout [-viaMap viaMapFileName]
```

When you use the `-viaMap` option then a via map file is created along with the Stream file. If you do not specify the `-viaMap` option, then a via map file is not created and only the Stream file is created.

After creating the via map file you can specify the via map file during XStream In using the following command:

```
strmin [-viaMap viaMapFileName]
```

XStream In checks for the via map file and if it exists, then the mapping information from the via map file is used for preserving vias. If you do not specify the `-viaMap` option, then the vias are not preserved and are translated as instances.

Sample Via Map File

```
viaMapTable = list( nil
  subMasterTable (
    ( "VIAGEN12S_0" "VIAGEN12S" nil )
    ( "VIAGEN12S_1" "VIAGEN12S" (("CutHeight" 8000 "int")) )
    ( "VIAGEN12S_2" "VIAGEN12S" (("CutWidth" 9000 "int")) )
    ( "VIAGEN23_3" "VIAGEN23" (("CutHeight" 5000 "int")) )
    ( "VIAGEN12S_4" "VIAGEN12S" (("CutWidth" 8000 "int")) )
    ( "VIAGEN12S_5" "VIAGEN12S" (("CutHeight" 7000 "int")) )
    ( "VIAGEN23_6" "VIAGEN23" nil )
    ( "VIAGEN23_7" "VIAGEN23" (("CutHeight" 10000 "int")) )
    ( "VIAGEN12B_8" "VIAGEN12B" nil )
    ( "VIAGEN12B_9" "VIAGEN12B" (("CutWidth" 8000 "int")) )
    ( "via1Array_10" "via1Array" (("CutHeight" 1000 "int") ("CutWidth" 1000
"int")) )
    ( "via2Array_11" "via2Array" nil )
    ( "via2Array_12" "via2Array" (("CutWidth" 1000 "int")) )
    ( "via1Array_13" "via1Array" nil )
  ) ;End of subMasterTable
) ;End of viaMapTable
```

Object Map File

An object map file is an ASCII file that maps OpenAccess objects and sub-objects to Stream layer and data type. OA2.2 objects such as boundary, blockage, marker, and row do not represent physical shapes that can be represented in the Stream format. The objects are design concepts used by Place and Route and other display-editing tools. You may want to preserve these objects while archiving design data or while exchanging data between different tools to complete a design cycle.

Design Data Translators Reference

Design Translation Using XStream Translator

The object map file maps the OpenAccess object and sub-objects to the Stream layer-datatype pair and decides which OpenAccess object should be translated as a shape on which Stream layer-datatype pair. You can provide multiple object map files for mapping.

If you do not provide an object map file, XStream will automatically search for the `techlibname.objectmap` file in the technology library. An object map file, if present in the technology library with some other name, is not detected.

Note: In case, `ignoreObjectMapFromTech` is specified `techlibname.objectmap` file is ignored during XStream Out.

Note: With XStream Out and XStream In sharing the same object map file from technology, both processes pick the same file from the technology library for translation. XStream In supports only a subset of objects in the objectMap file, therefore, warning '142' is displayed during translation. You can use `-noWarn 142` during XStream In to avoid these warning messages.

In case of Incremental Technology Database (ITDB) hierarchy, XStream will traverse through all the object map files in the hierarchy.

You can create an object map file using any text editor. In an object map file:

- Any line starting with # is considered a comment and is ignored. Blank lines are also ignored.
- Each mapping record should be a single line.
- Fields should be separated with one or more spaces or tabs.
- Each record in the object map file must contain atleast four fields.
 - OA object name
 - OA object sub-type
 - [OA layer name]
 - Stream layer number (0- 255)
 - Stream datatype (0 -255)
- If the same object type and object sub-type pair is specified in multiple records, then only the first record is considered; all subsequent records are ignored and warning messages issued.
- If you specify both the object map file and the layer map file during XStream Out and XStream In, then ensure that the Stream layer-datatype pairs specified in the object map file are not used in the layer map file. If the Stream layer-datatype pairs in the object map

Design Data Translators Reference

Design Translation Using XStream Translator

file are the same as the Stream layer-datatype pairs in the layer map file then the entries in the object map file are ignored and warning messages issued. However, the object will be created if the GDSII/Stream file was generated by XStream streamOut.

- If the layer map file is not specified during XStream Out then the layers are automatically mapped and the layer-datatype pairs specified in the object map file are not used.
- If the map file contains entries with invalid object type and sub-type mapped to correct Stream layer-datatype pair then the records are ignored and error messages issued.
- The OpenAccess Layer Name field is only valid for oaLayerBlockage objects not for other object types.
- The OpenAccess Layer Name field is optional only for XStream Out. If this field is not specified during XStream Out, XStream Out writes all the layer blockages to the same Stream layer-datatype pairs.

Enhanced Object Mapping for XStream In and XStream Out

The object map functionality preserves the color information of the `layerblockages` in XStream Out-XStream In round trip and vice versa. In addition, the object mapping functionality sets the color information on the `layerblockages` in case of third-party GDS or GDS created using XStream Out without the `-enableColoring` or `-noObjectProp` options.

The following enhancements have been made in the object map file format:

```
<Object Type>    <Object SubType>    [Layer Name]    <Stream Layer
Number>          <Stream DataType>    [Photomask Color]
```

where:

- Columns 1,2,4, and 5 are mandatory.
- Columns 3 and 6 are only for the `LayerBlockage` Object Type. These are optional for XStream Out. However, column 3, `Layer Name` is mandatory for XStream In.
- Valid values for `Photomask Color` column are `mask1Color`, `mask2Color`, `mask3Color`, `mask4Color`, `mask5Color`, `mask6Color`, `mask7Color`, `mask8Color`, `blackColor`, and `multiColor`.
- The object type `Stitch` can be used to map existing stitch objects in XStream Out.

Enhanced Layer Blockages for XStream In

XStream In considers all the color mapping entries in the object map file, if the *Enable Coloring* check box is selected. Otherwise, all the color mapping entries will be ignored and a warning message will be displayed.

XStream In follows the following order to create the layer blockages on the specific layer and to set the color information.

1. The object map is considered for the layer name and photo mask color.
2. In case the GDS was generated by XStream Out, the shape record property is considered for the layer name and photo mask color.

XStream Out will consider all the color mapping entries in the object map file if the Coloring Mode is enabled. Otherwise, all the color mapping entries will be ignored and a warning message will be displayed.

XStream Out follows the below mentioned order to map the layer blockages on a specific layer with a specific photo mask color:

1. Entry with layer name and photo mask color
2. Entry with only photo mask color
3. Entry with only layer name
4. Entry with no layer name and no photo mask color

Using Stitch Objects

The object type `Stitch` can be used to map existing stitch objects in XStream Out. An example of the object map file syntax is described below:

```
<OA Object Name> <OA object sub-type> <OA Layer Name> <Stream  
Layernumber> <Stream Datatype>
```

```
Stitch           None           M1           32           5
```

In the above example, the stitch region associated with stitch objects on layer `M1` is mapped as a shape on the Stream layer number `32` and Stream data type `5`.

When the GDS file streamed in, the layer mapping file is used to map the stitch region back to a specified purpose. In the example below, the purpose is `stitch`.

```
<OA Layer Name> <OA Purpose Name> <Stream Layernumber> <Stream  
Datatype>
```

For more information, see [Handling Stitches](#) in *Virtuoso Multi-Patterning Technology User Guide*.

Object Map File - OpenAccess Object Property Data

You can specify the object file by using the `-objectMap` command-line option. Also, you can load the object map file using the *Load* button in the XStream Out Object Map File Editor form. If you do not specify the object file, then only shapes are translated into the Stream file and OpenAccess objects, such as blockages, boundaries, rows, and markers are not translated. Any data related to the OpenAccess objects are dropped during translation.

If you load an object map file using the `-objectMap` option during XStream Out, then shapes as well as OpenAccess objects are translated as shapes. Also, the property data is written to the Stream file at the property attribute 126 on shapes that correspond to the OpenAccess objects. This property information is used during XStream In to preserve the OpenAccess objects.

To translate OpenAccess objects as shapes, but not write the property information to the property attribute 126, select the *Do Not Output Properties* option in the *Objects Map* section. You can also use the `-noObjectProp` option along with the `-objectMap` option during XStream Out. If you use the `-noObjectProp` option, then although the shapes and OpenAccess objects are translated, no property data is written to the Stream file for OpenAccess objects.

If the design contains area halos and the `-objectMap` option and the `-noObjectProp` options are not specified, then to preserve area halos in the XStream Out - XStream In cycle, the `-propMap` option is required. If you do not specify the `-propMap` option during XStream Out, the following message is displayed.

Property map file is needed to preserve `areaHalo` blockages. If the Property map file is not specified, then you need to specify the file at the `-propMap` option.

If XStream `streamOut` is used to generate a GDSII using an object map file, then there is no need of using an object file during `streamIn`. The objects would be automatically preserved in `streamOut-streamIn` round-trip as necessary information (on PROPATTR 126) would be written in the GDSII files in `streamOut` itself.



Ensure that you specify the option `-noObjectProp` only when using the `-objectMap` option. You will get an error message if you use the `-noObjectProp` option without the `-objectMap` option.

XStream Out Object Types and Sub-types

The following object type and object sub types are valid only for XStream Out.

Translating Boundaries

The OpenAccess boundary object defines an enclosed area, including a polygon outline. In OpenAccess there are four types of boundaries– area boundary, PR boundary, snap boundary, and cluster boundary. A cluster boundary has a cluster associated with it and this cluster must be specified while creating cluster boundaries. During XStream Out, however, the cluster information is lost and upon a subsequent XStream In, an empty cluster is created for each cluster boundary.

The following table lists the valid boundary object type and sub-types.

Object Type	Object Sub-type
Boundary	oaAreaBoundary
	oaPRBoundary
	oaSnapBoundary
	Cluster

Translating Blockages

The OpenAccess blockage object represents those areas in a design where specific object types cannot be located.

Blockages can be:

- **oaAreaBlockage** - represents an area. Most commonly used to prevent standard cells from being placed within this area.
- **oaLayerBlockage** - represents an area on a specific layer. Most commonly used to map layer blockages on different layers to different Stream layers and datatype pairs.
- **oaAreaHalo** - represents the area around the master's prBoundary. Most commonly used to prevent standard cells from being placed within a specific range of an instance.

The following table lists the valid blockage object type and sub-types.

Object Type	Object Sub-type

Design Data Translators Reference

Design Translation Using XStream Translator

AreaBlockage	placement
LayerBlockage	routing
	viaRouting
	feedthru
	fill
	pin
	screen
	slot
	wiring
AreaHalo	placement

Note: If the number of coordinates in a LayerBlockage exceeds 8000, it is divided into smaller objects with the number of coordinates less than or equal to 8000. This is done to ensure a valid GDS file.

Important

The AreaBlockage object type can only have placement as the sub-type, while the LayerBlockage object can have all other sub-types, except for the placement sub-type.

Design Data Translators Reference

Design Translation Using XStream Translator

Translating Markers

The OpenAccess marker object indicates design violations and the objects that cause the violations.

The following table lists the Marker object type and sub-types that are translated and preserved.

Object Type	Object Sub-type
Marker	acknowledgedWarning
	annotation
	criticalError
	error
	fatalError
	info
	signedOffCriticalError
	signedOffError
	warning

Note: If the number of coordinates in a Marker exceeds 8000, it is divided into smaller objects with number of coordinates less than or equal to 8000. This is done to ensure a valid GDS file.

Translating Rows

An OpenAccess row object represents a potential location for placing standard cells, macros, or devices. For the row object type, the valid object sub-type is `none`.

Note: Custom and mixed rows are not translated and preserved in this release.

Sample XStream Out Object Map File

```
# ObjectType  ObjectSubType  StreamLayer  StreamPurpose
Boundary     Area           0           1
Boundary     Cluster          0           2
Boundary     PR              0           3
Boundary     Snap            0           4
```

Design Data Translators Reference

Design Translation Using XStream Translator

LayerBlockage	viaRouting	metall1	1	1
LayerBlockage	routing	metal2	1	2
AreaBlockage	placement	1	3	
LayerBlockage	wiring	metall1	1	4
LayerBlockage	fill	metal2	1	5
LayerBlockage	slot	metall1	1	6
LayerBlockage	slot	metal2	3	7
LayerBlockage	pin	metall1	1	7
LayerBlockage	feedthru	metal2	1	8
LayerBlockage	screen	metall1	1	9
Marker	annotation	2	1	
Marker	info	2	2	
Marker	acknowledgedWarning	2	3	
Marker	warning	2	4	
Marker	signedOffError	2	5	
Marker	error	2	6	
Marker	signedOffCriticalError	2	7	
Marker	criticalError	2	8	
Marker	fatalError	2	9	
Row	none	3	10	

Creating OpenAccess Objects from Third Party GDSII

If the GDSII is from third-party (no information on PROPATTR 126) or the GDSII file is created by XStream Out using the `-noObjectProp` option, then users need to use object map files to create OpenAccess objects from BOUNDARY or BOX records in the GDSII file. This functionality only works for the OpenAccess objects (layerBlockage, PR boundary, Snap boundary, and Marker).

The object map file maps the Stream layer-datatype pair to OpenAccess object and sub-objects and decides which shape should be translated as an OpenAccess object.

XStream In Object Types and Sub-types

The following object type and object sub types are valid only for XStream In.

Translating Boundaries

The OpenAccess boundary object defines an enclosed area, including a polygon outline. In OpenAccess there are two types of boundaries, PR boundary and snap boundary.

Design Data Translators Reference

Design Translation Using XStream Translator

Following table lists the valid boundary object type and sub-types.

Object Type	Object Sub-type
Boundary	oaPRBoundary
	oaSnapBoundary

Translating Blockages

The OpenAccess blockage object represents those areas in a design where specific object types cannot be located.

Blockage can be `oaLayerBlockage` representing an area on a specific layer. It is most commonly used to map layer blockages on different layers to different Stream layers and datatype pairs. Following table lists the valid blockage object type and sub-types.

Object Type	Object Sub-type
LayerBlockage	routing
	viaRouting
	feedthru
	fill
	pin
	screen
	slot
	wiring

Translating Markers

The OpenAccess marker object indicates design violations and the objects that cause the violations. Following table lists the marker object type and sub-types that are translated and preserved.

Object Type	Object Sub-type

Design Data Translators Reference

Design Translation Using XStream Translator

Marker	acknowledgedWarning
	annotation
	criticalError
	error
	fatalError
	info
	signedOffCriticalError
	signedOffError
	warning

Sample XStream In Object Map File

```
Boundary pr    1 1
Boundary snap  1 2
layerBlockage fill  metal1    2    1
layerBlockage routing metal2    2    2
layerBlockage slot   metal1    1    6
layerBlockage slot   metal2    3    7
Marker error   3 1
Marker info    3 2
```

Note: For `layerBlockage` the object map file must contain all five fields, namely, OpenAccess object name, OpenAccess object sub-type, OpenAccess layer name, Stream layer number, and Stream datatype. The OpenAccess layer name field is mandatory in XStream In.

OpenAccess PathSegs to XStream/GDSII Record Mapping

OpenAccess pathSeg objects can be either orthogonal (0, 90, 180, or 270 degrees) or diagonal (45, 135, 225, or 315 degrees). All the orthogonal pathSegs will be written as PATH records in the Stream file during streamOut unless *Convert Path to Polygon* option is used or path extension is *custom* or *chamfer*. However, the diagonal pathSegs will always be written as polygons (BOUNDARY) in the Stream file.

OpenAccess PathSegs	Extensions	Object Type
Orthogonal pathseg	'truncate', 'extend', or 'variable'	paths (PATH)

Design Data Translators Reference

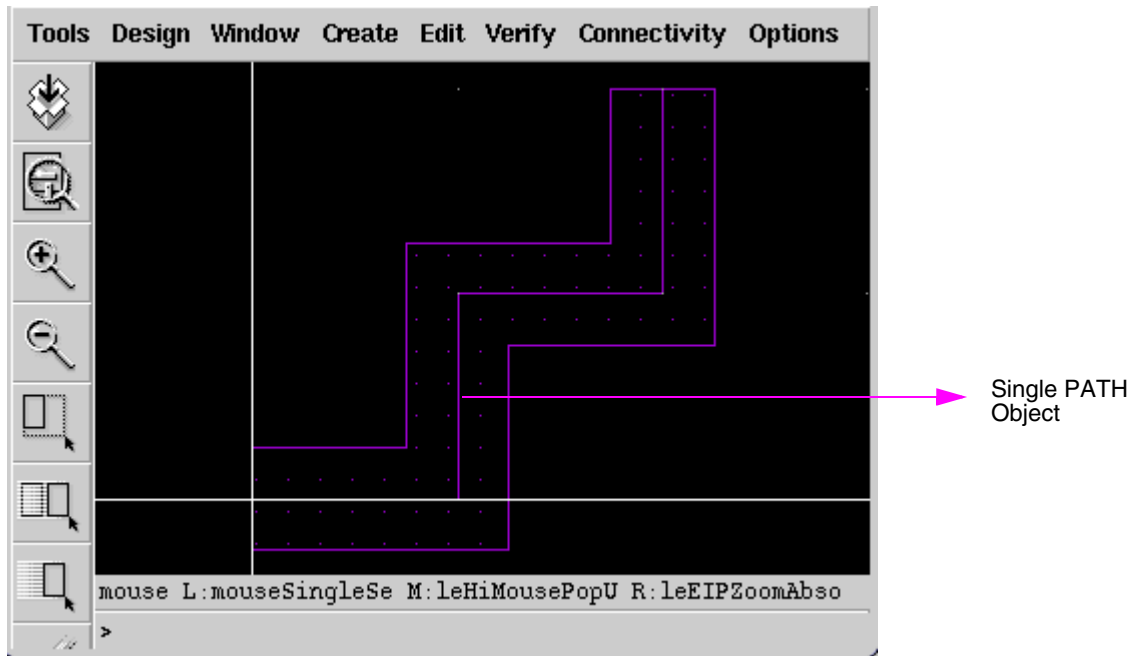
Design Translation Using XStream Translator

OpenAccess PathSegs	Extensions	Object Type
Orthogonal pathseg	'custom' or 'chamfer'	polygons (BOUNDARY)
Diagonal pathseg	always	polygons (BOUNDARY)

A diagonal pathSeg in the OpenAccess design will be written as a polygon (BOUNDARY) record in the Stream file during XStream Out. If the Stream file is imported back in OpenAccess using XStream In then the original diagonal pathSeg will become a polygon. The diagonal pathSeg will not be preserved during XStream Out/XStream In, only the geometry will be preserved.

In IC5141, a Path object is created in CDB database when a route segment is created. During XStream Out, this multi-point path is written as a single multi-point PATH record in the GDSII file.

Example 2-20 Single multi-point PATH Record



A multi-point path is written as a single multi-point PATH record in the GDSII file as:

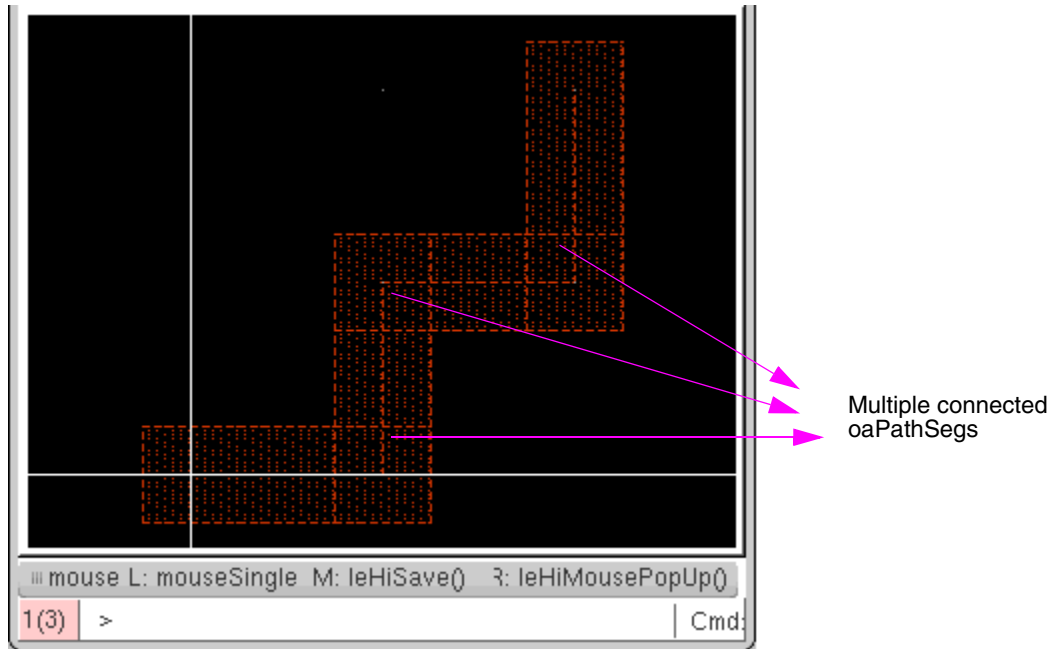
```
PATH LAYER=1 DATATYPE=0
PATHTYPE 0 WIDTH 1000 XY=(0,0)
(2000,0) (2000,2000) (4000,2000) (4000,4000) ENDEL
```

Design Data Translators Reference

Design Translation Using XStream Translator

In IC61, a collection of Path object are created in OpenAccess database when a route segment is created. During XStream Out, these multi-point path is written as a multiple 2-point PATH record in the GDSII file.

Example 2-21 Multiple 2-point PATH Record



A multi-point path is written as a multiple 2-point PATH record in the GDSII file as:

```
PATH LAYER=1 DATATYPE=0
PATHTYPE 2 WIDTH 1000 XY=(4000,2000) (4000,4000) ENDEL
PATH LAYER=1 DATATYPE=0
PATHTYPE 2 WIDTH 1000 XY=(2000,2000) (4000,2000) ENDEL
PATH LAYER=1 DATATYPE=0
PATHTYPE 2 WIDTH 1000 XY=(2000,0) (2000,2000) ENDEL
PATH LAYER=1 DATATYPE=0
PATHTYPE 2 WIDTH 1000 XY=(0,0) (2000,0) ENDEL
```

The `mergePathSegsToPath` option has been introduced to assemble all the connected `oaPathSegs` into a single `PATH` record during `streamOut`. This enables you to reduce the size of the GDSII. In addition, it will provide backward compatibility with IC5141.

Assumptions

- Only the `oaPathSegs`, which has the same width would be considered for merging.
- A `pathSeg` will not be considered for merging if:
 - It has the extensions, such as spike or gap on the edge.
 - The `beginStyle` or `endStyle` of an `oaPathSeg` is *custom* or *chamfer*
- End type of `PATH` would become variable if:
 - `BeginStyle` of the first `oaPathSeg` and `endStyle` of the last `pathSeg` are different. Only styles *truncate*, *extends*, and *variable* are considered.
- When property mapping is used, if there is a conflict between the properties of the `oaPathSegs` objects, then the first property value will be considered for writing in the GDSII and the second property value will be dropped with a warning.

This option must be used when required as it will have some performance overheads.

Preserving Connectivity Information

This section includes:

- Preserving Pin Connectivity Information
- Preserving Net Names

Preserving Pin Connectivity Information

The Stream format does not support pins and their connectivity information. To preserve pins in the round trip, use the *Pin Attribute Num* option.

During XStream Out, the value of the *Pin Attribute Num* option specifies the Stream attribute number that stores the pin information. If you specify a value for this option between 1-127, the following information related to pins is translated to the Stream file:

- Access direction

Possible directions are `Top (T)`, `Bottom (B)`, `Left (L)`, `Right (R)`. If a pin has more than one access direction, the access direction appears as any of the following

- ☐ `Top`
- ☐ `Bottom`
- ☐ `Left`
- ☐ `Right`

If the access direction is not specified, XStream Out writes a blank character to the Stream file during translation.

- Pin name
- Terminal name
- Terminal direction

Possible terminal directions are: `input`, `output`, `inputOutput`, `switch`, `jumper`, `unused`, and `tristate`.

- Instance name

If you specify 0 as the value of the *Pin Attribute Num* option, then pin information will not be translated to the Stream file.

During XStream In, the value of the *Pin Attribute Num* option specifies the Stream attribute number containing the pin information in the Stream file.

Example

If the pin name is 13, terminal name is VSS, the access direction is Bottom, and the terminal direction is inputOutput, then XStream Out will write B 13 VSS InputOutput as the value of the Stream attribute number in the *Pin Attribute Num* option. This value will then be assigned to the geometry corresponding to the pin in the Stream file.

Preserving Net Names

The Stream format does not support net names that are associated with various shapes. You can preserve net names to ensure proper design translation. During XStream Out, when you specify `netName` for an object in the property map file, the net name is saved in the Stream file as a value of the mapped attribute.

During XStream In, if any shape having a property on the attribute that is mapped to `netName`, is specified in the property map file, then the net name is verified in the database. If it exists, then it is associated with the shape. Otherwise, a net is created with the specified name and is associated with the shape.

Design Data Translators Reference

Design Translation Using XStream Translator

Sample Log File

```
*****
Product   : Virtuoso(R) XStream In
Program   : @(#) $CDS: strmin.exe version 5.0.2 11/17/2003 05:11 (cds125840) $
          : sub-version 5.02.41.60
Started at: 21-Nov-2003 10:42:52
User Name : ysharma
Host Name : silky
Directory : /hm/ysharma/XStream
CADENCE Design Systems, Inc.
*****

Info: Cellview Rev Num:59, Tech Rev Num:33

Summary of Options:
    library      mylib
    strmFile      /hm/ysharma/XStream/cds2.gds
    logFile       strmIn.log
    view          layout
    case          Preserve
    propSeparator,

1. Translating Cellview (v0001a_xr03d4 layout)
INFO (ID) :Creating a new layer L119 in the technology file.
INFO (ID) :Creating a new purposeP30 in the technology file.
INFO (ID) :Creating a new layer:purpose pair L119:P30 in the technology file.
...
2. Translating Cellview (v0001a_xr03d2 layout)
3. Translating Cellview (v0001a_xr03d1 layout)
...

Summary of Objects Translated:
    Instances:117198
    Arrays: 0
    Polygons:5423
    Paths: 0
    Rectangles:560782
    Text: 2573
    Node: 0
    Line: 0
    Cells: 165
Elapsed Time = 135.0s
```

Design Data Translators Reference

Design Translation Using XStream Translator

Sample Summary File

```
*****
Product      : Virtuoso(R) XStream Out
Program      : @(#)$CDS: strmount.exe version  5.0.2 11/17/2003 21:20
(cicser7s) $: sub-version vce3.3.112
Started at: 17-Nov-2003 21:24:18
User Name    : usisodia
Host Name    : cicser7s
Directory    : /servers/icistore/ test/translator/xstream/outFeatureTest1
CADENCE Design Systems, Inc.
*****

                        Individual Cell Statistics

CellView Rectangles Polygons Paths Ellipses Labels Instances Arrays Nodes
*****
lib/top/layout 0      0      0      0      0      4      0      0
lib/leaf1/layout 1     1      0      0      0      0      0      0
lib/leaf2/layout 1     2      1      0      0      0      0      0
lib2/top1/layout 0     0      0      0      0      4      0      0
lib/mosaic/layout 0    0      0      0      0      1      1      0

Statistics Of Layers

Cadence Cadence Stream Stream Rectangles Polygons Paths Ellipsis labels Nodes
layer   purpose layer datatype
-----
metall1 drawing 0      0      0      1      0      0      0      0
pin1    drawing2 1      0      0      1      0      0      0
poly1   drawing 2      0      1      0      1      0      0
poly2   drawing 3      0      1      1      0      0      0

Summary Of Stream Objects
-----
2 Rectangles
3 Polygons
1 Paths
0 Ellipses
0 Labels
9 Instances
1 Arrays
6 Cells
0 Nodes

Design Libraries:
DEFINE reflib2      reflib2
DEFINE cellMapLib2  cellMapLib2
```

Environment Variables in XStream

XStream supports following environment variables:

Design Data Translators Reference

Design Translation Using XStream Translator

- **XST_DEBUG** - this environment variable can be set to any value. Using this environment variable, XStream writes additional error or warning messages in the log file. These additional messages help you in debugging design information.
- **CDS_AUTO_CKIN** - XStream uses this environment variable to control the check in of design data to Design Management System (DM). For more information, see *Cadence Library Manager User Guide*.
- **CDS_AUTO_CKOUT** - XStream uses this environment variable to control the check out of design data from Design Management System (DM). For more information, see *Cadence Library Manager User Guide*.

Note: XStream does not support the environment variables CDS_PROMPT_CKOUT and CDS_PROMPT_CKIN.

- **CDS_EXP_PCELL_DIR** - this environment variable controls the directory to be used for caching the Pcells. You can specify either an absolute path or a path relative to your current working directory. Set this variable prior to launching the Virtuoso Studio design environment.
- **CDS_ENABLE_EXP_PCELL** - this environment variable controls the caching of evaluated SKILL Pcells. It can be set to true or false. The default value of this variable in a Virtuoso session is different from its default value while using a non-Virtuoso or third party application.

Design Data Translators Reference

Design Translation Using XStream Translator

Sample Template File

XStream Out Template File

```
*****
# Copyright (C) 2001 - 2018                                     *
# Cadence Design Systems Inc. All Rights Reserved.               *
#*****
#
#*****
#
# filename : streamOut.template
#
#*****

arrayInstToScalar          "false"          #- Convert Arrays to Scalar Instances
backupGdsLogFiles          "false"          #- Backup GDSII and Log Files
case                       "preserve"        #- preserve | upper | lower - Case Sensitivity
cellListFile               ""               #- The Cell List File Name
cellMap                    ""               #- Cell Map File Name
cellNamePrefix             ""               #- Prefix to be added to each STRUCT
cellNameSuffix             ""               #- Suffix to be added to each STRUCT
checkPolygon               "false"          #- Report Bad Polygons
#colorAnalysisGroup        ""               #- Name of the color analysis group
convertDot                 "node"           #- polygon | node | ignore - Convert Dots to
convertPcellPin            ""               #- geometry | text | geometryAndText | ignore - Convert Pcell Pin to
convertPin                 "geometry"        #- geometry | text | geometryAndText | ignore - Convert Pin to
dbuPerUU                   ""               #- DB Units Per User Units
diagonalPathToPolygon      "false"          #- Convert Paths with non-orthogonal segments to polygons
doNotPreservePcellPins    "false"          #- Do not Preserve PCell Pins
donutNumSides              "64"            #- Number of sides for the BOUNDARY of donut
ellipseNumSides            "64"            #- Number of sides for the BOUNDARY of ellipse
#enableColoring            "true"           #- Enable Coloring Support
flattenPcells              "false"          #- Convert PCells to Geometry
flattenPcellsExclusionFile ""               #- A file specifying list of Pcells to exclude from flattening
flattenVias                "false"          #- Convert Vias to Geometry
flattenViaShapesToViaPurp "false"          #- Flatten Vias' Shapes to Via Purpose
fontMap                    ""               #- Font Map File Name
gdsCellNameLength         ""               #- Truncate GDS Cell/STRUCT name length to specified value
hierDepth                  "32767"         #- (0-32767) - Hierarchy Depth Limit
ignoreCurrentTimestamp     "false"          #- Ignores current timestamp
ignoreLines                "false"          #- Ignores Lines During Translation
ignoreMissingCells         "false"          #- Ignores Missing cellViews During Translation and Continue
Translation
ignoreMissingVias          "false"          #- Ignores Missing standard/custom vias During Translation
and Continue Translation
ignoreObjectMapFromTech    "false"          #- Ignores Object Map file from Technology
ignorePcellEvalFail        "false"          #- Ignores the PCell Evaluation Failure
ignoreTopCellPrefixSuffix "false"          #- Ignores cellName Prefix and Suffix for topCell
ignoreZeroWidthPathSeg     "false"          #- Ignores zero width PathSeg during translation
infoToWarn                 ""               #- Treat the Specified Info Messages Ids as the Warnings
labelCase                  "preserve"       #- preserve | upper | lower - Label Case Sensitivity
labelDepth                 "1"              #- Hierarchical depth to add labels to
labelMap                   ""               #- Label Map File Name
layerMap                   ""               #- Layer Map File Name
library                    "LIB"            #- Library Name
logFile                    "strmOut.log"    #- Log File Name
#mapAllColorToLocked        "false"          #- Map all the Colored Shapes to Locked DataType
maxVertices                "200"           #- Maximum Vertices in Path/Polygon
```

Design Data Translators Reference

Design Translation Using XStream Translator

mergePathSegsToPath	"false"	#- Merge the Connected Pathsegs to Path
noConvertHalfWidthPath	"false"	#- Do not Convert The Half Width Path to Boundary
noInfo	""	#- Do not Print the Specified Info Messages Ids
noObjectProp	"false"	#- Do not Write Special OA Object Properties
noOutputTextDisplays	"false"	#- Ignores Text Display Objects
noOutputUnplacedInst	"false"	#- Ignores the Unplaced Instances
noWarn	""	#- Do not Print the Specified Warning Messages Ids
objectMap	""	#- Object Map File Name
outputDir	""	#- The Output Directory for Stream File
pathToPolygon	"false"	#- Convert Paths to Polygons
pinAttNum	""	#- (0-127) - Stream Attribute Number for Preserving Pins
propMap	""	#- Property Map File Name
propValueOnly	"false"	#- Output Property Values Only
rectToBox	"false"	#- Output Rectangle as BOX
refLibList	""	#- Reference Library List File Name
replaceBusBitChar	"false"	#- Replace "<>" With "[]"
reportPrecisionLoss	"false"	#- Report Precision Loss in Scaling
respectGDSIINameLimit	"false"	#- Respect GDSII limits
runDir	"."	#- Run Directory
scale	"1.00000"	#- Design Scaling Factor
snapToGrid	"false"	#- Snap XY to Grid Resolution
strmFile	"top.gds"	#- Stream File Name
strmTextNS	"cdba"	#- Text NameSpace
strmVersion	"5"	#- 3 5 6 - Stream Version
subMasterSeparator	"_CDNS_"	#- Separator to be used for sub-master naming
summaryFile	""	#- Summary File Name
techLib	"top"	#- Technology Library Name
topCell	""	#- Top Cell(s) to Translate
translateUnmappedLPP	"false"	#- Allow undefined LPP to be translated by incrementally
mapping LPP using layerMap file, SKILL hook poLayerMap,		and automatic layer mapping
userSkillFile	""	#- Users' Skill File Name
verbose	"false"	#- Generate detailed information in the log and summary files
viaCutArefThreshold	""	#- Write a STRUCT if the number of cuts are more the specified
viaMap	""	#- Via Map File Name
view	"layout"	#- View Name(s) to Translate
warnToErr	""	#- Treat the Specified Warning Messgaes Ids as the Errors
wildCardInCellMap	"false"	#- Wild Card in Cell Map

Note: The maximum length of a single line in the template file for topcell and other XStream Out options is 8192.

XStream In Template File

```

*****
# Copyright (C) 2001 - 2018 *
# Cadence Design Systems Inc. All Rights Reserved. *
*****
#
#*****
#
# filename : streamIn.template
#
#*****

arrayInstToScalar          "false"          #- Convert AREF to Scalar Instances
attachTechFileOfLib        ""                #- Attach Techfile of Library
case                       "preserve"         #- preserve | upper | lower - Case Sensitivity
cellMap                    ""                #- Cell Map File Name
checkPolygon               "false"           #- Report Bad Polygons

```

Design Data Translators Reference

Design Translation Using XStream Translator

compress	"false"	#- Allow libraries to be compressed
#compressLevel	"1"	#- Defines the compression level to use (default: 1)
convertPathToPathSeg	"false"	#- Convert Two Point Path(s) to PathSeg(s)
detectOverflow of shapes in a cellview	"false"	#- Enable overflow cellview creation on reaching maximum limit
dbuPerUU	""	#- DB Units Per User Units
#enableColoring	"false"	#- Enable Coloring Support
enableIncrementalImport	"false"	#- Enable incremental import of multiple Stream Files
enableLocking	"false"	#- Enable File Locking
enableNegativeViaExt	"false"	#- Enable Negative Extension for Via Detection
excludeMapToVia	""	#- Via exclusion Data file
fontMap	""	#- Font Map File Name
hierDepth	"32767"	#- (0-32767) - Hierarchy Depth Limit
ignoreBoxes	"false"	#- Ignore BOX records
ignoreObjectProp	"false"	#- Ignore Object Properties
ignoreZeroWidthPath	"false"	#- Ignore Zero Width Path(s)
infoToWarn	""	#- Treat the Specified Info Messgaes Ids as the Warnings
keepStreamCells	"false"	#- Keep Stream File Cells
labelCase	"preserve"	#- preserve upper lower - Label Case Sensitivity
layerMap	""	#- Layer Map File Name
library	"strmLib"	#- Library Name
loadTechFile	""	#- ASCII Technology File Name
logFile	"strmIn.log"	#- Log File Name
maxCellsInTargetLib	"30000"	#- Maximum Cells in Target Library
mergeUndefPurposToDrawing	"false"	#- Merges Undefined Purposes to "drawing" Purpose
noDetectVias	"false"	#- Do Not Detect possible viaDefs from Strem File and create vias
noInfo	""	#- Do not Print the Specified Info Messages Ids
noWarn	""	#- Do not Print the Specified Warning Messages Ids
objectMap	""	#- Object Mapping File Name
#overflowCellNameSuffix	"_xstOverflow"	#- Cell name suffix of overflow cellviews
#overflowViewName	"layout"	#- View name of overflow cellviews
#overwriteRootCells	"false"	#- Overwrite the top cells
pinAttNum	""	#- (0-127) - Stream Attribute Number To Preserve Pin
propMap	""	#- Property Map File Name
propSeparator	","	#- Property Separator Character
refLibList	""	#- Rerference Library List File
replaceBusBitChar	"false"	#- Replace "[]" With "<>"
reportPrecisionLoss	"false"	#- Report Precision Loss in Scaling
runDir	","	#- Run Directory
scale	"1.00000"	#- Design Scaling Factor
scaleTextHeight	"1.00000"	#- Text Height Scaling Value (+ve value only)
showCompletionMsgBox	"true"	#- Show Completion Message Box
skipUndefinedLPP	"false"	#- Skip Creating Undefined Layer-Purpose Pair in Technology
snapToGrid	"false"	#- Snap XY to Grid Resolution
strmFile	"test.gds"	#- Stream File Name
#strmTechGen	""	#- Generated the LPP information of specified stream file
strmTextNS	"cdba"	#- Text NameSpace
summaryFile	""	#- Summary File Name
techRefs	""	#- Ordered List of Technology Libraries
topCell	""	#- Top Cell(s) to Translate
translateNode	"false"	#- Convert Node(s) to Dot(s)
translateUnmappedLPP LPP using layerMap file, SKILL hook piLayerMap, and automatic layer mapping	"false"	#- Allow undefined LPP to be translated by incrementally mapping
userSkillFile	""	#- Users' Skill File Name
verbose	"false"	#- Generate detailed information in the log and summary files
viaMap	""	#- Via Map File Name
view	"layout"	#- View Name(s) to Translate
virtualMemory	"false"	#- Import to Virtual Memory
warnToErr	""	#- Treat the Specified Warning Messgaes Ids as the Errors
wildCardInCellMap	"false"	#- Wild Card in Cell Map
writeMode	"overwrite"	#- Write Mode for Existing Cells



During XStream In and XStream Out both `strmFile` and `library` fields are mandatory. However, null strings (“”) can be allowed while loading the template file.

Library Cell(s) Limit

Since the OpenAccess library structure complies to 5.x architecture, the total number of cells in a library can go only up to the maximum number of sub-directories that are supported by the UNIX filesystem(32,767). This limits XStream to support only those Stream files that have less than 32,767 cells, which are to be translated in a library.

To overcome this limit, XStream has been enhanced to keep a count of number of cells in the target library. Whenever the number of cells in the target library become 30,000 and then, if XStream needs to translate more cells, a new target library *TargetLib_#* (where # can be 1,2, or 3) is created.

If a new target library already exists, then, XStream creates another library instead of overwriting the existing one. For example, if a target library *TargetLib_1* already exists, then XStream creates *TargetLib_2*. The technology library for the new target library is attached to the technology library of the original target library.

Performance Guidelines for XStream

XStream In and XStream Out provide many options and features to make its use model extremely flexible in accordance with the increasing requirements of the customers.

The following section provides information on the known speed impact of some of the XStream In and XStream Out options, and some guidelines on extracting maximum performance by avoiding those options that reduce the speed of translation in specific customer designs.

Guidelines for XStream In

- In order to optimize the time consumed in searching for reference libraries during XStream In, use the methods described below:

- ☐ Reduce the `cds.lib` file size

The `cds.lib` file should consist of a minimal list of libraries for XStream In to perform faster. The minimum number of libraries listed in `cds.lib` should be the target and reference libraries. The cells of these libraries should be instantiated in

Design Data Translators Reference

Design Translation Using XStream Translator

the Stream file. If there are a large number of libraries mentioned in the `cds.lib` file, XStream In spends extra time in searching for different cellviews in those libraries and thus takes more time. Another way to resolve this issue without modifying the `cds.lib` file is by using the *Reference Library List* option.

- ❑ Use the *Reference Library List* option

Suppose you are having a number of reference libraries in `cds.lib` and a design or a sub design/module is using a small fraction of the libraries that are being used. In another case, the libraries defined in `cds.lib` may not get ordered in an optimal fashion to minimize the search for a referenced cell during XStream In. In such a case, use the *Reference Library List* option.

Example:

Consider a situation where you are using instances from five different libraries: A, B, C, D, and E. You know that the usage density (the maximum probability of finding the referenced cell from a library) is from C and E. In such a case, in order to speed up the XStream In process, first create a file, say `refLib1.list`. Add library names to this file in the sequence:

```
C
E
A
B
D
```

Next, use the `refLibList` option as shown below:

```
-refLibList refLib1
```

If you provide a proper reference library name for a structure (cellview) in the cell name map file, then XStream does not search for that cellview in the list of libraries. Therefore, a cell name map file can also provide speed advantage when you use the *Reference Library List* option.

- ❑ If the value of the *Convert Arrays to Scalar Instances* option is chosen as ON (a non-default value), it will explode each AREF record in a Stream file. This will be done by translating an AREF record to the corresponding individual instances in the OpenAccess database. In some specific cases, this can reduce the translation speed and increase the output database size.

- Ensure that there is enough virtual memory in your system. The virtual memory requirements for XStream might vary from 0.6 to 1.3 times the biggest struct size of the Stream file.

XStream In has an internal algorithm, which estimates the virtual memory requirements before starting the translation process. It gives appropriate warnings and errors in the

Design Data Translators Reference

Design Translation Using XStream Translator

beginning if the available memory on your system is either less than or close to its estimated requirement. This feature can save time as you can update the swap space on your system in the beginning of translation itself.

Note: This functionality is not available in the VCE 3.2 release.

- Use the *Disable file locking* option to disable file locking during XStream In. By default, file locking will not be disabled. The disabling of file locking reduces the run time of XStream In especially if the designs contain a large number of cellviews in their hierarchy or library.

Default Value: Deselected

For more information, see [Enable File Locking](#).

Guidelines for XStream Out

- Avoid the *Flatten Pcells* option. This option increases the XStream Out translation time because it has to flatten all Pcell instances in the design.
- When the *Convert Arrays to Scalar Instances* option is enabled, all arrays explode in OpenAccess. This is because each element of the array is translated to a corresponding SREF record in the Stream file. This significantly slows down the translation and increases the size of the output Stream file.
- For specific purposes, you can use a limited set of layers to be translated. This can be done by configuring the layer mapping table accordingly. Using the limited set of layers, you can speed up translation because the amount of data to be translated is reduced.
- Avoid using the *Output Paths as Polygons* option. This option converts each OpenAccess path to a polygon (boundary record in Stream format) during XStream Out. If there are large number of paths in a design, this option can reduce the speed of translation due to the additional overhead of converting each path to a polygon internally.

Note: XStream progress bar is not linear because its increment is based on the number of structures in the Stream file that is the number of cellviews present in the design. As the cellviews might vary in sizes, progress bar might seem to increase at variable speed. For example, if the last few cellviews being translated are of big sizes as compared to the other cellviews in the design, it might take a longer time to finish even after hitting 99% in some cases.

Performance Improvement in XStream Out Using Express Pcells

- While translating Pcell instances in designs, streamOut evaluates Pcell SKILL code for writing submasters in the GDSII/Stream file. The evaluation of Pcell SKILL code impacts the performance of streamOut in the case of designs with large number of complex Pcell instances. This is because for the evaluation of Pcell SKILL code, a design is opened each time. However, when *Express Pcells* feature is enabled while exporting GDSII/Stream, the streamOut performance is improved. On some of the benchmark designs the performance gain is even greater than 10X.

To manage the Pcell cache, you can use the UNIX environment variables, CDS_EXP_PCELL_DIR and CDS_ENABLE_EXP_PCELL. By default, these variables are set to false in the Virtuoso environment. These variables are read only once, during the Virtuoso session startup. If you want to use the Express Pcells feature and cache your Pcell submasters for better performance and interoperability, set this variable to true before starting a Virtuoso session.

- The *Express Pcell* feature, available in the Virtuoso environment, maintains a cache of the pre-evaluated pcells on the disk. Because this feature is based on a caching

Design Data Translators Reference

Design Translation Using XStream Translator

mechanism, Pcell code is not evaluated each time a design is opened. This offers an enormous performance in case of designs containing a large number of Pcell instances or even in cases of designs containing moderate number of instances of complex Pcells that involve computationally intensive SKILL code.

Chip Finishing Flow

Before the IC6.1.5 release, as part of the chip finishing flow, OpenAccess designs imported from the GDSII/Stream file were also stored on disk. These OpenAccess designs were read from the disk while they were edited in the Virtuoso layout editor, and the modified design after being exported to a GDSII/Stream file was saved on the disk. Because OpenAccess designs—especially the custom and mixed signals designs—can contain thousands of cellviews, reading and writing the designs from or to the disk, which is determined by the disk access speed, was a time consuming process.

In the IC6.1.5 release, to improve the speed at which OpenAccess designs are accessed, the following features have been introduced:

- Virtuoso XStream In translator reads a GDSII file from the disk and creates an OpenAccess design in the virtual memory.
- You can use the Virtuoso layout editor to view or edit an OpenAccess design stored in the virtual memory.

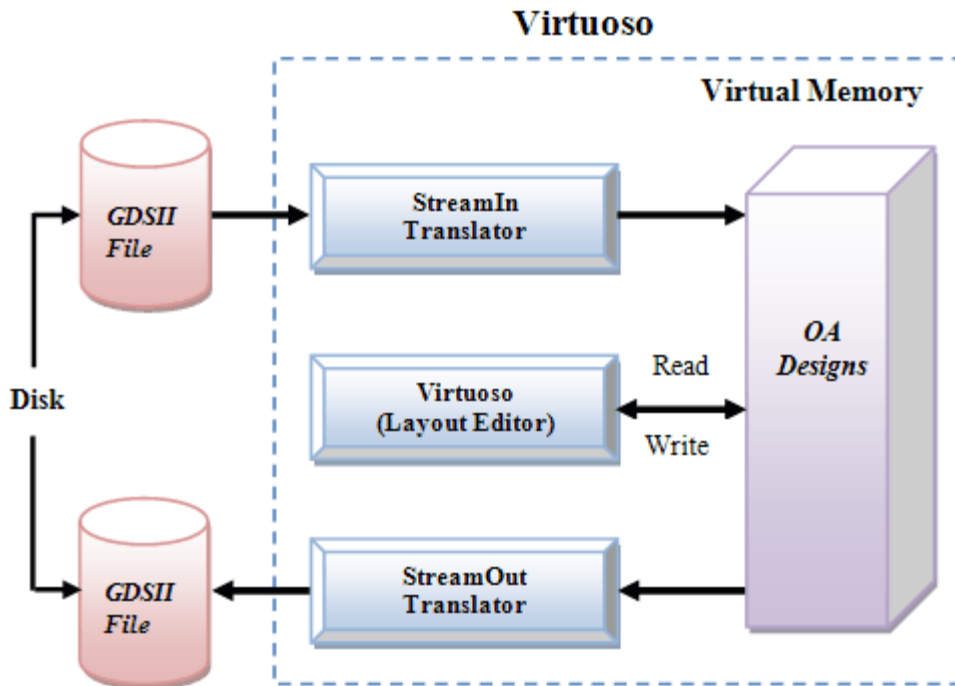
Note: You can also use DFII SKILL functions to access the OpenAccess design stored in the virtual memory.

- You can use the XStream Out translator to export an OpenAccess design from virtual memory to the GDSII/Stream file on the disk.

The figure below illustrates how XStream In and XStream Out work in the IC6.1.5 release.

Design Data Translators Reference

Design Translation Using XStream Translator



Note: You do not need a special license to use XStream In with virtual memory. XStream In uses the same license that is used by the Virtuoso layout editor.

To use XStream In with virtual memory, you need to select the Import to Virtual Memory check box in the XStream In form. Similarly, if you want to use XStream Out with virtual memory, you need to select the Stream Out from Virtual Memory check box.

Note: Run this flow in a new library and not in an existing library. This means that instead of running the chip-finishing flow in a pre-created library, you should allow XStream In create the new library.

Important

XStream In GDSII to virtual memory requires space for storing temporary data, though the OpenAccess designs are not saved on disk.

In Linux, the temporary data is stored in `/dev/shm`. If the available space in `/dev/shm` is less than 5 MB, temporary data is stored in `/tmp`. Moreover, if the available space in `/tmp` is less than 5 MB, the temporary data is stored in `.` / (the run directory).

Design Data Translators Reference

Design Translation Using XStream Translator

In Solaris and IBM, the temporary data is stored in `/tmp`. If the available space in `/tmp` is less than 5 MB, temporary data is stored in `.` / (the run directory).

The temporary data is deleted when the Virtuoso session is exited.

By default, information (INFO) and warning (WARNING) messages are not displayed during XStream In to virtual memory.

Note: If XStream In is done in virtual memory mode, the virtual library created by XStream In can be saved to the disk using the function, `xstInSaveVMLib`. You can also retrieve the list of virtual libraries created by XStream In using the function, `xstInGetVMLibs`.

For additional information about improving performance or space utilization during translation, refer to [GZip Support](#).



Video

For a video on Chip Finishing Flow, see [Introducing The Chip Finishing Flow](#) on Cadence Online Support.

For information about XStream SKILL functions, see the [*Design Data Translator's SKILL Reference*](#).

Design Data Translators Reference

Design Translation Using XStream Translator

Design Translation Using XOasis Translator

This chapter contains the following sections:

- [Overview](#)
- [Running the XOasis Translator](#)
- [Design Translation During XOasis In](#)
 - ❑ [XOasis In Use Model](#)
 - ❑ [Multithreading in XOasis In](#)
 - ❑ [XOasis In Form](#)
 - ❑ [XOasis In More Options Form](#)
 - ❑ [XOasis In Editor Forms](#)
 - ❑ [XOasis In Option Names in GUI and Template File](#)
 - ❑ [Seeding the XOasis In GUI](#)
 - ❑ [OASIS to OpenAccess Mapping](#)
 - ❑ [Mapping OASIS Records to OpenAccess Objects](#)
- [Design Translation During XOasis Out](#)
 - ❑ [XOasis Out Use Model](#)
 - ❑ [XOasis Out Form](#)
 - ❑ [XOasis Out More Options Form](#)
 - ❑ [XOasis Out Editor Forms](#)
 - ❑ [XOasis Out Names in GUI and Template File](#)
 - ❑ [Seeding the XOasis Out GUI](#)

Design Data Translators Reference

Design Translation Using XOasis Translator

- ❑ [OpenAccess to OASIS Mapping](#)
- ❑ [Unsupported Object Types](#)
- [Layer Mapping in XOasis](#)
- [Automatic Layer Mapping in XOasis](#)
- [Using Map Files During Translation](#)
- [Sample Log File](#)
- [Sample Summary File](#)
- [Sample Template File](#)
- [Performance Guidelines for XOasis](#)
- [Chip Finishing Flow](#)

Overview

OASIS format defines an encapsulation and interchange format for hierarchical integrated circuit mask layout information. This format also provides specifications to interchange mask data between systems such as EDA software, mask writing tools, and mask inspection repair tools. The OASIS format:

- Supports two-dimensional figure arrays
- Supports primitive shapes like vertical and horizontal trapezoid, square, and circle
- Uses variable-width integers (N bytes)
- Allows relative (Dx,Dy) coordinates in both step/repeat and figures
- Supports variable string length, including cell names
- Supports N-vertexes polygon
- Uses IEEE standard 64-bit floating point numbers
- Supports layer name and purpose information
- Supports embedded comments
- Supports embedded data compression

It consists of two modules, XOasis In and XOasis Out. XOasis In translates designs in the OASIS format to the OpenAccess (OA) database. XOasis Out translates designs from the OpenAccess database to the OASIS format.

Important

In this first production release of the OASIS format, we recommend that you verify your results when taping out a design. This can be achieved by translating the OA design database into OASIS versus GDSII, and then comparing these results by running a Boolean XOR with a physical verification tool, such as Physical Verification System (PVS).

This chapter provides information about:

- Running the XOasis translator
- XOasis In options
- XOasis Out options
- Mapping files.

Design Data Translators Reference

Design Translation Using XOasis Translator

- Sample log file, summary file, and the template file.
- Preserving OpenAccess objects during a cycle of XOasis Out and XOasis In.

Note: XOasis translators are based on OASIS SEMI P039-0308 standards. Contact SEMI for more information on OASIS standards.

Running the XOasis Translator

You can translate design files by running the XOasis translator either from the Textual User Interface (TUI) or from the Graphical User Interface (GUI). In this section, you will learn about the methods for running the XOasis translator by using TUI and GUI. Additionally, you will also read about the method for invoking help and setting the environment variables.

Running XOasis In and XOasis Out Using TUI

You can run XOasis In and XOasis Out in the batch mode by using the `oasisin` and `oasisout` commands. Use any of the two methods described below to run these commands.

- Specify the options as command-line arguments.

```
$oasisin -option1 value1 -option2 value2 -option3 value3
$oasisout -option1 value1 -option2 value2 -option3 value3
```

where:

<i>option1</i>	First option
<i>value1</i>	Value of the first option
<i>option2</i>	Second option
<i>value2</i>	Value of the second option
<i>option3</i>	Third Option
<i>value3</i>	Value of the third option

Example 3-1 Running XOasis In by specifying the `library`, `oasisFile`, and `logFile` options as command-line arguments.

```
$oasisin -library lib1 -oasisFile in.oasis -logFile oasisin.log
```

- Write the options and their values in a text file and then specify the text file as an argument to `oasisin` and `oasisout`.

```
$oasisin -templateFile textfileName
$oasisout -templateFile textfileName
```

where:

<i>-templateFile</i>	Name of the template file option
<i>textfileName</i>	Name of the template file containing option values

To view the template file, see [Sample Template File](#).

Example 3-2 Running XOasis Out by writing options and their values in the text file template.txt.

```
$oasisout -templateFile template.txt
```

The contents of text file appear in the format shown below:

```
option1      value1
option2      value2
```

For Boolean options, do not specify any value. When you specify a boolean option, its value is accepted as `true`. Otherwise, its value is `false`.

You can combine these two methods by specifying some options on the command line and other options in the template file. This method is a combination of the above two methods.

If you specify the same option in the template file and the command line, the option specified in the command line takes precedence.

```
$oasisin -option1 value1 -option2 value2 -templateFile textfileName
$oasisout -option1 value1 -option2 value2 -templateFile textfileName
```

where:

<i>option1</i>	First option
<i>value1</i>	Value of the first option
<i>option2</i>	Second option
<i>value2</i>	Value of the second option
<i>templateFile</i>	Name of the template file
<i>textfileName</i>	Name of the text file

Example 3-3 Running XOasis In by specifying library, oasisFile, and templateFile as command-line arguments.

```
$oasisin -library lib1 -oasisFile in.oasis -templateFile template.txt
```

In the above example, the `oasisFile` and `libName` options are taken from the command line and all other options are taken from `template.txt`.



You must specify a library name and an output filename during XOasis Out. If you do not specify these names, an error message is generated. Similarly, during XOasis In, you must specify the input OASIS filename and a library name, otherwise, an error message is generated.

Using Help

You can use the `-h` or `-help` option to display the description of the supported XOasis In or XOasis Out options. This information is displayed even when you type `oasisin` or `oasisout` without specifying any option.

Example 3-4 Using help during XOasis In and XOasis Out

```
oasisin [-h|-help]
oasisout [-h|-help]
```

Running XOasis In and XOasis Out from the GUI

You can run XOasis In by choosing the *File–Import–OASIS* command from the CIW. Similarly, you can run XOasis Out by choosing the *File–Export–OASIS* command from the CIW.

Note: `LD_LIBRARY_PATH` is now set automatically when you invoke XOasis from the command line or from GUI.



For a video on XOasis GUI, see [Introducing the New XOasis GUI](#) on Cadence Online Support.

Preserving GUI History in XOasis

The `-guihistory` command lets you use the last preserved GUI history at command line. These values can be overwritten at command line, if needed.

GUI history is saved only when the SKILL environment variable `xstEnableGUIHistory` is set to `t`. By default, the environment variable is set to `nil`, which means that the command is disabled.

Design Data Translators Reference

Design Translation Using XOasis Translator

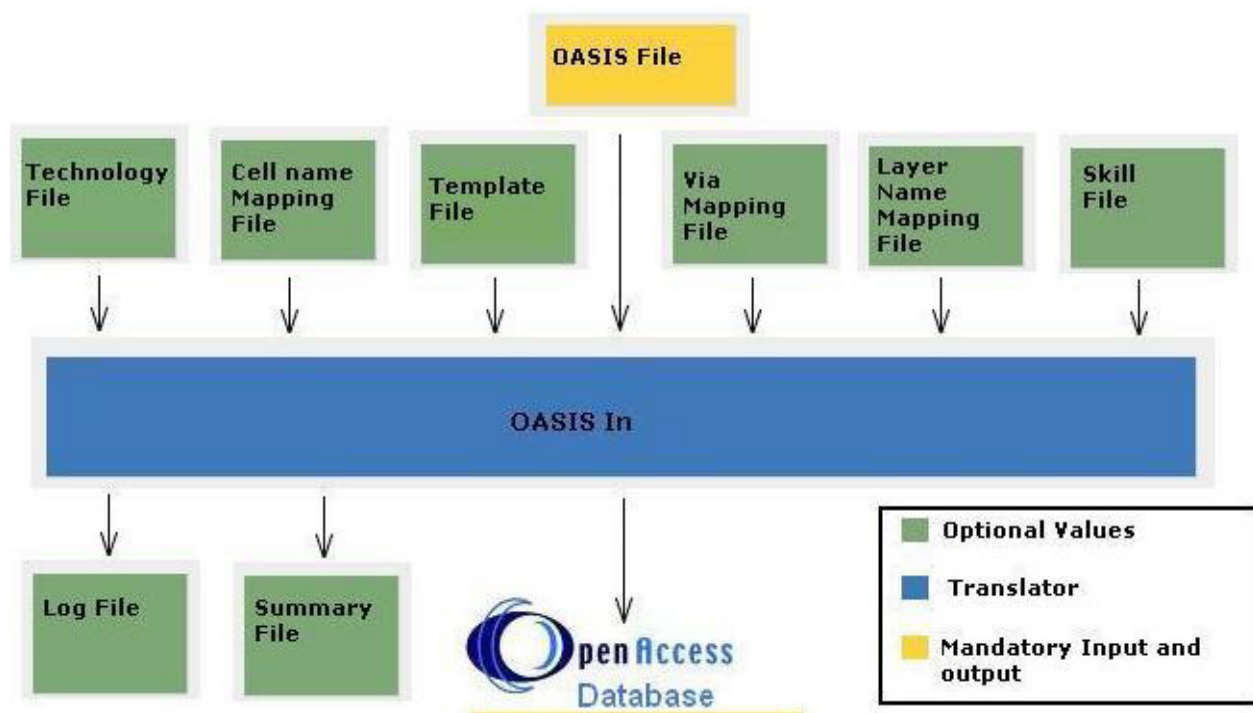
GUI history values are used across Virtuoso sessions. Therefore, when `xoasEnableGUIHistory` is enabled, the XOasis form also reads these values.

These values are also given the highest precedence. Therefore, if both pre-seeding and GUI history values are available, the XOasis form will use the values from the GUI history.

Design Translation During XOasis In

XOasis In Use Model

XOasis In converts design data in the OASIS format to designs in the OpenAccess database. The inputs to the XOasis In process are XOasis In options, optional mapping files, user-defined OASIS files, and technology file information. The log and summary files and the OpenAccess libraries are generated after the design translation.



The following table provides generic information about the files used during XOasis In.

File Type	Description
OASIS file	An OASIS formatted file that is to be translated to OpenAccess.

Design Data Translators Reference

Design Translation Using XOasis Translator

File Type	Description
Template file	A file that contains the option settings you want to use during XOasis In. In the XOasis In template file, '\$' is considered as a special character because it is a UNIX environment variable. However, if you want '\$' to be treated as a normal character, you can escape it by using '\' before it.
Technology file	A file used to compile the OpenAccess technology file for the OpenAccess database during XOasis In.
SKILL file	An optional file used for mapping cell names and layer names.
OpenAccess database	The output database
Log file	A file used for recording the translation process. It also keeps a log of all the messages.
Summary file	A file that contains a summary of the translated design.

For details on the map files, see [Design Translation During XOasis In](#).

Multithreading in XOasis In

XOasis In uses multithreading to improve performance during translation. For multithreading, a tool can use as many threads as there are available computing units on the system.

Note: One computing unit = One CPU core or hyper-thread

XOasis evaluates the machine configuration and then applies the best value to be used for multithreading based on available CPU cores.

Multithreading works differently when:

- [Import OASIS to Virtual Memory Mode is Enabled](#)
- [Import OASIS to Virtual Memory Mode is Disabled or XOasis is Run from Command Line](#)

Import OASIS to Virtual Memory Mode is Enabled

When [Import OASIS to Virtual Memory](#) is enabled, translation occurs in virtual memory. In this mode, by default XOasis one thread for translation.

Design Data Translators Reference

Design Translation Using XOasis Translator

You can also manually set number of cores to be used for multithreading with the XOASIS_IN_VM_NT environment variable. For example, the following sets number of cores to be used for multithreading in VM mode to 4 .

```
setenv XOASIS_IN_VM_NT 4
```

If you set the value as 0 or 1, it implies no cores will be used and multithreading will not take place.

Import OASIS to Virtual Memory Mode is Disabled or XOasis is Run from Command Line

When Import OASIS to Virtual Memory is disabled or XOasis is run from command line, a maximum of two CPU cores are used for translation.

Fast Oasis Out

You can use the *Export Oasis From VM* option located in the *File* menu of Layout Editor to export the currently open design in the layout window. When you choose this option, translation starts immediately in Virtual Memory mode and the XOasis Out GUI is not launched.

This process considers the following arguments during translation from the current layout window:

- The current library (`-library`).
- The current cell (`-topCell`).
- The current view (`-view`).
- The Oasis filename (`-oasisFile`), which is `<topCell.gds>` by default. The environment variables `oasisFilenamePrefix` and `oasisFilenameSuffix` control the prefix and the suffix of the GDS filename, respectively. If the GDS file already exists, it will be suffixed with `_1`, `_2` during export.
- The `layerMap` and `objectMap` files, if either one or both files are present in the technology, or if the self-technology library contains these files.
- The `enableColoring` SKILL environment variable controls the value for the Coloring Mode (`enableColoring`) option to be used during XOasis Out translation.

After the translation is complete, a message will be displayed, informing you whether the translation has completed successfully or not. A detailed log is saved in the `oasisOut.log` file.

XOasis In Form

The screenshot shows the XOasis In form interface. A red box highlights the 'OASIS File (required)' and 'Library (required)' fields, with a red arrow pointing to them from the text 'Mandatory Options'. Below these are fields for 'Top Cell', 'View' (set to 'layout'), and 'Template File'. A checkbox for 'Import to Virtual Memory' is also present. A red arrow points from the text 'Expand to view additional options' to the 'Layer Map' and 'Object Map' expandable sections. The 'Layer Map' section is expanded, showing a 'File Name' field and two checkboxes: 'Translate Unmapped LPP' and 'Use Automatic Layer Mapping'. At the bottom are buttons for 'Translate', 'Apply', 'Cancel', 'Reset All Fields', 'More Options', and 'Help'.

The main XOasis In form comprises mandatory options that are essential to run the translator including OASIS File, Library, Top Cell, View, and Template File. The form also comprises a few other most commonly used options, such as Layer Map and Object Map.

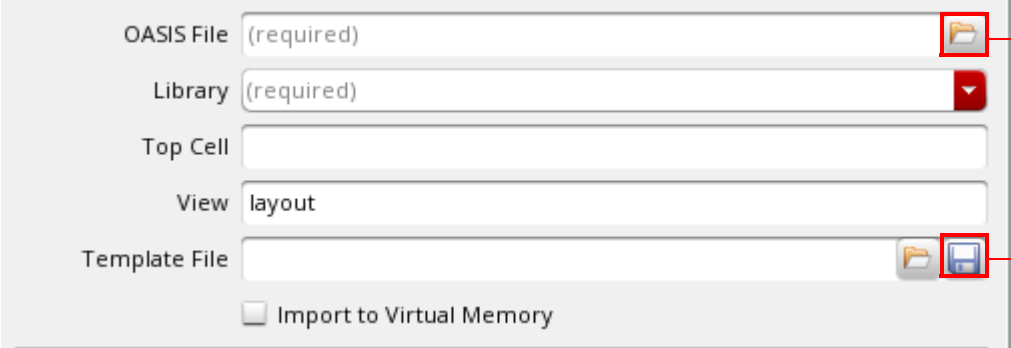
The following sections on the main form can be expanded to view additional options:

- Technology
- Layer Map
- Object Map
- Log File

Design Data Translators Reference

Design Translation Using XOasis Translator

Discussed below are the options available on the XOasis In form.



The screenshot shows the XOasis In form with the following fields and buttons:

- OASIS File**: A text field with "(required)" placeholder. A red box highlights the **File Browser** button (a folder icon) to its right. A red arrow points from the text "File Browser button" to this button.
- Library**: A text field with "(required)" placeholder. A red box highlights the drop-down arrow to its right.
- Top Cell**: A text field.
- View**: A text field with "layout" as the current value.
- Template File**: A text field. A red box highlights the **Save** button (a floppy disk icon) to its right. A red arrow points from the text "Save button" to this button.
- ☐ **Import to Virtual Memory**: A checkbox at the bottom.

OASIS File

Use the *OASIS File* field to specify the name of the OASIS file that is to be translated to OpenAccess. Alternatively, click the *File Browser* button to select the OASIS file. This field supports filenames that are relative to the run directory or are absolute.

You can either specify the absolute or relative path for the OASIS file. If the absolute path is specified, the OASIS file is picked up from the specified directory irrespective of the value specified for the run directory. However, in case of the relative path, the *OASIS* file is searched in the specified path relative to the run directory. By default, the run directory is the current directory.



Select an OASIS file before running XOasis In.

Command-line option: `oasisFile`

Library

Use the *Library* field to specify the target library where data is to be translated. The library name should be a logical name and follow the naming conventions supported by OpenAccess. You can also select an existing library as the target library from the drop-down list.

If the library does not exist, a new library is created in the run directory. If the target library already exists, data is translated to the library.



Select a destination library before running XOasis In.

Command-line option: `library`

Top Cell

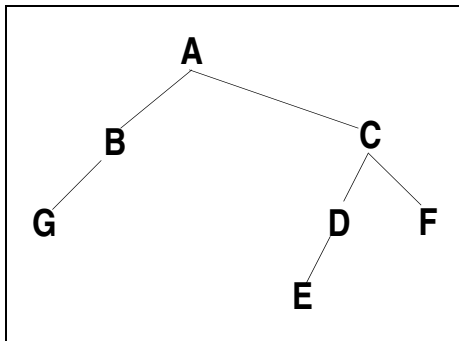
Use the *Top Cell* field to specify the name of the cell at the top of the hierarchy that you want to translate to OpenAccess. The OASIS format supports hierarchical designs. So, the *Top Cell* option determines the node from where the design is to be translated.

Default value: ""

Command-line option: `topCell`

Example 3-5 Translating cells by using the Top Cell option

Consider the hierarchical structure below where A, B, C, D, E, F, and G are cell names. You want to translate these cells using the *Top Level Cell* field.

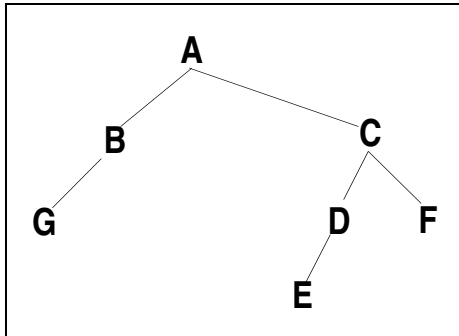


When C is specified as the top cell in the *Top Level Cell* field, cells C, D, E, and F are translated.

View

Use the *View* field to specify view names that are created in the target library during translation. If you specify a cellview name in the cell name map file or SKILL file, then the view name specified in these files takes precedence over the name specified in the *XOasis In* form.

Example 3-6 Translating cells C, D, E, and F into the layout1 view



In the figure above, if C is specified as the top cell and `layout1` as the view name, then the structures C, D, E, and F are translated with `layout1` as the view name.

Default value: `view`

Template File

Use the *Template File* field to specify an existing template file. You can also click the *File Browser* button to select the template file. When this file is loaded, the XOasis In form is updated with the option settings specified in it. If the correct template file is loaded, then all the GUI fields in the form are reset.

Click the *Save* button to save the current option settings to the template file.

Command-line option: `templateFile`

Import OASIS to Virtual Memory

Select the *Import OASIS to Virtual Memory* check box to read an OASIS file and create an OpenAccess design in the virtual memory. You can also specify the number of CPU cores that should be used for multithreading during translation. For more information, see [Multithreading in XOasis In](#).

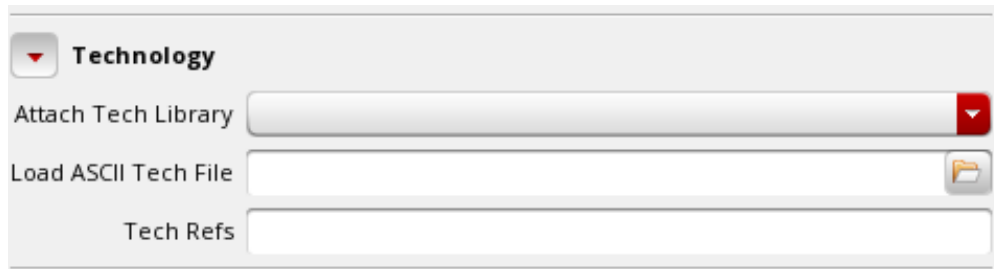
Additionally, this option is selected, [Disable Info Message IDs](#) and [Disable Warning Message IDs](#) fields are populated with `*`, to indicate that all the information (`INFO`) and warning (`WARNING`) messages are disabled for CIW and the Oasis In log file. If you manually change the value of these fields, or if you remove `*` from them, then if *Oasis In to Virtual Memory* check box is selected, information and warning messages, if there are any, are not displayed in the CIW. They are logged in the log file.

For more information, refer to the [Chip Finishing Flow](#) section.

Default value: Deselected

Technology

Expand *Technology* to specify a technology library, file, database that you would like to use during translation.



Attach Tech Library

Use the *Attach Tech Library* drop-down list box to attach an existing technology file to the target library. The *Attach Tech Library* drop-down list box lists all the technology libraries specified in the `cds.lib` file.

If you select this option, XOasis In attaches the technology file from the library specified in the *Attach Tech Library* field instead of creating a new one. This option takes precedence over the *Load ASCII Tech File* option.

If the target library already contains a technology file, then the *Attach Tech Library* option is ignored. The attached technology file is treated as read-only by XOasis In and its contents remain unaltered during XOasis In.

Note: It is essential to have a layer map file while using the *Attach Tech Library* option.

Default value: ""

Default value: `attachTechFileofLib`

Load ASCII Tech File

Use the *Load ASCII Tech File* field to create or merge a technology file in the target library. Alternatively, click the *File Browser* button to select the technology file. The technology file is created when the ASCII technology file is compiled during XOasis In. If the technology file already exists in the target library, then the technology file that has been created by compiling the ASCII technology file is merged with the existing one.

Design Data Translators Reference

Design Translation Using XOasis Translator

The ASCII technology file is searched with respect to the run directory if the complete path of the file is not specified.

Note: The technology file is processed before the contents of the OASIS file.

When a target library is attached to another technology library, XOasis In ignores this option and generates a warning message. If the target library is not attached to any other technology library, a technology file is created or merged with the technology file created from the compilation of the ASCII technology file.

If the absolute path to the current directory exists, then when you click the *File Browser* button, current directory opens in the File Browser. Otherwise, run directory opens.

Command-line option: `loadTechFile`

Tech Refs

Use the *Tech Refs* field to specify the technology databases that will be referenced by the technology database created by XOasis In. You can specify multiple technology databases in this field. Conflicts in the technology graph, mean that the technology database created by XOasis In and the referenced technology databases that you have specified in this field have conflicts. In this scenario, the tool stops and generates an error to specify that XOasis In has detected a conflict in the technology database graph.

Command-line option: `techRefs`

Layer Map



Use the *Layer Map* field to specify an existing layer map file. You can also click the *File Browser* button to select the layer map file. Next, click the *Editor* button to view the specified file in editor.

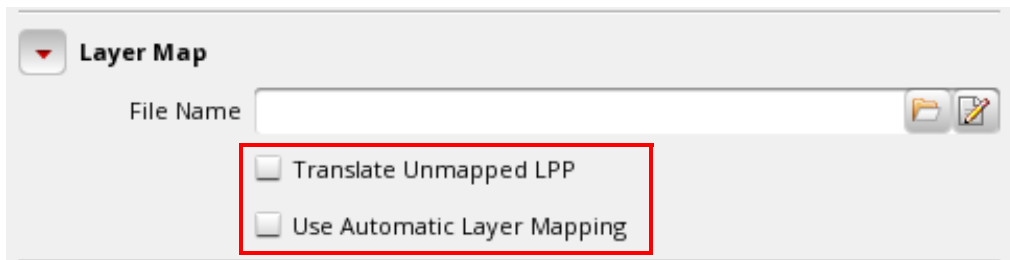
To create new layer mapping records or load a new layer map file, leave the *Layer Map* field blank, and click the *Editor* button to display the XOasis In Layer Map File Editor Form.

Command-line option: `layerMap`

Expand *Layer Map* to view additional options.

Design Data Translators Reference

Design Translation Using XOasis Translator



Translate Unmapped LPPs

Use the *Translate Unmapped LPP* option to map unmapped layers. This option works only when a different user-defined layer mapping, layer map file or SKILL function, has been specified. When you enable *Translate Unmapped LPP*, first, the user-specified layer mapping is applied, and then, automatic layer mapping is performed to translate unmapped layers. For more information, see [Layer Mapping when Translate Unmapped LPP is Selected](#).

Default value: Deselected

Command-line option: `translateUnmappedLPP`

Use Automatic Layer Mapping

Use the *Use Automatic Layer Mapping* option to map layers automatically. This option works only if no layer mapping has been specified. Otherwise, this option is ignored and layers are mapped with the specified mapping. For more information about automatic layer mapping, see [Automatic Layer Mapping in XOasis](#).

Default value: Deselected

Object Map

Object mapping file maps the object type and sub type to the Oasis layer-datatype pair. The object mapping file determines which OpenAccess object should be translated as a shape on which Oasis layer-datatype.



Design Data Translators Reference

Design Translation Using XOasis Translator

Use the *Object Map* field to specify an existing object map file. You can also click the *File Browser* button to select the object map file. Next, click the Editor button to view the specified file in editor.

Command-line option: `objectMap`

To create new object mapping records or load a new object map file, leave the Object Map field blank, click the Editor button to display the XOasis In Object Map File Editor Form.

Expand *Object Map* to view the additional option.

Ignore Object Properties

Use the *Ignore Object Properties* option to ignore property information stored in OASIS used to recreate OA objects during XOasis In.

Property information is written by using an object-map during XOASIS Out to translate objects as shapes in OASIS. XOasis Out, by default would write special properties to these shapes which are used by XOasis In to automatically recreate objects from these shapes. Ignoring these properties means that these objects, which are shapes with special properties in OASIS, will now be translated as shapes in OA.

Default value: Deselected

Command-line option: `ignoreObjectProp`

Log File

Use the *Log File* field to specify log file to be used during XOasis In translation.

Design Data Translators Reference

Design Translation Using XOasis Translator

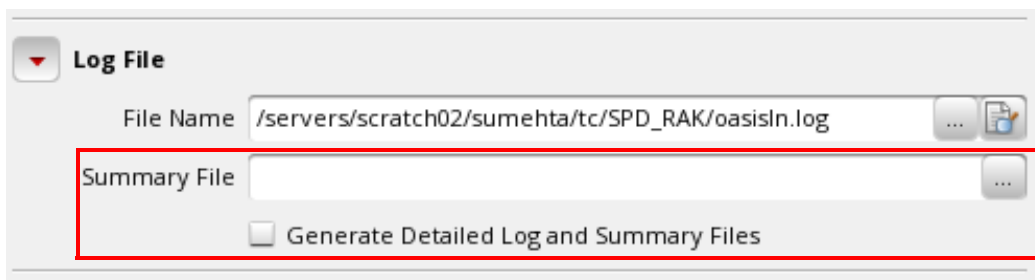
Use the *Log File* field to specify a name for the log file. Alternatively, click the *Browse* button to select the log file. A log file records the translation process steps and the messages generated by XOasis In. After the translation is completed, this file is displayed in a separate read-only text window. However, if the log file size is greater than 200 MB, a message will be displayed mentioning the path to the log file.

The path specified for the log file can be absolute or relative to the run directory. To view the log file, see [Sample Log File](#).

Default filename: oasisIn.log

Command-line option: logFile

Expand Log File to view additional options.



Summary File

Use the *Summary File* field to specify a file that writes the summary of the translated design. If you do not specify a filename in the *Summary File* field before translation, the summary file will not be generated.

The path specified for the summary file can be absolute or relative to the run directory. To view the summary file, see [Sample Summary File](#).

Command-line option: summaryFile

Generate Detailed Log and Summary Files

Use the *Generate Detailed Log and Summary Files* option to include additional summary information in the XOasis In log file and XOasis In summary file.

With this option is enabled, following sections with additional information are added to the XOasis In summary file:

Design Data Translators Reference

Design Translation Using XOasis Translator

- **Individual Cell Statistics:** Cellview summary added to this section is sorted alphabetically according to library/cell/view.

The bBox column is also added. This column lists the boundary boxes of the translated cells in the following format:

```
((lowerLeft.x lowerLeft.y) (upperRight.x upperRight.y))
```

Here, *lowerLeft* and *upperRight* are the lower-left and upper-right coordinates of the boundary box.

Individual Cell Statistics

Library/Cell/View	Instance	Array	Polygon	Rect	Path	Text	Dot	Ellipse	bBox
oasisInLib1/cell1/layout	0	0	0	5	0	0	0	0	((-
5.900 -18.300) (35.200 20.000))									
oasisInLib1/cell2/layout	0	0	0	5	0	0	0	0	((-
5.900 -18.300) (35.200 20.000))									
oasisInLib1/top/layout	4	0	0	5	0	0	0	0	((-
145.100 -234.900) (110.700 200.000))									

- **Statistics of Layers:** Layer summary added to this section is sorted in the same order in which layer-purpose pairs are defined in the technology file or the layer pellet.

Statistics of Layers

Cadence Layer	Cadence Purpose	Oasis Layer	Oasis Datatype	Polygon	Rect	Path	PathSeg	ZeroAreaPolygon	Text	Dot	Ellipse
L0	drawing	0	0	0		3	0	0	0 0	0	0
L1	drawing	1	0	0		3	0	0	0 0	0	0
L2	drawing	2	0	0		3	0	0	0 0	0	0
L3	drawing	3	0	0		3	0	0	0 0	0	0
L4	drawing	4	0	0		3	0	0	0 0	0	0

- **Top Cells:** This section lists the referenced top cells

Top Cells

Cell Name

```
topCell1
topCell2
topCell3
topCell4
```

Design Data Translators Reference

Design Translation Using XOasis Translator

- **Missing Cells:** This section lists the cells that are referenced but are missing from the input OASIS file and the referenced library, if it is specified.

Missing Cells

Cell Name

missingcell1
missingcell2
missingcell3

- **Dropped Layers:** This section lists the `OasisLayer:OasisDataType` pair for which no mapping is found. Shapes on these layer-purpose pairs are not translated.

Dropped Layers

Oasis Oasis
Layer Datatype

81 0
82 0
83 0

- **Individual Advanced Object Statistics:** This section lists the shapes translated as Place and Route objects during XOasis In. It lists the object type, sub-type, Oasis Layer, Oasis name, layer name (for layerBlockage), datatype, and the count of the objects translated.

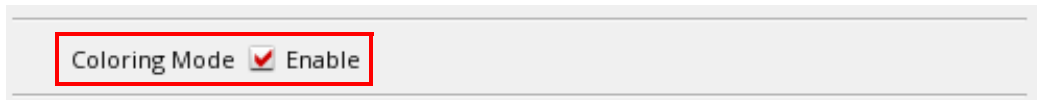
Individual Advanced Object Statistics

Object Type	Object Sub-type	Oasis Layer	Oasis Name	Layer	Datatype	Count
Boundary	Cluster			104	104	3
Boundary	PR			100	100	3
Boundary	Area			102	102	6
Boundary	Snap			106	106	3
Marker	error			110	110	6
Marker	info			108	108	3
AreaBlockage	routing			114	114	3
AreaHalo	routing			116	116	3
LayerBlockage	routing	M4		112	112	3

Default value: Deselected

Command-line option: verbose

Coloring Mode



To enable color mapping during XOasis In, select the *Enable* check box for the *Coloring Mode* option.

Default value: Selected

Command-line option: `enableColoring`

Important

The layer map file, which is either specified through the *Layer Map* field or available in the technology library, is required if you enable *Coloring Mode*. Otherwise, translation stops and an error message will be displayed.

Technology library is mandatory when *Coloring Mode* is enabled. Otherwise, translation stops and an error message is displayed.

Technology library can be specified using the following XOasis In options:

- ☐ *Attach Tech Library* (`attachTechFileOfLib`)
- ☐ *Load ASCII Tech File* (`loadTechFile`)
- ☐ *Tech Refs* (`techRef`)

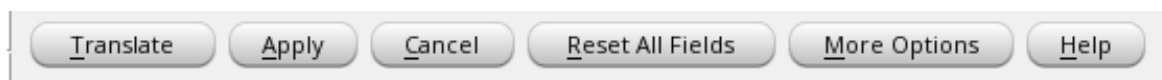
Technology can be either attached to the existing target library or the target library has its own technology.

Currently, coloring of the vias is not supported in the case of the *Via Map File* option.

Note: If the layer map and object map files do not have any color entry, the `-enableColoring` option is not considered during translation.

Design Data Translators Reference

Design Translation Using XOasis Translator



Translate

Use the *Translate* button to translate the OASIS file to OpenAccess. If you select the *Translate* button without selecting *Load File*, the input data will be validated and translated directly.

Apply

Use the *Apply* button to translate the XOasis file to OpenAccess. After the translation is completed, the *XOasis In* form does not disappear.

Important

Clicking the *Translate* or the *Apply* button, saves the values you specify in the XOasis In form and when you start a new session of XOasis In, these last applied changes will be visible in the form. If you make any updates to the values but do not apply the changes by clicking the *Translate* or *Apply* button, these new updates will not be visible the next time you open the XOasis In form. The form will continue displaying values from the last applied changes.

Progress Bar



The *Progress bar* is displayed, when you click the *Translate* or *Apply* button. It moves left to right to indicate that the translation is in progress stops when the translation is completed. You can also click the *Stop* button to stop the process.

Cancel

Use the *Cancel* button to exit the *XOasis In* form.

Reset All Fields

Use the *Reset All Fields* button to set the GUI fields to their default values, which are displayed when the tool is invoked.

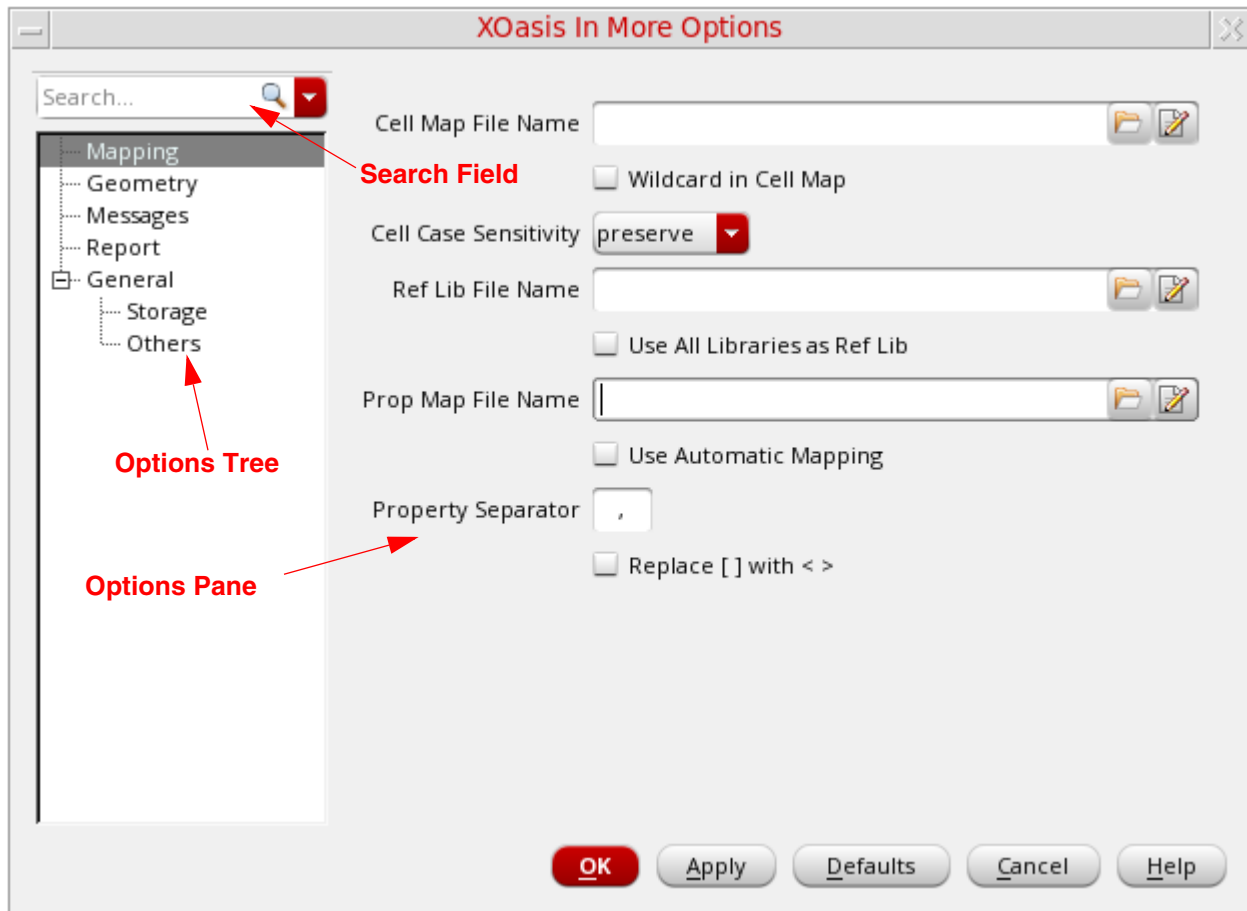
More Options

Use the *More Options* button to display the XOasis In More Options form which features additional options that you can set before starting XOasis In translation.

Help

Use the Help button to invoke online help.

XOasis In More Options Form

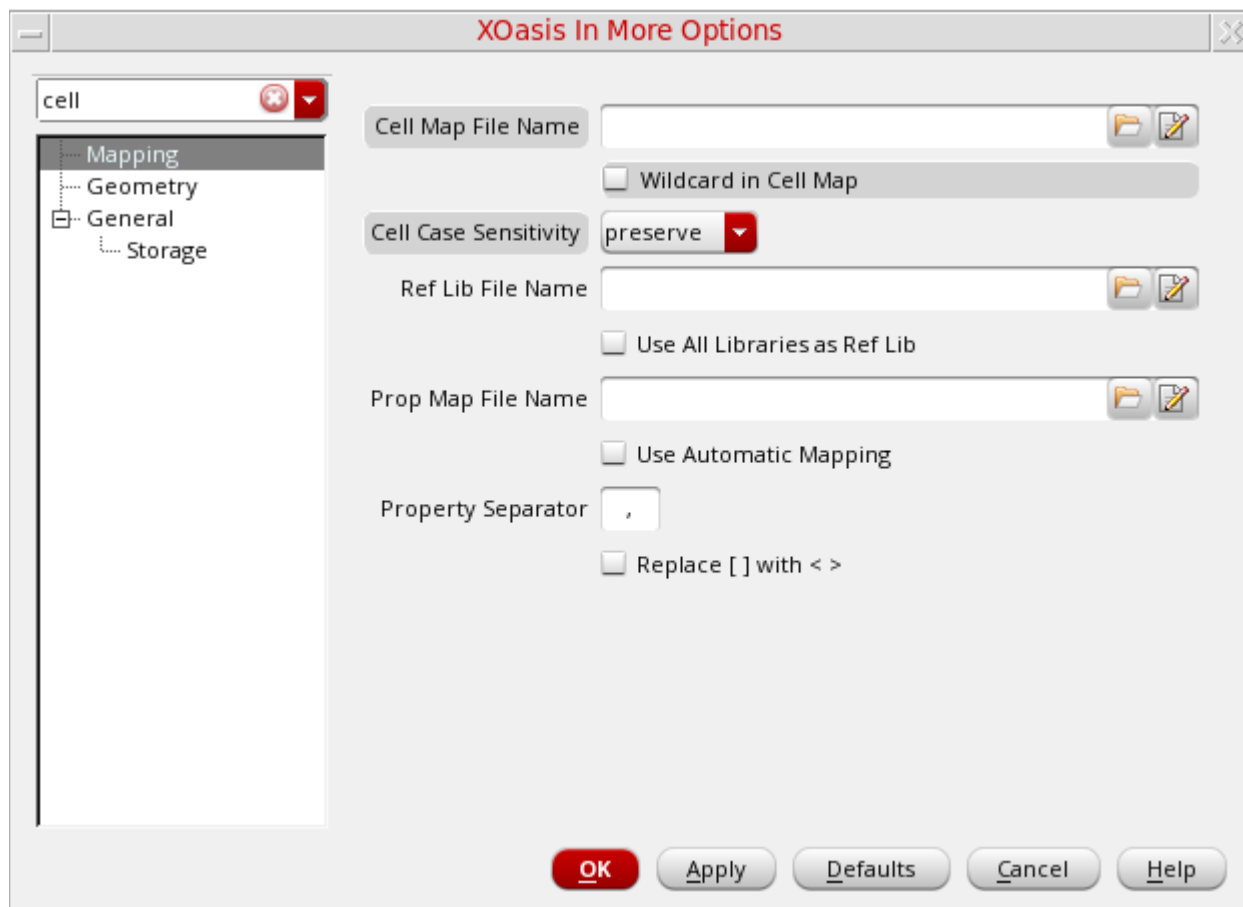


Design Data Translators Reference

Design Translation Using XOasis Translator

Search Field

Use the *Search* field to quickly look for options you want to access or the options related to the value you type in this field. For example, if you type `cell`, only the options related to cell will be visible in the options tree. The related options will also be highlighted in the options pane.



Options Tree

The Options tree enables you to access the following option panes where you can perform additional settings.

- [Mapping Options](#)
- [Geometry Options](#)
- [Messages Options](#)
- [Report Options](#)

Design Data Translators Reference

Design Translation Using XOasis Translator

- General: Storage Options
- General: Others Options

Options Panes

Options panes comprise different commands related to the option currently selected in the tree.

OK

Click *OK* to save your current settings for the More Options form and exit.

Apply

Click *Apply* to save your settings on the More Options form. When you click *Apply*, the form is not closed.

Defaults

Click *Defaults* to set default values for all options in the More Options form.

Cancel

Click *Cancel* to close the More Options form without saving any changes.

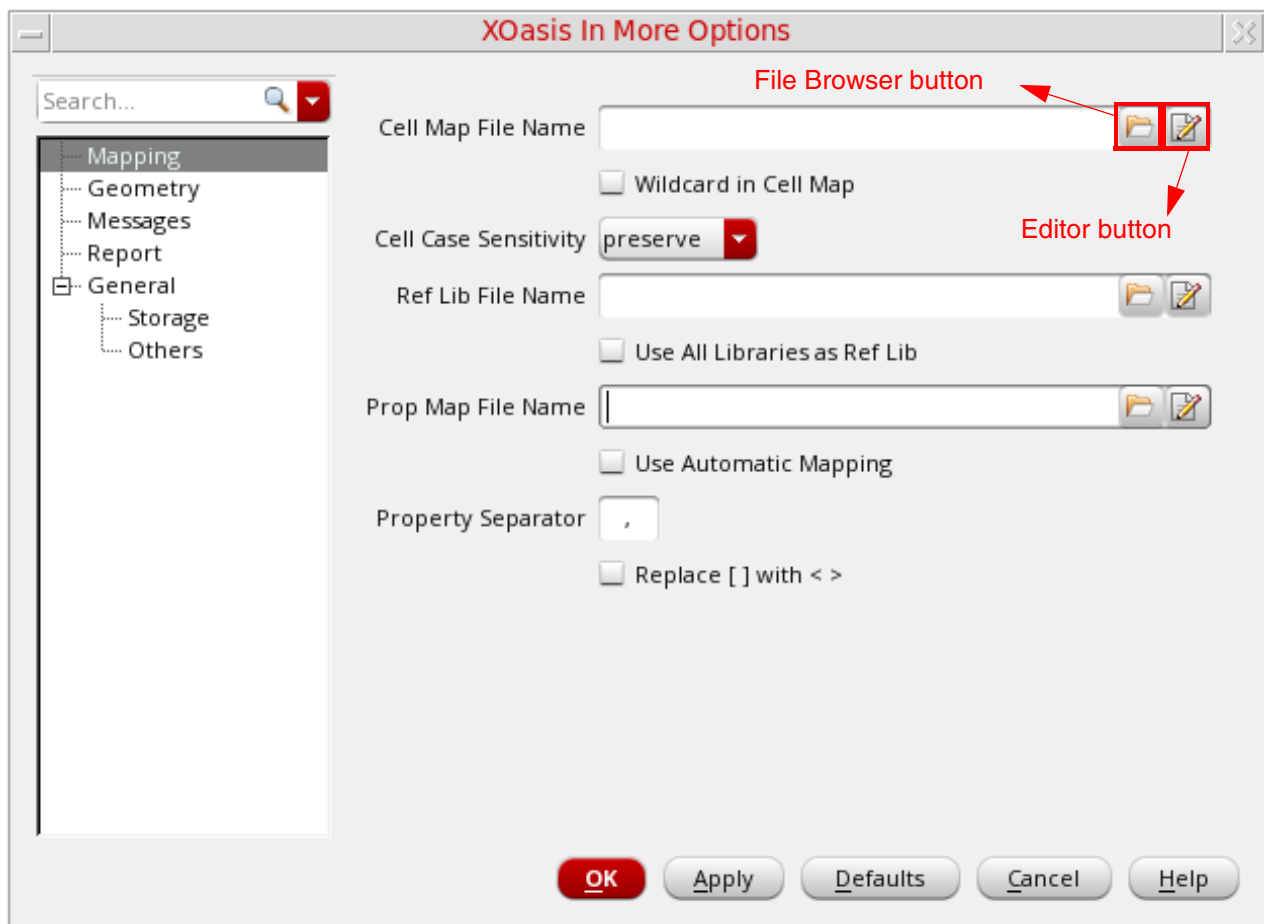
Design Data Translators Reference

Design Translation Using XOasis Translator

This section covers the option panes that you can access from the Options tree in the XOasis In More Options form.

Mapping Options

Use the *Mapping* options pane to specify additional mapping files to be used during XOasis In translation.



This section describes options on the *Mapping* options pane.

Cell Map File Name

Use the *Cell Map File Name* field to specify an existing cell map file. You can also click the File Browser button to select the cell map file. Next, click the *Editor* button to view the specified file in editor.

Design Data Translators Reference

Design Translation Using XOasis Translator

Command-line option: `cellMap`

To create new cell mapping records or load a new cell map file, leave the *Cell Map File* field blank, and click the Editor button to display the XOasis In Cell Map File Editor Form.

Wild Card in Cell Map

Use the *Wild Card in Cell Map* option to use wild cards in XOasis In. The option supports the characters '*' and '?' as wild cards and '\' as the escape character. If you don't use this option, wild card will be considered as a special character during translation.

Consider the following example:

```
*          INV_*          *          INV\*_*_ver1
```

Here, when you use XOasisIn, the created cellview will be `INV_NAND`. If the structure in the GDS is `INV*_NAND_ver1`. In this example, in `INV\*_*_ver1`, the first '*' is treated as special character and the second '*' as the wild card.

If you don't use *Wild Card in Cell Map*, the default behavior will be same as a normal cell map.

Default value: Deselected

Command-line option: `wildCardInCellMap`

Cell Case Sensitivity

The *Cell Case Sensitivity* option determines whether XOasis In changes the case of letters in cell names. XOasis In ignores this option when cell mapping is done. You can map cell names in a cell name map table or in a SKILL file.

preserve

The *preserve* option translates cell names without changing the case. For example, the name `Obj1` remains `Obj1`.

upper

The *upper* option translates all cell names to uppercase. For example, the name `Obj1` becomes `OBJ1`.

lower

The *lower* option translates all cell names to lowercase. For example, the name `Obj1` becomes `obj1`.

Default value: `preserve`

Command-line option: `case`

Ref Lib File Name

Use the *Ref Lib File Name* field to specify an existing reference library name. You can also click the *File Browser* button to select the reference library. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `refLibList`

To select a library name from the list of available libraries to be used as a reference libraries, leave the *Ref Lib File Name* field blank, and click the *Editor* button to display the XOasis In Reference Library File Editor Form.

Use All Libraries as Ref Lib

Use the *Use All Libraries as Ref Lib* option to select all available libraries as reference libraries. The `refLibList` option exports only instances to cellviews from these reference libraries. It does not export any cellviews from them.

Note: To ensure that the missing or referenced cellviews are found in these libraries, you need to provide same references libraries during XOasis In.

If you invoke XOasis In from the command line, you can write the list of reference libraries in a text file and provide this file as the value of the `refLibList` option. If you want to consider all the libraries defined in the `cds.lib` file as reference libraries, use the `OASIS_CDS_LIB` keyword. You can provide this keyword as a value of the `reflibList` option from the command line or write it in a text file containing the list of libraries.

Default value: Deselected

Prop Map File Name

Use the *Prop Map File Name* field to specify an existing properties mapping file. You can also click the *File Browser* button to select the properties mapping file. Next, click the *Editor* button to view the specified file in editor.

Design Data Translators Reference

Design Translation Using XOasis Translator

Command-line option: `propMap`

To create new properties mapping records or load a new properties mapping file, leave the *Prop Map File Name* field blank, and click the *Editor* button to display the XOasis In Property Map File Editor Form.

Use Automatic Mapping

Select the *Use Automatic Mapping* option to enable automatic property mapping. XOasisIn now supports the XOAS_AUTO_PM keyword. You can assign this keyword to the `-propMap` option for automatic property mapping.

Default value: Deselected

Property Separator

Use the *Property Separator* format specifier to specify a property separator. If you specify a property separator in the property map file, XOasis In ignores the property separator defined in the form.

Default value: The default property separator is a comma ‘,’.

Command-line option: `propSeparator`

Replace [] with <>

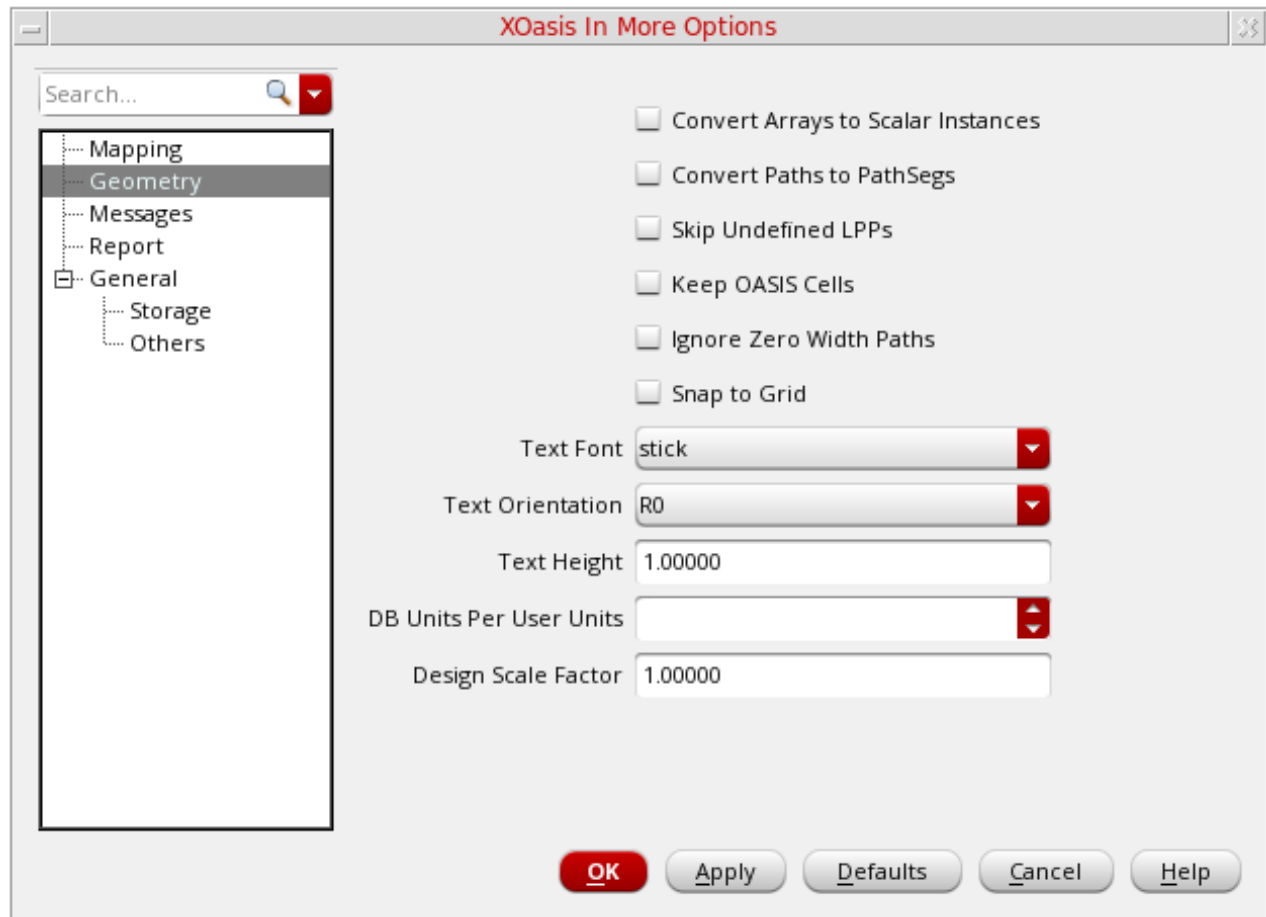
The *Replace [] with <>* option replaces the character [with < and] with > in text records during the translation from the Oasis file to a DFII library.

Default value: Deselected

Command-line option: `replaceBusBitChar`

Geometry Options

This section describes options on the Geometry options pane.



Convert Arrays to Scalar Instances

The *Convert Arrays to Scalar Instances* option translates OASIS arrays to ROWS * COLS scalar instances in OpenAccess. If this option is not selected, each element in AREF translates to a single array in OpenAccess.

Default value: Deselected

Command-line option: `arrayInstToScalar`

Convert Paths to Pathsegs

Use the *Convert Paths to Pathsegs* field to convert the two-point orthogonal paths in the input OASIS file to `pathSeg` objects in the OA database.

Default value: Deselected

Command-line option: `convertPathToPathSeg`

All non-orthogonal paths (PATH) in the input OASIS file are imported as OA path objects even when the *Convert Paths to Pathsegs* option is used during XOasis In.

Skip Undefined LPPs

The *Skip Undefined LPPs* option ignores the layer-purpose pairs that are not existing in the technology file. This option is selected only when you use XOasis In on an existing target library in a design. When a user selects the *Skip Undefined LPPs* option, a warning message will be displayed mentioning that the OASIS layer datatype pair is mapped to the layer-purpose pair, which does not exist in the technology library.

The *Skip Undefined LPPs* option is ignored when a new target library is created. This is because there is no pre-existing technology information. If the technology file of the target library is write-protected, then this option is automatically selected during XOasis In.

The *Skip Undefined LPPs* option is considered when you run XOasis In, in a new library using an ASCII technology file. In this case, any undefined layer-purpose pairs will not be created in the OA technology for the target library.

Default value: Deselected

Command-line option: `skipUndefinedLPP`

Keep OASIS Cells

The *Keep OASIS cells* option provides a way of referring first to the information pertaining to a particular cell in the OASIS file instead of a reference library.

If the *Keep OASIS cells* option is not used, the OASIS file will not be searched while creating instances. If *Write Mode* is set as *overwrite*, this option is ignored and a warning message is generated.

Default value: Deselected

Command-line option: `keepOasisCells`

Ignore Zero Width Paths

Use the *Ignore Zero Width Paths* option to ignore the translation of the zero-width path. If you select this option, zero-width paths are ignored, otherwise they are translated as lines.

Default value: Deselected

Command-line option: `ignoreZeroWidthPath`

Snap to Grid

The *Snap to Grid* option adjusts the coordinates of geometric objects to coincide with the grid resolution. However, this option can shift data from the original XOasis grid resulting in a loss of precision.

Default value: Deselected

Command-line option: `snapToGrid`

Text Font

Use the *Text Font* field to select the default text font. The available font options are stick, euroStyle, gothic, math, roman, script, fixed, swedish, and milSpec.

Default value: stick

Command-line option: `textFont`

Text Orientation

Use the *Text Orientation* field to select the default text orientation.

Default value: R0

Command-line option: `textOrientation`

Text Height

Use the *Text Height* field to control the height of text labels. Default value is 1. For example, you can specify the text height as two or half so that it will double or reduce the label height to half.

Note: This option is ignored for the labels having text attribute property set. For these labels, text height is considered from the text attribute property value.

Default value: 1.00

Command-line option: `textHeight`

DB Units Per User Units

Use the *DB Units per User Units* field to specify the database units per user units, which should be used during translation. This option allows you to scale the target design library as per the new `DBUPerUU`.

Note: This option is considered only if new technology information is created in the target library. This technology information contains the `DBUPerUU` value specified at the option.

Valid values: Greater than or equal to 1

Command-line option: `dbuPerUU`

Design Scale Factor

Use the *Design Scale Factor* option to scale an imported design by the factor set here. This option will accept any positive floating value. The `scale` option will not have any impact on existing options or behavior because scaling is done internally. It will work in addition to the settings in the `DBUPerUU` option.

For example, consider `DBUPerUU` is 1000 and OASIS polygon size is 3000 db units (user units). If you run the command `strmin -scale 4 -dbuPerUU 2000`, the resultant polygon size will be calculated as below:

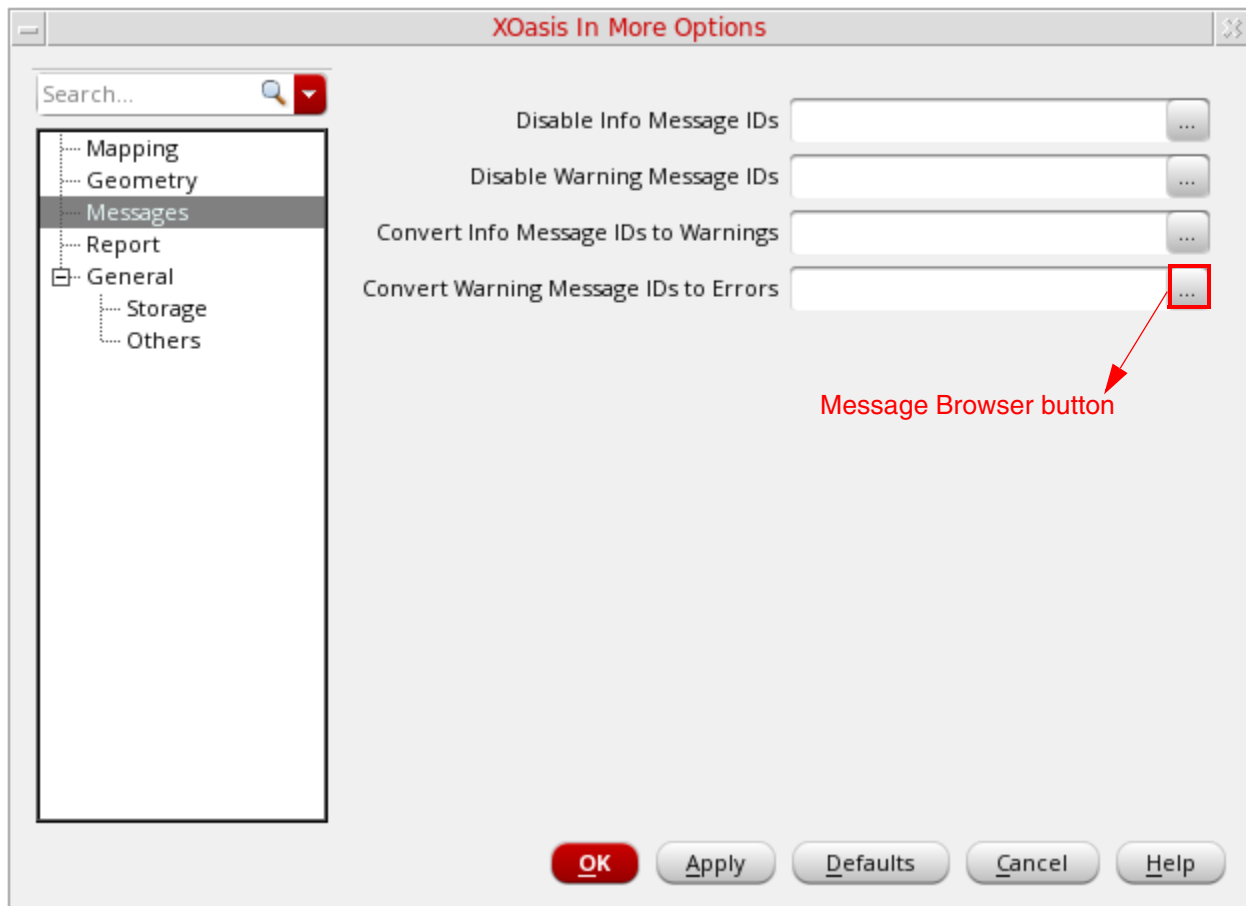
$3000 \times (2000/1000) \times 4 = 24000$ db units (user units)

Default value: 1.00

Command-line option: `scale`

Messages Options

This section describes options on the Messages options pane.



Disable Info Message IDs

Use the *Disable Info Message IDs* field to specify list of information message IDs that you do not want to print. You can also specify message IDs of dependent tools in this field.

The information message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 3075. The list of information message IDs must be separated by a comma or white space.

Command-line option: `noInfo`

Alternatively, click the *Message Browser* button to display the XOasis In Info Messages form to select or specify the information messages that you do not want to print.

Disable Warning Message IDs

Use the *Disable Warning Message IDs* field to specify warning message IDs that you do not want to print. You can also specify message IDs of dependent tools in this field.

The warning message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 3075. The list of warning message IDs should be separated by a comma or white space.

Command-line option: `noWarn`

Alternatively, click the *Message Browser* button to display the [XOasis In Warning Messages](#) form to select or specify the warning messages that you do not want to print.

Convert Info Message IDs to Warnings

Use the *Convert Info Message IDs to Warnings* field to specify list of information message IDs that you want to convert to warning messages. You can also specify message IDs of dependent tools in this field.

The information message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 4374. The list of information message IDs must be separated by a comma or white space.

Command-line option: `infoToWarn`

Alternatively, click the *Message Browser* button to display the [XOasis In Info Messages](#) form to select or specify the information messages that you want to convert to warning messages.

Convert Warning Message IDs to Errors

Use the *Convert Warning Message IDs to Errors* field to specify the IDs of warning messages that you want to convert to error messages. You can also specify message IDs of dependent tools in this field.

The warning message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 3075. The list of warning message IDs should be separated by a comma or white space.

Command-line option: `WarnToErr`

Alternatively, click the *Message Browser* button to display the [XOasis In Warning Messages](#) form to select or specify the warning messages that you want to convert to error messages.

XOasis In Messages Forms

This sections describes the following XOasis In Message forms that are displayed when you click the respective Message Browser button.

- [XOasis In Info Messages Form to Disable Info Messages](#)
- [XOasis In Warning Messages Form to Disable Warning Messages](#)
- [XOasis In Info Messages Form to Convert Info Messages to Warnings](#)
- [XOasis In Warning Messages Form to Convert Warnings to Errors](#)

XOasis In Info Messages Form to Disable Info Messages

This section describes options on the XOasis In Info Messages form that is displayed when you click the *Message Browser* button for the Disable Info Message IDs option.

Select info messages to disable

Only XOasis messages are listed below. To select message IDs of dependent tools, specify the message IDs on the Messages page or in the Other IDs field in the table below.

Id	Message
☐ Other IDs	
☐ 30	Layer map file %s exists in the technology library but is not readable. Automatic layer mapping will...
☐ 60	Creating a new purpose %s in the technology file.
☐ 61	Creating a new layer %s in the technology file.
☐ 62	Creating a new layer:purpose pair %s:%s in the technology file.
☐ 69	All the libraries specified in 'cds.lib' will be considered as reference libraries. This is because refLi...
☐ 70	The 'Reference Library list' and the 'Keep Oasis Cells' options have been used together. Any cell d...
☐ 138	The 'enableLocking' option is selected so the output cellViews in the Cadence database will be sep...
☐ 163	You have not used the viaMap option. If a viaMap file corresponding techfile or technology library...
☐ 171	The 'noObjectProp' and the 'objectMap' options have been used together. Therefore no property...
☐ 173	Object map file %s exists in the technology library but is not readable. No object mapping will be d...

☐ Select All

OK Apply Cancel Help

Select check boxes for the XOasis information messages that you want to disable.

Other IDs

Use the *Other IDs* field to specify information message IDs of other dependent tools.

Select All

Click the *Select All* check box to select all message IDs.

Cancel

Use the *Cancel* button to exit the XOasis In Info Messages form without saving any changes.

XOasis In Warning Messages Form to Disable Warning Messages

This section describes options on the XOasis In Warning Messages form that is displayed when you click the *Message Browser* button for the Disable Warning Message IDs option.

Id	Message
☐ Other IDs	
☐ 18	Performing automatic layer mapping because either the layer mapping information is missing or t...
☐ 20	Output Oasis file '%s' already exists. It will be overwritten.
☐ 25	The layer map file '%s' contains an invalid record at line number '%d' and will be ignored. This is b...
☐ 27	The property map file '%s' contains an invalid record at line number '%d' and will be ignored. This ...
☐ 28	The cell map file '%s' contains an invalid record at line number '%d' and will be ignored. This is bec...
☐ 29	refLib file contains invalid record at line number '%d'. This record in the refLib file will be ignored.
☐ 31	Layer map file '%s' contains a duplicate entry for Oasis layer:datatype '%d:%d' at line '%d'. Therefo...
☐ 32	The layer map file '%s' contains unknown layer '%s' at line number '%d' and will be ignored. This is ...
☐ 33	The layer map file '%s' contains unknown purpose '%s' at line number '%d' and will be ignored. Thi...
☐ 34	The objects in the layer purpose pair '%u:%u' are ignored. This is because either the layer purpos...

Select check boxes of the XOasis warning messages listed in this form.

Other IDs

Use the *Other IDs* field to specify warning message IDs of other dependent tools.

Select All

Click the *Select All* check box to select all message IDs.

Cancel

Use the *Cancel* button to exit the XOasis In Warning Messages form without saving any changes.

XOasis In Info Messages Form to Convert Info Messages to Warnings

This section describes options on the XOasis In Info Messages form that is displayed when you click the *Message Browser* button for the Convert Info Message IDs to Warnings option.

Id	Message
☐ Other IDs	
☐ 30	Layer map file %s exists in the technology library but is not readable. Automatic layer mapping will...
☐ 60	Creating a new purpose %s in the technology file.
☐ 61	Creating a new layer %s in the technology file.
☐ 62	Creating a new layer:purpose pair %s:%s in the technology file.
☐ 69	All the libraries specified in 'cds.lib' will be considered as reference libraries. This is because refLi...
☐ 70	The 'Reference Library list' and the 'Keep Oasis Cells' options have been used together. Any cell d...
☐ 138	The 'enableLocking' option is selected so the output cellViews in the Cadence database will be sep...
☐ 163	You have not used the viaMap option. If a viaMap file corresponding techfile or technology library...
☐ 171	The 'noObjectProp' and the 'objectMap' options have been used together. Therefore no property...
☐ 173	Object map file %s exists in the technology library but is not readable. No object mapping will be d...

Select check boxes of the XOasis information messages listed in this form.

Other IDs

Use the *Other IDs* field to specify information message IDs of other dependent tools.

Cancel

Use the *Cancel* button to exit the XOasis In Info Messages form without saving any changes.

XOasis In Warning Messages Form to Convert Warnings to Errors

This section describes options on the XOasis In Warning Messages form that is displayed when you click the *Message Browser* button for the Convert Warning Message IDs to Errors option.

Select warning messages to convert to errors

Only XOasis messages are listed below. To select message IDs of dependent tools, specify the message IDs on the Messages page or in the Other IDs field in the table below.

Id	Message
☐ Other IDs	
☐ 18	Performing automatic layer mapping because either the layer mapping information is missing or t...
☐ 20	Output Oasis file '%s' already exists. It will be overwritten.
☐ 25	The layer map file '%s' contains an invalid record at line number '%d' and will be ignored. This is b...
☐ 27	The property map file '%s' contains an invalid record at line number '%d' and will be ignored. This ...
☐ 28	The cell map file '%s' contains an invalid record at line number '%d' and will be ignored. This is bec...
☐ 29	refLib file contains invalid record at line number '%d'. This record in the refLib file will be ignored.
☐ 31	Layer map file '%s' contains a duplicate entry for Oasis layer:datatype '%d:%d' at line '%d'. Therefo...
☐ 32	The layer map file '%s' contains unknown layer '%s' at line number '%d' and will be ignored. This is ...
☐ 33	The layer map file '%s' contains unknown purpose '%s' at line number '%d' and will be ignored. Thi...
☐ 34	The objects in the layer purpose pair '%u:%u' are ignored. This is because either the layer purpos...

Select check boxes of the XOasis information messages listed in this form.

Other IDs

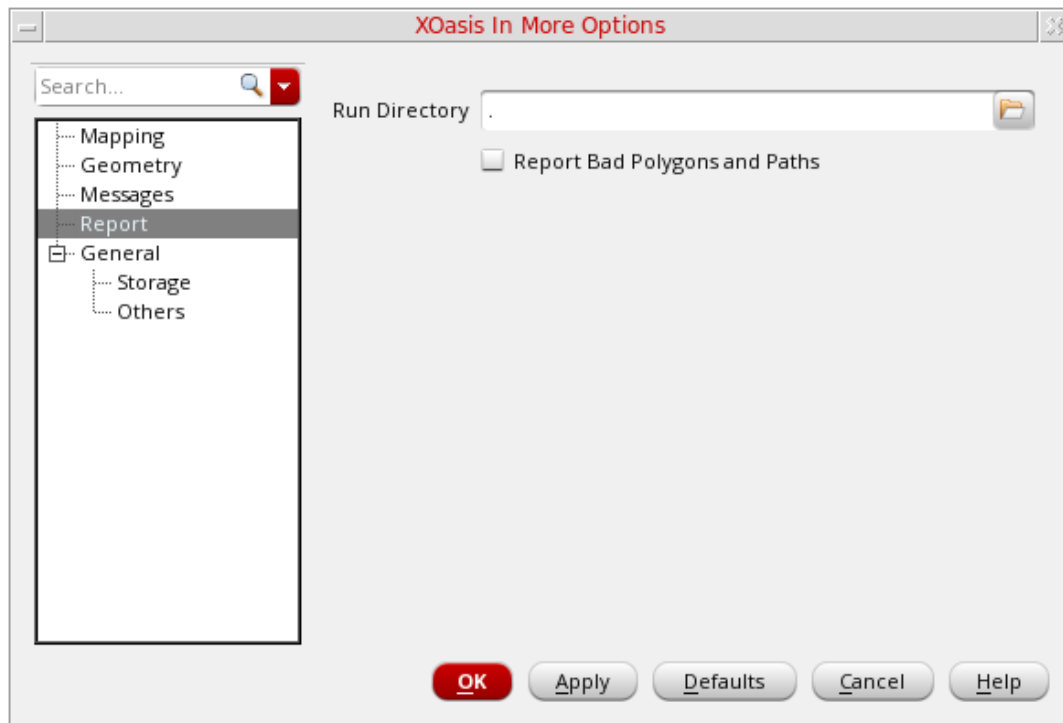
Use the *Other IDs* field to specify information message IDs of other dependent tools.

Cancel

Use the *Cancel* button to exit the XOasis In Info Messages form without saving any changes.

Report Options

This section describes options on the Report options pane.



Run Directory

The *Run Directory* field accepts the default directory for all the files you specify in the *XOasis In* form. The run directory will be created automatically, provided you have the appropriate permission, if it does not exist during XOasis In or XOasis Out. If you want to read from or write to a file that is not in the run directory, type the absolute file path in the field.

Default value: Current working directory

Command-line option: `runDir`

Report Bad Polygons and Paths

The *Report Bad Polygons and Path* option provides a way for reporting self intersecting paths; coincident or collinear paths; paths that have less than two coordinates; path with an invalid begin and end extension value; an open polygon; coincident or collinear polygons; and a polygon with self-intersecting points. However, using this option impacts performance.

Design Data Translators Reference

Design Translation Using XOasis Translator

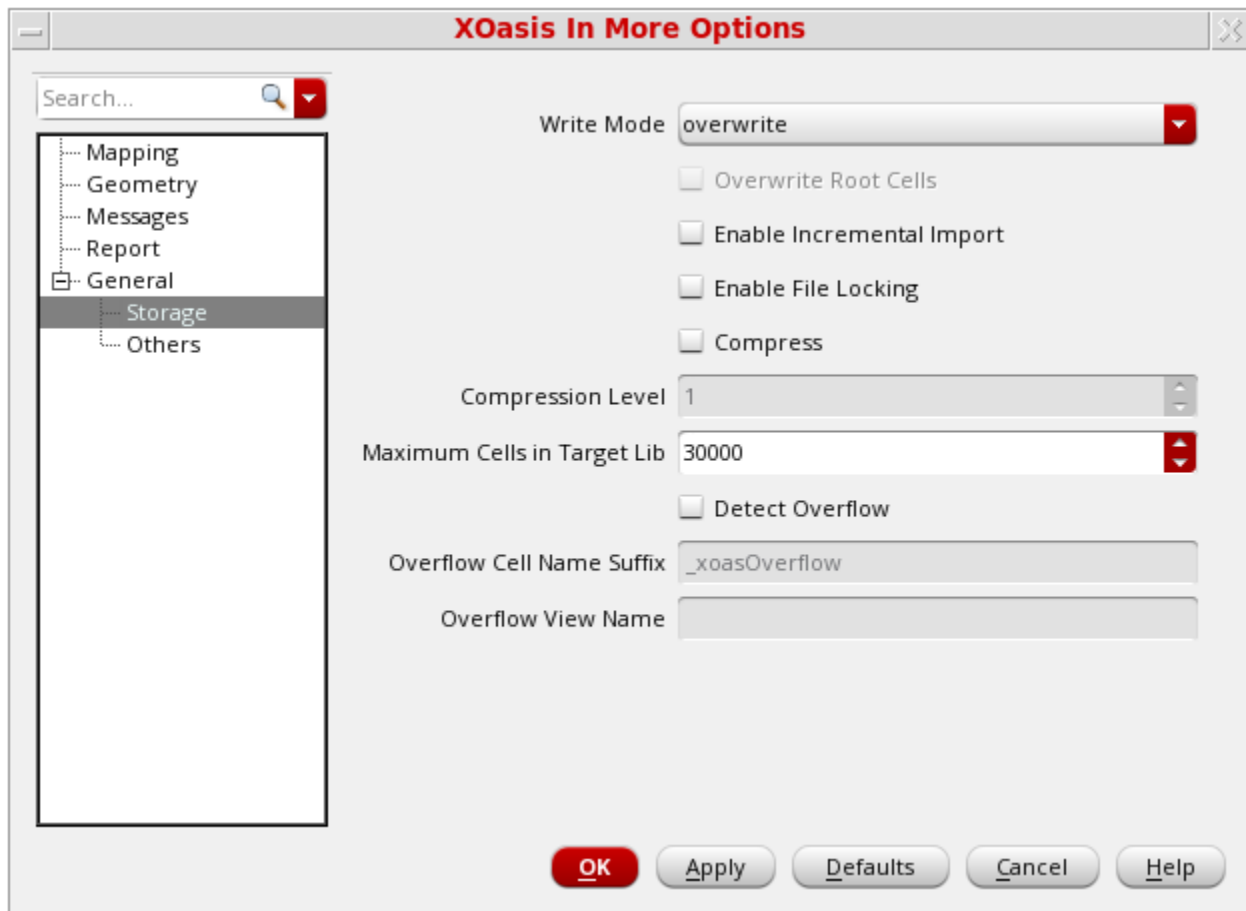
If an OASIS file has PATH records with PATHTYPE1 that has a circular path end, then according to the OASIS format, these PATH records are translated as paths with octagonal extensions. If there is an issue, it should be corrected at the source of the OASIS file.

Default value: Deselected

Command-line option: `checkPolygon`

General: Storage Options

This section describes options on the General: Storage options pane.



Write Mode

Use the *Write Mode* field to select the write mode during the translation of the existing cells. The following options are available:

overwrite

Select the *overwrite* option to overwrite the existing cells in the target library during translation.

noOverwrite

Select the *noOverwrite* option to prevent existing cells in the target library from being overwritten during translation.

append

Select the *append* option to append data to an existing database during translation. If this option is selected, in case the OASIS file contains shapes or instances that already exist in the database, they will be appended to the existing database. Therefore, the database will have duplicates of these shapes and instances.

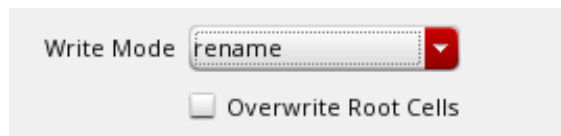
rename

Select the *rename* option to rename the duplicate cells in the target library during translation.

Default value: overwrite

Command-line option: `writeMode`

Overwrite Root Cells



This option is available only when *Write Mode* is set to *rename*. When this option is used, all the root cells (un-instantiated cells in the OASIS file) replace the duplicate cells in the user-defined top-cell hierarchy; and the cells present in the hierarchy of the root cell are renamed.

Default value: Deselected

Command-line option: `overwriteRootCells`

Detect Overflow

Use the *Detect Overflow* option to detect overflow cellviews during translation.

OpenAccess limits the number of individual object types (like shapes, instances and properties) in a cellview. By default, the XOasis In translator fails to translate an OASIS file to OpenAccess if the maximum capacity of a cellview is reached for an individual object type. The *Detect Overflow* option enables XOasis In to translate an OASIS file even if the number of individual objects is more than the maximum capacity of a single cellview. In this scenario, overflow cellviews are created on reaching the maximum capacity of a cellview, with the remaining objects translated in the overflow cell. An instance of this overflow cell is then placed inside the main cell exceeding the maximum capacity.

Design Data Translators Reference

Design Translation Using XOasis Translator

Overflow cells are created with the name of the original cell suffixed with the value of the Overflow Cell Name Suffix option, and a view-name specified by the value of option Overflow View Name option.

Default value: Deselected

Command-line option: `detectOverflow`

Overflow Cell Name Suffix

Use the *Overflow Cell Name Suffix* to specify the string to be appended to the name of the overflow cells.

Default value: `_xoasOverflow`

Command-line option: `overflowCellNameSuffix`

Overflow View Name

Use the *Overflow View Name* to specify a view name for the overflow cells.

Default value: If no value is specified, value of the View option is used.

Command-line option: `overflowViewName`

Example 3-7 Multiple OASIS files; Single XOasis In

G1. oas	G2. oas	G3. oas	G4. oas
TopCell Definition SREF A SREF B SREF C A Definition SREF C B Definition SREF C C Definition	A Definition SREF C C Definition	B Definition SREF C C Definition	C Definition

In the above example, there are multiple definitions for cells across OASIS files.

If the *Write Mode* is set to *overwrite*, then the expected XOasis In output is the shown below.

Example 3-8 XOasis In output: Write mode is overwrite

```
TopCell1 Definition
```

```
SREF A2
```

```
SREF B3
```

```
SREF C4
```

```
A2 Definition
```

```
SREF C4
```

```
B3 Definition
```

```
SREF C4
```

```
C4 Definition
```

If the *Write Mode* is set to *noOverwrite*, then the expected XOasis In output is the shown below.

Example 3-9 XOasis In output: Write mode is noOverwrite

```
TopCell1 Definition
```

```
SREF A1
```

```
SREF B1
```

```
SREF C1
```

```
A1 Definition
```

```
SREF C1
```

```
C1 Definition
```

```
B1 Definition
```

```
SREF C1
```

If the *Write Mode* is set to *append*, then the expected XOasis In output is the shown below.

Design Data Translators Reference

Design Translation Using XOasis Translator

Example 3-10 XOasis In output: Write mode is append

TopCell1 ₁ Definition	
SREF A ₁₊₂	A ₁₊₂ : Cellname is 'A' and
SREF B ₁₊₃	it has shapes in cell 'A'
SREF C ₁₊₂₊₃₊₄	of G1. oas and G2. oas
A ₁₊₂ Definition	
SREF C ₁₊₂₊₃₊₄	
SREF C ₁₊₂₊₃₊₄	
B ₁₊₃ Definition	
SREF C ₁₊₂₊₃₊₄	
SREF C ₁₊₂₊₃₊₄	
C ₁₊₂₊₃₊₄ Definition	

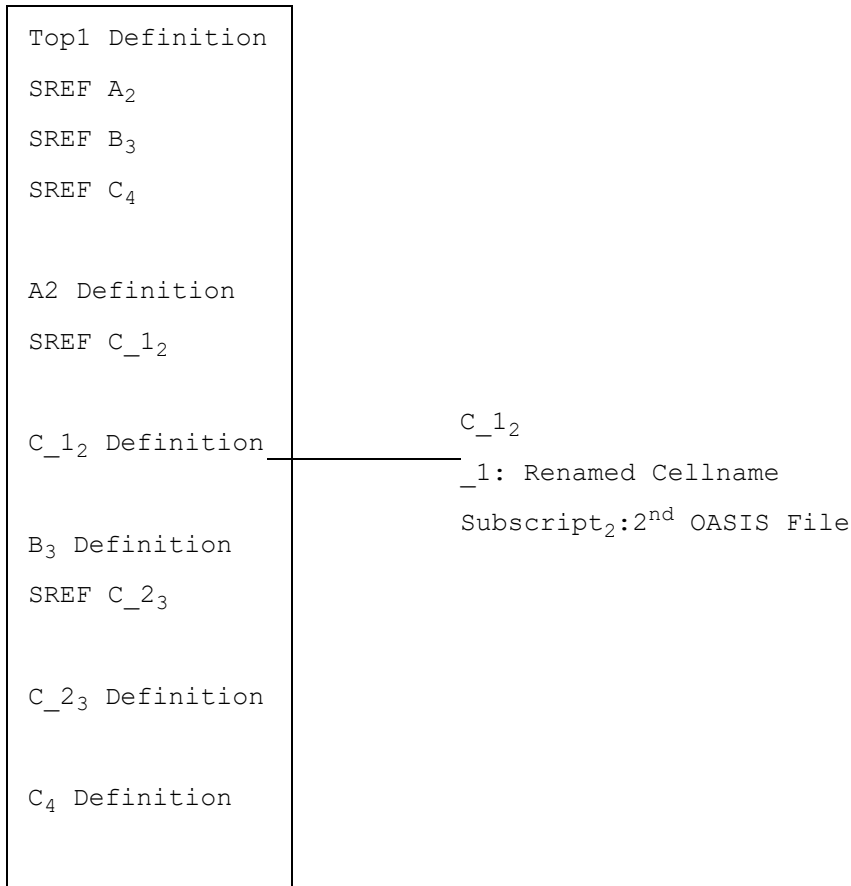
If the *Write Mode* is set to *rename*, then the expected XOasis In output is the shown below.

Example 3-11 XOasis In output: Write mode is rename

TopCell ₁ Definition	
SREF A ₁	
SREF B ₁	
SREF C ₁	
A ₁ Definition	
SREF C ₁	
B ₁ Definition	
SREF C ₁	
C ₁ Definition	
A ₁₂ Definition	C ₁₂
SREF C ₁₂	₁ : Renamed Cellname
	Subscript ₂ :2 nd OASIS File
C ₁₂ Definition	
B ₁₃ Definition	
SREF C ₂₃	
C ₂₃ Definition	
C ₃₄ Definition	

If the *Write Mode* is set to *rename* and the *Overwrite Root Cells* option is selected, then the expected XOasis In output is the shown below.

Example 3-12 XOasis In output: Write mode (rename) and Overwrite Root Cells



Note: If you set the `xstInOverwriteLibPopup` SKILL variable to `t` and *Write* mode to *overwrite* or *append*, then when you try to perform XOasis In translation to an existing library, a message will be displayed to warn you that cells in the target library will be overwritten or appended to in this process.

Enable Incremental Import

Use the *Enable Incremental Import* option to import multiple OASIS files into the same design library. The two methods to import the OASIS files are:

- Specify all the OASIS files in one XOasis In translation.
- Specify a subset of OASIS files at a time. For this, multiple XOasis In translations are needed to import the OASIS files. This is called incremental import mode. For incremental import mode, select the *Enable Incremental Import* option during the first XOasis In translation cycle.

Default value: Deselected

XOasis In creates a temporary file to maintain incremental import data at the following location:

```
.cadence/geoTrans/.incrementalImport
```

A limitation for this option is that one directory can handle only one incremental import. However, you can perform XOasis In translation, with the *Enable Incremental Import* option deselected, in the same directory.

Command-line option: `enableIncrementalImport`

Enable File Locking

Use the *Enable File Locking* option to enable file locking during XOasis In. This option allows other applications to access the target library when XOasis In is running. The additional synchronization overhead required for shared access by multiple applications increases the translation time.

Default value: Deselected

Command-line option: `enableLocking`

Compress

Use the *Compress* option to enable compression during XOasis In.

Default value: Deselected

Command-line option: `compress`

Compression Level

Use the *Compression Level* option to specify the level of compression to be applied during XOasis In. You can specify a value from 1 to 9 for this option. Value 1 signifies the lowest level of compression and 9 signifies the highest level of compression. If you do not specify any value in the *Compression Level* option, value 1 is considered as the compression level during XOasis In. This option is enabled only if the *Compress* option is selected.

Valid values: 1 to 9

Default value: 1

Command-line option: `compressLevel`

Maximum Cells in Target Lib

Use the *Maximum Cells in Target Lib* field to specify the maximum number of cells that you want to include in the target library. If the number of cells are more than the cells you have specified, then the other library will be created with the same name followed by “_1”. For example, if the number of cells in the target library, `trglib`, exceeds the limit of the value specified in the *Maximum Cells in Target Lib* field, then the new library will be created as, `trglib_1`. The number following the filename will be incremented by 1 depending on the number of libraries that are created. For example, another library will be created as, `trglib_2`.

Valid values: 1-1024000

Default value: 30,000

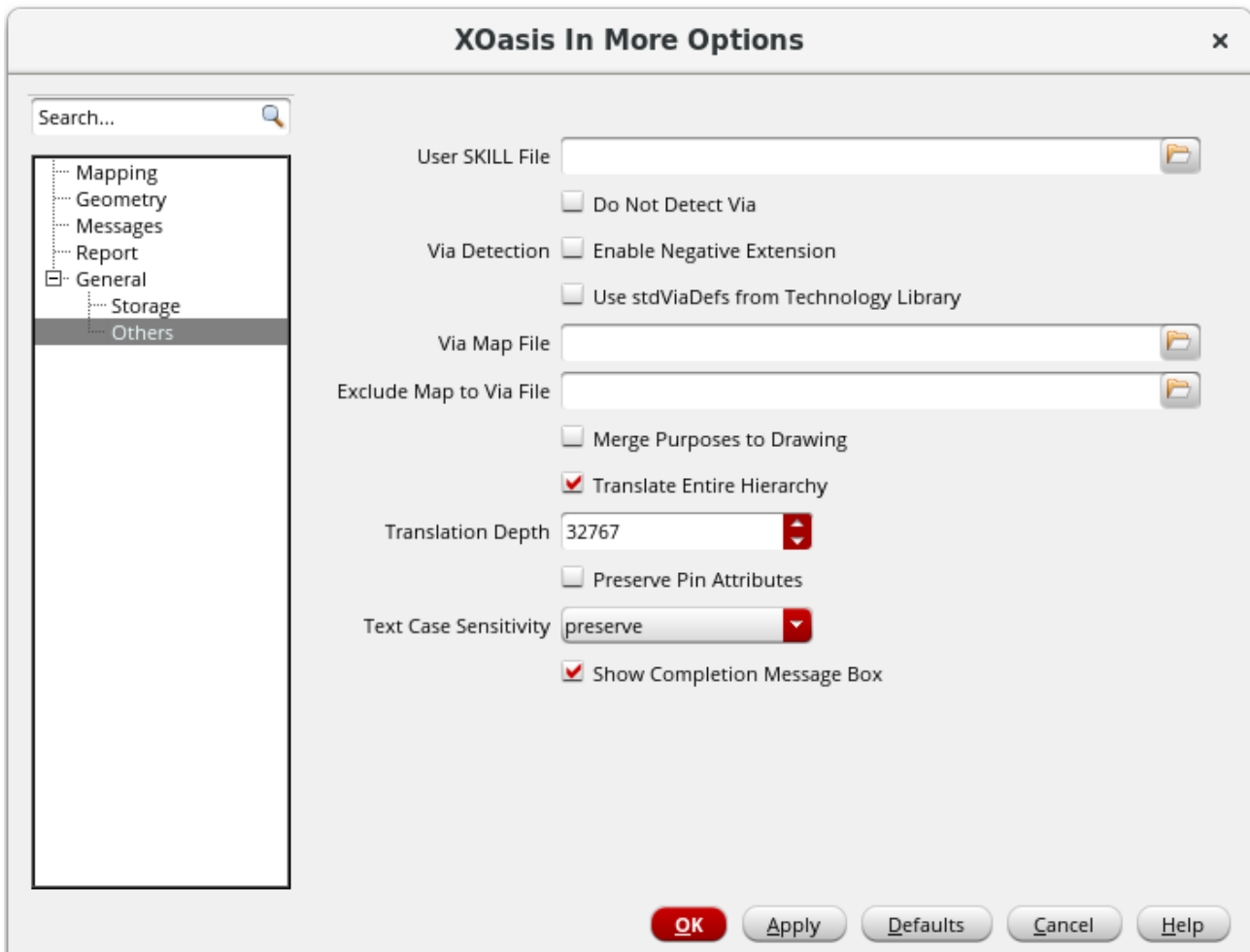
Command-line option: `maxCellsInTargetLib`

Design Data Translators Reference

Design Translation Using XOasis Translator

General: Others Options

This section describes the options on the General: Others option pane.



User SKILL File

Use the *User SKILL File* field to specify a file that consists of user-defined SKILL routines. You can specify mapping mechanism for cell names, layers, and fonts through the use of programming routines written in SKILL in the SKILL file.

In order to load all the required SKILL procedures in a single file, use only one SKILL file during translation. The SKILL file is searched with respect to the run directory if the complete path of the SKILL file is not specified.

UserSkillFile will always be loaded before the libInit.il file. The libInit.il file is usually present in the technology file.

Note: XOasis In supports only following SKILL Core, `db`, `dd`, `tech`, and `cst` SKILL functions. Application specific functions, such as `le`, `hi`, `sch`, `lx`, `via` and so on are not supported.

Command-line option: `userSkillFile`

Do Not Detect Via

Select the *Do Not Detect Via* option to disable via detection during translation.

In case, there is no via map file and the *Do Not Detect Via* option is not specified, the tool evaluates the system and if it is possible, performs via detection. This process creates standard and custom vias during XOasis In.

If you do not have a via map file, and the option *Do Not Detect Via* is selected, then the translator will produce multiple OpenAccess designs to represent a via. This affects the performance of XOasis In and other applications, which work on these OpenAccess designs adversely.

Via detection is a color-aware process. Therefore, when the *Coloring Mode* option is enabled, color information on vias is appropriately set during translation.

Note: The vias are not detected automatically for cdsGen vias.

Default value: Deselected

Command-line option: `noDetectVias`

Via Detection

Via detection is enabled by default during XStream In translation. You can specify the additional following options to be considered during translation.

■ ***Enable Negative Extension***

Use the *Enable Negative Extension* option to detect vias with negative metal extension during via detection. Vias with negative extension have cut shapes that are partially overlapping with metal shapes.

Default value: Deselected

Command-line option: `enableNegativeViaExt`

■ **Use stdViaDefs from Technology Library**

Use the Use stdViaDefs from Technology Library option to enable using the standard via definitions from the technology database for via detection.

Default value: Deselected

Command-line option: `useExistingViaDefsForDetection`

Via Map File

Use the *Via Map File* field to specify a via map file. The via map file maps the Structure names from the OASIS file to the corresponding viaDef names in the input technology library. For more information on the via map file, see [Via Map File](#).

Command-line option: `viaMap`

Exclude Map To Via File

Use the *Exclude Map To Via File* field to specify a file name that contains the list of STRUCTs. In XOasis In, vias are detected by default. Therefore, when the *Exclude Map To Via File* option is specified, XOasis In does not convert the STRUCT names as Viadefs.

The via exclusion file contains one OASIS structure per line. This option supports the use of following wildcards:

Wildcard	Action
\$	Matches one unknown character.
*	Matches zero or more unknown characters.
\	Works as the escape character. Use this when you want a wildcard character to be treated as a normal character.

Sample Via Exclusion File

```
# List of structures to be excluded from via mapping
M1M2Wire      # Do not map M1M2Wire to a via
M2M3Wire*     # Do not map cells starting with M2M3Wire to a via
```

Command-line option: `excludeMapToVia`

Merge Purposes to Drawing

Use the *Merge Purposes to Drawing* option to map all OASIS datatypes to the purpose `drawing` during XOasis In. This option is applicable only if XOasis In does not find a layer map file or a SKILL file. For more information about automatic layer mapping, see [Layer Map File](#).

Default value: Deselected

Command-line option: `mergeUndefPurposToDrawing`

Translate Entire Hierarchy

Use the *Translate Entire Hierarchy* option to translate the entire design hierarchy.

Default value: Selected

Translation Depth

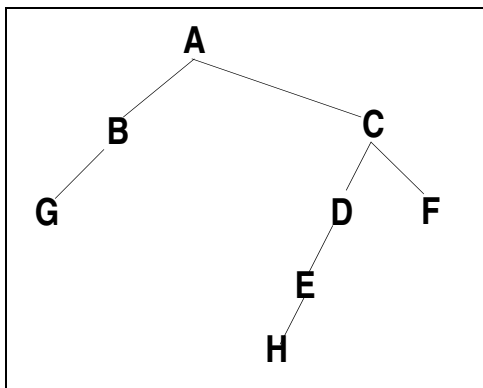
Use the *Translation Depth* option to specify the depth of the design hierarchy during XOasis In. XOasis In stops the translation process at the hierarchy level indicated by this option. The hierarchy depth value applies to all the top cells getting translated to OpenAccess. This option is enabled only if the *Translate Entire Hierarchy* option is deselected.

Value values: 0 to 32767

Default value: 32767

Command-line option: `hierDepth`

Example 3-13 Translating cells C, D, and F by using the Translation Depth option



The hierarchy above shows cells A, B, C, D, E, F, G, and H placed at different levels. When cell C is selected as the top cell and the hierarchy depth is selected as 1, then the cells C, D, and F are translated.

An instance of E is created in the structure D, but its structure information is ignored. Cell H is ignored and is not translated to OpenAccess.

Preserve Pin Attributes

Use the *Preserve Pin Attributes* field to preserve pin connectivity information.

Default value: Deselected

Command-line option: `preservePinAtt`

Text Case Sensitivity

Use the *Text Case Sensitivity* option to specify if XOasis In will change the casing of text labels. The following options are available:

- *preserve* - The *preserve* option translates text labels without changing case. For example, the label `Obj1` remains `Obj1`.
- *upper* - The *upper* option translates the text label to uppercase. For example, the label `Obj1` becomes `OBJ1`.
- *lower* - The *lower* option translates the text labels to lowercase. For example, the label `Obj1` becomes `obj1`.

Default value: `preserve`

Design Data Translators Reference

Design Translation Using XOasis Translator

Command-line option: `textCase`

Show Completion Message Box

The Show Completion Message Box check box lets you to display the *Oasis In translation complete* message box with the number of errors and warnings after the translation is completed. The state of this check box depends on the value of the `xoasShowCompletionMsgBox` SKILL variable, which is `t` by default.

XOasis In Editor Forms

This section discusses the following XOasis In Editor Forms:

- [XOasis In Layer Map File Editor Form](#)
- [XOasis In Object Map File Editor Form](#)
- [XOasis In Cell Map File Editor Form](#)
- [XOasis In Reference Library File Editor Form](#)
- [XOasis In Property Map File Editor Form](#)

XOasis In Layer Map File Editor Form

This section describes options on the XOasis In Layer Map File Editor form that is displayed when you click the *Editor* button for the [Layer Map](#) option.

OASIS Layer	OASIS Datatype	Layer Name	Purpose Name	Material Type	Mask Number	Photo Mask Color	Color State

Add Row button Delete Row button

New File

Load Save As and Exit **Save and Exit** Cancel Help

Add Row

Click the *Add Row* button to create a new layer mapping record.

For more information on columns in a layer mapping record, see [Layer Map File](#).

OASIS Layer

Use the *OASIS Layer* field to specify the OASIS layer name that you want to map to the OpenAccess layer.

OASIS Datatype

Use the *OASIS Datatype* field to specify the OASIS datatype that you want to translate to OpenAccess purpose.

Layer Name

Use the *Layer Name* field to specify the OpenAccess layer name to which you want to map the OASIS layer.

Purpose Name

Use the *Purpose Name* field to specify the OpenAccess purpose name to which you want to map the OASIS datatype.

Material Type

Use the *Material Type* field to specify the OpenAccess material type to which you want to map the OASIS datatype. This will be used to create the layer, if it does not exist. To hide *Material Type* column, right-click on the column heading row and clear the *Material Type* check box.

Mask Number

Use the *Mask Number* field to specify the OpenAccess mask number to which you want to map the OASIS layer. To hide the *Mask Number* column, right-click on the column heading row and clear the *Mask Number* check box.

Photo Mask Color

Use the *Photo Mask Color* field to specify the photo mask color of the layer.

Color State

Use the *Color State* field to specify the color state of the layer.

Note: The *Photo Mask Color* and *Color State* fields are visible only if the *Coloring Mode* option is enabled.

Note: If you want to populate and select the values corresponding to the fields mentioned above, then you need to enter the valid OASIS file in the *OASIS File* field. In case you want to add or modify the values in any of these fields then you need to double-click in the cells.

Delete Row

Use the *Delete Row* button to delete an existing layer mapping record.

Load

Use the *Load* button to load a new layer map file. You can specify multiple layer name map files by using the *Load* file option iteratively. A layer map file maps OASIS layer number and OASIS data type pairs to OpenAccess layer name and purpose name pairs. If you specify an invalid filename, a warning message is generated and the filename is ignored.

The path specified for the layer map file can be absolute or relative to the run directory. For more information, see [Layer Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the mapping information in a new file and exit the layer map file editor.

Save and Exit

Use the *Save and Exit* button to save the mapping information in the current file and exit the layer map file editor.

Cancel

Use the *Cancel* button to exit the XOasis In Layer Map File Editor form without saving any changes.

Design Data Translators Reference

Design Translation Using XOasis Translator

XOasis In Object Map File Editor Form

This section describes options on the XOasis In Object Map File Editor form that is displayed when you click the *Editor* button for the Object Map option.

Add Row

Click the *Add Row* button to create a new object mapping record.

For more information on columns in an object mapping record, see Object Map File.

Object Name

Use the *Object Name* field to select the name of the object to be mapped.

Sub Object Name

Use the *Sub Object Name* field to select the name of the sub object to be mapped.

Layer Name

The *Layer Name* field is displayed only when the Object Name is specified as layerBlockage. However, *N/A* will be displayed in the *Layer Name* field in case of any other Object Name.

OASIS Layer

Use the *OASIS Layer* field to specify the Oasis layer number for the OpenAccess object. By default, the Oasis layer number is specified as 0.

OASIS Type

Use the *OASIS Type* field to specify the Oasis datatype for the OpenAccess object. By default, the Oasis datatype is specified as 0.

Photo Mask Color

Use the *Photo Mask Color* field to specify the photo mask color of the layer.

The *Photo Mask Color* field is visible only if *Coloring Mode* is enabled.



If the Coloring Mode is not enabled and if either layer or object map file has color-related entries, a warning is displayed and the color-related entries are not visible in the layer and object map file editors. Also, these entries are ignored during translation. These entries are visible only when Coloring Mode is enabled.

Delete Row

Use the *Delete Row* button to delete an existing object mapping record.

Load

Use the *Load* button to load the object map file. An object map file is used for preserving OpenAccess objects. If you specify an invalid file name, a warning message is generated and the file name is ignored.

The path specified for the object map file can be absolute or relative to the run directory. For more information, see Object Map File.

Save As and Exit

Use the *Save As* and *Exit* button to save the object mapping information in a new file and exit the object map file editor.

Save and Exit

Use the *Save and Exit* button to save the object mapping information in the current file and exit the object map file editor.

Design Data Translators Reference

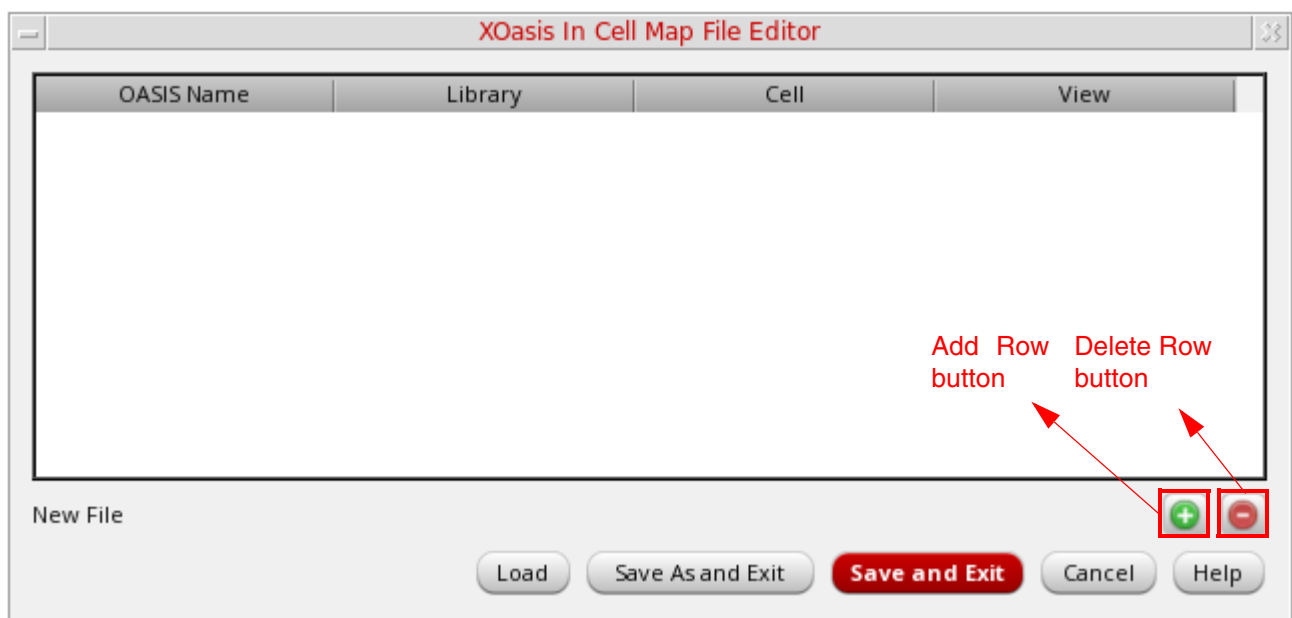
Design Translation Using XOasis Translator

Cancel

Use the *Cancel* button to exit the XOasis In Object Map File Editor form without saving any changes.

XOasis In Cell Map File Editor Form

This section describes options on the XOasis In Cell Map File Editor form that is displayed when you click the *Editor* button for the Cell Map File Name option.



Add Row

Click the *Add Row* button to create a new cell mapping record.

For more information on columns in a cell mapping record, see Cell Name Map File.

OASIS Name

Use the *OASIS Name* field to specify a name for the OASIS file that you want to translate to OpenAccess.

Library

Use the *Library* field to select the library that you want to translate to OpenAccess. The *Library* drop-down list lists all the available libraries defined in `cds.lib` as well as `XOASIS_TARGET_LIB`.

Cell

Use the *Cell* field to specify the cell name that you want to translate to OpenAccess.

View

Use the *View* field to specify the view name of the specified cell that you want to translate to OpenAccess.

Note: To add or modify the values in any of these fields double-click the cells.

Delete Row

Use the *Delete Row* button to delete an existing mapping record.

Load

Use the *Load* button to load a new cell name map file. A cell name map file contains a table that maps structure names in the OASIS file to cellview names in the OpenAccess database.

The path specified for the cell name map file can be absolute or relative to the run directory.

Save As and Exit

Use the *Save As and Exit* button to save the object mapping information in a new file and exit the cell map file editor.

Save and Exit

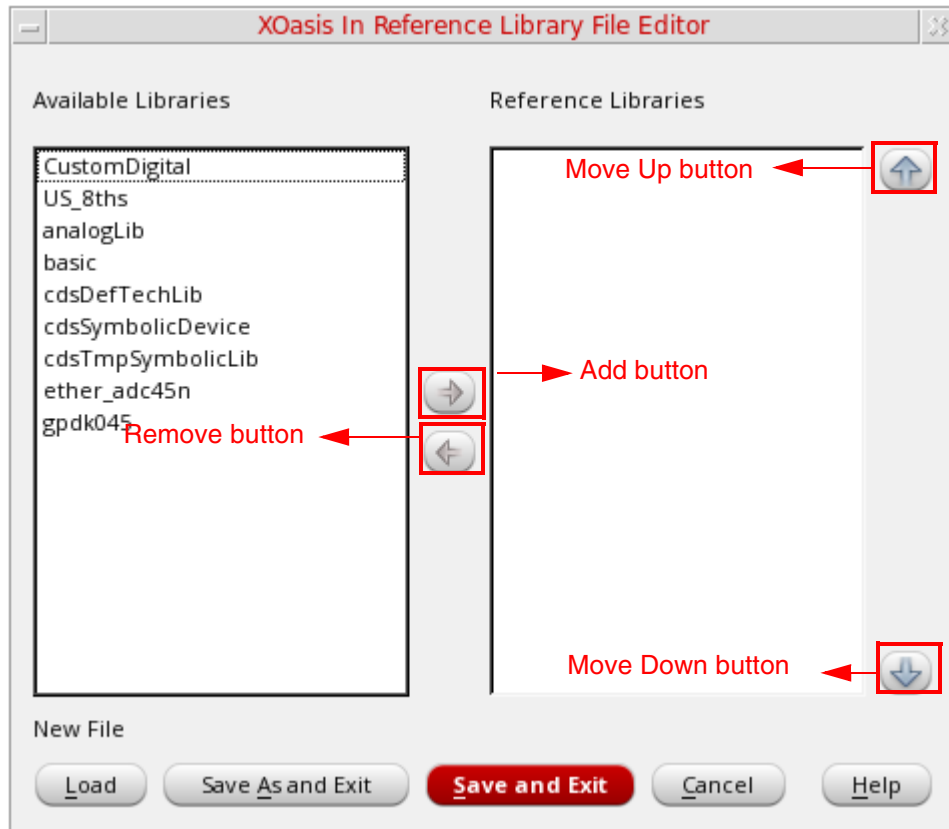
Use the *Save and Exit* button to save the object mapping information in the current file and exit the cell map file editor.

Cancel

Use the *Cancel* button to exit the XOasis In Cell Map File Editor form without saving any changes.

XOasis In Reference Library File Editor Form

This section describes options on the XOasis In Reference Library File Editor form that is displayed when you click the *Editor* button for the Ref Lib File Name option.



Note: If you want to specify the target library name as reference library then you need to use the `XOASIS_TARGET_LIB` variable in the *Available Libraries* field.

Add

Use the *Add* button to add libraries to the *Reference Libraries* field.

The *Reference Libraries* field is used to specify logical library names that you want to consider as reference libraries. The libraries specified in this field should be present in the `cds.lib` file. Otherwise, they are not considered as reference libraries.

During XOasis In, when you search for master cellviews of instances, reference libraries are searched in the same order as that specified in the *Reference Libraries* field. If any of the specified libraries do not exist in the `cds.lib` file, an error message is displayed.

Remove

Use the *Remove* button to remove libraries from the *Reference Libraries* field.

Move Up

Use the *Move Up* button to move up selected reference libraries in the Reference Libraries list on the right. This helps you arrange reference libraries as per your order of preference.

Move Down

Use the *Move Down* button to move down selected reference libraries in the Reference Libraries list on the right. This helps you arrange reference libraries as per your order of preference.

Load

Use the *Load* button to load the text file containing the list of reference libraries. When you specify the text file, the path and the text filename are displayed in the blank field above the *Load* button.

Save As and Exit

Use the *Save As and Exit* button to save the reference library information in a new file and exit the reference library map file editor.

Save and Exit

Use the *Save and Exit* button to save the reference library information in the current file and exit the reference library map file editor.

Cancel

Use the *Cancel* button to exit the XOasis In Reference Library File Editor form without saving any changes.

XOasis In Property Map File Editor Form

This section describes options on the XOasis In Property Map File Editor form that is displayed when you click the *Editor* button for the Prop Map File Name option.

The screenshot shows the 'XOasis In Property Map File Editor' window. It features a table with columns for 'Property ID', 'Object Type', and 'Property Name'. Below the table, there are 'Add Row' and 'Delete Row' buttons. Further down, there is a 'New File' section with a 'Property Value Only' checkbox and a 'Property Separator' field. At the bottom, there are 'Load', 'Save As and Exit', 'Save and Exit', 'Cancel', and 'Help' buttons.

Add Row

Click *Add Row* button to create a new property mapping record.

For more information on columns in a property mapping record, see Property Map File.

Property ID

(Optional) Use the *Property ID* field to specify an ID for the OASIS file. This ID is used to import GDSII-Stream-style properties from the OASIS file. The ID can be any number from 0 to 125.

Note: GDSII-Stream-style property is used for compatibility with GDSII Stream properties in OASIS files. By default, an OASIS file stores the properties in the `property-name:property-value` format. However, if the field *Property ID* is provided, the properties are stored similar to GDSII Stream format, where a property is stored for the specified property ID.

Object Type

Use the *Object Type* option to select an object that you want to translate to OpenAccess.

Property Name

Use the *Property Name* field to specify the property name that you want to map to OpenAccess.

Note: To add or modify the values in any of the above mentioned fields, double-click in the corresponding cells.

Property Value Only

Use the *Property Value Only* format specifier to import the property value of a Property ID from the Oasis file as the value of the specified propertyName of an objectType.

Default value: Deselected

Property Separator

Use the *Property Separator* format specifier to specify a property separator. If you specify a property separator in the property map file, XOasis In ignores the property separator defined in the form.

Default value: The default property separator is a comma ‘,’.

Note: Values for *Property Value Only* and *Property Separator* are populated using the values specified in the special option line in the Property Map File.

Delete Row

Click the *Delete Row* button to delete an existing property mapping record.

Load

Use the *Load* button to load the property map file. A property map file is used for preserving properties associated with OpenAccess objects. If you specify an invalid file name, a warning message is generated and the file name is ignored.

The path specified for the property map file can be absolute or relative to the run directory. For more information, see Property Map File.

Save As and Exit

Use the *Save As and Exit* button to save the property map information in a new file and exit the property map file editor.

Save and Exit

Use the *Save and Exit* button to save the property map information in the current file and exit the property map file editor.

Cancel

Use the *Cancel* button to exit the XOasis In Property Map File Editor form without saving any changes.

Design Data Translators Reference

Design Translation Using XOasis Translator

XOasis In Option Names in GUI and Template File

The following table provides a list of all the XOasis In GUI options and their corresponding names in the template file. To view the sample template file, see [XOasis In Template File](#).

Name in Template File	XOasis In GUI Options
arrayInstToScalar	<u>Convert Arrays to Scalar Instances</u>
attachTechFileofLib	<u>Attach Tech Library</u>
case	<u>Cell Case Sensitivity</u>
cellMap	<u>Cell Map File Name</u>
checkPolygon	<u>Report Bad Polygons and Paths</u>
compress	<u>Compress</u>
compressLevel	<u>Compression Level</u>
convertPathToPathSeg	<u>Convert Paths to Pathsegs</u>
dbuPerUU	<u>DB Units Per User Units</u>
detectOverflow	<u>Detect Overflow</u>
enableColoring	<u>Coloring Mode</u>
enableIncrementalImport	<u>Enable Incremental Import</u>
enableLocking	<u>Enable File Locking</u>
enableNegativeViaExt	<u>Enable Negative Extension</u>
excludeMapToVia	<u>Exclude Map To Via File</u>
hierDepth	<u>Translation Depth</u>
ignoreObjectProp	<u>Ignore Object Properties</u>
ignoreZeroWidthPath	<u>Ignore Zero Width Paths</u>
infoToWarn	<u>Convert Info Message IDs to Warnings</u>
keepOasisCells	<u>Keep OASIS Cells</u>
layerMap	<u>Layer Map</u>
library	<u>Library*</u>
loadTechFile	<u>Load ASCII Tech File</u>
logFile	<u>Log File</u>

Design Data Translators Reference

Design Translation Using XOasis Translator

Name in Template File	XOasis In GUI Options
maxCellsInTargetLib	<u>Maximum Cells in Target Lib</u>
mergeUndefPurposToDrawing	<u>Merge Purposes to Drawing</u>
noDetectVias	<u>Do Not Detect Via</u>
noInfo	<u>Disable Info Message IDs</u>
noWarn	<u>Disable Warning Message IDs</u>
oasisFile	<u>OASIS File*</u>
objectMap	<u>Object Map</u>
overflowCellNameSuffix	<u>Overflow Cell Name Suffix</u>
overflowViewName	<u>Overflow View Name</u>
overwriteRootCells	<u>Overwrite Root Cells</u>
preservePinAttr	<u>Preserve Pin Attributes</u>
propMap	<u>Prop Map File Name</u>
propSeparator	<u>Property Separator</u>
refLibList	<u>Ref Lib File Name</u>
replaceBusBitChar	<u>Replace [] with <></u>
runDir	<u>Run Directory</u>
scale	<u>Design Scale Factor</u>
textHeight	<u>Text Height</u>
showCompletionMsgBox	<u>Show Completion Message Box</u>
skipUndefinedLPP	<u>Skip Undefined LPPs</u>
snapToGrid	<u>Snap to Grid</u>
summaryFile	<u>Summary File</u>
techRefs	<u>Tech Refs</u>
templateFile	<u>Template File</u>
textCase	<u>Text Case Sensitivity</u>
textFont	<u>Text Font</u>
textOrientation	<u>Text Orientation</u>

Design Data Translators Reference

Design Translation Using XOasis Translator

Name in Template File	XOasis In GUI Options
topCell	<u>Top Cell</u>
translateUnmappedLPP	<u>Translate Unmapped LPPs</u>
userSkillFile	<u>User SKILL File</u>
useExistingViaDefsForDetection	<u>Use stdViaDefs from Technology Library</u>
verbose	<u>Generate Detailed Log and Summary Files</u>
viaMap	<u>Via Map File</u>
view	<u>View</u>
virtualMemory	<u>Import OASIS to Virtual Memory</u>
WarnToErr	<u>Convert Warning Message IDs to Errors</u>
wildCardInCellMap	<u>Wild Card in Cell Map</u>
writeMode	<u>Write Mode</u>

* Mandatory parameter

Note: You can include any UNIX environment variable in template file options. For example, if you use \$TECH_DIR, \$CDS_WORKAREA, \$USER variables in a template file and the variables are correctly resolved by XOasis.

You can also access and input values in XOasis In GUI using SKILL commands. To populate values in the *XOasis In* form, you need to enter the xoasInSetField command in CIW. Similarly, to access values from the *XOasis In* form, you need to enter the xoasInGetField command in CIW.

Note: All the options listed above are valid for xoasInSetField, xoasInGetField, template file, and oasisInKeys. All the options except `virtualMemory` and `showCompletionMsgBox` are valid for command line executable or batch process.

Seeding the XOasis In GUI

You can use the following methods to set default values for a user:

■ Using the XOASIS_IN environment variable

You can assign the XOasis In template file as a value of the XOASIS_IN shell environment variable.

Design Data Translators Reference

Design Translation Using XOasis Translator

For example:

```
envSetVal( "xoasis" "XOASIS_IN" 'string "demo.txt" )
setenv XOASIS_IN "demo.txt":
```

Sample XOasis In option template file:

```
=====
pinAttNum          "100"
summaryFile        "xOasisIn.sum"
replaceBusBitChar
=====
```

■ Using oasisInKeys SKILL list

You can define a set of option values to load into the XOasis In form in the `oasisInKeys` SKILL list. For this, define the pre-seeding options in the `.cdsinit` file because the system automatically loads this file when you start Virtuoso.

A sample `oasisInKeys` SKILL list is shown below:

```
;*****
; Copyright (C) 2013 - 2018                                     *
; Cadence Design Systems Inc. All Rights Reserved.             *
;*****
;
;*****
;
; filename : oasisIn.il
;
;*****

oasisInKeys=list(nil
'hierDepth          "32767"          ;-(0-32767) - Hierarchy Depth Limit
'maxCellsInTargetLib "30000"         ;-(Maximum Cells in Target Library
'textOrient         "R0"             ;-( R0 | R90 | R180 | R270 | MY |, MYR90 MX | MXR90 - Default
Text Orientation
'arrayInstToScalar  nil              ;-(Convert AREF to Scalar Instances
'attachTechFileOfLib ""              ;-(Attach Techfile of Library
'case               "preserve"        ;-(preserve | upper | lower - Case Sensitivity
'cellMap            ""               ;-(Cell Map File Name
'checkPolygon       nil              ;-(Report Bad Polygons
'compress           nil              ;-(Allow libraries to be compressed
'compressLevel      "1"              ;-(Defines the compression level to use (default: 1)
'convertPathToPathSeg nil            ;-(Convert Two Point Path(s) to PathSeg(s)
'detectOverflow     nil              ;-(Enable overflow cellview creation on reaching maximum limit
of shapes in a cellview
'dbuPerUU          "1000"            ;-(DB Units Per User Units
'enableLocking      nil              ;-(Enable File Locking
'enableColoring     nil              ;-(Enable Coloring Support
'enableIncrementalImport nil          ;-(Enable incremental import of multiple OASIS Files
'enableNegativeViaExt nil            ;-(Enable Negative Extension for Via Detection
'excludeMapToVia    ""               ;-(Via exclusion Data file
'ignoreZeroWidthPath nil            ;-(Ignore Zero Width Path(s)
'ignoreObjectProp   nil              ;-(Ignore Object Properties
'infoToWarn         ""               ;-(Treat the Specified Info Messgaes Ids as the Warnings
'keepOasisCells     nil              ;-(Keep Oasis File Cells
'layerMap           ""               ;-(Layer Map File Name
```

Design Data Translators Reference

Design Translation Using XOasis Translator

```

'library                "oasisInLib"          ;- Library Name
'loadTechFile           ""                    ;- ASCII Technology File Name
'logFile                "oasisIn.log"         ;- Log File Name
'mergeUndefPurposToDrawing nil                ;- Merges Undefined Purposes to "drawing" Purpose
'noDetectVias           nil                    ;- Do Not Detect possible viaDefs from Oasis File and create vias
'noInfo                 ""                    ;- Do not Print the Specified Info Messages Ids
'noWarn                 ""                    ;- Do not Print the Specified Warning Messages Ids
'oasisFile              "test.oasis"         ;- Oasis File Name
'objectMap              ""                    ;- Object Mapping File Name
'overflowCellNameSuffix "_xoasOverflow"       ;- Cell name suffix of overflow cellviews
'overflowViewName       "layout"             ;- View name of overflow cellviews
'overwriteRootCells     nil                    ;- Overwrite top Cells in case write mode is rename
'preservePinAtt         nil                    ;- Preserve Pin Connectivity Information
'propMap                ""                    ;- Property Map File Name
'propSeparator          ", "                 ;- Property Separator Character
'refLibList              ""                    ;- Rerference Library List File
'replaceBusBitChar      nil                    ;- Replace "[" With "<"
'runDir                 "."                   ;- Run Directory
'scale                  "1.00000"            ;- Design Scaling Factor
'showCompletionMsgBox   t                     ;- Show the Completion Message Box
'skipUndefinedLPP       nil                    ;- Skip Creating Undefined Layer-Purpose Pair in Technology
'snapToGrid             nil                    ;- Snap XY to Grid Resolution
'summaryFile            ""                    ;- Summary File Name
'techRefs               ""                    ;- Ordered List of Technology Libraries
'textCase               "preserve"           ;- preserve | upper | lower - Text Case Sensitivity
'textFont               "euroStyle"         ;- euroStyle | gothic | math | roman | script | stick | fixed
| swedish | milSpec
'textHeight             "1.00000"           ;- Text Height (+ve value only)
'topCell                ""                    ;- Top Cell(s) to Translate
'translateUnmappedLPP   nil                    ;- Allow undefined LPP to be translated by incrementally
mapping LPP using layerMap file, SKILL hook piLayerMap, and automatic layer mapping
'useExistingViaDefsForDetection nil          ;- Enable the usage of standard viaDefs from technology database
for via detection
'userSkillFile          ""                    ;- Users' Skill File Name
'verbose                nil                    ;- Generate detailed information in the log and summary files
'viaMap                 ""                    ;- Via Map File Name
'view                   "layout"             ;- View Name(s) to Translate
'virtualMemory          nil                    ;- Do OASIS In into Virtual Memory
'warnToErr              ""                    ;- Treat the Specified Warning Messgaes Ids as the Errors
'wildCardInCellMap      nil                    ;- Wild Card in Cell Map
'writeMode              "overwrite"         ;- Write mode of existing cells
)

```

It is possible to combine both the pre-seeding methods but Cadence recommends to avoid using both methods at the same time. If you combine both the methods, `oasisInKeys` SKILL list takes precedence over template file.

To prepare a SKILL list for XOasis Out, see [Seeding the XOasis Out GUI](#).

Handling Non-Manhattan Angle Rotation and Magnification

When the source OASIS file contains SREF/AREF with either non-Manhattan angle rotation or non-default magnification, xOasis In translates and generates variant of the original master cell. The instance in the translated OA design refers to the variant, instead of the original master cell.

Design Data Translators Reference

Design Translation Using XOasis Translator

Each variant has the same content as the original cell, except that the content is either rotated or magnified, or both. If original cell contains instances or vias, variants are generated recursively for their masters. The cell name of the variant is different from the original master cell name. The variant name follows the below convention:

```
variantName = originalCellName[_m<mag>][_a<angle>][_v<version>]
```

Here, the *mag* and *angle* are floating-point values written in the `printf` format of `.15g`. The suffix, *version* is appended to the name if a cellview with the same name already exists in the library. In case of such a conflict, versions will be generated regardless of the value of `-writeMode` option.

Additionally, AREF with non-Manhattan angle rotation is translated as an array of scalar instances. These instances refer to the rotated (and maybe further scaled) variant. Accordingly, the array itself is rotated and scaled.

Finally, when the via-detection option is enabled, REFs that either point to any-angle or magnified master, or both, are not detected as vias. The `-detectVias` option has no effect on such rotated or magnified variants.

OASIS to OpenAccess Mapping

This section describes the mapping among OASIS and OpenAccess constructs. The XOasis In translator performs semantic checking while processing the records contained in an OASIS file.

Terminology Mapping

The following table includes information about Terminology mappings between OASIS and OpenAccess constructs:

OASIS Construct	OpenAccess Equivalent
CELL	oaBlock
PLACEMENT	oaInst
PROPERTY	oaProp
LAYER	oaLayerNum
DATATYPE	oaPurposeNum

Modal Variables

The OASIS standard defines Modal variables for compaction purposes. The XOasis translator supports Modal variables implicitly. For more information on Modal variables, refer to the OASIS documentation.

Mapping OASIS Records to OpenAccess Objects

The following table includes information about mappings between OASIS records and OpenAccess objects:

OASIS Record	OpenAccess Equivalent	Description
LAYERNAME	layer/datatype and layer/texttype	The OASIS standard does not utilize layer name anywhere else in the standard.
CELL	oaBlock in an oaDesign	
Cellname	oaBlock	The reference numbers can be implicit or explicit; One CELLNAME record cannot use an implicit as well as explicit reference.
TEXT	oaText	As per OASIS standard: ■ The TEXT may reference a TEXTSTRING by reference number or the text may be specified in the record ■ The x and y coordinates are signed integers representing either the absolute or relative location of the text element.
TEXTSTRING	oaText	The reference number is an unsigned integer which is either implicitly or explicitly assigned to the cell. Implicit assignment uses a sequential numbering scheme starting with 0 and is incremented for each successive TEXTSTRING record
RECTANGLE	oaRect	If the rectangle is a square, the record can further compresses the data.
POLYGON	oaPolygon	
TRAPEZOID	oaPolygon	A TRAPEZOID record represents a trapezoid figure, that is, a polygon with four vertexes having at least two opposite sides parallel and parallel to either the x- or the y- axis. Note: A trapezoid can also specify a rectangular shape and could be mapped to an oaRect or an oaPathSeg

Design Data Translators Reference

Design Translation Using XOasis Translator

CTRAPEZOID	oaPolygon	A CTRAPEZOID record represents a compact representation by assuming that two sides of TRAPEZOID are parallel to either the X or Y axis and the remaining two sides form either a 45° or 90° angle with them. The CTRAPEZOID can represent a rectangle, square, or triangle.
CIRCLE	oaEllipse	The CIRCLE record specifies the radius from which the bounding box of ellipse is calculated for creating the oaEllipse.
PLACEMENT		A PLACEMENT record describes one or more instances of the referenced cell within the current cell. X and y coordinates represent either the absolute or relative location of the placement. There are two ways to specify the placement: ■ The first form assumes a magnification value of 1.0 and a counter-clockwise rotation which is an integral multiple of 90°. ■ The other form allows specification of a magnification and angle.
PROPERTY		A property may be associated with the entire OASIS file, a CELL, a PLACEMENT, an element (such as a CIRCLE), or a name (such as a CELLNAME) record. A PROPERTY record specifies the number of property value fields in the record and whether the record contains a standard or non-standard (user) property. PROPERTY records that are placed directly after the START record are associated globally with the entire OASIS file.
PROPNAME		A PROPNAME record associates the name of a property with a unique reference number. This approach enables PROPERTY records to avoid redundancy by not storing the text of the property name.

Mapping Path

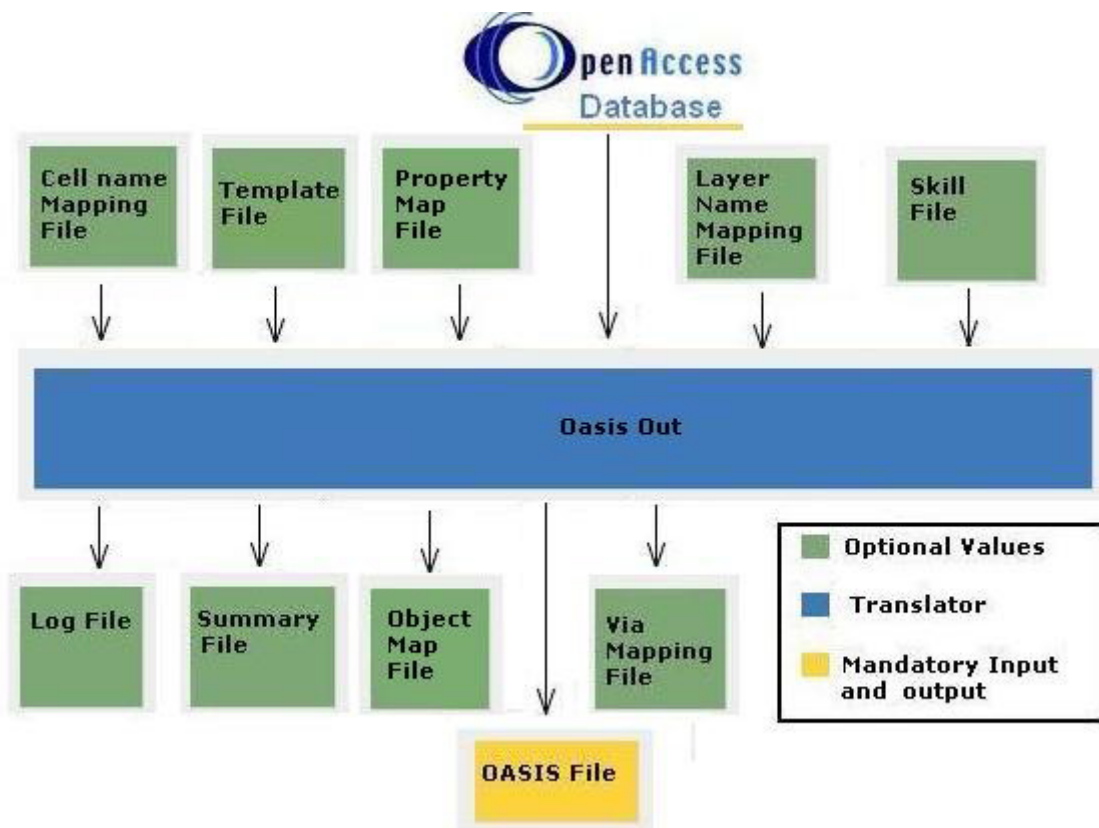
The following table defines the mapping among OASIS PATH and OpenAccess path:

OASIS PATH	OpenAccess Path
Flush extension	oacTruncatePathStyle
Half-width extension	oacExtendPathStyle
Explicit path extension	oacVariablePathStyle

Design Translation During XOasis Out

XOasis Out Use Model

XOasis Out translates designs from the OpenAccess database to the OASIS format. The inputs to the XOasis Out process are the OpenAccess database, XOasis Out options, the optional mapping files, and the user-defined SKILL files. Using these inputs, XOasis Out generates an OASIS file, a log file, and an optional summary file. The XOasis Out use model is shown below:



The following table provides generic information about input and output files during XOasis Out.

File Type	Description
OASIS File	The output OASIS file.

Design Data Translators Reference

Design Translation Using XOasis Translator

File Type	Description
Template File	A file that contains the parameter settings you want to use to run XOasis Out. In the XOasis Out template file, '\$' is considered as a special character because it is a UNIX environment variable. However, if you want '\$' to be treated as a normal character, you can escape it by using '\' before it.
SKILL File	A file used to map cell names, fonts, and layer names.
OpenAccess database	The database you want to translate to a OASIS file.
Log file	A file used for recording the translation process. It also keeps a log of all the messages.
Summary File	A file that has a summary of the translated design.

For details on map files, see [Design Translation During XOasis In](#).

Once you have selected the option to run XOasis Out, click the *OK* button. The *XOasis Out* form will be displayed.

XOasis Out Form

You can export the OASIS file from the GUI by choosing the *File–Export–OASIS* option from the CIW. This method will enable you to display the *XOasis Out* form.

The screenshot shows the 'XOasis Out' dialog box. A red box highlights the 'OASIS File (required)' and 'Library (required)' fields, with an arrow pointing to them from the text 'Mandatory Options'. Another red box highlights the 'Layer Map' section, which is expanded to show 'File Name', 'Translate Unmapped LPP', and 'Use Automatic Layer Mapping' checkboxes, with an arrow pointing to it from the text 'Expand to view additional options'. The 'Layer Map' section is also highlighted by a red box. The 'Object Map' section is also visible. At the bottom are buttons for 'Translate', 'Apply', 'Cancel', 'Reset All Fields', 'More Options', and 'Help'.

For a list of equivalent template file options, see [XOasis Out Names in GUI and Template File](#).

The XOasis Out form comprises mandatory options that are essential to run the translation. The form also features a few other most commonly used options, such as Layer Map and Object Map.

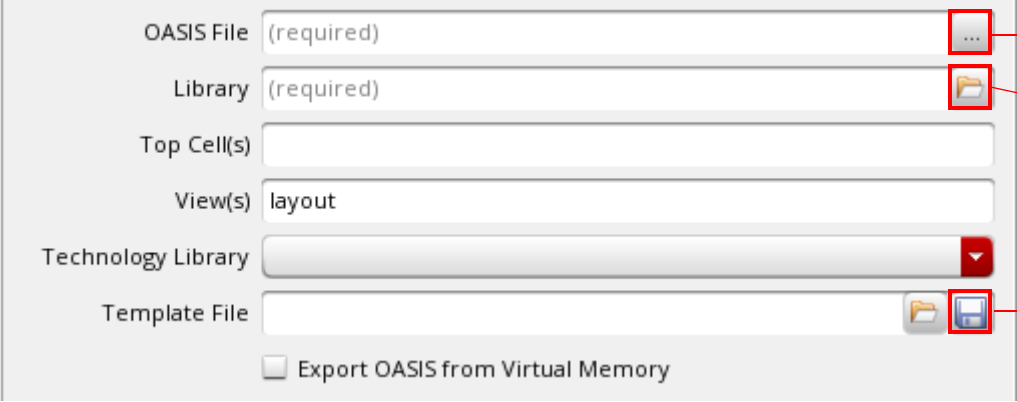
The following sections on the main form can be expanded to view additional options:

- [Layer Map](#)
- [Object Map](#)
- [Log File](#)

Design Data Translators Reference

Design Translation Using XOasis Translator

Discussed below are XOasis Out options available on the main form.



The screenshot shows the XOasis Translator main form with the following fields and buttons:

- OASIS File** (required): A text field with a **Browse button** (three dots icon) to its right.
- Library** (required): A text field with a **Library Browser button** (folder icon) to its right.
- Top Cell(s)**: A text field.
- View(s)**: A text field containing the value "layout".
- Technology Library**: A dropdown menu with a red arrow icon.
- Template File**: A text field with a **Save button** (floppy disk icon) to its right.
- ☐ **Export OASIS from Virtual Memory**: A checkbox at the bottom.

Red arrows point from the text labels to the corresponding buttons in the form.

OASIS File

This is the output OASIS file. Use the *OASIS File* field to specify the name of the output OASIS file. Alternatively, click the *Browse* button to select the OASIS file. The path specified for the output file can be absolute or relative to the run directory. However, if the user has not specified the OASIS filename then it will be automatically generated from the top cell. This is a mandatory field.

Command-line option: `oasisFile`

Library

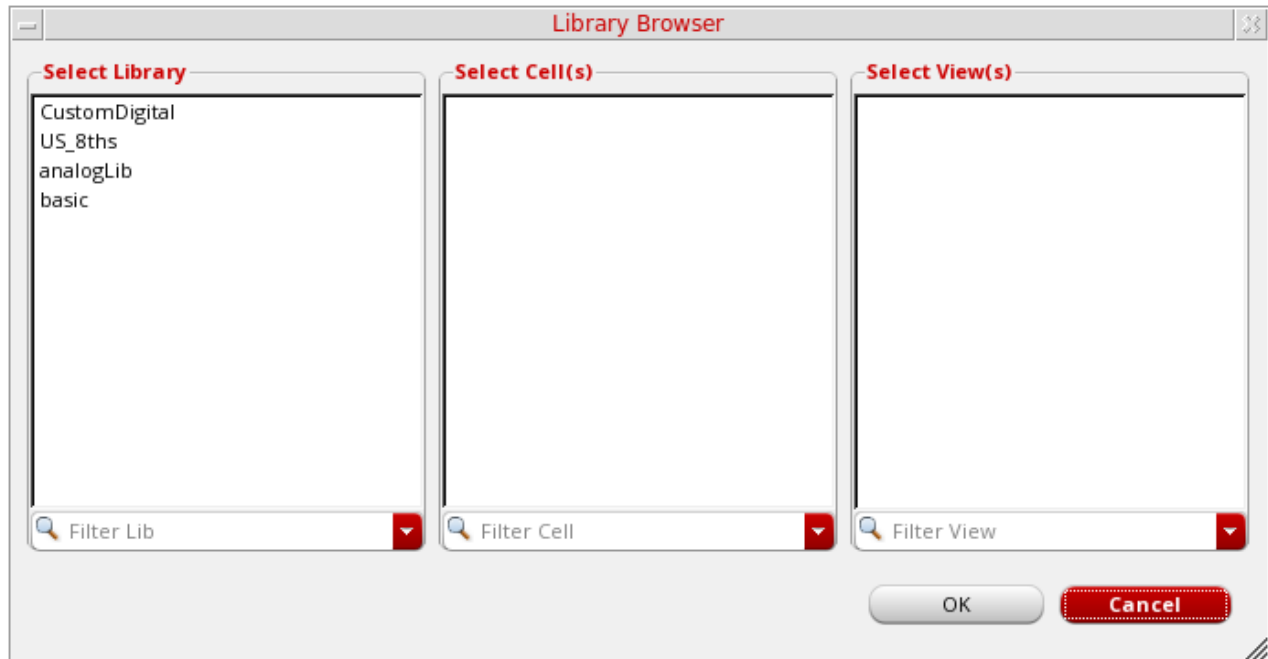
Use the *Library* option to specify library from which a design is to be translated to the OASIS format. The library name should be a logical name supported in the `cds.lib` file. The library name should also follow the naming conventions supported by OpenAccess. You can also click the *Library Browser* button to display the Library Browser.

Command-line option: `library`

Design Data Translators Reference

Design Translation Using XOasis Translator

You can use multiple search strings and wildcard characters, * and ?, to perform your search. The form also saves the search history. Hidden cells are not displayed in this form.



The **Library Browser** dialog box contains three main sections: **Select Library**, **Select Cell(s)**, and **Select View(s)**. The **Select Library** section lists four libraries: CustomDigital, US_8ths, analogLib, and basic. Below each section is a search bar with a magnifying glass icon and a dropdown arrow. The search bars are labeled "Filter Lib", "Filter Cell", and "Filter View". At the bottom right, there are two buttons: "OK" and "Cancel".

Note: To hide cells, in Library Browser, right-click on a cell name and select *Hide Cell*.

Note: You can also specify the `-cdslib` option to access the `cds.lib` file from any other directory. For example, to access the `cds.lib` file from any other location, use the syntax:
`virtuoso -cdslib <cds.lib file path>`

Top Cell(s)

Use the *Top Level Cell(s)* field to specify the name of the cell at the top of the hierarchy that you want to translate to the OASIS file.

By default, all cells in the specified library are translated.

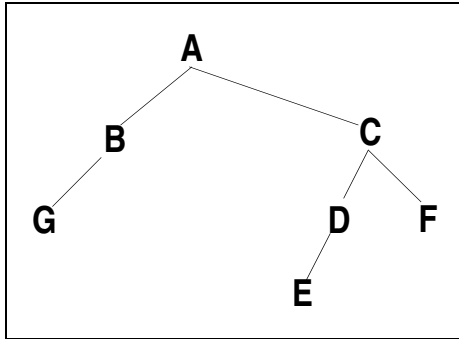
Command-line option: `topCell`

Note: A Pcell supermaster can also be specified at the top level cell. It will be written in the OASIS file during XOasis Out.

After you enter the top level cell in the *Top Level Cell* field, the output file will reflect the filename as `<Top Cell>.<file extension>`. The default file extension name for the output file is `.oasis`. You can change the default file extension name from `.oasis` to any other string by using the `oasisOutExt` SKILL variable.

Example 3-14 Translating cells by using the Top Cell option

Consider the hierarchical structure below where A, B, C, D, E, F, and G are cell names. You want to translate these cells using the *Top Level Cell(s)* field.



When C is specified as the top cell in the *Top Level Cell(s)* field, cells C, D, E, and F are translated. For instance, if you specify the top cell name as B and C, cells B, G, C, D, E, and F are translated. Below is the tabular representation on how the cells will be translated.

Top Level Cell field	Cells translated
B C	B G C D E F
C	C D E F
B D	B G D E

Note: If the *Top Level Cell(s)* field is left blank, then all the cells will be translated.

Important

When using the command line, you need to give the cell names in quotes. For example "B G".

Important

The multiple top cells option is yet to be implemented.

View(s)

Use the *View(s)* field to specify a view name for the top cell. If you do not specify a view name, then the `layout` view is considered as the default view. If you specify any other name as the view name, then the view name that you specify and the hierarchy below is translated.

Design Data Translators Reference

Design Translation Using XOasis Translator

Note: After you open the *XOasis Out* form, the lib/cell/view names of the active layout window will be populated automatically.

Command-line option: `view`

Technology Library

Use the *Technology Library* option for getting information about layers and purposes used in the design. By default, the technology file of the source library is used to get information about the layers and purposes used in the design.

Command-line option: `techLib`

Template File

Use the *Template File* field to specify an existing template file. You can also click the *File Browser* button to select the template file. When this file is loaded, the XOasis Out form is updated with the option settings specified in it.

Command-line option: `templateFile`

Save

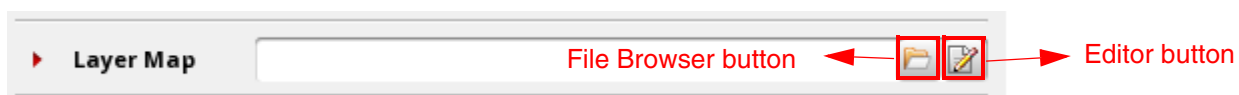
Click the *Save* button to save the current option settings to the template file.

Export OASIS from Virtual Memory

Select the *Export OASIS from Virtual Memory* check box to export OpenAccess designs from the virtual memory to an OASIS file on disk.

Note: If the *Export OASIS from Virtual Memory* check box is selected, information (INFO) and warning (WARNING) messages are not displayed in the CIW. They are logged in the log file.

Layer Map



Design Data Translators Reference

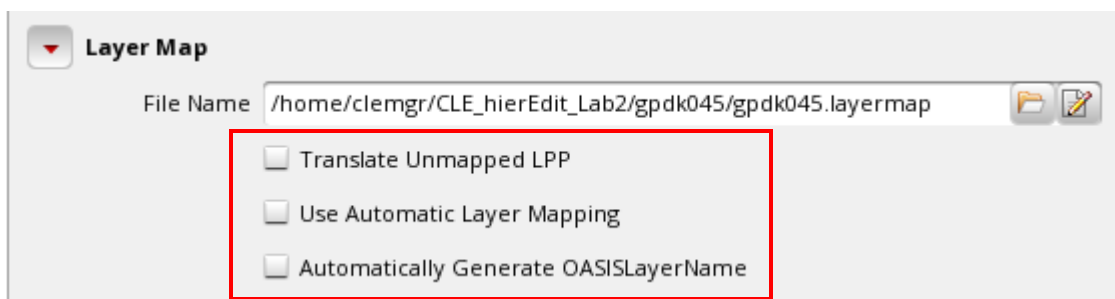
Design Translation Using XOasis Translator

Use the *File Name* field to specify an existing layer map file. You can also click the *File Browser* button to select the layer map file. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `layerMap`

To create new layer mapping records or load a new layer map file, leave the *Layer Map* field blank, and click the *Editor* button to display the XOasis Out Layer Map File Editor Form.

Expand *Layer Map* to view additional options.



The screenshot shows a dialog box titled "Layer Map". It has a "File Name" field containing the path "/home/clemgr/CLE_hierEdit_Lab2/gpdk045/gpdk045.layermap". To the right of the field are two icons: a folder icon and a document icon. Below the field, there are three checkboxes, all of which are unchecked. These checkboxes are enclosed in a red rectangular box. The checkboxes are labeled: "Translate Unmapped LPP", "Use Automatic Layer Mapping", and "Automatically Generate OASISLayerName".

Translate Unmapped LPPs

Use the *Translate Unmapped LPP* option to map unmapped layers. This option works only when a different user-defined layer mapping, layer map file or SKILL function, has been specified. When you enable *Translate Unmapped LPP*, first the user-specified layer mapping is applied, and then, automatic layer mapping is performed to translate unmapped layers. For more information, see Layer Mapping when Translate Unmapped LPP is Selected.

Default value: Deselected

Command-line option: `translateUnmappedLPP`

Use Automatic Layer Mapping

Use the *Use Automatic Layer Mapping* option to map layers automatically. This option works only if no layer mapping has been specified. Otherwise, this option is ignored and layers are mapped with the specified mapping. For more information about automatic layer mapping, see Automatic Layer Mapping in XOasis.

Default value: Deselected

Automatically Generate OASISLayerName

Use the *Automatically Generate OASISLayerName* option to generate `OASISLayerName` for all the entries in the layer map and object map files except for those entries for which the `OASISLayerName` qualifier is already specified. You can use this option to generate `OASISLayerName` for all types of layer mapping, including mapping using layer map file, `userSkillFile`, and automatic layer mapping.

Format of the generated `OASISLayerName` is:

- **LayerMap Format:** `LayerName_purposeName`
- **ObjectMap Format:** `ObjectType_ObjectSubType`

Consider the following entries in a layer map file:

```
Metall  drawing 3  0
```

```
Metall  net      3  0
```

In this example, the generated `OASISLayerName` is `Metall_drawing` for the first entry in the layer map file for the pair `OASISLayer:OASISType`, which is `3:0`.

Command-line option: `autoGenOASISLayerName`

Related Topic

[OASISLayerName](#)

Object Map



Use the *Object Map* field to specify an existing object map file. You can also click the *File Browser* button to select the object map file. Next, click the *Editor* button to view the specified file in editor.

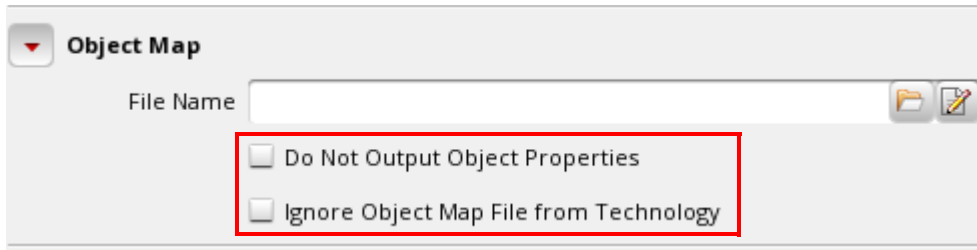
Command-line option: `objectMap`

To create new object mapping records or load a new object map file, leave the *Object Map* field blank, and click the *Editor* button to display the [XOasis Out Object Map File Editor Form](#).

Design Data Translators Reference

Design Translation Using XOasis Translator

Expand *Object Map* to view additional options.



Do Not Output Object Properties

Use the *Do Not Output Object Properties* option to translate OA objects as shapes, but not write the property information to the property attribute 126. If you select the *Do Not Output Properties* option, then although the shapes and OA objects are translated, no property data is written to the OASIS file for OA objects.

Default value: Deselected

Command-line option: `noObjectProp`

Ignore Object Map File from Technology

Use the *Ignore Map File from Technology* to ignore PDK-installed object map file during XOasis Out.

Default value: Deselected

Command-line option: `ignoreObjectMapFromTech`

Log File

Use the *Log File* field to specify log file to be used during XOasis Out translation.



A log file is used to record the translation process steps and the messages generated by XOasis Out. After the translation is completed, the log file is displayed in a separate read-only

Design Data Translators Reference

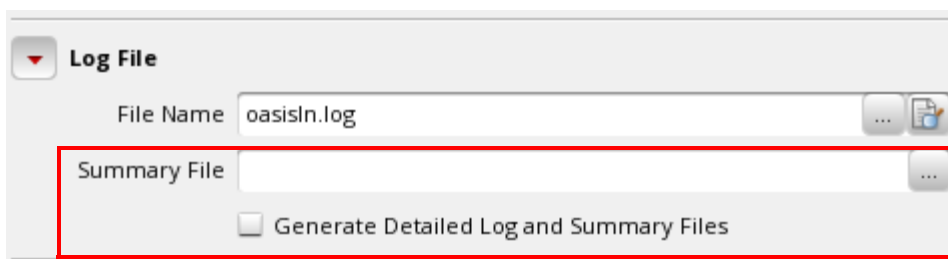
Design Translation Using XOasis Translator

text window. Use the *Log File* field to specify the log filename. Alternatively, click the Browse button to select the log file.

The path specified for the log file should be a valid file path. It can be absolute or relative to the run directory. For more information, see [Sample Log File](#).

Default filename: oasisOut.log

Command-line option: logFile



Summary File

Use the *Summary File* field to specify a file that will provide a summary of the translated design. If you do not specify a filename in the *Summary File* option before translation, the summary file will not be generated.

The path specified for the summary file can be absolute or relative to the run directory. To view the sample summary file, see [Sample Summary File](#).

Command-line option: summaryFile

Generate Detailed Log and Summary Files

Use the *Generate Detailed Log and Summary Files* option to include additional summary information in the XOasis Out log file and XOasis Out summary file.

Command-line option: summaryFile

With this option is enabled, following sections with additional information are added to the XOasis Out summary file:

- **Individual Cell Statistics:** Individual cell statistics for *Basic Objects* and *Advance Objects* is alphabetically sorted order based on Library/Cell/View.

Design Data Translators Reference

Design Translation Using XOasis Translator

- ❑ **Bounding box for the cells:** *Individual Cell Statistics for basic objects* prints additional column titled *bBox*. This column lists the boundary box of the translated cell in the following format:

`((lowerLeft.x lowerLeft.y) (upperRight.x upperRight.y))`

Here, *lowerLeft* and *upperRight* are the lower-left and upper-right coordinates of the boundary box.

Individual Cell Statistics - Basic Objects

Library/Cell/View	Struct Name	Instance	Array	Polygon	Rect	Path	PathSeg	Text	Line	Dot	Arc
Donut	Ellipse	bBox									
LIB/cell1/layout	cell1	0	0	0	1	0	0	0	0	0	0
0		((-10.0000 0.8000)	(-3.5000 3.9000))								
LIB/cell2/layout	cell2	0	0	0	1	0	0	0	0	0	0
0		((-5.4000 -9.3000)	(17.9000 -2.9000))								
LIB/cell3/layout	cell3	0	0	0	1	0	0	0	0	0	0
0		((-14.4000 -7.2000)	(11.5000 5.2000))								
LIB/top/layout	top	5	0	3	7	1	4	0	0	2	0
5	12	((-2073.2000 -5623.0000)	(10231.9700 1410.5100))								

Individual Cell Statistics - Advanced Objects

Library/Cell/View	Struct Name	PRBdy	OtherBdy	AreaBlkg	LayerBlkg	AreaHalo	Row	Marker	CustVia
StdVia									
LIB/cell1/layout	cell1	1	0	0	0	0	0	0	0
LIB/cell2/layout	cell2	0	0	1	0	0	0	0	0
LIB/cell3/layout	cell3	0	0	1	0	0	0	0	0
LIB/top/layout	top	0	0	0	1	0	0	0	0

- **Statistics of Layers:** Layer summary added to this section is sorted in the same order in which layer-purpose pairs are defined in the technology file or the layer pellet.

Statistics of Layers

Cadence Layer	Cadence Purpose	Oasis Layer	Oasis Datatype	Polygon	Rect	Path	PathSeg	Text	Line	Dot	Arc	Donut	Ellipse
Metal1 drawing	7	0	0	4	0	0	0	0	0	0	0	0	0
Metal2 drawing	9	0	3	0	0	0	0	0	0	0	0	0	0
Metal3 drawing	11	0	0	0	0	0	0	0	0	0	0	0	4
Metal4 drawing	31	0	0	0	0	0	0	0	0	0	0	0	8
Metal5 drawing	33	0	0	0	0	0	0	0	0	0	0	5	0
Metal7 drawing	38	0	0	0	1	0	0	0	0	0	0	0	0
Metal8 drawing	40	0	0	0	0	4	0	0	0	0	0	0	0
Metal9 drawing	42	0	0	3	0	0	0	0	0	2	0	0	0

Design Data Translators Reference

Design Translation Using XOasis Translator

- **Missing Cells:** This section lists the cells that are referenced but are missing in the OpenAccess database.

Missing Cells

Library/Cell/View

LIB/missingcell/layout

- **Dropped Layers:** This section lists the layer-purpose pairs for which no mapping is found. Shapes on these layer-purpose pairs are not translated.

Dropped Layers

Cadence Layer	Cadence Purpose
Oxide	drawing
Poly	drawing
Pimp	boundary

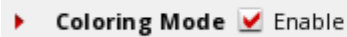
- **Individual Advanced Object Statistics:** This section lists the Place and Route objects translated during XOasis Out. It lists the object-type, sub-object type, Oasis layer name, Oasis Name, layer name (for layerBlockage), datatype, and the count of the objects translated.

Individual Advanced Object Statistics

Object Type	Object Sub-type	Oasis Layer	Oasis Name	Layer	Datatype	Count
Boundary	Area			102	102	6
Boundary	PR			100	100	3
Boundary	Snap			106	106	3
Boundary	Cluster			104	104	3
Row	none			110	110	6
AreaBlockage	placement			114	114	3
AreaHalo	placement			116	116	3
Marker	info			108	108	3
LayerBlockage	routing	M4		112	112	3

Coloring Mode

To enable color mapping during XOasis Out, select the *Enable* check box for the *Coloring Mode* option.



Default value: Selected

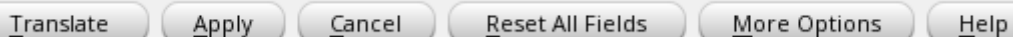
Command-line option: `enableColoring`

Important

The layer map file, which is either specified through the *Layer Map* field or is available in the technology, is required if you select the enable *Coloring Mode* (`-enableColoring`). Otherwise, translation is terminated and an error message is displayed.

In case both *Merge Connected PathSegs* and *Coloring Mode* options are enabled, the connected pathSegs on colorable layers are not merged and each pathSeg is written as a separate PATH record in the OASIS file. The *Merge Connected PathSegs* option is *not considered for the pathSegs on colorable layers*.

If the layer map and object map files do not have any color entry, the `-enableColoring` option is not considered during translation.



Translate

Use the *Translate* button to translate OpenAccess files to the OASIS files. This option, if used without loading the hierarchy, enables faster translation.

If you have opened cellviews that are not yet saved, the XOasis Out GUI displays a warning message to save all the opened cellviews before translating.

Apply

Use the *Apply* button to translate OpenAccess files to the OASIS files. After the translation is completed, the *XOasis Out* form does not disappear.

Cancel

Use the *Cancel* button to exit the *XOasis Out* form.

Reset All Fields

Use the *Reset All Fields* button to set the GUI fields to their default values, which are displayed when the tool is invoked.

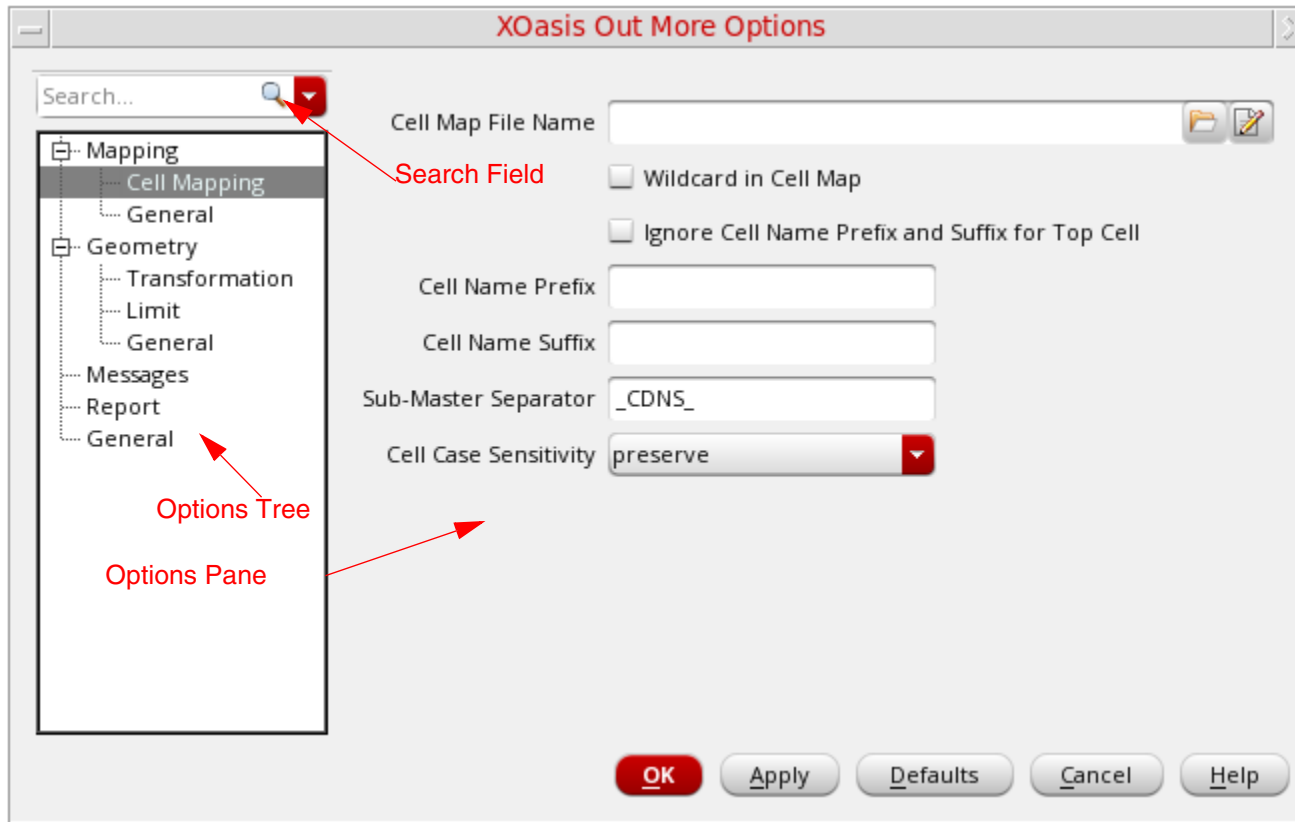
More Options

Use the *More Options* button to display the XOasis Out More Options form that comprises additional options that you can set before starting XOasis Out translation.

Help

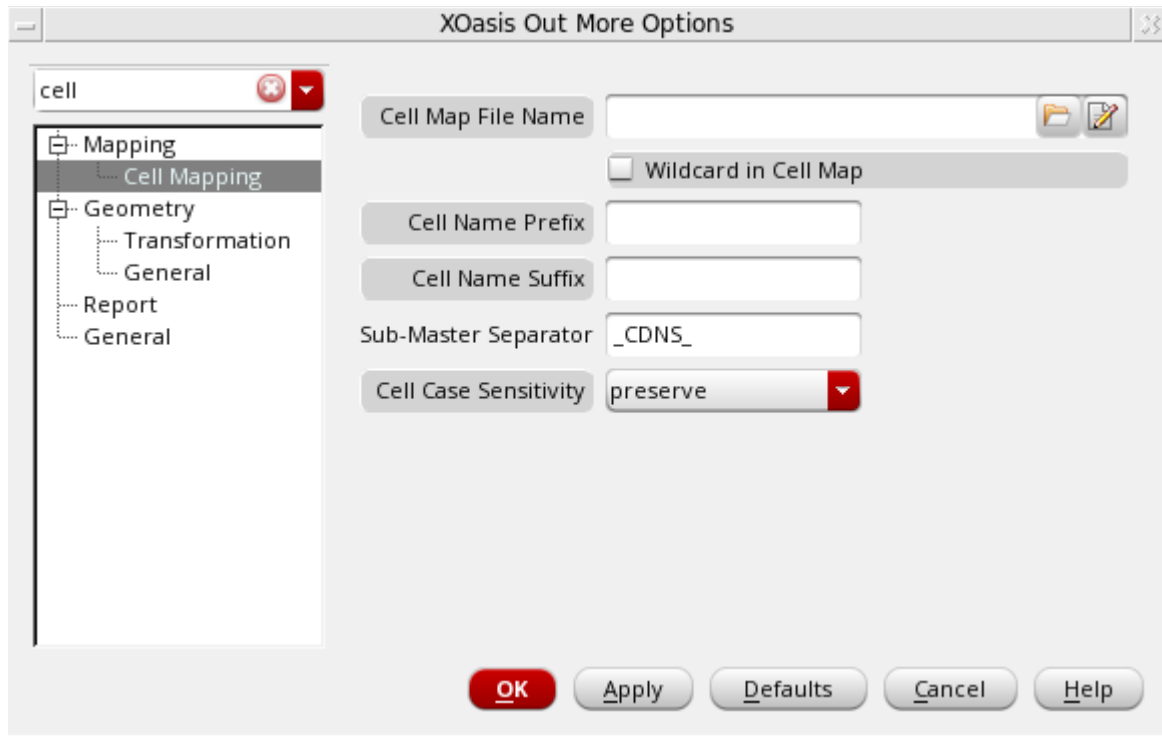
Use the *Help* button to invoke online help.

XOasis Out More Options Form



Search Field

Use the Search field to quickly look for options you want to access or the options related to the value you type in this field. For example, if you type `cell`, only the options related to cell will be visible in the options tree. The related options will also be highlighted in the options pane.



Options Tree

The Options tree enables you to access the following option panes that where you can perform additional settings.

- Mapping Options
 - ☐ [Mapping: Cell Mapping Options](#)
 - ☐ [Mapping: General Options](#)
- Geometry Options
 - ☐ [Geometry: Transformation Options](#)
 - ☐ [Geometry: Limit Options](#)

- ☐ Geometry: General Options
- Messages Options
- Report Options
- General Options

Options Panes

Options panes comprise different options related to option currently selected in the tree.

OK

Click *OK* to save your current settings for the More Options form and exit.

Apply

Click *Apply* to save your settings on the More Options form. When you click Apply, the form is not closed.

Defaults

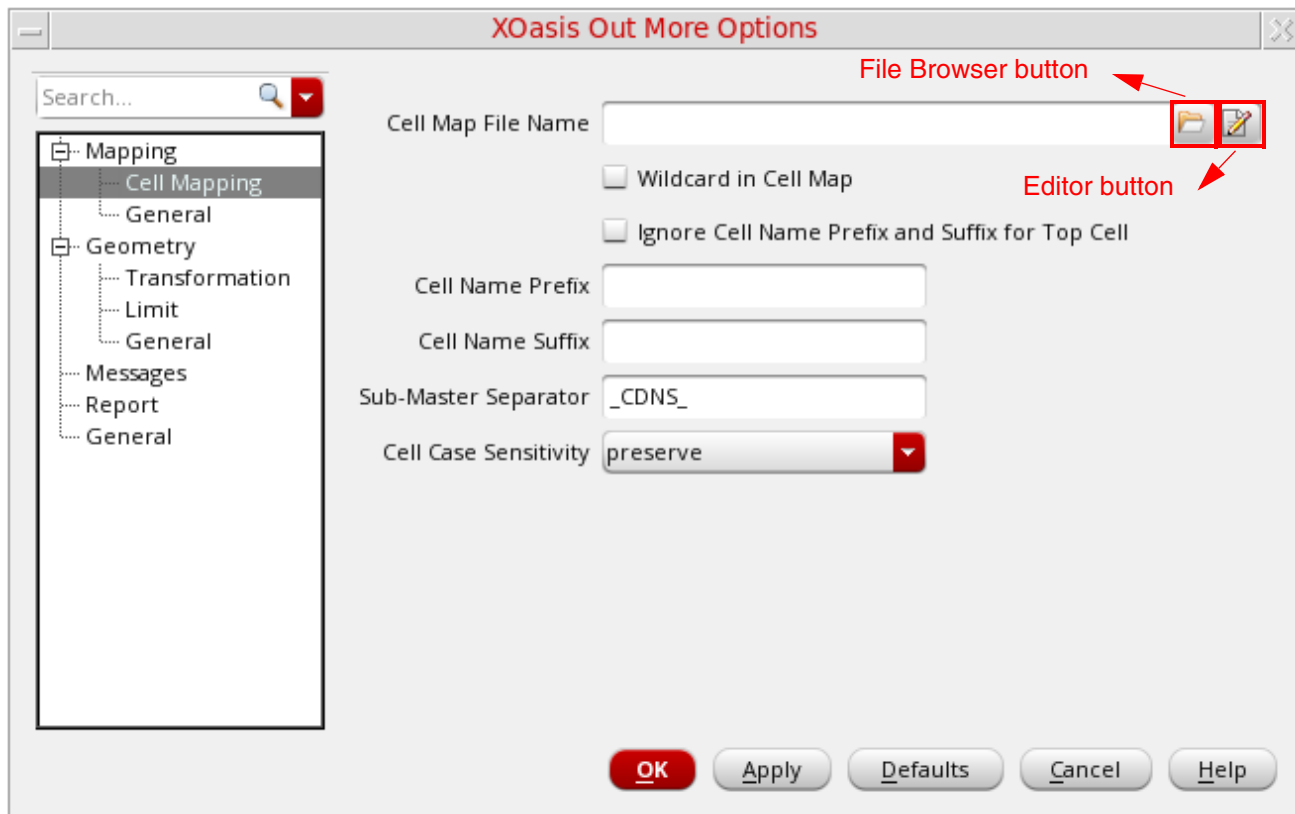
Click *Defaults* to set default values for all options in the More Options form.

Cancel

Click *Cancel* to close the More Options form without saving any changes.

Mapping: Cell Mapping Options

This section describes options on the Mapping: Cell Mapping options pane.



Cell Map File Name

Use the *Cell Map File Name* field to specify an existing cell map file. You can also click the *File Browser* button to select the cell map file. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `cellMap`

To create new cell mapping records or load a new cell map file, leave the *Cell Map* field blank, and click the *Editor* button to display the XOasis Out Cell Map File Editor Form.

Wildcard in Cell Map

Use the *Wildcard in Cell Map* option to use wild card in XOasis Out. The option supports the characters '*' and '?' as wild cards and '/' as the escape character. If you don't use this option, wild card will be considered as a special character during translation.

Design Data Translators Reference

Design Translation Using XOasis Translator

Consider the following example:

* INV_* * INV*_*_ver1

Here, when you use XOasis Out, the created structure will be INV*_NAND_ver1 if the design cellName is INV_NAND. In this example, in INV*_*_ver1, the first '*' is treated as special character and the second '*' as the wild card.

Note: If you don't use *Wild Card in Cell Map*, the default behavior will be same as a normal cell map.

Command-line option: wildCardInCellMap

Ignore Cell Name Prefix and Suffix for Top Cell

Use the *Ignore Cell Name Prefix and Suffix for Top Cell* if you want to ignore the Cell Name Prefix and Cell Name Suffix for top level cell(s).

Default value: Deselected

Command-line option: ignoreTopCellPrefixSuffix

Cell Name Prefix

This option enables you to add a prefix in all the cell names in OA. For example, if a prefix, _pi is added the cellname, *cell11*, then the cellname in OA would be _picell11. The *Cell Case Sensitivity* option will be applied on the modified name.

Command-line option: cellNamePrefix

Cell Name Suffix

This option enables you to add suffix in all the cell names in OA. For example, if a suffix, _si is added the cellname, *cell11* then the cellname in OA would be *cell11_si*. The *Cell Case Sensitivity* option will be applied on the modified name.

Note: If cell name mapping is provided either through a cellMap file or through a SKILL function, *poCellNameMap()*, then the prefix and suffix will be added to names that are already mapped.

If you want to the ignore cell name prefix or suffix for top cell, use the -cellMap option with the -wildCardInCellMap option. The cell map file will look like the following:

Design Data Translators Reference

Design Translation Using XOasis Translator

```
* top layout top
* * layout prefix_*_suffix
```

You can also use the `poCellNameMap` SKILL function to rename the cell based on your requirement. You can specify this SKILL command in a `.il` file and pass it as a value to the `-userSkillFile`.

Command-line option: `cellNameSuffix`

Sub-Master Separator

Use the *Sub-Master Separator* to change the default value of the separator added to Pcells during translation. The default separator is `_CDNS_`. The following naming convention is used:

`pCell/via master name + subMasterSeparator + timestamp + number`

Here,

- ❑ **subMasterSeparator** is configured using the `subMasterSeparator` option
- ❑ **timestamp** is an 11 digit number, where the first nine digits are the number of seconds elapsed since January 1, 1970 and the last 2 digits are micro seconds
- ❑ **number** is an incrementing number.

Command-line option: `subMasterSeparator`

Cell Case Sensitivity

The *Cell Case Sensitivity* option specifies whether XOasis Out changes the case of letters in cell names. XOasis Out uses this option when you do not supply cell name mapping information for the cell. You can map cell names in a cell name map table or in a SKILL file.

preserve

The *preserve* option translates the cell names without changing case. For example, the name `Obj1` remains `Obj1`.

lower

The *lower* option translates all cell names to lowercase. For example, the name `Obj1` becomes `obj1`.

Design Data Translators Reference

Design Translation Using XOasis Translator

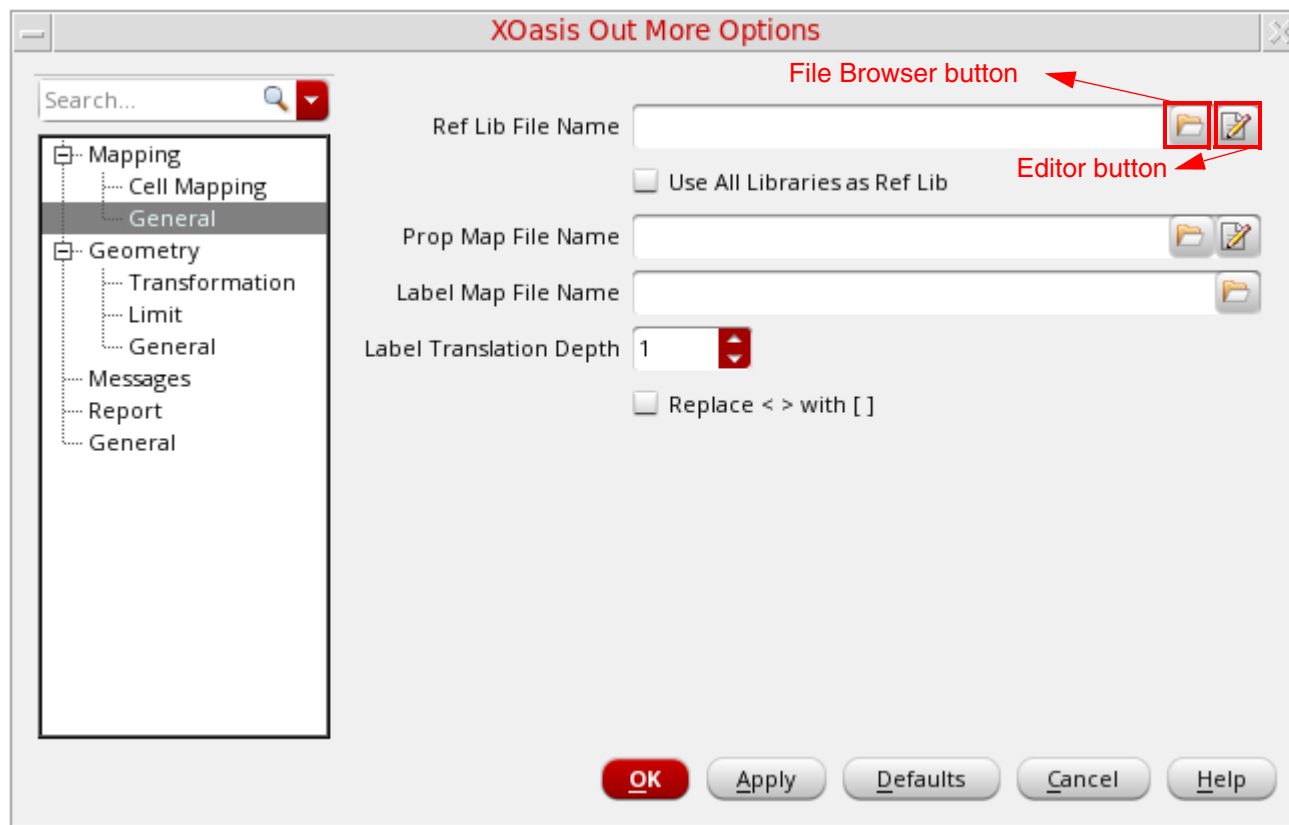
upper

The *upper* option translates all cell names to uppercase. For example, the name `Obj1` becomes `OBJ1`.

Command-line option: `case`

Mapping: General Options

This section describes options on the Mapping: General options pane.



Ref Lib File Name

Use the *Ref Lib File Name* field to select a library name from the list of available libraries that are to be used as a reference libraries. You can also click the *File Browser* button to select the file containing list of reference libraries. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `refLibList`

Design Data Translators Reference

Design Translation Using XOasis Translator

To select a library name from the list of available libraries to be used as a reference libraries, leave the *Ref Lib File Name* field blank, and click the *Editor* button to display the XOasis Out Reference Library File Editor Form.

Use All Libraries as Ref Lib

Use the *Use All Libraries as Ref Lib* option to select all available libraries as reference libraries.

Note: If you invoke XOasis Out from the command line, you can write the list of reference libraries in a text file and provide this filename with the `refLibList` option. If you want to consider all the libraries defined in the `cds.lib` file as reference libraries, use the `XOASIS_CDS_LIB` keyword. You can provide this keyword as a value of the `reflibList` option from the command line or write it in a text file containing the list of libraries.

Prop Map File Name

Use the *Prop Map File Name* field to specify an existing properties mapping file. You can also click the *File Browser* button to select the properties mapping file. Next, click the *Editor* button to view the specified file in editor.

Command-line option: `propMap`

To create new properties mapping records or load a new properties mapping file, leave the *Prop Map File Name* field blank, and click the *Editor* button to display the XOasis Out Property Map File Editor Form.

Label Map File

Use the *Label Map File* field to specify a label map file that determines the object type for which labels are created and determines the mapping of shapes on an `oaLayer` and `oaPurpose` pair to a OASIS layer and OASIS data type pair.

The object type specified in the label map file is either a terminal or a net.

Command-line option: `labelMap`

Label Translation Depth

Use the *Label Translation Depth* field to specify the depth of the design hierarchy for which labels are generated. If you leave this field as is (default value 1), labels are generated for only the top level of the design.

Design Data Translators Reference

Design Translation Using XOasis Translator

Note: The `-labelDepth` option works only when the `-labelMap` option is specified.

The term *Labels* in the *Label Translation Depth* option name refers to the `net/term/pin` name. This is different from the label created using the *Layout XL > Create > Label* command.

Value values: 1 to 32767

Default value: 1

Command-line option: `labelDepth`

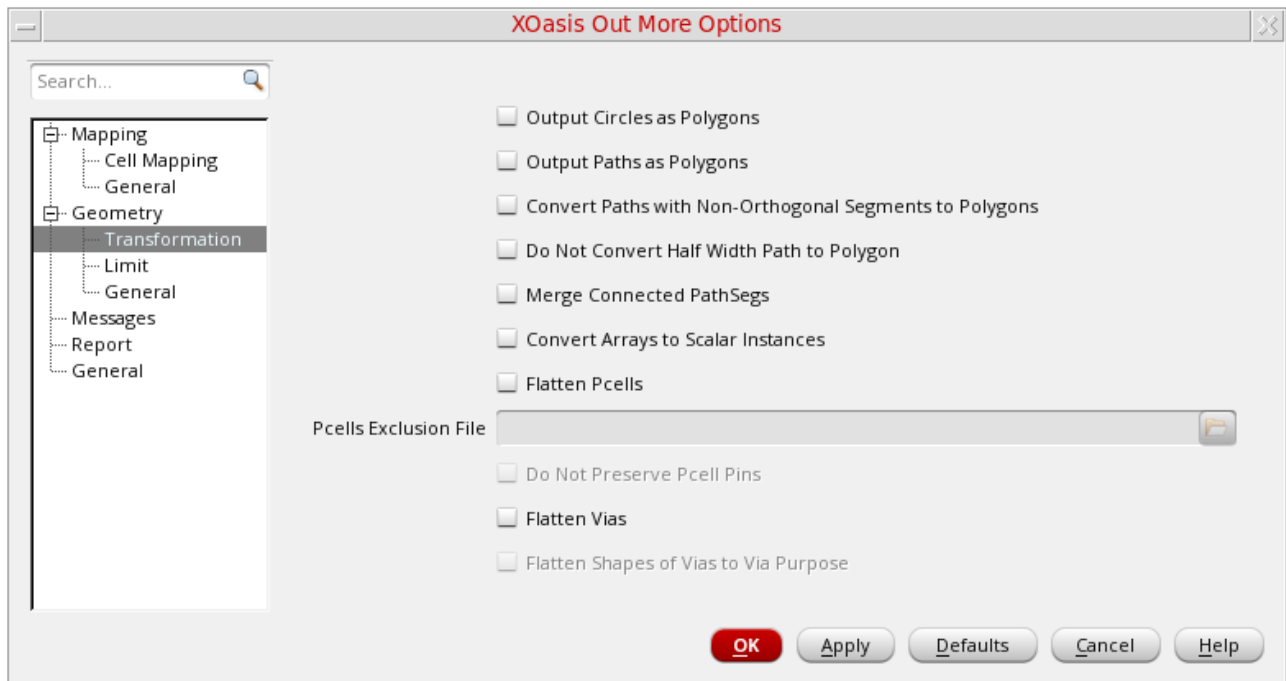
Replace <> with []

The *Replace <> with []* option replaces the character `<` with `[` and `>` with `]` in text records during the translation from the DFII library to an OASIS file.

Command-line option: `replaceBusBitChar`

Geometry: Transformation Options

This section describes options on the Geometry: Transformation options pane.



Output Circles as Polygons

Use the *Output Circles as Polygons* option to translate all circles or ellipses in OpenAccess as polygon records in the OASIS format.

Command-line option: `circleToPolygon`

Output Paths as Polygons

Use the *Output Paths as Polygons* option to translate all paths in OpenAccess as polygon records in the OASIS format.

However, there are the following instances where path is converted as a polygon even if the *path as polygon* option is not used:

- Path is less than 1/2 width
- Path with acute angle
- Path with rounded extension

Command-line option: `pathToPolygon`

Convert Paths with Non-Orthogonal Segments to Polygons

Select the *Convert Paths with Non-Orthogonal Segments to Polygons* check box to export paths with non-orthogonal segments as `BOUNDARY` records. When this check box is not selected, all paths with non-orthogonal segments are translated as `PATH` records.

Default Value: Deselected

Command-line option: `diagonalPathToPolygon`

Do Not Convert Half Width Path to Polygon

Select the *Do Not Convert Half Width Path to Polygon* check box to translate half width path as a path and not as a polygon. When this check box is not selected, all half width paths are translated as `BOUNDARY` records.

Default Value: Deselected

Command-line option: `noConvertHalfWidthPath`

Merge Connected PathSegs

Use the *Merge Connected PathSegs* option to assemble all the connected pathsegs into a single `PATH` record during XOasis Out.

Command-line option: `mergePathSegsToPath`

Convert Arrays to Scalar Instances

Select the *Convert Arrays to Scalar instances* check box to translate an array as an Oasis array (`AREF`) or break an array into individual instances (`SREF`). When you select this option, an array in the OpenAccess database is translated as multiple `SREFs` in the OASIS file. For performance issues related to this option, see [Guidelines for XOasis Out](#).

Note: The *Convert Arrays to Scalar instances* option works when the [*Enable Coloring*](#) option is enabled.

Default value: Deselected

Command-line option: `arrayInstToScalar`

Flatten Pcells

Use the *Flatten Pcells* option if you want to flatten the Pcell instances in the designs.

Default value: Deselected

Command-line option: `flattenPcells`

Pcells Exclusion File

Use the *Pcells Exclusion File* option to specify the name of the exclusion file that contains a list of Pcells that need to be excluded during Pcell flattening. The *Pcells Exclusion File* option is available only if you select the *Flatten Pcells* option.

Example of a Pcell Exclusion file:

```
;Specify either library, library and cell, or library, cell, and view
(libraryB)
(libraryA "pcellA")
(libraryA "pcellB")
(libraryA "pcellC" "layout1") ("LibraryC")
```

Default value: Not specified

Command-line option: `flattenPcellsExclusionFile`

Do Not Preserve Pcells Pins

Select this check box if you do not want to retain the connectivity information of pins in flattened Pcell instances. The *Do Not Preserve Pcell pins* option is available only if you select the *Flatten Pcells* option.

Command-line option: `doNotPreservePcellPins`

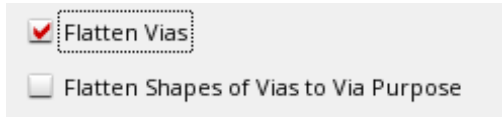
Flatten Vias

Use the *Flatten Vias* option if you want to flatten all standard via and custom vias in the design.

Default value: Deselected

Command-line option: `flattenVias`

Flatten Shapes of Vias to Via Purpose

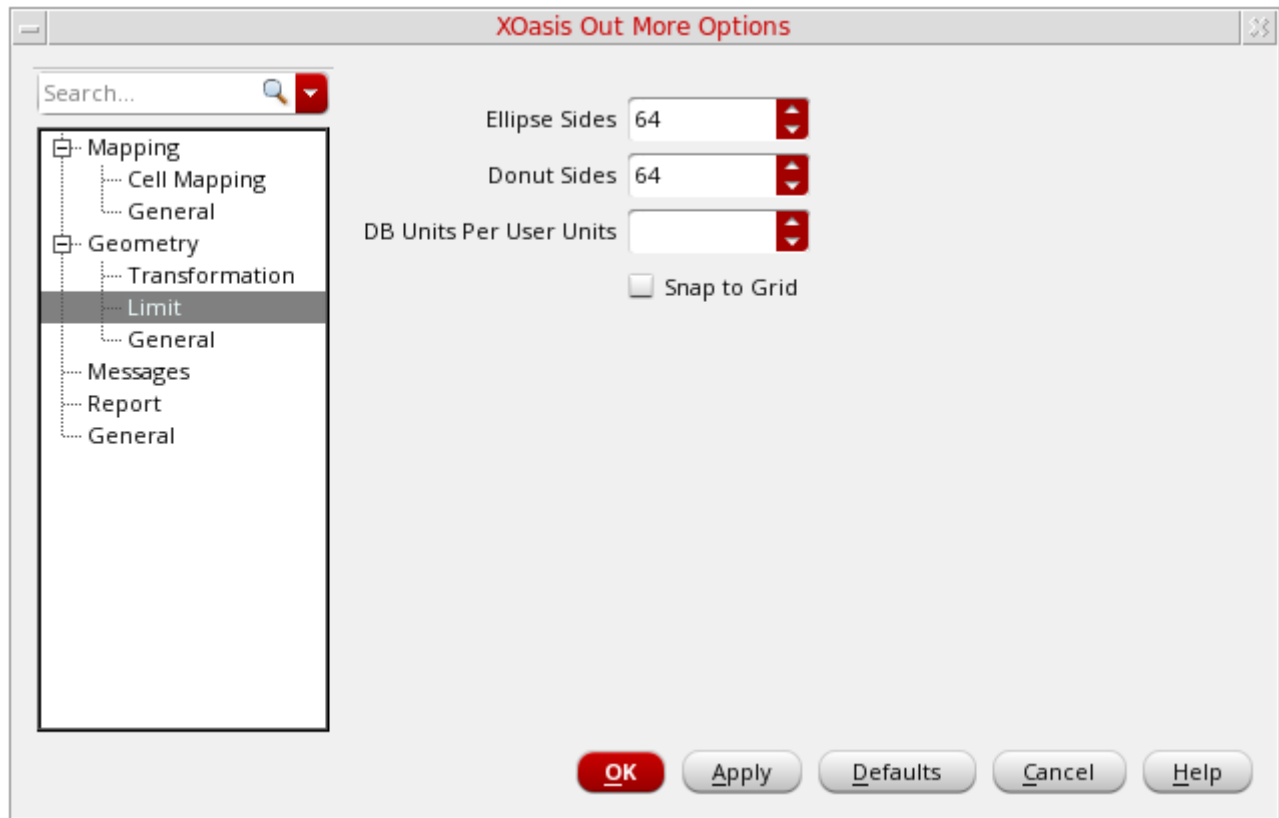


Select the *Flatten Shapes of Vias to Via Purpose* check box to flatten the shapes of vias and set the original purpose that was defined for the vias. This option is available only when the *Flatten Vias* option is selected.

Command-line option: `flattenViaShapesToViaPurp`

Geometry: Limit Options

This section describes options on the Geometry: Limit options pane.



Ellipse Sides

Specifies the number of sides for the boundary of ellipse.

Valid values: Any value between 4 and 131072 (inclusive of both values)

Default value: 64

Command-line option: `ellipseNumSides`

Donut Sides

Specifies the number of sides for the boundary of donut objects.

Valid values: Any value between 4 and 131072 (inclusive of both values)

Design Data Translators Reference

Design Translation Using XOasis Translator

Default value: 64

Command-line option: `donutNumSides`

DB Units Per User Units

Use the *DB Units per User Units* field to specify the database units per user units, which should be used during translation. This option allows you to scale the target design library as per the new `DBUPerUU`.

This option is considered only if new technology information is created in the target library. This technology information contains the `DBUPerUU` value specified at the option.

Valid values: Greater than or equal to 1

Command-line option: `dbuPerUU`

Snap to Grid

The *Snap to Grid* option adjusts the coordinates of geometric objects to coincide with the grid resolution. However, this option can shift data from the original OASIS grid, resulting in a loss of precision.

Default value: Deselected

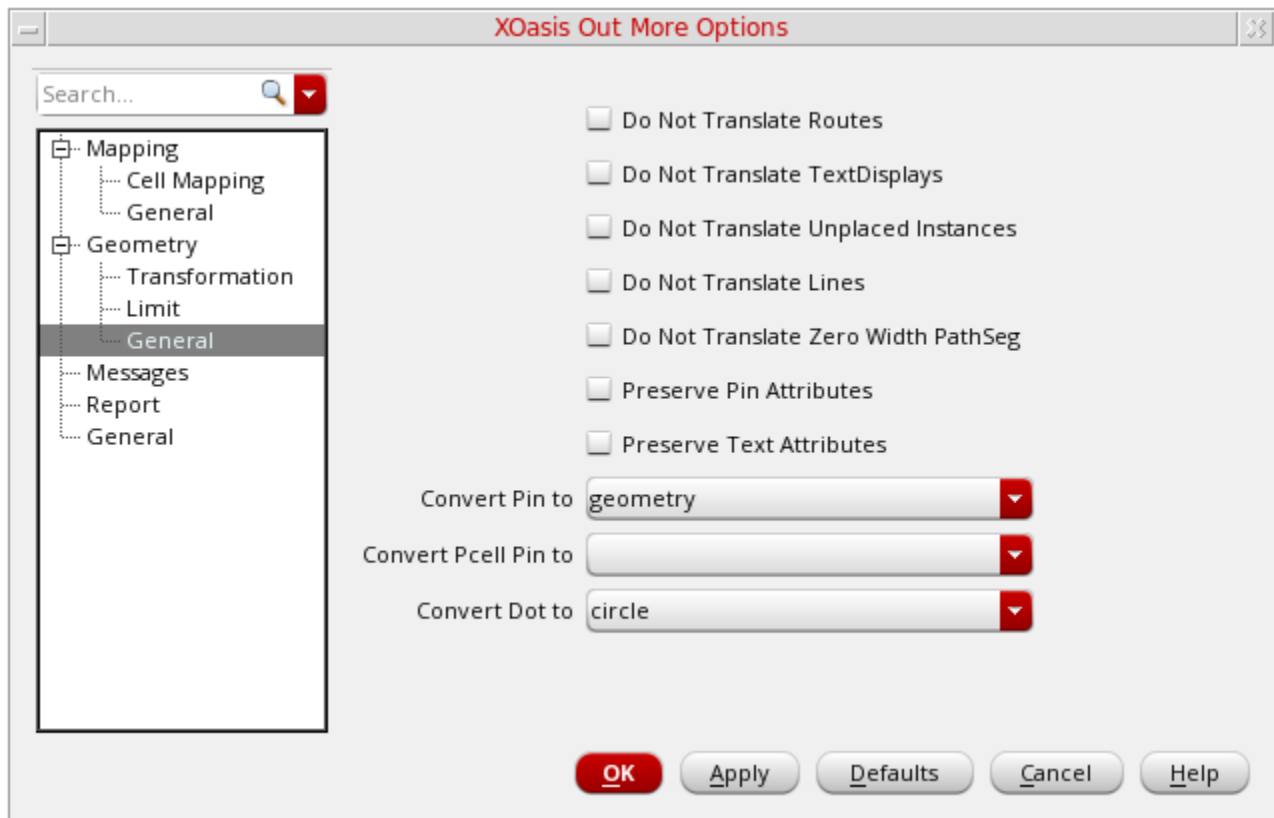
Command-line option: `snapToGrid`

Design Data Translators Reference

Design Translation Using XOasis Translator

Geometry: General Options

This section describes options on the Geometry: General options pane.



Do Not Translate Routes

Use the *Do Not Translate Routes* option to ignore the pathsegs and vias that are part of routes.

Command-line option: `ignoreRoutes`

Do Not Translate Text Displays

Use the *Do Not Translate Text Displays* option if you do not want to translate text displays into OASIS files. By default, XOasis Out translates the text displays into OASIS files.

Command-line option: `noOutputTextDisplays`

Do Not Translate Unplaced Instances

Unplaced instances are not visible in the Layout Editor. Therefore, if needed, select the *Do Not Translate Unplaced Instances* option to prevent these instances from getting translated during XOasis Out.

Default value: Deselected

Command-line option: `noOutputUnplacedInst`

Do Not Translate Lines

Use the *Do Not Translate Lines* option if you want to ignore the lines during translation. By default, lines are translated as zero-width paths.

Default value: Deselected

Command-line option: `ignoreLines`

Do Not Translate Zero Width PathSeg

Use the *Do Not Translate Zero Width PathSeg* option to ignore the translation of zero-width path. If you select this option, zero-width paths are ignored, otherwise they are translated as lines.

Default value: Deselected

Command-line option: `ignoreZeroWidthPathSeg`

Preserve Pin Attributes

Use the *Preserve Pin Attributes* option to preserve pin connectivity information.

Command-line option: `preservePinAtt`

Preserve Text Attributes

Use the *Preserve Text Attributes* option to preserve text attributes as properties while exporting using XOasis Out. This option preserves the font, justification, rotation, height, overbar, drafting, and visible attributes. These attributes are restored when you import by using XOasis In.

Design Data Translators Reference

Design Translation Using XOasis Translator

Command-line option: `preserveTextAtt`

Convert Pin to

The *Convert pin to* option controls the translation of pins present in an OpenAccess design to OASIS format. During XOasis Out, pins can either be translated as geometries or text, or be ignored. Use the options described in the table below to specify whether pins are translated as labels or instances, or are ignored by the translator.

Command-line option: `convertPin`

Note: The following table also summarizes the output when a label map file is specified simultaneously with the Convert Pin to options and when it is not.

Table 3-1 Convert Pin to Options

Convert Pin To	Label Map File	Output
geometry	No	■ Pins are translated as the corresponding shape or instance.
	Yes	■ Geometry as well as text is translated.
text	No	■ Pins are translated as labels. ■ The label string consists of the net name associated with the shape.
	Yes	■ The corresponding entry in the label map file takes precedence over the Convert Pin to option. ■ Only text labels are translated. ■ A warning message informing that the mapping will be based on the label map file displays, and the translation continues to run.

Design Data Translators Reference

Design Translation Using XOasis Translator

Convert Pin To	Label Map File	Output
geometryAndText	No	■ Pins are translated into shapes or instances and text labels.
	Yes	■ The label map file takes precedence over the Convert Pin to option ■ Geometry as well as text is translated ■ A warning message informing that the mapping will be based on the label map file displays, and the translation continues to run.
ignore	No	■ Pins are not translated.
	Yes	■ The label map file is ignored. ■ No record is translated. ■ A warning message informing that the label map option will be ignored displays, and XOasis Out continues to run.

Convert Pcell Pin to

Use the *Convert Pcell Pin to* option to control the translation of the pins present in the Pcell submaster to OASIS format. This option overrides the value of `convertPin` for the Pcell submaster. However, if the `convertPcellPin` option is not specified, value of `convertPin` is considered.

During XOasis Out, pins can either be translated as geometries or text, or be ignored. For information on specifying whether the pins in the Pcell submaster are translated as labels or instances, or are ignored by the translator, see the [Convert Pin to Options](#) table.

Command-line option: `convertPcellPin`

Convert Dot to

Use the *Convert Dot to* option if you want to convert dots to circles or polygons.

Circle

The *Circle* option converts dots to circles.

Polygon

The *Polygon* option converts dots to polygons.

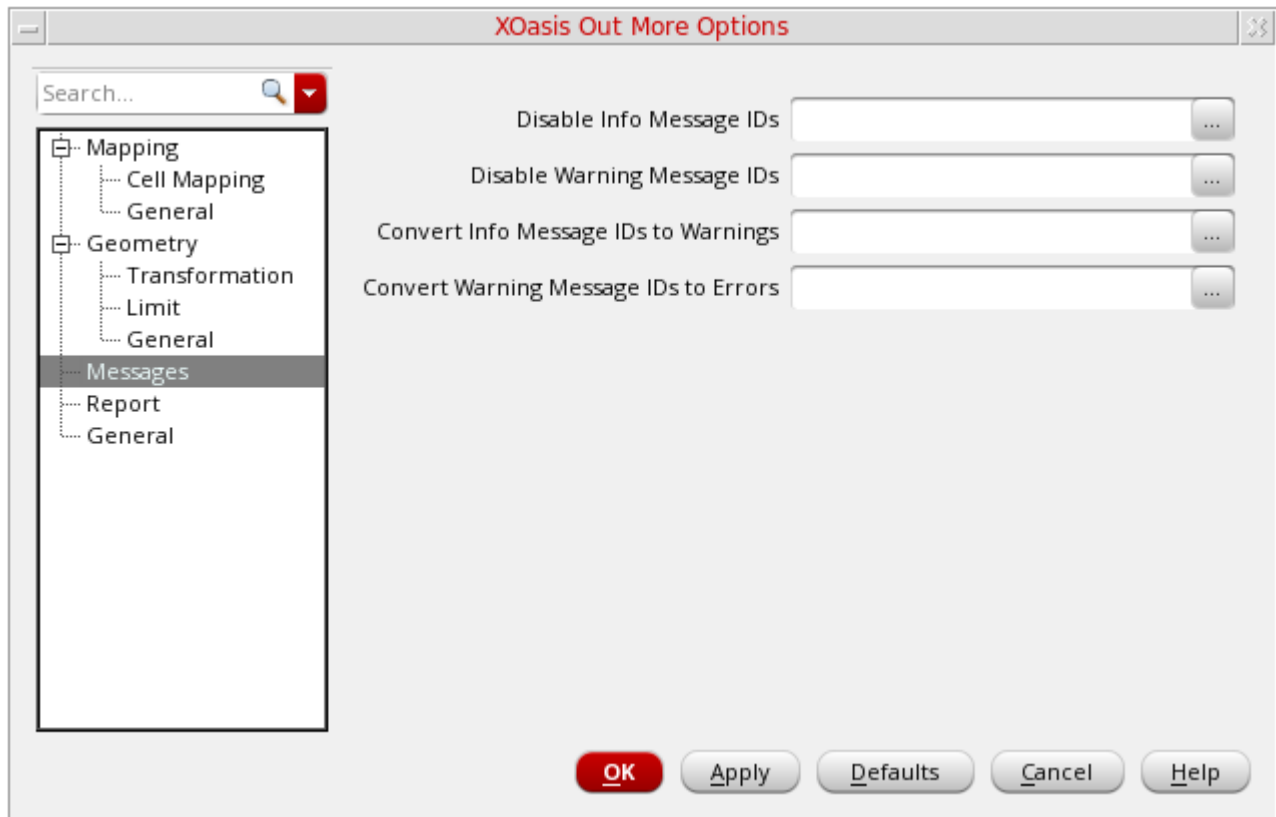
Ignore

The *Ignore* option ignores the dots during translation

Command-line option: `convertDot`

Messages Options

Use the Messages options pane to specify the information and warning message IDs that you want to disable or convert.



Disable Info Message IDs

Use the *Disable Info Message IDs* field to specify list of information message IDs that you do not want to print. You can also specify message IDs of dependent tools in this field.

The information message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 4374. The list of information message IDs should be separated by a white space.

Command-line option: `noInfo`

Alternatively, click the *Message Browser* button to display the XOasis Out Info Messages form to select or specify the information messages that you do not want to print.

Disable Warning Message IDs

Use the *Disable Warning Message IDs field* to specify list of warning message IDs, separated by comma that you do not want to print. You can also specify message IDs of dependent tools in this field.

The warning message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 3075. The list of warning message IDs should be separated by a comma or white space.

Command-line option: `noWarn`

Alternatively, click the *Message Browser* button to display the XOasis Out Warning Messages form to select or specify the warning messages that you do not want to print.

Convert Info Message IDs to Warnings

Use the *Convert Info Message IDs to Warnings field* to specify list of information message IDs that you want to convert to warning messages. You can also specify message IDs of dependent tools in this field.

The information message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 4374. The list of information message IDs should be separated by a comma or white space.

Command-line option: `infoToWarn`

Alternatively, click the *Message Browser* button to display the XOasis Out Info Messages form to select or specify the information messages that you want to convert to warning messages.

Convert Warning Message IDs to Errors

Use the *Convert Warning Message IDs to Errors field* to specify the IDs of warning messages that you want to convert to error messages. You can also specify message IDs of dependent tools in this field.

The warning message IDs should be specified in numeric format and they should not be prefixed by any other character. For example, 3075. The list of warning message IDs must be separated by a comma or white space.

Command-line option: `WarnToErr`

Design Data Translators Reference

Design Translation Using XOasis Translator

Alternatively, click the *Message Browser* button to display the XOasis Out Warning Messages form to select or specify the warning messages that you want to convert to error messages.

Design Data Translators Reference

Design Translation Using XOasis Translator

XOasis Out Messages Forms

This section describes the XOasis In Message forms that are displayed when you click the respective *Message Browser* button.

XOasis Out Info Messages Form to Disable Info Messages

This section describes options on the XOasis Out Info Messages form that is displayed when you click the *Message Browser* button for the Disable Info Message IDs option.

Id	Message
☐ Other IDs	
☐ 30	Layer map file %s exists in the technology library but is not readable. Automatic layer mapping will...
☐ 60	Creating a new purpose %s in the technology file.
☐ 61	Creating a new layer %s in the technology file.
☐ 62	Creating a new layer:purpose pair %s:%s in the technology file.
☐ 69	All the libraries specified in 'cds.lib' will be considered as reference libraries. This is because refLi...
☐ 70	The 'Reference Library list' and the 'Keep Oasis Cells' options have been used together. Any cell d...
☐ 138	The 'enableLocking' option is selected so the output cellViews in the Cadence database will be sep...
☐ 163	You have not used the viaMap option. If a viaMap file corresponding techfile or technology library...
☐ 171	The 'noObjectProp' and the 'objectMap' options have been used together. Therefore no property...
☐ 173	Object map file %s exists in the technology library but is not readable. No object mapping will be d...

☐ Select All

OK Apply Cancel Help

Select check boxes for the XOasis information messages that you want to disable.

Other IDs

Use the *Other IDs* field to specify information message IDs of other dependent tools.

Select All

Click the *Select All* check box to select all message IDs.

Design Data Translators Reference

Design Translation Using XOasis Translator

Cancel

Use the *Cancel* button to exit the XOasis Out Info Messages form without saving any changes.

XOasis Out Warning Messages Form to Disable Warning Messages

This section describes options on the XOasis Out Warning Messages form that is displayed when you click the *Message Browser* button for the Disable Warning Message IDs option.

Id	Message
☐ Other IDs	
☐ 18	Performing automatic layer mapping because either the layer mapping information is missing or t...
☐ 20	Output Oasis file '%s' already exists. It will be overwritten.
☐ 25	The layer map file '%s' contains an invalid record at line number '%d' and will be ignored. This is b...
☐ 27	The property map file '%s' contains an invalid record at line number '%d' and will be ignored. This ...
☐ 28	The cell map file '%s' contains an invalid record at line number '%d' and will be ignored. This is bec...
☐ 29	refLib file contains invalid record at line number '%d'. This record in the refLib file will be ignored.
☐ 31	Layer map file '%s' contains a duplicate entry for Oasis layer:datatype '%d:%d' at line '%d'. Therefo...
☐ 32	The layer map file '%s' contains unknown layer '%s' at line number '%d' and will be ignored. This is ...
☐ 33	The layer map file '%s' contains unknown purpose '%s' at line number '%d' and will be ignored. Thi...
☐ 34	The objects in the layer purpose pair '%u:%u' are ignored. This is because either the layer purpos...

Select check boxes of the XOasis warning messages listed in this form that you want to disable.

Other IDs

Use the *Other IDs* field to specify warning message IDs of other dependent tools.

Design Data Translators Reference

Design Translation Using XOasis Translator

Select All

Click the *Select All* check box to select all message IDs.

Cancel

Use the *Cancel* button to exit the XOasis Out Warning Messages form without saving any changes.

XOasis Out Info Messages Form to Convert Info Messages to Warnings

This section describes options on the XOasis Out Info Messages form that is displayed when you click the *Message Browser* button for the Convert Info Message IDs to Warnings option.

Id	Message
☐ Other IDs	
☐ 30	Layer map file %s exists in the technology library but is not readable. Automatic layer mapping will...
☐ 60	Creating a new purpose %s in the technology file.
☐ 61	Creating a new layer %s in the technology file.
☐ 62	Creating a new layer:purpose pair %s:%s in the technology file.
☐ 69	All the libraries specified in 'cds.lib' will be considered as reference libraries. This is because refLi...
☐ 70	The 'Reference Library list' and the 'Keep Oasis Cells' options have been used together. Any cell d...
☐ 138	The 'enableLocking' option is selected so the output cellViews in the Cadence database will be sep...
☐ 163	You have not used the viaMap option. If a viaMap file corresponding techfile or technology library...
☐ 171	The 'noObjectProp' and the 'objectMap' options have been used together. Therefore no property...
☐ 173	Object map file %s exists in the technology library but is not readable. No object mapping will be d...

Select check boxes of the XOasis information messages listed in this form to covert to warnings.

Design Data Translators Reference

Design Translation Using XOasis Translator

Other IDs

Use the *Other IDs* field to specify information message IDs of other dependent tools.

Cancel

Use the *Cancel* button to exit the XOasis Out Info Messages form without saving any changes.

XOasis Out Warning Messages Form

This section describes options on the XOasis Out Warning Messages form that is displayed when you click the *Message Browser* button for the Convert Warning Message IDs to Errors option.

Id	Message
☐ Other IDs	
☐ 18	Performing automatic layer mapping because either the layer mapping information is missing or t...
☐ 20	Output Oasis file '%s' already exists. It will be overwritten.
☐ 25	The layer map file '%s' contains an invalid record at line number '%d' and will be ignored. This is b...
☐ 27	The property map file '%s' contains an invalid record at line number '%d' and will be ignored. This ...
☐ 28	The cell map file '%s' contains an invalid record at line number '%d' and will be ignored. This is bec...
☐ 29	refLib file contains invalid record at line number '%d'. This record in the refLib file will be ignored.
☐ 31	Layer map file '%s' contains a duplicate entry for Oasis layer:datatype '%d:%d' at line '%d'. Therefo...
☐ 32	The layer map file '%s' contains unknown layer '%s' at line number '%d' and will be ignored. This is ...
☐ 33	The layer map file '%s' contains unknown purpose '%s' at line number '%d' and will be ignored. Thi...
☐ 34	The objects in the layer purpose pair '%u:%u' are ignored. This is because either the layer purpos...

Select check boxes of the XOasis warning messages listed in this form that you want to convert to errors.

Design Data Translators Reference

Design Translation Using XOasis Translator

Other IDs

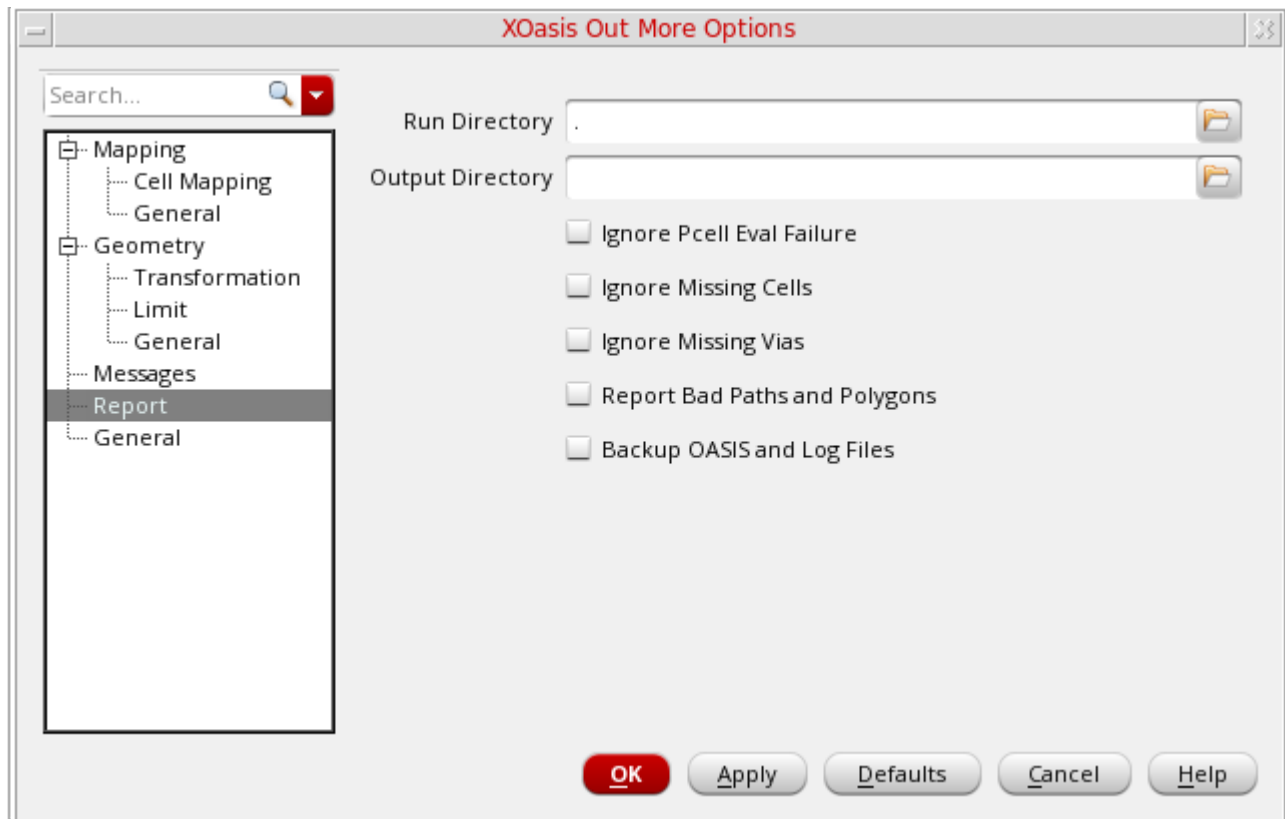
Use the *Other IDs* field to specify information message IDs of other dependent tools.

Cancel

Use the *Cancel* button to exit the XOasis Out Info Messages form without saving any changes.

Report Options

This section describes options on the Report options pane.



Run Directory

The *Run Directory* field accepts the default directory for all the files you specify in the *XOasis Out* form. If you want to read from or write to a file that is not present in the run directory, type the full path in the field for the file.

Use the directory browser to select the path of the run directory.

Default value: Current working directory

Command-line option: `runDir`

Output Directory

The *Output Directory* field enables you to specify the path of the output directory.

Default value: Run directory

Command-line option: `outputDir`

Ignore Pcell Eval Failure

Select the *Ignore Pcell Eval Failure* option if you do not want XOasis to halt upon any Pcell evaluation failure. If you do not select this option, XOasis will halt upon any Pcell evaluation failure and a error message will be displayed.

Default value: `nil`

Command-line option: `ignorePcellEvalFail`

Ignore Missing Cells

Select the *Ignore Missing Cells* option to ignore missing cells during translation and continue translation with a warning message displayed in the CIW.

Default: Deselected

Command-line option: `ignoreMissingCells`

Ignore Missing Vias

Select the *Ignore Missing Vias* option to ignore missing via definitions during translation and continue translation with a warning message displayed in the CIW.

Default: Deselected

Command-line option: `ignoreMissingVias`

Report Bad Polygons and Paths

The *Report Bad Polygons and Paths* option provides a way for reporting self-intersecting paths, coincident or collinear paths, paths that have less then two coordinates, paths with an invalid begin and end extension value, an open polygon, coincident or collinear polygons, and a polygon with self-intersecting points. However, using this option can degrade the performance.

Design Data Translators Reference

Design Translation Using XOasis Translator

If in an OpenAccess design with octagonal extension, paths are mapped to PATHTYPE1, which is a circular path end according to the OASIS format, you can retain the geometry of these paths by using the *Output Paths as Polygon* option.

Note: A begin or end path with negative extension is truncated to zero if the length of the new path segment is greater than or equal to the half-path width. A warning is added to the log file.

Default value: Deselected

Command-line option: `checkPolygon`

Backup OASIS and Log Files

Select the *Backup OASIS and Log Files* option if you want to create a backup of the OASIS and log files.

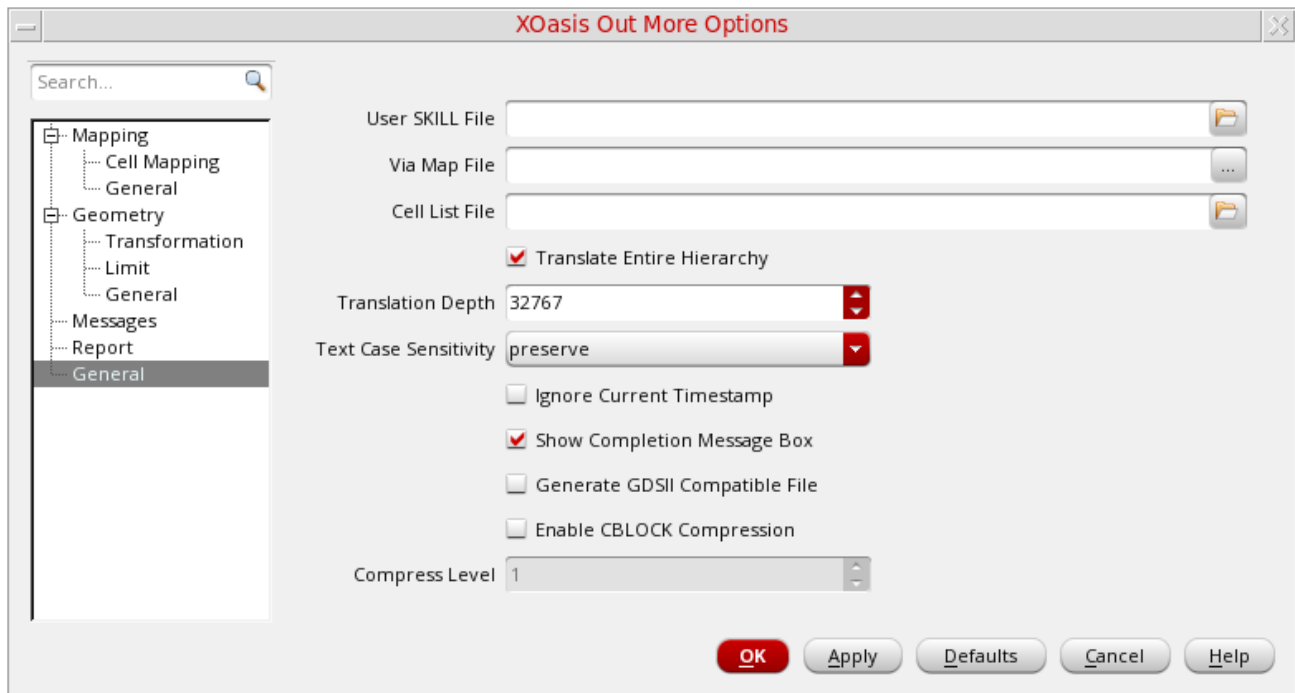
Command-line option: `backOasisLogFiles`

Design Data Translators Reference

Design Translation Using XOasis Translator

General Options

This section describes options on the General options pane.



User SKILL File

The *User SKILL File* field lets you specify a file that consists of user-defined SKILL routines. Using this file, you can specify a mapping mechanism for cell names, layers, and fonts through the use of programming routines written in SKILL.

In order to load all the required SKILL procedures in a single file, use only one SKILL file during translation. The SKILL file is searched with respect to the run directory if complete path of the SKILL file is not specified.

Note: XOasis Out supports only those SKILL functions that are essential to evaluate a Pcell. These are SKILL Core, `db`, `dd`, `tech`, `cst`, `abe`, `vfo` (`vfo.cxt`), and `pas` (`pdkutils.cxt`) SKILL functions. Application specific functions such as `le`, `hi`, `sch`, `lx`, and `via` are not supported.

Command-line option: `userSkillFile`

Via Map File

Use the *Via Map File* field to specify a via map file. The via map file maps the structure names from the OASIS file to the corresponding viaDef names in the input technology library. This via map file can be used to recreate the original vias during subsequent XOasis Ins. For more information on the via map file, see [Via Map File](#).

Command-line option: `viaMapFile`

Cell List File

The *Cell List File* field enables you to specify a file containing a list of cell names. The cell names in a cell list file is displayed in the following format:

```
<libName> <cellName> <viewName> <output OASIS file name>
```

For example, the cell list file can be specified as:

```
<Lib1> <cell1> <view1> <outOASIS1>
```

```
<*> <cell2> <*> <*>
```

Note: Asterisk “*” can be specified for all the fields, except `<cellName>`. However, if you only specify the `<cellName>`, XOasis Out will consider this entry as a valid entry.

Command-line option: `cellListFile`

Translate Entire Hierarchy

Use the *Translate Entire Hierarchy* option to translate the entire design hierarchy.

Default value: Selected

Translation Depth

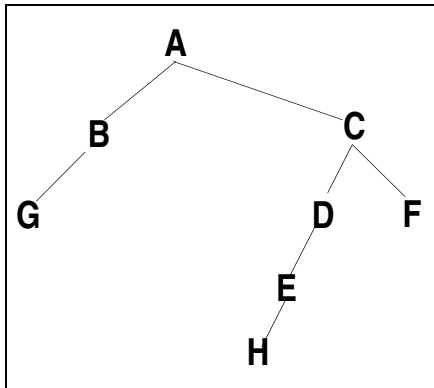
Use the *Translation Depth* field to control the depth of the design hierarchy during XOasis Out. XOasis Out stops the translation process at the hierarchy level indicated by this option. The hierarchy depth value applies to all the top cells getting translated. This option is enabled only if the *Translate Entire Hierarchy* option is deselected.

Value values: 0 to 32767

Default value: 32767

Command-line option: `hierDepth`

Example 3-15 Translating cells by using the Translation Depth option



The above hierarchy shows structures A, B, C, D, E, F, G, and H placed at different levels. When structure C is selected as the top cell name and the hierarchy depth is selected as 1, the structures C, D, and F are translated.

Instance of E will first be translated as SREF inside structure D and then be translated to the OASIS file, but its structure information is ignored. Structure H is ignored and is not translated.

By default, the entire hierarchy is translated.

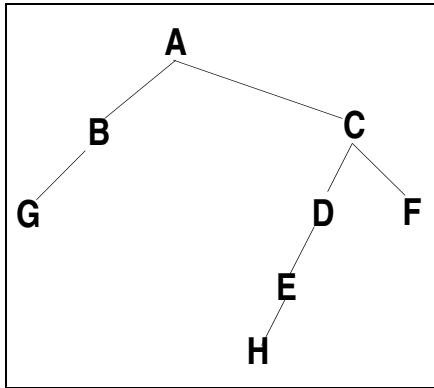
Text Case Sensitivity

Use the *Text Case Sensitivity* option to specify if XOasis changes the casing of letters in the text labels. The following options are available:

- *preserve* - The *preserve* option translates text labels without changing case. For example, the label Obj1 remains Obj1.
- *lower* - The *lower* option translates the text labels to lowercase. For example, the label Obj1 becomes obj1.
- *upper* - The *upper* option translates the text label to uppercase. For example, the label Obj1 becomes OBJ1.

Command-line option: `textCase`

Example 3-16 Translating cells by using the Translation Depth option



The above hierarchy shows structures A, B, C, D, E, F, G, and H placed at different levels. When structure C is selected as the top cell name and the hierarchy depth is selected as 1, the structures C, D, and F are translated.

Instance of E will first be translated as SREF inside structure D and then be translated to the Oasis file, but its structure information is ignored. Structure H is ignored and is not translated.

By default, the entire hierarchy is translated.

Ignore Current Timestamp

Use the *Ignore Current Timestamp* option to ignore the current timestamp while exporting an OASIS file.

When you select this option:

- Time stamp is not added as a suffix to the standard via and Pcell names. As a result, their names do not change every time you generate the OASIS file.
- Exported OASIS file is time-independent. This means that time information is not added to the exported OASIS file and therefore, contents of the OASIS file do not change every time you run XOasis Out.

Command-line option: `ignoreCurrentTimestamp`

Show Completion Message Box

The *Show Completion Message Box* check box enables you to display the *XOasis Out translation complete* message box along with the number of errors and warnings after the

translation is completed. The state of this check box depends on the value of the `xoasShowCompletionMsgBox` SKILL variable, which is `t` by default.

Generate GDSII Compatible File

Select the *Generate GDSII Compatible File* option to generate the XOasis file with limits specific to GDSII files. When you select this option, the limits listed below are followed:

- Cell names are mapped to unique 32-character names.

Polygons are limited to 200 vertexes. If the number of vertexes or points exceeds 200, a warning message is displayed and translation continues.

Command-line option: `GDSIICompatible`

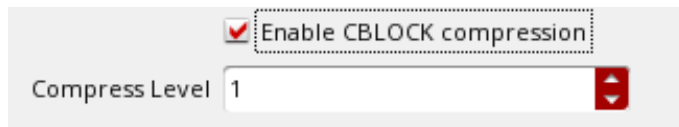
Enable CBLOCK Compression

Select the *Enable CBLOCK Compression* option if you want CBLOCK compression in the OASIS file.

Command-line option: `compress`

Compress Level

Use the *Compression Level* option to specify the level of CBLOCK compression to be applied during XOasis Out. You can specify a value from 1 to 9 for this option. Value 1 signifies the lowest level of compression and 9 signifies the highest level of compression. If you do not specify any value in the *Compression Level* option, value 1 is considered as the compression level during XOasis Out. This option is enabled only if the *Compress* option is selected.



The screenshot shows a user interface for the XOasis translator. At the top, there is a checked checkbox labeled "Enable CBLOCK compression". Below this, there is a label "Compress Level" followed by a spin box containing the value "1". The spin box has red up and down arrow buttons on its right side.

Valid values: 1 to 9

Default value: 1

Command-line option: `compressLevel`

XOasis Out Editor Forms

This section discusses the following XOasis Out Editor Forms:

- [XOasis Out Layer Map File Editor Form](#)
- [XOasis Out Object Map File Editor Form](#)
- [XOasis Out Cell Map File Editor Form](#)
- [XOasis Out Reference Library File Editor Form](#)
- [XOasis Out Property Map File Editor Form](#)

XOasis Out Layer Map File Editor Form

This section describes options on the XOasis Out Layer Map File Editor form that is displayed when you click the *Editor* button for the [Layer Map](#) option.

Layer Name	Purpose Name	OASIS Layer	OASIS Type	Material	Mask Number	Qualifier	Photo Mask Color	Color State

New File

Load Save As and Exit **Save and Exit** Cancel Help

Add Row

Click the *Add Row* button to create a new layer mapping record.

For more information on columns in a layer mapping record, see [Layer Map File](#).

OASIS Layer

Use the *OASIS Layer* field to specify the OASIS layer name to which DFII layer name has to be translated.

OASIS Datatype

Use the *OASIS Datatype* field to specify the OASIS datatype to which the DFII purpose name has to be translated.

Layer Name

Use the *Layer Name* field to specify a DFII layer that you want to translate to the OASIS format. The number of layers present in this option are taken from the technology file.

Purpose Name

Use the *Purpose Name* field to specify a DFII purpose that you want to translate to the OASIS format. The number of purposes present in this option are taken from the technology file.

Material

Use the *Material* field to specify the OpenAccess material type to which you want to map the OASIS type. This *Material* will be used to create the layer, if the layer does not exist.

Mask Number

Use the *Mask Number* field to specify the OpenAccess mask number to which you want to map the OASIS layer.

Qualifier

Use the *Qualifier* field to specify the OpenAccess qualifier to which you want to map the OASIS type.

Photo Mask Color

Use the *Photo Mask Color* field to specify the photo mask color of the layer.

Color State

Use the *Color State* field to specify the color state of the layer.

The *Photo Mask Color* and *Color State* fields are visible only if the *Enable Coloring* check box is selected on the *General* tab.

Delete Row

Use the *Delete Row* button to delete an existing layer mapping record.

Load

Use the *Load* option to load a new layer map file. You can specify multiple layer name map files by using the *Load* option, iteratively. Using a layer map file, you can map OpenAccess layer name and purpose name pairs to OASIS layer number and OASIS datatype pairs. If you specify an invalid filename, a warning message is generated and the filename is ignored.

Design Data Translators Reference

Design Translation Using XOasis Translator

The path specified for the layer map file can be absolute or relative to the run directory. For more information, see [Layer Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the mapping information in a new file and exit the layer map file editor.

Save and Exit

Use the *Save and Exit* button to save the mapping information in the current file and exit the layer map file editor.

Cancel

Use the *Cancel* button to exit the XOasis Out Layer Map File Editor form without saving any changes.

XOasis Out Object Map File Editor Form

This section describes options on the XOasis Out Object Map File Editor form that is displayed when you click the *Editor* button for the [Object Map](#) option.

Object Name	Sub Object Name	Layer Name	OASIS Layer	OASIS Type	Qualifier	Photo Mask Color

New File

Load Save As and Exit **Save and Exit** Cancel Help

Add Row button Delete Row button

Add Row

Click the *Add Row* button to create a new object mapping record.

For more information on columns in an object mapping record, see [Object Map File](#).

Object Name

Use the *Object Name* field to select the name of the object to be mapped.

Sub Object Name

Use the *Sub Object Name* option to select the name of the subobject to be mapped.

Layer Name

The *Layer Name* field is displayed only when the *Object Name* is specified as layerBlockage. Use the *Layer Name* field to specify the OpenAccess layer name to which you want to map the OASIS layer.

OASIS Layer

Use the *OASIS Layer* field to specify the OASIS layer number for the OA object.

OASIS Type

Use the *OASIS Type* field to specify the OASIS datatype for the OA object.

Qualifier

Use the *Qualifier* field to specify the `OASISLayerName` for the OA object. The name is specified in a specific format. For example, `OASISLayerName:boundary_area_0_1`.

Photo Mask Color

Use the *Photo Mask Color* field to specify the photo mask color of the layer.

Note: The *Photo Mask Color* field is visible only if the [Coloring Mode](#) is enabled.



If the layer and object map files have color-related entries however, Coloring Mode is not enabled, then these entries are not visible in the respective editors and a warning is displayed to inform you that these entries will be ignored during translation. These entries are visible only when Coloring Mode is enabled.

Delete Row

Use the *Delete Row* button to delete an existing object mapping record.

Load

Use the *Load* button to load the object map file. An object map file is used for preserving OpenAccess objects. If you specify an invalid filename, a warning message is generated and the filename is ignored.

The path specified for the object map file can be absolute or relative to the run directory. For more information, see [Object Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the object mapping information in a new file and exit the object map file editor.

Save and Exit

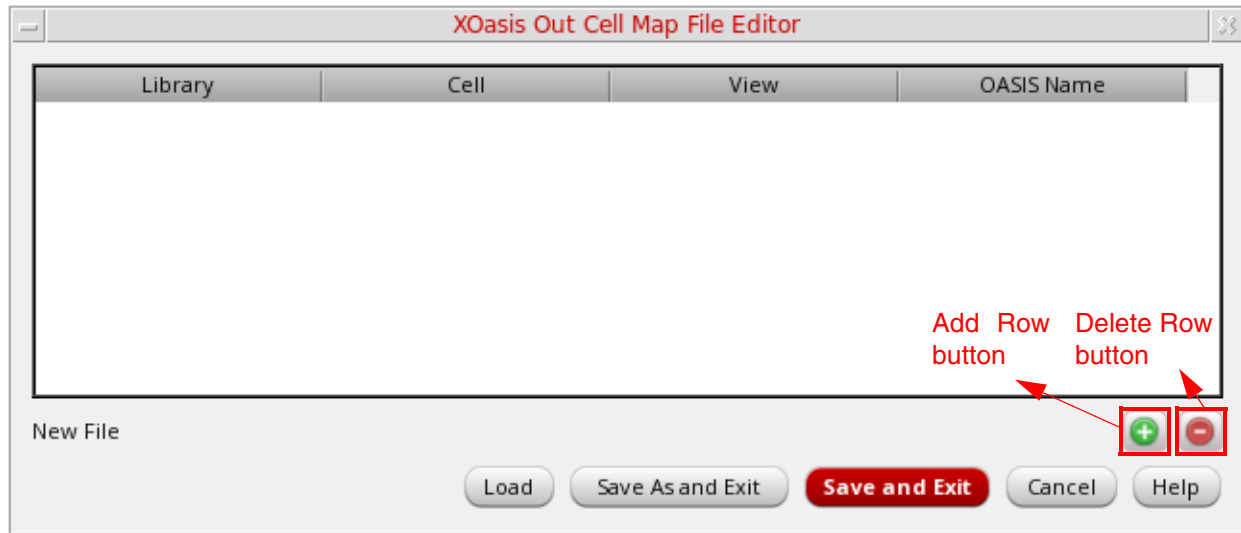
Use the *Save and Exit* button to save the object mapping information in the current file and exit the object map file editor.

Cancel

Use the *Cancel* button to exit the XOasis Out Object Map File Editor form without saving any changes.

XOasis Out Cell Map File Editor Form

This section describes options on the XOasis Out Cell Map File Editor form that is displayed when you click the *Editor* button for the Cell Map File Name option.



Add Row

Click the *Add Row* button to create a new cell mapping record.

For more information on columns in a cell mapping record, see Cell Name Map File.

Library

Use the *Library* field to select a library from which you want to translate a design to OpenAccess. All libraries present in the *Library* field are a list of reference libraries present in the `cds.lib` file.

Cell

Use the *Cell* option to select a library cell that you want to translate to the OASIS format.

View

Use the *View* option to select a view of the specified cell that you want to translate to the OASIS format. In the above figure, view `via` of cell `M1_M2` will be translated to the OASIS format.

OASIS Name

Use the *OASIS Name* field to specify the OASIS filename in which you want to translate the selected library, cell, and view.

Delete Row

Use the *Delete Row* button to delete an existing cell mapping record.

Load

Use the *Load* button to load a new cell name map file. A cell name map file contains a table that maps cellview names in the OpenAccess database to structure names in the OASIS file.

The path specified for the cell name map file can be absolute or relative to the run directory. For more information, see [Cell Name Map File](#).

Save As and Exit

Use the *Save As and Exit* button to save the cell mapping information in a new file and exit the cell map file editor.

Save and Exit

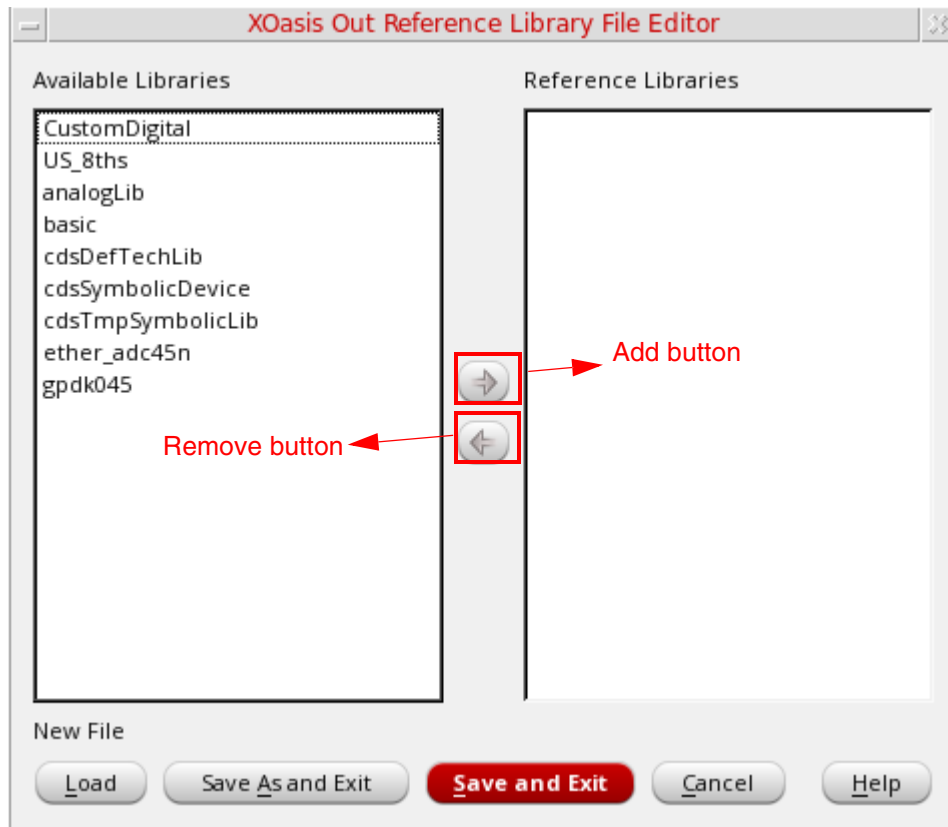
Use the *Save and Exit* button to save the cell mapping information in the current file and exit the cell map file editor.

Cancel

Use the *Cancel* button to exit the XOasis Out Cell Map File Editor form without saving any changes.

XOasis Out Reference Library File Editor Form

This section describes options on the XOasis Out Reference Library File Editor form that is displayed when you click the *Editor* button for the Ref Lib File Name option.



Add

Use the *Add* button to add libraries to the *Reference Libraries* field.

The *Reference Libraries* field is used to specify logical library names that you want to consider as reference libraries. The libraries specified in this field should be present in the `cds.lib` file. Otherwise, they are not considered as reference libraries.

During XOasis Out, when you search for master cellviews of instances, reference libraries are searched in the same order as specified in the *Reference Libraries* field. If any of the specified libraries do not exist in the `cds.lib` file, an error message is displayed.

Design Data Translators Reference

Design Translation Using XOasis Translator

Remove

Use the *Remove* button to remove libraries from the *Reference Libraries* field.

Load

Use the *Load* button to load a text file containing the list of reference libraries. The path and name of the text file is displayed in the blank text field above this button.

Save As and Exit

Use the *Save As and Exit* button to save the reference library information in a new file and exit the reference library file editor.

Save and Exit

Use the *Save and Exit* button to save the reference library information in the current file and exit the reference library file editor.

Cancel

Use the *Cancel* button to exit the XOasis Out Reference Library File Editor form without saving any changes.

XOasis Out Property Map File Editor Form

This section describes options on the XOasis Out Property Map File Editor form that is displayed when you click the *Editor* button for the Prop Map File Name option.

The screenshot shows the 'XOasis Out Property Map File Editor' window. It features a table with three columns: 'Property Name', 'Object Type', and 'Property ID'. Below the table, there are two buttons: 'Add Row button' (a green square with a white plus sign) and 'Delete Row button' (a red square with a white minus sign). Below these buttons is a section labeled 'New File' with a sub-section 'Option Line' containing a checkbox 'Property Value Only' and a 'Property Separator' field with a comma. At the bottom are five buttons: 'Load', 'Save As and Exit', 'Save and Exit' (highlighted in red), 'Cancel', and 'Help'.

Add Row

Click *Add Row* button to create a new property mapping record.

For more information on columns in a property mapping record, see Property Map File.

Property Name

Use the *Property Name* field to specify the property name that you want to map to the OASIS format.

Object Type

Use the *Object Type* option to select the object that you want to translate to OASIS format. The supported object types are: Arc, Dot, Ellipse, Line, Path, Polygon, Rect, Text, ArrayInst, ScalarInst, PathSeg, Route, Marker, Row, AreaBoundary, PRBoundary, SnapBoundary, ClusterBoundary, AreaBlockage, LayerBlockage, AreaHalo, Pin, stdVia, and customVia.

Property ID

(Optional) Use the *Property ID* field to specify the ID of the property to which you want to map the selected object type and the property name. Property ID is specified if this

property has to be stored as a GDSII-Stream-style property in the OASIS file. The ID can be any number from 0 to 125.

Note: GDSII-Stream-style property is used for compatibility with GDSII Stream properties in OASIS files. By default, an OASIS file stores the properties in the `property-name:property-value` format. However, if the field *Property ID* is provided, the properties are stored similar to GDSII Stream format, where a property is stored for the specified property ID.

Property Value Only

Use the *Property Value Only* format specifier to export the specified value of `propertyName` of an `objectType` as the value of the corresponding property ID specified in the mapping record to the Oasis file. If *Property Value Only* is specified in the property map file, XOasis Out ignores the *Property Value Only* defined in the form.

Default value: Deselected

Property Separator

Use the *Property Separator* format specifier to specify a property separator.

Default value: The default property separator is a comma ‘,’.

Note: Values for *Property Value Only* and *Property Separator* are populated using the values specified in the special option line in the Property Map File.

Delete Row

Use the *Delete Row* button to delete an existing property mapping record.

Load

Use the *Load* button to load a property map file. A property map file preserves properties associated with OpenAccess objects. If you specify an invalid filename, a warning message is generated and the filename is ignored.

The path specified for the log file should be a valid file path. It can be absolute or relative to the run directory. For more information, see Property Map File.

Save As and Exit

Use the *Save As and Exit* button to save the property map information in a new file and exit the property map file editor.

Save and Exit

Use the *Save and Exit* button to save the property map information in the current file and exit the property map file editor.

Cancel

Use the *Cancel* button to exit XOasis Out Property Map File Editor form without saving any changes.

XOasis Out Names in GUI and Template File

The following table provides a list of all the XOasis Out GUI options and their corresponding names in the template file. For information on the sample template file, see [XOasis Out Template File](#).

Name in the Template File	XOasis Out GUI Options
arrayInstToScalar	<u><i>Convert Arrays to Scalar Instances</i></u>
autoGenOASISLayerName	<u><i>Automatically Generate OASISLayerName</i></u>
backupOasisLogFiles	<u><i>Backup OASIS and Log Files</i></u>
case	<u><i>Cell Case Sensitivity</i></u>
cellListFile	<u><i>Cell List File</i></u>
cellMap	<u><i>Cell Map File Name</i></u>
cellNamePrefix	<u><i>Cell Name Prefix</i></u>
cellNameSuffix	<u><i>Cell Name Suffix</i></u>
checkPolygon	<u><i>Output Circles as Polygons</i></u>
circleToPolygon	<u><i>Output Circles as Polygons</i></u>
compress	<u><i>Enable CBLOCK Compression</i></u>
compressLevel	<u><i>Compress Level</i></u>
convertDot	<u><i>Convert Dot to</i></u>
convertPcellPin	<u><i>Convert Pcell Pin to</i></u>
convertPin	<u><i>Convert Pin to</i></u>
dbuPerUU	<u><i>DB Units Per User Units</i></u>
diagonalPathToPolygon	<u><i>Convert Paths with Non-Orthogonal Segments to Polygons</i></u>
doNotPreservePcellPins	<u><i>Do Not Preserve Pcells Pins</i></u>
donutNumSides	<u><i>Donut Sides</i></u>
ellipseNumSides	<u><i>Ellipse Sides</i></u>
enableColoring	<u><i>Coloring Mode</i></u>
flattenPcells	<u><i>Flatten Pcells</i></u>

Design Data Translators Reference

Design Translation Using XOasis Translator

Name in the Template File	XOasis Out GUI Options
flattenPcellsExclusionFile	<u>Pcells Exclusion File</u>
flattenVias	<u>Flatten Vias</u>
flattenViaShapesToViaPurp	<u>Flatten Shapes of Vias to Via Purpose</u>
GDSIICompatible	<u>Generate GDSII Compatible File</u>
hierDepth	<u>Translation Depth</u>
ignoreLines	<u>Do Not Translate Lines</u>
ignoreMissingCells	<u>Ignore Missing Cells</u>
ignoreMissingViass	<u>Ignore Missing Vias</u>
ignoreObjectMapFromTech	<u>Ignore Object Map File from Technology</u>
ignorePcellEvalFail	<u>Ignore Pcell Eval Failure</u>
ignoreRoutes	<u>Do Not Translate Routes</u>
ignoreCurrentTimestamp	<u>Ignore Current Timestamp</u>
ignoreTopCellPrefixSuffix	<u>Ignore Cell Name Prefix and Suffix for Top Cell</u>
ignoreZeroWidthPathSeg	<u>Do Not Translate Zero Width PathSeg</u>
infoToWarn	<u>Convert Info Message IDs to Warnings</u>
labelDepth	<u>Label Translation Depth</u>
labelMap	<u>Label Map File</u>
layerMap	<u>Layer Map</u>
library	<u>Library*</u>
logFile	<u>Log File</u>
mergePathSegsToPath	<u>Merge Connected PathSegs</u>
noConvertHalfWidthPath	<u>Do Not Convert Half Width Path to Polygon</u>
noInfo	<u>Disable Info Message IDs</u>
noObjectProp	<u>Do Not Output Object Properties</u>
noOutputTextDisplays	<u>Do Not Translate Text Displays</u>
noOutputUnplacedInst	<u>Do Not Translate Unplaced Instances</u>

Design Data Translators Reference

Design Translation Using XOasis Translator

Name in the Template File	XOasis Out GUI Options
noWarn	<u>Disable Warning Message IDs</u>
oasisFile	<u>OASIS File*</u>
objectMap	<u>Object Map</u>
objectMap	<u>Object Map File</u>
outputDir	<u>Output Directory</u>
pathToPolygon	<u>Output Circles as Polygons</u>
preserveTextAtt	<u>Preserve Text Attributes</u>
preservePinAtt	<u>Preserve Pin Attributes</u>
propMap	<u>Prop Map File Name</u>
refLibList	<u>Ref Lib File Name</u>
replaceBusBitChar	<u>Replace <> with []</u>
runDir	<u>Run Directory</u>
showCompletionMsgBox	<u>Show Completion Message Box</u>
snapToGrid	<u>Snap to Grid</u>
subMasterSeparator	<u>Sub-Master Separator</u>
summaryFile	<u>Summary File</u>
techlib	<u>Technology Library</u>
textCase	<u>Text Case Sensitivity</u>
templateFile	<u>Template File</u>
topCell	<u>Top Cell(s)</u>
translateUnmappedLPP	<u>Translate Unmapped LPPs</u>
userSkillFile	<u>User SKILL File</u>
verbose	<u>Generate Detailed Log and Summary Files</u>
viaMap	<u>Via Map File</u>
view	<u>View(s)</u>
virtualMemory	<u>Export OASIS from Virtual Memory</u>
warnToErr	<u>Convert Warning Message IDs to Errors</u>

Design Data Translators Reference

Design Translation Using XOasis Translator

Name in the Template File	XOasis Out GUI Options

wildCardInCellMap	<u>Wildcard in Cell Map</u>

You can also access and input values in XOasis Out GUI using SKILL commands. Using SKILL commands, you will be able to set or clear fields in the GUI form. To populate values in the *XOasis Out* form, you need to enter the xoasOutSetField command in CIW. To access values from the *XOasis Out* form, you need to enter the xoasOutGetField command in CIW.

Note: All the options listed above are valid for xoasOutSetField, xoasOutGetField, template file, and oasisOutKeys. All the options except `virtualMemory` and `showCompletionMsgBox` are valid for command line executable or batch process.

Seeding the XOasis Out GUI

You can use the following methods to set default values for a user:

■ Using the XOASIS_IN environment variable

You can assign the XOasis In template file as a value of the XOASIS_OUT shell environment variable.

For example:

```
envSetVal ( "xoasis" "XOASIS_OUT" 'string "demo.txt" )
setenv XOASIS_OUT "demo.txt":
```

Sample XOasis Out option template file:

```
=====
pinAttNum          "100"
summaryFile         "xOasisOut.sum"
replaceBusBitChar
=====
```

■ Using oasisOutKeys SKILL list

You can define a set of option values to load into the XOasis In form in the `oasisOutKeys` SKILL list. For this, define the pre-seeding options in the `.cdsinit` file because the system automatically loads this file when you start Virtuoso,.

It is possible to combine both the pre-seeding methods but Cadence recommends to avoid using both methods at the same time. If you combine both the methods, `oasisOutKeys` SKILL list takes precedence over template file.

The following is a sample `oasisOutKeys` SKILL list:

Design Data Translators Reference

Design Translation Using XOasis Translator

```
;*****
; Copyright (C) 2013 - 2018                                     *
; Cadence Design Systems Inc. All Rights Reserved.               *
;*****
;
;*****
;
; filename : oasisOut.il
;
;*****

oasisOutKeys=list(nil
'hierDepth          "32767"          ;-(0-32767) - Hierarchy Depth Limit
'GDSIICompatible    nil              ;- Generate the oasis file with GDSII specific limits
'arrayInstToScalar  nil              ;- Convert Arrays to Scalar Instances
'autoGenOASISLayerName nil          ;- Automatically Generate OASISLayerName for layers and objects
mappings
'backupOasisLogFiles nil             ;- Backup OASIS and Log Files
'case               "preserve"       ;- preserve | upper | lower - Case Sensitivity
'cellListFile       ""               ;- The Cell List File Name
'cellMap            ""               ;- Cell Map File Name
'cellNamePrefix     ""               ;- Prefix to be added to each STRUCT
'cellNameSuffix     ""               ;- Suffix to be added to each STRUCT
'checkPolygon       nil              ;- Report Bad Polygons
'circleToPolygon    nil              ;- Convert Circles/Ellipses to Polygons
'colorAnalysisGroup ""               ;- Name of the color analysis group
'compress           nil              ;- Enable CBLOCK compression
'compressLevel      "1"              ;- Set Compression level for CBLOCK Compression
'convertDot         "polygon"        ;- polygon | circle | ignore - Convert Dots to
'convertPcellPin    ""               ;- geometry | text | geometryAndText | ignore - Convert Pcell Pin to
'convertPin         "geometry"       ;- geometry | text | geometryAndText | ignore - Convert Pin to
'dbuPerUU           "1000"          ;- DB Units Per User Units
'diagonalPathToPolygon nil           ;- Convert Paths with non-orthogonal segments to polygons
'doNotPreservePcellPins nil          ;- Do Not Preserve the PCell Pins
'donutNumSides      "64"             ;- Number of sides for the BOUNDARY of donut
'ellipseNumSidess   "64"             ;- Number of sides for the BOUNDAR of circle/ellipse
'enableColoring     nil              ;- Enable Coloring Support
'flattenPcells      nil              ;- Flatten the PCell Instances
'flattenPcellsExclusionFile ""        ;- A file specifying list of Pcells to exclude from flattening
'flattenViaShapesToViaPurp nil       ;- Flatten Vias' Shapes to Via Purpose
'flattenVias        nil              ;- Flatten the Vias
'ignoreCurrentTimestamp nil          ;- Ignores current timestamp
'ignoreLines        nil              ;- Ignore Lines During Translation
'ignoreMissingCells nil              ;- Ignore Missing cellViews during Translation and continue
Translation
'ignoreMissingVias  nil              ;- Ignore Missing standard/custom vias during Translation and
continue Translation
'ignoreObjectMapFromTech nil         ;- Ignore object map file from technology
'ignorePcellEvalFail nil             ;- Ignore the PCell Evaluation Failure
'ignoreRoutes       nil              ;- Ignore Routes During Translation
'ignoreTopCellPrefixSuffix nil       ;- Ignores cellName Prefix and Suffix for topCell
'ignoreZeroWidthPathSeg nil          ;- Ignore zero width PathSeg during translation
'infoToWarn         ""               ;- Treat the Specified Info Messgaes Ids as the Warnings
'labelDepth         "1"              ;- Hierarchical depth to add labels to
'labelMap           ""               ;- Label Map File Name
'layerMap           ""               ;- Layer Map File Name
'library            "oasisOutLib"    ;- Library Name
'logFile            "oasisOut.log"    ;- Log File Name
'mergePathSegsToPath nil             ;- Merge the Connected Pathsegs to Path
'noConvertHalfWidthPath nil          ;- Do not Convert The Half Width Path to Boundary
'noInfo             ""               ;- Do not Print the Specified Info Messages Ids
'noObjectProp       nil              ;- Do not Write Special OA Object Properties
```

Design Data Translators Reference

Design Translation Using XOasis Translator

```

'noOutputTextDisplays      nil          ; - Ignore Text Display Objects
'noOutputUnplacedInst      nil          ; - Ignore the Unplaced Instances
'noWarn                    ""           ; - Do not Print the Specified Warning Messages Ids
'oasisFile                  "test.oasis" ; - Oasis File Name
'objectMap                  ""           ; - Object Map File Name
'outputDir                  "."          ; - The Output Directory for Oasis File
'pathToPolygon              nil          ; - Convert Paths to Polygons
'preserveTextAtt            nil          ; - Preserve Text Attributes as Properties
'preservePinAtt             nil          ; - Preserve Pin Connectivity Information
'propMap                    ""           ; - Property Map File Name
'refLibList                 ""           ; - Reference Library List File Name
'replaceBusBitChar          nil          ; - Replace "<>" With "[]"
'runDir                     "."          ; - Run Directory
'showCompletionMsgBox       t           ; - Show the Completion Message Box
'subMasterSeparator         "_CDNS_"     ; - Separator to used for sub-master naming
'snapToGrid                 nil          ; - Snap XY to Grid Resolution
'summaryFile                ""           ; - Summary File Name
'techLib                    ""           ; - Technology Library Name
'textCase                   "preserve"   ; - preserve | upper | lower - Label Case Sensitivity
'topCell                    "top"        ; - Top Cell(s) to Translate
'translateUnmappedLPP       nil          ; - Allow undefined LPP to be translated by incrementally
mapping LPP using layerMap file, SKILL hook poLayerMap, and automatic layer mapping
'userSkillFile              ""           ; - Users' Skill File Name
'verbose                    nil          ; - Generate detailed information in the log and summary files
'viaMap                     ""           ; - Via Map File Name
'view                       "layout"     ; - View Name(s) to Translate
'virtualMemory              nil          ; - Do OASIS Out from Virtual Memory
'warnToErr                  ""           ; - Treat the Specified Warning Messgaes Ids as the Errors
'wildCardInCellMap          nil          ; - Wild Card in Cell Map
)

```

Variant Generation during XOasis Out

XOasis Out generates the color variant that is used to support unlocked color shapes. The effective color of the unlocked color shapes is hierarchy and context dependent. This implies that the unlocked color shapes in a cell can have different colors based on the hierarchy at the topCell and its context. In addition, a cell with unlocked color shapes can have many color variants based on the effective color of the unlocked shapes, and XOasis Out has to write all those color variants in order to preserve the effective color.

Therefore, to have a OASIS with effective color, the following options are required:

- The `-topCell` option to specify the hierarchy and context.
- The `-enableColoring` option to enable the coloring mode.
- The layer map file with the unlocked color mapping.

The array instances can be written as multiple SREF records in the OASIS file whenever there is a need for variant generation. This is because both the `-enableColoring` and `-topCell` options are specified and the layer map file has unlocked mapping entries. Therefore, different color variants might need to be generated for each element of an array.

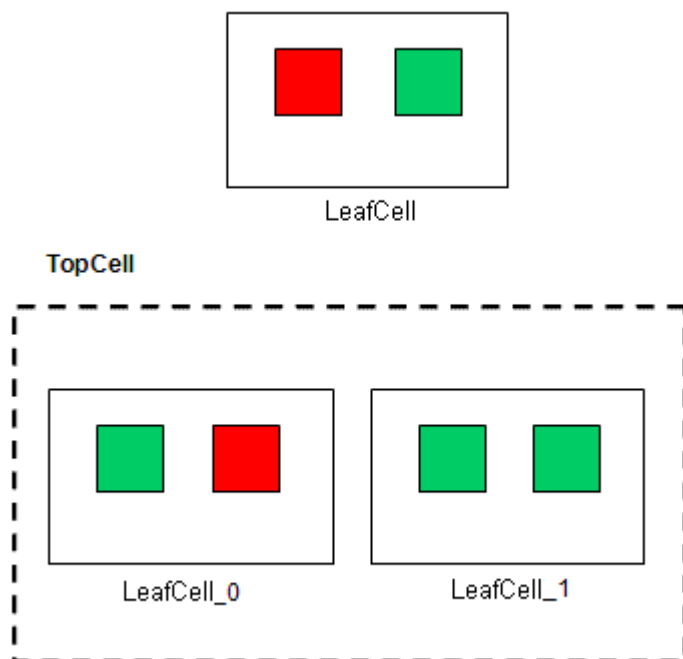
Design Data Translators Reference

Design Translation Using XOasis Translator

In addition, the OASIS can be generated without effective color or hierarchy or context-specific color for the unlocked color shapes. This is because the `-topCell` option is not specified but the `-enableColoring` option is specified and the layer map file has unlocked mapping entries. Therefore, the hierarchy context should be defined using the `-topCell` option.

In case of the unlocked color state, XOasis Out also saves the information related to the color variant. Let us consider a scenario where a cell has two rectangles with unlocked green and red colors. Now, in the top cell, the placement of this cell may cause violation of the minimum color distance constraint. Due to this, the effective color of the shapes may change and color variants of the same instance are generated accordingly. XOasis Out will be able to save this information and write color variants as different structs in the OASIS file.

Consider the following layout where the rectangle is drawn on the `Metal1:drawing` layer-purpose pair.



Consider the following entries in the layer map file:

Metal1	drawing	1	1	mask1Color	unlocked
Metal1	drawing	2	1	mask1Color	unlocked

XOasis Out will write color variants of the instance, `LeafCell`, as different structs in the following OASIS file:

Design Data Translators Reference

Design Translation Using XOasis Translator

STRNAME LeafCell_0

BOUNDARY LAYER=1 DATATYPE=1 XY=... ENDEL

BOUNDARY LAYER=2 DATATYPE=1 XY=... ENDEL

ENDSTR

STRNAME LeafCell_1

BOUNDARY LAYER=1 DATATYPE=1 XY=... ENDEL

BOUNDARY LAYER=1 DATATYPE=1 XY=... ELDEL

ENDSTR

STRNAME TopCell

SREF SNAME=LeafCell_0 X=... Y=... ENDEL

SREF SNAME=LeafCell_1 X=... Y=... ENDEL

ENDSTR

Hierarchical Coloring during XOasis Out

In XOasis Out, hierarchical mode is the mode when you are translating a fully colored hierarchical chip and are exporting colors on all the shapes. In a hierarchical design, effective colors is shifted and color variants are generated for the cells.

To specify the hierarchical coloring mode during XOasis Out, specify `-enableColoring -topCell <cellname>` during translation, and include both the locked and unlocked mapping specifications in the layer map file.

If the shape that is being translated belongs to the pre-colored shape categories, these shapes will be mapped according to the locked specifications in the layer map file.

Note: The two user command-line options `-enableColoring` and `-topCell`, and the layer map file determine the multi-patterning mode and behavior of translation using XOasis Out.

Pre-coloring during XOasis Out

In XOasis Out, if you are translating a full chip starting with the top-cell instead of individual cells, you can also export coloring on a set of pre-colored shapes, and map those shapes based on the locked color specifications specified in the layer map file.

The pre-colored shapes are categorized as:

- Locked shapes, which are the shapes with `locked` color state. These shapes are user-specified colored and locked shapes.
- Anchored shapes

When translating the coloring information on a hierarchical chip, XOasis Out, uses the effective color of the shapes by default. One exception of to the default behavior is when the color state of the shape is `locked`. In this case, the effective color on the shape should always be the same as the assigned color, so that the translator can use the assigned color instead.

To specify the pre-coloring mode during XOasis Out, specify `-enableColoring -topCell <cellname>` during translation and include the `locked` mapping specifications in the layer map file.

OpenAccess to OASIS Mapping

This section expresses the relationship between OpenAccess objects and OASIS records. For additional information about the Oasis Out translator, refer to the [Design Translation During XOasis Out](#) section.

Mapping OpenAccess Shape Objects

Most OpenAccess `oaShapes` have a one-to-one mapping with an OASIS equivalent records. For shapes that do not have an equivalent record, the POLYGON record is used. The following table provides a one-to-one mapping information for supported shapes:

OpenAccess Object	OASIS Record	Description
oaArc	No OASIS equivalent	Not supported

Design Data Translators Reference

Design Translation Using XOasis Translator

oaDonut	No OASIS equivalent	An oaDonut is mapped to a POLYGON with 64 sides. The object will appears as an oaPolygon in round trip.
oaDot	No OASIS equivalent	oaDot can be converted to CIRCLE, POLYGON or ignored depending on the option provided. The object appears as a zero-sized oaEllipse in round trip.
oaEllipse	CIRCLE (If the oaEllipse has a square bounding box)	If the ellipse is not square, the figure is approximated using a POLYGON record with 64 sides.
oaLine	No OASIS equivalent	An oaLine will map to a zero width PATH record
oaPolygon	POLYGON	
oaRect	RECTANGLE	
oaText	TEXT	
oaEvalText	No OASIS equivalent	The oaEvalText is mapped to a TEXT record. The object will appears as oaText in round trip.

Mapping Paths

The following table defines the mapping among OASIS path extensions and OpenAccess path styles:

OpenAccess Path Style	OASIS Path Extension
oacTruncatePathStyle	Flush extension
oacExtendPathStyle	Half-width extension
oacVariablePathStyle	Explicit path extension
oacRoundPathStyle	Polygon

Mapping Pathsegs

The following table defines the mapping among OASIS and OpenAccess begin and end styles:

oaPathSeg Shapes	OASIS Path Extension
oacTruncateEndStyle	Flush extension
oacExtendEndStyle	Half-width extension
oacVariableEndStyle	Explicit path extension
oacChamferEndStyle or oacCustomEndStyle	Polygon

Mapping Text Displays

OpenAccess Text Display	OASIS Mapping
oaPropDisplay	TEXT
oaAttrDisplay	TEXT
oaTextOverride	TEXT
oaInstPropDisplay	TEXT
oaInstAttrDisplay	TEXT

Note: The *noTextDisplay* command line option enables you to stop text display mapping output in OASIS file.

Other Figures

All other figures do not have an OASIS equivalent record. XOasis supports object mapping for some of these objects through an object mapping file. Since these objects are mapping to an OASIS geometry element, all resulting records have OASIS layer and datatype numbers.

Design Data Translators Reference

Design Translation Using XOasis Translator

oaBlockage

The following table lists the OASIS records created through object mapping.

OpenAccess Object	Object Type	OASIS Record
oaAreaBlockage	PLACEMENT	POLYGON
oaAreaHalo	PLACEMENT	POLYGON corresponding to the offsets around the block's PRBoundary.
oaLayerBlockage		RECTANGLE/POLYGON
oacRoutingBlockageType	routing	
oacViaBlockageType	viaRouting	
oacFeedthruBlockageType	feedthru	
oacFillBlockageType		
oacPinBlockageType	fill	
oacScreenBlockageType	pin	
oacSlotBlockageType	screen	
oacWiringBlockageType	slot	
	wiring	
oaLayerHalo		POLYGON corresponding to the offsets around the associated instance or master's PRboundary.
oacRoutingBlockageType	routing	
oacViaBlockageType	viaRouting	
oacFeedthruBlockageType	feedthru	
oacFillBlockageType		
oacPinBlockageType	fill	
oacScreenBlockageType	pin	
oacSlotBlockageType	screen	
oacWiringBlockageType	slot	
	wiring	

Design Data Translators Reference

Design Translation Using XOasis Translator

oaBoundary

The following table lists the OASIS records created through object mapping:

OpenAccess Object	Object Type	OASIS RECORD
oaAreaBoundary	area	RECTANGLE / POLYGON
oaPRBoundary	PR	RECTANGLE / POLYGON
oaSnapBoundary	snap	RECTANGLE
oaClusterBoundary	cluster	RECTANGLE / POLYGON

oaMarker

The Oasis Out translator maps oaMarker objects with a layer mapping to OASIS POLYGON records. The object mapping functionality should be used to restore the POLYGON record

Design Data Translators Reference

Design Translation Using XOasis Translator

back to the corresponding oaMarker. The following table provides a detailed description on mapping oaMarker objects:

OpenAccess Object	Object Type	OASIS RECORD
oaMarker with severity		RECTANGLE / POLYGON
oacAcknowledgedWarningMarkerSeverity	acknowledgedWarning	
oacAnnotationMarkerSeverity	annotation	
oacCriticalErrorMarkerSeverity	criticalError	
oacErrorMarkerSeverity	error	
oacFatalErrorMarkerSeverity	fatalError	
oacInfoMarkerSeverity	info	
oacSignedOffCriticalErrorMarkerSeverity	signedOffCriticalError	
oacSignedOffErrorMarkerSeverity	signedOffError	
oacWarningMarkerSeverity	warning	

oaRow

As described in the following table, the Oasis Out translator maps an oaRow with a layer mapping to an POLYGON record.

OpenAccess Object	OASIS RECORD
oaRow	RECTANGLE / POLYGON

Unsupported Object Types

The following OpenAccess objects types are not supported:

- oaFigGroup
- oaGuide
- oaRoute
- oaSteiner

Layer Mapping in XOasis

Layer mapping during XOasis translation can be done by using the following ways:

- By using user-defined layer map file. (see [Layer Map File](#))
- By using the user-defined [piLayerMap/poLayerMap](#) function in SKILL file
- By performing automatic layer mapping. (see [Automatic Layer Mapping in XOasis](#))
- By using the `techlibname.layermap` file, if it is defined in the technology

Here, the layer map file and the SKILL file are user-defined. While, `techlibname.layermap` file is system defined and is available in the technology by default. The translation process gives priority to user-defined input.

Default Precedence for Layer Mapping

By default, XOasis applies only one type of layer mapping at a time. Given below is the sequence in which it gives precedence or priority to the different methods used for layer mapping:

1. If you provide a layer map file, the layer map file is used for layer mapping.
2. If you provide a user-defined SKILL file, layer mapping is done using the [piLayerMap/poLayerMap](#) function defined in the SKILL file. This type of layer mapping is also referred to as dynamic layer mapping.

Note: If you provide a layer map file as well as a SKILL file for layer mapping, the layer map file will take precedence over the SKILL file. The [piLayerMap/poLayerMap](#) SKILL function will not be considered in this case.

3. If you do not provide a layer map file or SKILL file, XOasis will automatically search for the `techlibname.layermap` file in the technology library. Additionally, in case of Incremental Technology Database (ITDB) hierarchy, XOasis will traverse through all the layer map files in the hierarchy. If a layer map file is present in the technology library with some other name, it will not be detected by default.
4. If both the layer map file and the SKILL file are not found, automatic layer mapping takes place.

Note: If you want to force XOasis to map layers automatically and ignore the `techlibname.layermap` file, you can override the default behavior of XOasis by specifying the `XOASIS_AUTO_LM` keyword for the `-layerMap` option. When you specify the `XOASIS_AUTO_LM` keyword:

- ❑ Even if a layer map file with the name `techlibname.layermap` exists, it will be ignored.

- ❑ XOasis will use its internal layer mapping scheme.

Layer Mapping when Translate Unmapped LPP is Selected

By default, XOasis applies only one type of mapping during translation. Therefore, if you specify the layer map file or the SKILL file, the tool will refer to it, and perform the translation. In this case, any unmapped layers are ignored and not translated.

However, if you choose to translate unmapped LPPs, the tool will first look for and apply any user-defined mapping (layer file or SKILL file), based on the default precedence. Next, the tool will perform automatic layer mapping, which will then translate all unmapped layers also.

Automatic Layer Mapping in XOasis

XOasis uses number-based mapping to perform automatic layer mapping in XOasis In and XOasis Out.

Automatic Layer Mapping in XOasis In

While performing automatic layer mapping, XOasis In tries to use the `oasisLayer/oasisType` number for the `dfIILayer:dfIIPurpose` number and creates `L<oasisLayer>/P<oasisLayer>` (layer:purpose), if required.

For the reserved `dfIILayer:dfIIPurpose` number, it creates `L<oasisLayer>_XSTIN/P<oasisType>_XSTIN` `dfIILayer/dfIIPurpose`.

During XOasis In, if you do not provide a layer map file, then for each Oasis layer or Oasis datatype present in the OASIS file, a new layer-purpose pair is created in the technology file. For example, if Oasis layer is 2 and Oasis datatype is 3, then layer L2 and purpose P3 is created in the technology file. For every new layer-purpose pair, XOasis In populates the `techDisplays` section of the technology file. The display packets **y0-y9** defined in the default `display.drf` file are used to display the newly created layer-purpose pair.

Also, if you select the Merge Purposes To Drawing option then all the undefined purposes are mapped to drawing.

Automatic Layer Mapping in XOasis Out

By default, while performing automatic layer mapping, XOasis Out tries to use the `dfIILayer:dfIIPurpose` number for the `oasisLayer/oasisType` number. For `dfIILayer:dfIIPurpose` `L<oasisLayer>_XSTIN/P<oasisType>_XSTIN` it uses `oasisLayer/oasisType`.

Automatic layer mapping works the same way when you use the `-translateUnmappedLPP` option to map unmapped layers during XOasis Out.

For example, if there is an unmapped layer:

`M1/drawing1` with the `dfIILayer/dfIIPurpose` number as `30/1`, then, all shapes on `M1/drawing1` will be mapped to `oasisLayer/oasisType 30/1`

However, if there is an unmapped layer:

L100_XSTIN: P100_XSTIN with a random dfIILayer/dfIIPurpose number, then, all shapes on L100_XSTIN/P100_XSTIN will be mapped to oasisLayer/oasisType 100/100.

After XOasis Out, the mapping information is available in the `xOasisOut_layermap.txt` file. During automatic layer mapping, for each (layer, purpose) pair in OA design, if the layer number and purpose number are within the valid range for xOasis (0 to $2^{16}-1$), they are mapped to the corresponding xOasis layer and datatype. Purpose drawing is mapped to purpose 0.

Also, those layer numbers and purpose numbers that are in the range of 2^{16} to $2^{32}-1$, are mapped to the available xOasis layer and purpose in the range, 0 to $2^{16}-1$.

Note: If `xOasisOut_layermap.txt` is not writable, then a new file `xOasisOut_layermap.userName.txt` is generated. However, if `xOasisOut_layerMap.userName.txt` is also not writable, a new file `xOasisOut_layermap.userName.n.txt` is generated automatically and the mapping information is added to this file. Here, *n* stands for 1, 2 and so on.

Using Map Files During Translation

Map files control the behavior of XOasis In and XOasis Out. Using map files, you can map layers, cellview names, fonts, and properties in the OpenAccess database to the OASIS layers, structure names, OASIS fonts, and attributes in the OASIS file to each other. The mapping support by using the SKILL file is yet to be implemented.

The various types of map files that are used during translation are:

- Layer Map File
- Label Map File
- Property Map File
- Cell Name Map File
- Via Map File
- Object Map File

Layer Map File

A layer map file is an ASCII file that maps the layer numbers and data types used in the OASIS file to the layer-purpose pairs in OpenAccess, and vice-versa. You can provide multiple layer map files for mapping. In addition to mapping layers for translation, you can also map layers that merge many layers into a single layer.

Layer Map File Format and Conventions

A layer map file has the following format:

```
<Layer Name> <Purpose Name> <OASIS Layer Number> <OASIS DataType>  
[Material Type] [Mask Number] [Qualifier] [Photomask Color] [Color  
State]
```

A layer map file must follow the conventions described below.

- It must have the following four fields:
 - ❑ OpenAccess layer name
 - ❑ OpenAccess purpose name
 - ❑ OASIS layer number
 - ❑ OASIS data type

Design Data Translators Reference

Design Translation Using XOasis Translator

- It can also include the following optional fields:
 - ❑ Material Type
 - ❑ Mask Number
 - ❑ Qualifier
 - ❑ Photomask Color
 - ❑ Color State
- The fields must be separated by one or more spaces or tabs.
- Each mapping record must be a single line.
- The OASIS layer number and the OASIS data type must be a positive number including 0.
- Any record starting with a # is considered a comment and is ignored.

Using Mandatory Fields In the Layer Map File

You can map a DFII layer-purpose pair to more than one OASIS layer and data type pair. To do this, separate the ranges with hyphens, and the individual numbers and ranges with commas. The layer map file must contain all the layers that occur in the data translation. *Export - OASIS* converts only layers listed in the layer map file. (It does not convert layers that are not listed in the layer map file.) The following is an example of an *Export - OASIS* layer mapping file:

# DFII #Layer Name	DFII Layer-Purpose	OASIS Layer Number	OASIS Data Type
metal1	drawing	1, 2	0
metal1	drawing	3, 4-9	0
text, diff	drawing	5	0
metal, pwell	drawing	6	0

Design Data Translators Reference

Design Translation Using XOasis Translator

The first entry in the above example, is the same as the following two entries:

# DFII #Layer Name	DFII Layer-Purpose	OASIS Layer Number	OASIS Data Type
metal1	drawing	1	0
metal1	drawing	2	0

Similarly, you can map more than one OASIS layer to one DFII layer-purpose pair. The following is an example of an *Import - OASIS* layer map file.

# DFII #Layer Name	DFII Layer-Purpose	OASIS Layer Number	OASIS Data Type
metal	drawing	13, 10, 18	1
poly	boundary	40	0-63

The first entry in the above example, is the same as the following three entries:

# DFII #Layer Name	DFII Layer-Purpose	OASIS Layer Number	OASIS Data Type
metal	drawing	13	1
metal	drawing	10	1
metal	drawing	18	1

Sample Layer Map File

#Open Access Layer Name	Open Access Purpose Name	OASIS Layer	OASIS DataType
text	drawing	5	0

Important

The same OASIS layer number and data type pair might be mapped to more than one DFII on OpenAccess layer-purpose pairs. In this scenario, XOasis In translates all the entries for that OASIS layer to only the first DFII layer-purpose pair specified in the layer map file.

If you want XOasis In to retain the different layer-purpose pairs mapped to the same OASIS layer number and data type pair, update the layer name map file used during the XOasis Out stage itself. Change either the OASIS layer number or the data type for all such layer-purpose pairs. Use the updated layer name file to again run XOasis Out and then XOasis In to obtain the desired results.

For example, if the layer map file looks like:

#OpenAccess Layer Name	OpenAccess Purpose Name	OASIS Layer Number	OASIS DataType
metal1	drawing	2	0
metal2	boundary	2	0
metal3	boundary	2	1

where, `metal1` and `metal2` have the same OASIS layer number and data type. Even though `metal3` has the same OASIS layer number but it differs from `metal1` and `metal2` in the OASIS data type. Thus, XOasis In translates the OASIS layer number 2 with data type 0 only to the first layer-purpose pair mapped to this OASIS layer and data type, that is, `metal1` drawing. XOasis In translates OASIS layer number 2 with data type 1 to `metal3` boundary.

Using Optional Fields in Layer Map File

A layer map file can have `Material Type`, `Mask Number`, `Qualifier`, `Photomask Color`, and `Color State` optional fields.

- The layer map file format with optional fields work with both the command line and GUI. However, in GUI the inline editing support will be there only for a layer and purpose.
- The translation of shapes is based on the `layerMap` entries.
- Material type and Mask number are considered during XOasis In and ignored during XOasis Out.

Design Data Translators Reference

Design Translation Using XOasis Translator

- The qualifier is considered only during XOasis Out and is ignored during XOasis In. Valid values for the `Qualifier` column are `pin`, `floating`, `cutSize:<float>:<float>`, `OASISLayerName`, and `ignoreLPP`.

Here, qualifier:

- ❑ `pin` ensures that the specified shape is a pin. This qualifier supports pins onto a different OASIS layer datatype.
(see [Example for qualifier pin](#))
- ❑ `floating` is only valid for purpose, `fill` and `fillOPC`. It fills onto a different OASIS layer data type. This qualifier is considered during XOasis Out and ignored during XOasis In.

- ❑ `cutSize` is only valid for material, `cut`. The qualifier enables mapping the via cuts with different cut sizes onto different OASIS layer and OASIS data type. The cut sizes (height:width) are calculated from the bounding box of the shapes on a cut layer, whose material type is `cut`.

(see [Example for qualifier cutsize](#))

- ❑ `OASISLayerName` creates a user-specified name to be associated with the given `layername:datanum` in the OASIS file.

`OASISLayerName` does not affect actual layer mapping. This information is used only to create the `LAYERNAME` record.

OASIS layer name is generated automatically when `OASISLayerName` qualifier is not used but [Automatically Generate OASISLayerName](#) (`autoGenOASISLayerName`) is enabled.

(see [Example for qualifier OASISLayerName](#))

- ❑ `ignoreLPP` ignores the specified LPPs during mapping and does not display a missing LPP warning for them. This qualifier is cannot be used with any other optional field.

(see [Example for Qualifier IgnoreLPP](#))

- The `Photomask Color` and `Color State` optional fields are supported both in XOasis In and XOasis Out.

- ❑ Valid values for `Photomask Color` are `mask1Color`, `mask2Color`, `mask3Color`, `mask4Color`, `mask5Color`, `mask6Color`, `mask7Color`, and `mask8Color`.
- ❑ Valid values for the `Color State` field are `locked` and `unlocked`.

Allowed Combinations for the Mandatory and Optional Fields

- ❑ <Material>
- ❑ <Material> <MaskNumber>
- ❑ <Material> <MaskNumber> <Qualifier>
- ❑ <Material> <Qualifier>
- ❑ <Qualifier>
- ❑ <Material Type> <PhotoMaskColor> <ColorState>
- ❑ <Material Type> <MaskNumber> <PhotoMaskColor> <ColorState>
- ❑ <Material Type> <MaskNumber> <Qualifier> <PhotoMaskColor> <ColorState>
- ❑ <Material Type> <Qualifier> <PhotoMaskColor> <ColorState>
- ❑ <Qualifier> <PhotoMaskColor> <ColorState>

The following table illustrates the layer map file format with both the mandatory and optional fields:

# Layer Name	Purpose Name	OASIS Layer Number	OASIS DataType	[Material]	[Mask Number]	[Qualifier]
Metal1	drawing	1	1			
Via1	drawing	2	1	cut		cutSize: 1.0:1.0
Metal2	drawing	3	1	metal		ignoreLPP
Via2	drawing	4	1	cut	4	
Metal3	fill	5	1	metal		floating
Via3	drawing	6	1	cut	6	cutSize: 1.0:1.0
Metal4	drawing	7	1			pin
Metal4	drawing	8	1			

Example for qualifier cutsize

Translation of shapes will be based on the layerMap entries. Cases using different layerMap entries with the cut size qualifier are listed below:

Design Data Translators Reference

Design Translation Using XOasis Translator

- Let us assume, a design has the following shapes on the cut layer, `Via1`, and the purpose, `drawing`:
- R1: a rectangle with a bounding box of `0.5x0.5`
- R2: a rectangle with a bounding box of `1.0x1.0`
- P1: a polygon with a bounding box of `0.5x0.5`
- P2: a polygon with a bounding box of `1.0x1.0`
- P3: a polygon with a bounding box of `1.0x0.5`
- PA1: a path with a bounding box of `0.5x0.5`
- PA2: a path with a bounding box of `1.0x1.0`

■ Case 1:

```
Via1      drawing      2          1          cut
```

In this case, all the shapes (R1, R2, P1, P2, P3, PA1, and PA2) are written on `OASISlayer:datatype, 2:1`.

■ Case 2:

```
Via1      drawing      2          1          cut
```

```
Via1      drawing      2          2          cut      cutSize:1.0:1.0
```

In this case, all the shapes with the bounding box `1.0x1.0` (R2, P2, and PA2) are written on `OASISlayer:datatype, 2:2`. All other shapes use the default mapping, therefore, R1, P1, P3, and PA1 will be written on `OASISlayer:datatype, 2:1`.

■ Case 3:

```
Via1      drawing      2          2          cut      cutSize:1.0:1.0
```

In this case, all the shapes with the bounding box `1.0x1.0` (R2, P2, and PA2) are written on `OASISlayer:datatype, 2:2`. All other shapes are dropped with a one-time warning message because default mapping is not provided.

■ Case 4:

No mapping provided for `Via1 drawing`.

In this case, all the shapes are dropped with a one-time warning message because no mapping is provided.

Example for qualifier pin

Let us assume you have pin shapes and some regular shapes on an LPP (Layer-Purpose pair). Now, if you want XOasis Out to translate the pin shapes on to a different OASIS layer, you can use the `pin` qualifier. For example, if layer map is:

```
M4 drawing 8 1
```

then, all `<M4 drawing>` data should go on `<8:1>`. However, if you want all the pin shapes on the same layer `<M4 drawing>` should go onto the OASIS layer-datatype, `<7:1>`, then the following entry needs to be specified in the layer map.

```
M4 drawing 7 1 pin
```

Cases using different layerMap entries with the pin qualifier are listed below:

■ **Case 1:**

Let us assume you have the following entries in the layer map file.

```
M4 drawing 8 1
```

```
M4 drawing 7 1 pin
```

Then, all regular shapes will go to the OASIS layer, `<8:1>`. However, all pin shapes will go to `<7:1>`.

■ **Case 2:**

Let us assume you only have the following entry in the layer map file.

```
M4 drawing 8 1
```

Then, all the shapes including pin shapes will go to the OASIS layer, `<8:1>`.

■ **Case 3:**

Let us assume you only have the following entry in the layer map file.

```
M4 drawing 7 1 pin
```

Then, all pin shapes will go to the OASIS layer, `<7:1>`. All other shapes will be dropped.

Example for qualifier OASISLayerName

■ **Case 1:**

Consider the following entries in the layer map file:

Design Data Translators Reference

Design Translation Using XOasis Translator

```
Metal1 drawing 1 10 OASISLayerName:metal1
```

```
Metal2 drawing 2 10 OASISLayerName:metal2_mask1_locked mask1Color  
locked
```

These layer map file entries convey the following information:

- ❑ Shapes on Metal1:drawing should be mapped to 1:10 during translation,
- ❑ Shapes on Metal2:drawing, mask1Color, locked should be mapped to 2:10 during translation,
- ❑ Record LAYERNAME metal1 3(1) 3(10) to be created in the OASIS file.
- ❑ Record LAYERNAME metal2_mask1_locked 3(2) 3(10) to be created in the OASIS file.

■ Case 2:

If the same layer-number and data-number are mapped to different OASISLayerName, only the first entry is considered to create the LAYERNAME record in the OASIS file. All subsequent entries for this layer number and data number are ignored with a warning. However, the object map entry is still considered valid for object mapping. Now consider the following entries in the layer map file:

```
M1 drawing 5 5 OASISLayerName:Metal1
```

```
M2 drawing 5 5 OASISLayerName:Metal2
```

For the above layer map entries, only one LAYERNAME record will be created in the OASIS file as follows:

```
LAYERNAME name="Metal1" layers=3(5:5) datatypes=3(5:5)
```

However, layer-mappings `Metal1:drawing to 5:5` and `Metal2:drawing to 5:5` are still valid. All `Metal1:drawing` and `Metal2:drawing` shapes will be mapped to `5:5` in the OASIS file.

Example for Qualifier IgnoreLPP

Consider there are three shapes in a design, one each on:

`m1 drawing`

`m2 drawing`

`m3 drawing`

Assume you have the following entries in the layer map file

`m2 drawing 1 2`

`m3 drawing 3 4 ignoreLPP`

Then, the shape on `m2 drawing` will be translated successfully and the shapes on the `m1 drawing` and `m3 drawing` LPPs will not be translated. A warning informing that objects in a specific LPP are ignored, will be displayed only for the shape on the `m1 drawing`. The warning will not be displayed for the shape on `m3 drawing` because the `ignoreLPP` qualifier is specified for this entry in the layer map file.

Enhanced Layer Mapping for XOasis In

The layer-map functionality is enhanced to set the color information on the created shape based on the mapping entry used for `OASISLayer` or `OASISType`. XOasis In supports color mapping for both locked and unlocked color states, so that created shapes can have either the locked or unlocked color state.

In the layer-map file, XOasis In considers all the color mapping entries, if the `-enableColoring` option is specified. Otherwise, all the color mapping entries will be ignored and the warning message will be displayed.

The Photo stmask color information, which is specified in an entry of the `layerMap` file is validated against the number of the photo mask color in the technology library for the particular `oaLayerName`. In case there is any inconsistency in the entry of the `layerMap` file, the entry will be ignored and a warning message will be displayed.

In addition, XOasis In uses the color information of the mapped entry to set the color information on the created shapes. Valid values for the `Color State` field are `locked` and `unlocked`. Therefore, in XOasis In both the locked color shapes and unlocked color shapes can be created. Also, coloring can be done on the third-party OASIS with the color state as either locked or unlocked. Here, the key to differentiate the color information is `OASISLayer/OASISType`.

Enhanced Layer Mapping for XOasis Out

The layer-map functionality is enhanced to map color shapes to `OASISLayer` or `OASISType` based on the color information of the shape. XOasis Out supports color mapping for both locked and unlocked color shapes, according to the mappings specified in the layer map file.

In the layer map file, XOasis Out considers all the color mapping entries, if the `enableColoring` option is specified. Otherwise all the color mapping entries will be ignored and a warning message will be displayed.

The photo mask color information, which is specified in an entry of the `layerMap` file is validated against the number of photo mask color in the technology library for the particular `oaLayerName`. In case there is any inconsistency in the entry of the `layerMap` file, then the entry will be ignored and a warning message will be displayed.

If XOasis Out encounters an unlocked color shape, then it will try to use non-color mapping and the one-time warning message will be displayed for it. If non-color mapping is also not available for that `oaLPP`, then a warning will be displayed and the un-mapped shapes will be ignored.

XOasis Out follows the following order to map the locked color shape on a specific `OASISLayer/OASISType`.

1. The entry with the Photo Mask Color and Color State fields of the shape will be considered. If such color-aware mapping is not available for the `oaLPP` with specific color, then a warning will be displayed on per-shape basis.
2. The entry without the Photo Mask Color and Color State fields will be considered. If non-color aware mapping is not available for that `oaLPP`, then this un-mapped shape will be ignored and a warning will be displayed on a per shape basis.

The color-aware mapping will use the following order to consider the *qualifier* field:

1. **Color Pin Mapping:** The mapping entry with `Photomask Color`, `Color State`, and `Pin` as `Qualifier`.
2. **Color cutSize Mapping:** The mapping entry with `Photomask Color`, `Color State`, and `cutSize:<float>:<float>` as `Qualifier`.

- 3. Normal color mapping:** The mapping entry with `Photomask Color` and `Color State`.

Non-Color aware mapping will use the following order to consider the *qualifier* field:

- 1. Pin Mapping:** The mapping entry with *Pin* as *Qualifier*.
- 2. CutSize Mapping:** The mapping entry with `cutSize:<float>:<float>` as *Qualifier*.
- 3. Normal/default mapping:** The mapping entry with only four must fields and without *Qualifier* field.

Label Map File

The label map file is an ASCII file that generates labels for the top-level nets and terminals of a design. Use the `-labelMap` option to specify the labels are included during XOasis Out. You can use the `-labelDepth` option with XOasis Out to specify the hierarchy level that can be used for procuring the label information. The following table describes the two options:

Option	Description
<code>-labelDepth <i>number</i></code>	Specifies the depth of the design hierarchy labels that are generated. If the <code>-hierDepth</code> option is also used, and it is less in value, labels are generated for all hierarchy levels. If the <code>-labelDepth</code> option is not specified, labels are generated for only the top level of the design.
<code>-labelMap <i>filename</i></code>	Specifies a label map file that determines the object type for which labels are created and the mapping of shapes on an <code>oaLayer</code> and <code>oaPurpose</code> pair to a OASIS layer and OASIS data type pair. The object type specified in the label map file is either a terminal or a net.

Labels are created for each top level cell, or each top level cell and its children down to the specified level of hierarchy, if the `-labelDepth` option is used. Labels are created for all nets at the top level. If a net has a terminal and the terminal name differs from the net name, the label that is created corresponds to the terminal name. If a label map file includes more than one mapping for a combination of object, `oaLayer`, and `oaPurpose`, the first of these entries in the label map file is used and a warning is displayed.

Design Data Translators Reference

Design Translation Using XOasis Translator

The label map file format uses one line per mapping where the five fields are each separated from others by tabs or spaces. Lines beginning with the # character are comment lines and are ignored by the translator.

An example of a label map file is shown below.

#	Object	oaLayer	oaPurpose	OASIS Layer Number	OASIS Data Type
	Term	Metal5	drawing	131	0
	Term	Metal6	drawing	132	0
	Net	Metal5	drawing	131	1
	Net	Metal6	drawing	132	1
	Term	Metal5	drawing	131	2
	Term	Metal6	drawing	132	2
	Net	Metal5	drawing	131	3
	Net	Metal6	drawing	132	3
	Term	Metal5	drawing	131	4
	Term	Metal6	drawing	132	4
	Net	Metal5	drawing	131	5
	Net	Metal6	drawing	132	5

The label map file uses five required fields, listed in the following table.

Field Name	Value Type
Object	[Net Term]
Layer	An oaLayer name.
Purpose	An oaPurpose name.
OASIS Layer	A numeric value representing a OASIS layer.
OASIS Data Type	A numeric value representing a OASIS data type.

Property Map File

A property map file is an ASCII file that is used to map object types and property names from the OpenAccess database to the OASIS file. You can create a property map file using any text editor.

In a property map file,

- Each mapping record must be a single line.
- Fields in the mapping record must be separated with one or more spaces or tabs.
- Records starting with # are considered comments and are ignored.
- Multiple properties separated by a comma can be specified in the same mapping record.
- There is one option record and several mapping records.
- Option record is the first line and it is optional. The syntax for an option record is:

```
[-f<Format_Specifier>] [-s '<Separator_Character>']
```

Here, the *Format_Specifier* lets you specify the format in which properties and their values are written to the Oasis file.

For example, consider a property map record:

```
12 rect accessDir
```

This *Format_Specifier* can have the following values:

- -fv: Format the output to show the property values only. The property value is written as a value of the property ID specified in the mapping record. For the given example, this property will be exported as `inputOutput` in the Oasis file.
- -fp (default setting): Format the output to show the name-value pair of the property. For the given example, this property will be exported as `accessDir=inputOutput` in the Oasis file.

The *Separator_Character* specifies the special character to be used as property separator.

- The special character works as property separator in case multiple properties are mapped to the same Oasis property ID. For example, if there are two properties on a record, `prop1=val1` and `prop2=val2`, mapped to same Oasis property ID and property separator is `;`, then this record is exported as `prop1=val1;prop2=val2` (when -fp is also specified), or exported as `val1;val2` (when -fv is specified).

Note: You can specify either *Format_Specifier*, *Separator_Character*,

or both.

- A mapping record in the property map file consists of three fields: property ID, object type and property name, where,

<i>propertyID</i>	Specifies the property ID that is used to store the value of the property. This can be any number from 0 to 125. This field is optional and is specified to import or export GDSII-Stream-style properties in the OASIS file.
<i>objectType</i>	Specifies the object type for which a particular property is to be translated. Valid values of OA object types are Arc, Donut, Dot, Ellipse, Line, Path, Polygon, Rect, Text, ArrayInst, ScalarInst, PathSeg, Route, Marker, Row, AreaBoundary, PRBoundary, SnapBoundary, ClusterBoundary, AreaBlockage, AreaHalo, LayerBlockage, LayerHalo, and Pin.
<i>propertyName</i>	Specifies the property name to be preserved. Valid values are any ASCII string.

In OpenAccess, database objects, such as instances and shapes, can have certain attributes attached to them. By default, these attributes are not translated during XOasis Out. A property map file provides a mechanism to translate and preserve these attributes during XOasis Out and XOasis In. It is also used to map object types and attributes names from the OpenAccess database to the OASIS property names in the OASIS file.

During XOasis Out, sufficient information is written to the OASIS file so that the properties are preserved if you translate these properties as property name-value pairs. Therefore, while using XOasis In with the same information, you must not specify a property map file. If the same object name is associated with different user property names of the same object, then several properties records are written to the output OASIS file.

Sample Property Map File

```
#ObjectType      #PropName
-fp -s ';'
rect             propName1
polygon          propName2,propName10
net              prop12,netName
scalarInst       instName
```

Design Data Translators Reference

Design Translation Using XOasis Translator

The property map file supports certain keywords that are not OpenAccess properties. These keywords are described below.

- `instName`

You can translate and preserve instance names during XOasis Out and XOasis In. This can be done by specifying `instName` as a property for the object type `scalarInst` and `oaArrayInst` in the property map file.

- `netName`

The `netName` keyword is used to preserve information pertaining to net names of `oaShape` objects. Ideally, property ID should be same for all type of objects. This is because OASIS supports fewer objects and different property IDs can cause issue in preserving the `netName`.

Cell Name Map File

A cell name map file is an ASCII file that stores mapping information between the OpenAccess cellview names and OASIS cell names. A cell name map file can be created by using any text editor. You can also specify the cell name mapping information in a SKILL file. In a cell name map file:

- Each mapping record must be a single line.
- Fields must be separated with one or more spaces or tabs.
- Any line starting with # is considered a comment and is ignored. Blank lines are also ignored.
- Each record must contain the following four fields:
 - Library name
 - OpenAccess cell name
 - OpenAccess view name
 - OASIS structure name

If any record in the cell name map file is not in the proper format, the record is ignored and a warning message is generated. You can map cell names by providing a cell name map file, a user-defined SKILL file, or by using the *Case* option. The *Case* option is used when the cell name map file or the user-defined SKILL file is not used for cell mapping.

While cellview names can change by using a cell name map file or a SKILL file, they can also change automatically. The situations in which cellview names are automatically changed are described below.

- When the OpenAccess design consists of two cellviews that have the same name but are in different libraries. Similarly, two cellviews can have same cell names but different view names. In such a case, cellview names are automatically changed because the OASIS format does not support two structures having same names.

Example 3-17 When two libraries have same cell names

Consider a situation where a library `lib1` has a cell `cell1` and a view `view1`. Similarly, another library `lib2` has a cell `cell1` and a view `view1`. During XOasis Out, `cell1` is automatically mapped and assigned a name because the OASIS format does not support two cells having same names.

Example 3-18 When the library names and cell names are same but view names are different

Consider a situation where a library `lib1` has a cell `cell1` and views `view1` and `view2`. During XOasis Out, `cell1` is automatically mapped and assigned a new name because the OASIS format does not support two cells with the same name.

- When an OpenAccess cell name consists of a character that is not supported in the OASIS format, it is replaced by an underscore. Valid characters (with hexadecimal codes) supported by the OASIS format are:

```
!"#$%&'()*+,-./0123456789:;<=>?@      [21-40]
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_'          [41-60]
□      abcdefghijklmnopqrstuvwxyz{|}~      [61-7E]
```

Sample Cell Name Map File

#Library Name	OpenAccess Cell Name	OpenAccess View Name	OASIS Structure Name
reflib1	cell1	view1	STCELL1
*	cell2	view2	STCELL2
XOASIS_TARGET_LIB	cell3	view3	STCELL3
XOASIS_TARGET_LIB	cell4	view4	STCELL4

During XOasis Out, the cell mapping information of the cells, which are defined in the source library or any other library, and have been automatically mapped, is recorded in the `oasisOut_cellMap.txt` file. All the cells that are present in the source library and are automatically mapped have library name like `XOASIS_TARGET_LIB`. The `XOASIS_TARGET_LIB` will be replaced with the library name specified using the `-lib` option. The `*` keyword denotes that this mapping is applied to the specified cellview present in any reference library.

Via Map File

A via map file stores via mapping information during the XOasis Out - XOasis In round trip. A via represents a physical connection between two adjacent shapes that are on different layers. By default, during XOasis Out vias, are translated as SREFs in the OASIS file and recreated as instances during subsequent XOasis Ins.

To preserve and recreate vias in OA, XOasis requires the `viaDef` names from the technology file and the via parameters. This information can be stored in the via map file during XOasis

Out. Via map file allows you to preserve via information during XOasis Out - XOasis In round trips.

Note: If viadefs is not available, the record in the via map file is ignored and a warning message is issued.

During XOasis Out if you use the `-viaMap` option, then XOasis creates a OASIS file and a via map file. The via map file maps the structure names from the OASIS file to the corresponding viaDef names in the input technology library. This via map file is then used to recreate the original vias during subsequent XOasis Ins.

Important

The viaMap file is meant to preserve the via mapping information only on a XOasis Out - XOasis In round-trip.

Also, when using OASIS files from third-party vendors, you may want some structures to be treated as vias during XOasis In. To do this, you can create viaDefs in the technology library and manually create a via map file using any text editor.

Note: Processing vias is much faster as compared to processing instances. You can have designs containing millions of via-like structures. Therefore, using a via map file during XOasis In ensures that the layout editing and processing tools work faster for designs created by XOasis In.

In a via map file:

- Each mapping record must be a SKILL list.
- Any line starting with # is considered a comment and is ignored. Blank lines are also ignored.
- Each record must contain the following three fields:
 - sub master name
 - viaDef name
 - list of non-default parameter value
- If any record in the via map file is not in the proper format the record is ignored and a warning message is generated.
- For preserving vias ensure that the original technology file in which viaDefs are defined exists. If you do not specify the technology library during XOasis In and the target library does not have a technology file associated with it, then the `-viaMap` option is not available during XOasis In.

Design Data Translators Reference

Design Translation Using XOasis Translator

- Before starting the translation the via map file is loaded and for each entry in the via map file the corresponding viaDefs are searched in the technology file. If a viaDef does not exist in the technology file, then the record in the via map file is ignored and a warning message issued.
- In addition if you instantiate a via with the same parameter values multiple times in a design, then only one entry is created in the via map file during XOasis Out. If the same sub master names are specified multiple times in the via map file, only the first record is considered during XOasis In; all subsequent records are ignored and warning messages issued.

You can specify the name of the via map file from the command line by using the `-viaMap` option during XOasis Out, as shown below.

```
oasisout [-viaMap viaMapFileName]
```

When you use the `-viaMap` option, a via map file is created along with the corresponding OASIS file. If you do not specify the `-viaMap` option, the via map file is not created.

After creating the via map file you can specify the via map file during XOasis In using the following command:

```
oasisin [-viaMap viaMapFileName]
```

XOasis In checks for the via map file and if it exists, the mapping information from the via map file is used for preserving vias. If you do not specify the `-viaMap` option, the vias are not preserved and are translated as instances.

Sample Via Map File

```
viaMapTable = list( nil
  subMasterTable (
    ( "VIAGEN12S_0" "VIAGEN12S" nil )
    ( "VIAGEN12S_1" "VIAGEN12S" (("CutHeight" 8000 "int")) )
    ( "VIAGEN12S_2" "VIAGEN12S" (("CutWidth" 9000 "int")) )
    ( "VIAGEN23_3" "VIAGEN23" (("CutHeight" 5000 "int")) )
    ( "VIAGEN12S_4" "VIAGEN12S" (("CutWidth" 8000 "int")) )
    ( "VIAGEN12S_5" "VIAGEN12S" (("CutHeight" 7000 "int")) )
    ( "VIAGEN23_6" "VIAGEN23" nil )
    ( "VIAGEN23_7" "VIAGEN23" (("CutHeight" 10000 "int")) )
    ( "VIAGEN12B_8" "VIAGEN12B" nil )
    ( "VIAGEN12B_9" "VIAGEN12B" (("CutWidth" 8000 "int")) )
    ( "via1Array_10" "via1Array" (("CutHeight" 1000 "int") ("CutWidth" 1000
"int")) )
    ( "via2Array_11" "via2Array" nil )
```

Design Data Translators Reference

Design Translation Using XOasis Translator

```
( "via2Array_12" "via2Array" (("CutWidth" 1000 "int")) )  
( "via1Array_13" "via1Array" nil )  
);End of subMasterTable  
);End of viaMapTable
```

Object Map File

An object map file is an ASCII file that maps OpenAccess objects and sub-objects to OASIS layer and data type. OA2.2 objects such as boundary, blockage, marker, and row do not represent physical shapes that can be represented in the OASIS format. The objects are design concepts used by Place and Route and other display-editing tools. You may want to preserve these objects while archiving design data or while exchanging data between different tools to complete a design cycle.

The object map file maps the OA object and sub-objects to the OASIS layer-datatype pair and decides the OA object that should be translated as a shape on a specific OASIS layer-datatype pair. You can provide multiple object map files for mapping.

If you do not provide an object map file, XOasis will automatically search for the `techlibname.objectmap` file in the technology library. A object map file, if present in the technology library with some other name, is not detected. In case of Incremental Technology Database (ITDB) hierarchy, XOasis will traverse through all the object map files in the hierarchy.

You can create an object map file using any text editor. In an object map file:

- Any line starting with # is considered a comment and ignored. Blank lines are also ignored.
- Each mapping record should be a single line.
- Fields should be separated with one or more spaces or tabs.
- Each record in the object map file must contain at least four fields.
 - ❑ OA object name
 - ❑ OA object sub-type
 - ❑ [OA layer name]
 - ❑ OASIS layer number
 - ❑ OASIS datatype
 - ❑ [Qualifier]

❑ [Photomask Color]

- If the same object type and object sub-type pair is specified in multiple records, then only the first record is considered; all subsequent records are ignored and warning messages issued.
- If you specify both the object map file and the layer map file during XOasis Out, then ensure that the OASIS layer-datatype pairs specified in the object map file are not used in the layer map file. If the OASIS layer-datatype pairs in the object map file are the same as the OASIS layer-datatype pairs in the layer map file then the entries in the object map file are ignored and warning messages issued. However, the object will be created if the OASIS file was generated by XOasis Out and attributes information was saved in properties.
- If the layer map file is not specified during XOasis Out, the layers are automatically mapped and the layer-datatype pairs specified in the object map file are not used.
- If the object map file is specified during xOasis In, only mapped objects are recreated from properties information. If the object map file is not specified during xOasis In, all objects with attributes saved in properties are automatically recreated.
- If the map file contains entries with invalid object type and sub-type mapped to correct OASIS layer-datatype pair then the records are ignored and error messages are issued.
- The OA Layer Name field is optional and is available only for oaLayerBlockage objects.
- The OA Layer Name field is optional only for XOasis Out. If this field is not specified during XOasis Out, XOasis Out writes all the layer blockages to the same Oasis layer-datatype pairs.
- You can use the optional Qualifier field to specify is *OASISLayerName* for OpenAccess object. See, [Example for qualifier OASISLayerName](#).

Example for qualifier OASISLayerName

■ **Case 1:**

Consider the following entries in the object map file:

```
Boundary Snap 1 10 OASISLayerName:boundary_snap
```

```
Boundary Area 2 10 OASISLayerName:boundary_area
```

These object map file entries convey the following information:

Objects on `Boundary:Snap`, should be mapped to `1:10` during translation,

Objects on Boundary:Area, should be mapped to 2:10 during translation,

Record LAYERNAME boundary_snap 3(1) 3(10) to be created in the OASIS file.

Record LAYERNAME boundary_area 3(2) 3(10) to be created in the OASIS file.

■ **Case 2:**

The OASISLayerName qualifier field is considered for first unique combination of OASIS-Layer and OASIS-Type and it is ignored with a warning for all duplicate combinations of OASIS-Layer and OASIS-Type. However, the object map entry is still considered valid for object mapping.

Now consider the following entries in the layer map file:

Boundary Area 5 5 OASISLayerName:boundary_area

Boundary Snap 5 5 OASISLayerName:boundary_snap

For the above object map entries, only one LAYERNAME record will be created in the OASIS file as follows:

LAYERNAME name="boundary_area" layers=3(5:5) datatypes=3(5:5)

However, object-mappings Boundary:Area to 5:5 and Boundary:Snap to 5:5 are still valid. All Boundary:Area and Boundary:Snap objects will be mapped to 5:5 in the OASIS file.

Enhanced Object Mapping for XOasis Out

The object map functionality is enhanced to preserve the color information of the layerblockages in XOasis Out-XOasis In round trip. In addition, the object mapping functionality is enhanced to set the color information on the layerblockages in case of third-party OASIS or OASIS created using XOasis Out without the -enableColoring or -noObjectProp options.

The following enhancements have been made in the object map file format:

```
<Object Type>    <Object SubType>  [Layer Name]    <Oasis Layer  
Number>          <Oasis DataType> [Qualifier] [Photomask Color]
```

where:

- Columns 1,2,4, and 5 are mandatory.
- Columns 3 and 6 are only for the LayerBlockage Object Type. These are optional for XOasis Out.

Design Data Translators Reference

Design Translation Using XOasis Translator

- Valid values for the `Photomask Color` column are `mask1Color`, `mask2Color`, `mask3Color`, `mask4Color`, `mask5Color`, `mask6Color`, `mask7Color`, and `mask8Color`.
- The object type `Stitch` can be used to map existing stitch objects in XOasis Out.

Enhanced Layer Blockages for XOasis Out

XOasis Out will consider all the color mapping entries in the object map file if the *Enable Coloring* check box is selected. Otherwise, all the color mapping entries will be ignored and a warning message will be displayed.

XOasis Out follows the below mentioned order to map the layer blockages on a specific layer with a specific photo mask color:

1. Entry with layer name and photo mask color
2. Entry with only photo mask color
3. Entry with only layer name
4. Entry with no layer name and no photo mask color

Using Stitch Objects

The object type `Stitch` can be used to map existing stitch objects in XOasis Out. An example of the object map file syntax is described below:

```
<OA Object Name> <OA object sub-type> <OA Layer Name> <Oasis  
Layernumber> <Oasis Datatype>
```

```
Stitch           None           M1           32           5
```

In the above example, the stitch region associated with stitch objects on layer `M1` is mapped as a shape on the oasis layer number `32` and oasis data type `5`.

When the OASIS file is input back into OA using XOasis In, the layer mapping file is used to map the stitch region back to a specified purpose. In the example below, the purpose is `stitch`.

```
<OA Layer Name> <OA Purpose Name> <Oasis Layernumber> <Oasis  
Datatype>
```

```
M1           stitch           32           5
```

For more information, see [Handling Stitches](#) in *Virtuoso Multi-Patterning Technology User Guide*.

Object Map File - OA Object Property Data

You can specify the object file by using the `-objectMap` command-line option. If you do not specify the object map file, shapes are translated into the OASIS file and OA objects, such as blockages, boundaries, rows, and markers are not translated. Any data related to the OA objects are dropped during translation.

If you specify an object map file using the `-objectMap` option during XOasis Out, shapes as well as OA objects are translated as shapes. Also, the property data is written to the OASIS file as property data on shapes that correspond to the OA objects. This property information is used during XOasis In to preserve the OA objects.

The default property name for storing P&R objects attributes data is `oaObjAttr`. This property contains a set of object type and subtype specific attribute values in the form of ASCII strings separated by the ':' symbol. Instance name required for `areaHalo` roundtrip is automatically preserved by using the `oaObjAttr` property, which is associated with SREF or AREF record.

To translate OA objects as shapes, you can use the `-noObjectProp` option along with the `-objectMap` option during XOasis Out. If you use the `-noObjectProp` option, then although the shapes and OA objects are translated, no property data is written to the OASIS file for OA objects.

If the design contains area halos and the `-objectMap` and the `-noObjectProp` options are not specified, then to preserve area halos in the XOasis Out - XOasis In round trip, the `-propMap` option is required. If you do not specify the `-propMap` option during XOasis Out, the following message is displayed.

Property map file is needed to preserve `areaHalo` blockages. If the property map file is not specified, then you need to specify the file at the `-propMap` option.



Ensure that you specify the option `-noObjectProp` only when using the `-objectMap` option. You will get an error message if you use `-noObjectProp` option without the `-objectMap` option.

XOasis Out Object Types and Sub-types

The following object type and object sub types are valid only for XOasis Out.

Translating Boundaries

The OA boundary object defines an enclosed area, including a polygon outline. In OpenAccess there are four types of boundaries: area, PR, snap, and cluster. A cluster boundary has a cluster associated with it and this cluster must be specified while creating cluster boundaries. During XOasis Out, however, the cluster information is lost and upon a subsequent XOasis In, an empty cluster is created for each cluster boundary.

The following table lists the valid boundary object type and sub-types.

Object Type	Object Sub-type
Boundary	oaAreaBoundary
	oaPRBoundary
	oaSnapBoundary
	Cluster

Translating Blockages

The OA blockage object represents those areas in a design where specific object types cannot be located.

Blockages can be:

- **oaAreaBlockage** - represents an area. Most commonly used to prevent standard cells from being placed within this area.
- **oaLayerBlockage** - represents an area on a specific layer. Most commonly used to map layer blockages on different layers to different OASIS layers and datatype pairs.
- **oaAreaHalo** - represents the area around the master's prBoundary. Most commonly used to prevent standard cells from being placed within a specific range of an instance.

The following table lists the valid blockage object type and sub-types.

Object Type	Object Sub-type
AreaBlockage	placement

Design Data Translators Reference

Design Translation Using XOasis Translator

LayerBlockage	routing
	viaRouting
	feedthru
	fill
	pin
	screen
	slot
	wiring
AreaHalo	placement

Important

The AreaBlockage object type can only have placement as the sub-type while the LayerBlockage object can have all other sub-types, except for the placement sub-type.

Translating Markers

The OA marker object indicates design violations and the objects that cause the violations. Following table lists the marker object type and sub-types that are translated and preserved.

Object Type	Object Sub-type
Marker	acknowledgedWarning
	annotation
	criticalError
	error
	fatalError
	info
	signedOffCriticalError
	signedOffError
	warning

Translating Rows

An OA row object represents a potential location for placing standard cells, macros, or devices. For the row object type, the valid object sub-type is `none`.

Note: Custom and mixed rows are not translated and preserved in this release.

Sample XOasis Out Object Map File

```
# ObjectType  ObjectSubType  OasisLayer  OasisPurpose
Boundary      Area          0          1
Boundary      Cluster        0          2
Boundary      PR            0          3
Boundary      Snap          0          4
LayerBlockage viaRouting      1          1
LayerBlockage routing      1          2
AreaBlockage  placement    1          3
LayerBlockage wiring       1          4
LayerBlockage fill         1          5
LayerBlockage metall slot   1          6
LayerBlockage metal2 slot   3          7
LayerBlockage pin          1          7
LayerBlockage feedthru      1          8
LayerBlockage screen       1          9
Marker        annotation    2          1
Marker        info          2          2
Marker        acknowledgedWarning  2          3
Marker        warning       2          4
Marker        signedOffError    2          5
Marker        error          2          6
Marker        signedOffCriticalError  2          7
Marker        criticalError    2          8
Marker        fatalError       2          9
Row           none          3         10
```

Creating OA Objects from Third Party OASIS

If the OASIS file is created by using a third-party tool or by using the `-noObjectProp` option in XOasis Out, then you will have to use object map files to create OA objects from the records in the OASIS file. This functionality works only for the following OA objects: layer-blockage, PR-boundary, snap-boundary, and marker.

The object map file maps the OASIS layer-`datatype` pair to OA object and sub-objects and decides which shape should be translated as an OA object.

XOasis In Object Types and Sub-types

The following object type and object sub types are valid only for XOasis In.

Translating Boundaries

The OA boundary object defines an enclosed area, including a polygon outline. In OpenAccess, there are two types of boundaries, PR boundary and snap boundary.

Following table lists the valid boundary object type and sub-types.

Object Type	Object Sub-type
Boundary	oaPRBoundary
	oaSnapBoundary

Translating Blockages

The OA blockage object represents those areas in a design where specific object types cannot be located.

Blockages can be:

- `oaAreaBlockage` - represents an area. Most commonly used to prevent standard cells from being placed within this area.
- `oaLayerBlockage` - represents an area on a specific layer. Most commonly used to map layer blockages on different layers to different Oasis layers and datatype pairs.

Following table lists the valid blockage object type and sub-types:

Object Type	Object Sub-type
AreaBlockage	placement

Design Data Translators Reference

Design Translation Using XOasis Translator

Object Type	Object Sub-type
LayerBlockage	routing
	viaRouting
	feedthru
	fill
	pin
	screen
	slot
	wiring

Important

The `areaBlockage` object type can only have placement as the sub-type while the `layerBlockage` object can have all other sub-types, except for the placement sub-type.

Translating Markers

The OA marker object indicates design violations and the objects that cause these violations. Following table lists the marker object type and sub-types that are translated and preserved:

Object Type	Object Sub-type
Marker	acknowledgedWarning
	annotation
	criticalError
	error
	fatalError
	info
	signedOffCriticalError
	signedOffError
	warning

Sample XOasis In Object Map File

```
Boundary pr 1 1
Boundary snap 1 2
layerBlockage fill metall 2 1
layerBlockage routing metal2 2 2
layerBlockage slot metall 1 6
layerBlockage slot metal2 3 7
Marker error 3 1
Marker info 3 2
```

Note: For layerBlockage the object map file must contain all five fields, namely, OA object name, OA object sub-type, OA layer name, OASIS layer number, and OASIS datatype. The OA layer name field is mandatory in XOasis In.

OpenAccess PathSegs to XOasis/OASIS Record Mapping

OpenAccess pathSeg objects can be either orthogonal (0, 90, 180, or 270 degrees) or diagonal (45, 135, 225, or 315 degrees). All the orthogonal pathSegs will be written as PATH records in the OASIS file during XOasis Out unless the *Convert Path to Polygon* option is

Design Data Translators Reference

Design Translation Using XOasis Translator

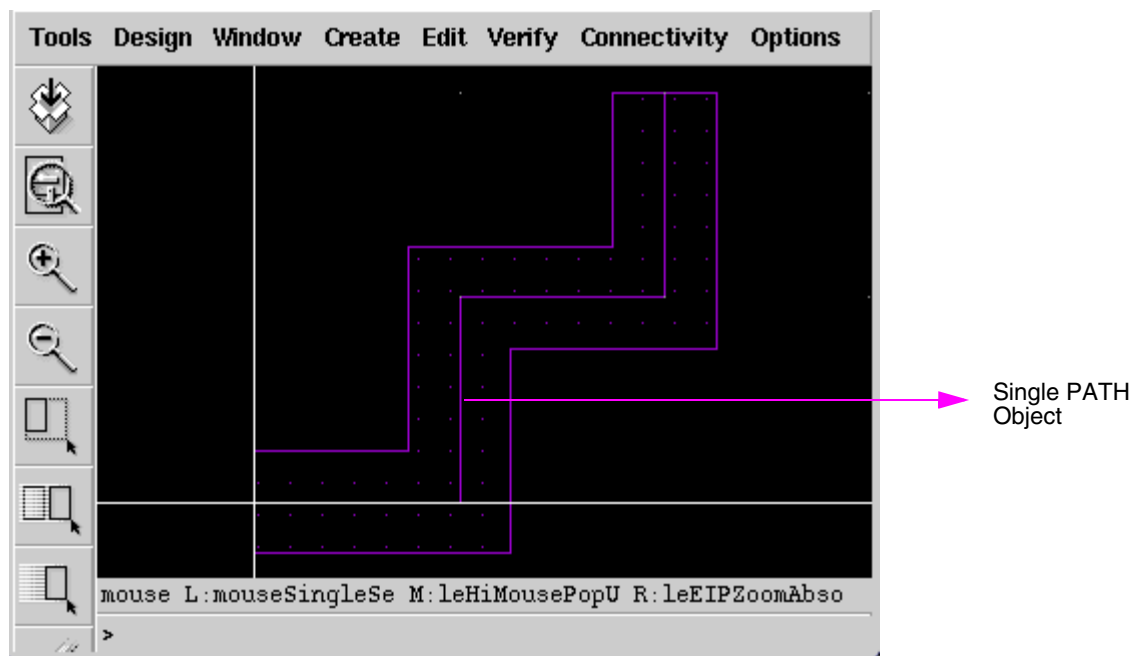
used, or the path extension is *custom* or *chamfer*. However, the diagonal pathSegs will always be written as polygons (BOUNDARY) in the OASIS file.

OA PathSegs	Extensions	Object Type
Orthogonal pathseg	'truncate', 'extend', or 'variable'	paths (PATH)
Orthogonal pathseg	'custom' or 'chamfer'	polygons (BOUNDARY)
Diagonal pathseg	always	polygons (BOUNDARY)

A diagonal pathSeg in the OA design will be written as a polygon (BOUNDARY) record in the OASIS file during XOasis Out. If the OASIS file is imported back in OA using XOasis In then the original diagonal pathSeg will become a polygon. The diagonal pathSeg will not be preserved during XOasis Out/XOasis In; only the geometry will be preserved.

In IC5141, a Path object is created in the CDB database when a route segment is created. During XOasis Out, this multi-point path is written as a single multi-point PATH record in the OASIS file.

Example of a Single multi-point PATH Record



Design Data Translators Reference

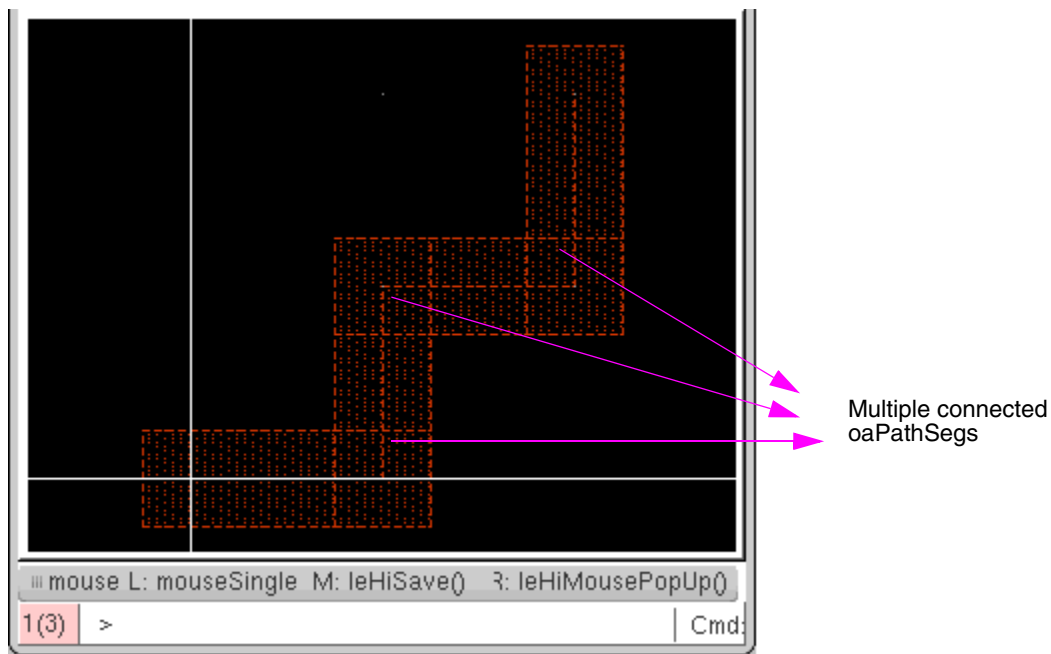
Design Translation Using XOasis Translator

A multi-point path is written as a single multi-point PATH record in the OASIS file as:

```
PATH LAYER=1 DATATYPE=0
PATHTYPE 0 WIDTH 1000 XY=(0,0)
(2000,0) (2000,2000) (4000,2000) (4000,4000) ENDEL
```

In IC61, a collection of Path objects are created in the OA database when a route segment is created. During XOasis Out, these multi-point paths are written as a multiple 2-point PATH records in the OASIS file.

Example of Multiple 2-point PATH Records



A multi-point path is written as a multiple 2-point PATH record in the OASIS file as:

```
PATH LAYER=1 DATATYPE=0
PATHTYPE 2 WIDTH 1000 XY=(4000,2000) (4000,4000) ENDEL
PATH LAYER=1 DATATYPE=0
PATHTYPE 2 WIDTH 1000 XY=(2000,2000) (4000,2000) ENDEL
PATH LAYER=1 DATATYPE=0
PATHTYPE 2 WIDTH 1000 XY=(2000,0) (2000,2000) ENDEL
PATH LAYER=1 DATATYPE=0
```

Design Data Translators Reference

Design Translation Using XOasis Translator

```
PATHTYPE 2 WIDTH 1000 XY=(0,0) (2000,0) ENDEL
```

The `mergePathSegsToPath` option has been introduced to assemble all the connected `oaPathSegs` into a single **PATH** record during XOasis Out. This enables you to reduce the size of the OASIS. In addition, it will provide backward compatibility with IC5141.

Assumptions

- Only the `oaPathSegs` that have the same width would be considered for merging.
- A `pathSeg` will not be considered for merging if:
 - It has the extensions, such as spike or gap, on the edge.
 - The `beginStyle` or `endStyle` of an `oaPathSeg` is *custom* or *chamfer*
- End type of **PATH** would become variable if:
 - `BeginStyle` of the first `oaPathSeg` and `endStyle` of the last `pathSeg` are different. Only styles, such as truncate, extends, and variable are considered.
- When property mapping is used, if there is a conflict between the properties of the `oaPathSegs` objects, the first property value will be considered for writing in the OASIS and the second property value will be dropped with a warning.

This option must be used only when required because it will have some performance overheads.

Design Data Translators Reference

Design Translation Using XOasis Translator

Sample Log File

```
*****
Product    : Virtuoso(R) XOasis In
Program    : @(#)$CDS: oasisin version 6.1.5 06/16/2010 12:55 (lnx05) $
           : sub-version 6.1.5
Started at: 18-Jun-2010 17:37:05
User Name  : tkgeorge
Host Name  : lnx05
Directory  : /tmp/tkgeorge/testcases/test/xoasis/demo
CADENCE Design Systems, Inc.
*****
Info: Cellview Rev Num:99, Tech Rev Num:59
```

Summary of Objects Translated :

Instances:	0
Arrays:	0
Polygons:	5
Paths:	2
PathSegs:	0
Rectangles:	2
Text:	4
Dot:	0
ZeroAreaPolygon:	0
Ellipse:	0
Line:	0
Cells:	3

Elapsed Time = 0.1s

Elapsed Time: 0.5s User Time: 0.5s CPU Time: 0.1s Peak VM: 279KB
INFO (XOasis-234): Translation completed. '0' error(s) and '0' warning(s) found.

Design Data Translators Reference

Design Translation Using XOasis Translator

Sample Summary File

```
*****
Product   : Virtuoso(R) XOasis In
Program   : @(#) $CDS: oasisin version 6.1.5 06/16/2010 12:55 (lnx05) $
           : sub-version 6.1.5
Started at: 18-Jun-2010 17:37:05
User Name : tkgeorge
Host Name : lnx05
Directory : /tmp/tkgeorge/testcases/test/xoasis/demo
CADENCE Design Systems, Inc.
*****
```

```
Info: Cellview Rev Num:99, Tech Rev Num:59
      Individual Cell Statistics
```

Library/Cell/View	Instance	Array	Polygon	Rect	Path	Text	Node
oasis_in/ellipse/layout	0	0	1	0	0	0	0
oasis_in/oasis_cell/layout	0	0	2	1	1	2	0
oasis_in/oasis_cell1/layout	0	0	2	1	1	2	0

Statistics of Layers

Cadence	Cadence	Oasis	Oasis								
Layer	Purpose	Layer	Datatype	Polygon	Rect	Path	PathSeg	Zero	Text	Node	Ellipse
								Area			
								Polygon			
y3	drawing	210	0	3	0	0	0	0	0	0	0
y0	drawing	207	0	0	2	0	0	0	2	0	0
y1	drawing	208	0	2	0	0	0	0	0	0	0
y2	drawing	209	0	0	0	0	0	0	0	0	0
y4	drawing	211	0	0	0	2	0	0	0	0	0
y5	drawing	212	0	0	0	0	0	0	2	0	0

```
Summary of Objects Translated:
  Instances:0
  Arrays: 0
  Polygons:5
  Paths: 2
  PathSegs:0
```

Design Data Translators Reference

Design Translation Using XOasis Translator

```
Rectangles:2
Text:      4
Dot:       0
Line:      0
ZeroAreaPolygon:0
Ellipse:0
Cells:     3
Elapsed Time = 0.1s
Design Libraries:
DEFINE oasis_in /tmp/tkgeorge/testcases/test/xoasis/demo
```

Design Data Translators Reference

Design Translation Using XOasis Translator

Sample Template File

XOasis Out Template File

```
*****
# Copyright (C) 2013 - 2018 *
# Cadence Design Systems Inc. All Rights Reserved. *
*****
#
#*****
#
# filename : oasisOut.template
#
#*****

#hierDepth                "32767"                #- (0-32767) - Hierarchy Depth Limit
#GDSIICompatible          #- Generate the oasis file with GDSII specific limits
#arrayInstToScalar        #- Convert Arrays to Scalar Instances
#autoGenOASISLayerName    #- Automatically Generate OASISLayerName for layers and objects
mappings
#backupOasisLogFiles      #- Backup OASIS and Log Files
case                       "preserve"              #- preserve | upper | lower - Case Sensitivity
cellListFile              ""                       #- The Cell List File Name
cellMap                   ""                       #- Cell Map File Name
cellNamePrefix            ""                       #- Prefix to be added to each STRUCT
cellNameSuffix            ""                       #- Suffix to be added to each STRUCT
#checkPolygon             #- Report Bad Polygons
#circleToPolygon          #- Convert Circles/Ellipses to Polygons
#colorAnalysisGroup       ""                       #- Name of the color analysis group
#compress                 #- Enable CBLOCK compression
#compressLevel            "1"                     #- Set the compression level for CBLOCK compression
convertDot                "polygon"               #- polygon | circle | ignore - Convert Dots to
convertPcellPin           ""                       #- geometry | text | geometryAndText | ignore - Convert Pcell Pin to
convertPin                "geometry"              #- geometry | text | geometryAndText | ignore - Convert Pin to
#dbuPerUU                 "1000"                 #- DB Units Per User Units
#diagonalPathToPolygon    #- Convert Paths with non-orthogonal segments to polygons
#doNotPreservePcellPins   #- Do Not Preserve the PCell Pins
#donutNumSides            "64"                   #- Number of sides for the BOUNDARY of donut
#ellipseNumSides          "64"                   #- Number of sides for the BOUNDAR of circle/ellipse
#enableColoring           #- Enable Coloring Support
#flattenPcells            #- Flatten the PCell Instances
#flattenPcellsExclusionFile ""                   #- A file specifying list of Pcells to exclude from flattening
#flattenViaShapesToViaPurp #- Flatten Vias' Shapes to Via Purpose
#flattenVias              #- Flatten the Vias
#ignoreCurrentTimestamp   "false"                #- Ignores current timestamp
#ignoreLines              #- Ignore Lines During Translation
#ignoreMissingCells       #- Ignore Missing cellViews during Translation and continue Translation
#ignoreMissingVias        #- Ignore Missing standard/custom vias during Translation and
continue Translation
#ignoreObjectMapFromTech  #- Ignore object map file from technology
#ignorePcellEvalFail      #- Ignore the PCell Evaluation Failure
#ignoreRoutes             #- ignore Routes During translation
ignoreTopCellPrefixSuffix "false"                #- Ignores cellName Prefix and Suffix for topCell
#ignoreZeroWidthPathSeg   #- Ignore zero width PathSeg during translation
infoToWarn                ""                     #- Treat the Specified Info Messgaes Ids as the Warnings
labelDepth                "1"                    #- Hierarchical depth to add labels to
labelMap                  ""                     #- Label Map File Name
layerMap                  ""                     #- Layer Map File Name
library                   "oasisOutLib"          #- Library Name
```


Design Data Translators Reference

Design Translation Using XOasis Translator

logFile	"oasisOut.log"	#- Log File Name
#mergePathSegsToPath		#- Merge the Connected Pathsegs to Path
#noConvertHalfWidthPath		#- Do not Convert The Half Width Path to Boundary
noInfo	""	#- Do not Print the Specified Info Messages Ids
#noObjectProp		#- Do not Write Special OA Object Properties
#noOutputTextDisplays		#- Ignore Text Display Objects
#noOutputUnplacedInst		#- Ignore the Unplaced Instances
noWarn	""	#- Do not Print the Specified Warning Messages Ids
oasisFile	"test.oasis"	#- Oasis File Name
objectMap	""	#- Object Map File Name
outputDir	""	#- The Output Directory for Oasis File
#pathToPolygon		#- Convert Paths to Polygons
#preserveTextAtt		#- Preserve Text Attributes as Properties
#preservePinAtt		#- Preserve Pin Connectivity Information
propMap	""	#- Property Map File Name
refLibList	""	#- Reference Library List File Name
#replaceBusBitChar		#- Replace "<>" With "[]"
runDir	""	#- Run Directory
#snapToGrid		#- Snap XY to Grid Resolution
subMasterSeparator	"_CDNS_"	#- Separator to be used for sub-master naming
summaryFile	""	#- Summary File Name
techLib	""	#- Technology Library Name
textCase	"preserve"	#- preserve upper lower - Label Case Sensitivity
topCell	"top"	#- Top Cell(s) to Translate
translateUnmappedLPP	"false"	#- Allow undefined LPP to be translated by incrementally mapping
LPP using layerMap file, SKILL hook poLayerMap, and automatic layer mapping		
userSkillFile	""	#- Users' Skill File Name
verbose	"false"	#- Generate detailed information in the log and summary files
viaMap	""	#- Via Map File Name
view	"layout"	#- View Name(s) to Translate
warnToErr	""	#- Treat the Specified Warning Messgaes Ids as the Errors
#wildCardInCellMap		#- Wild Card in Cell Map

Note: The maximum length of a single line in the template file for the topcell and other XOasis Out options is 8192.

XOasis In Template File

```

*****
# Copyright (C) 2013 - 2018                               *
# Cadence Design Systems Inc. All Rights Reserved.         *
*****
#
# *****
#
# filename : oasisIn.template
#
# *****

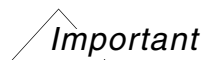
```

arrayInstToScalar	"false"	#- Convert AREF to Scalar Instances
attachTechFileOfLib	""	#- Attach Techfile of Library
case	"preserve"	#- preserve upper lower - Case Sensitivity
cellMap	""	#- Cell Map File Name
checkPolygon	"false"	#- Report Bad Polygons
compress	"false"	#- Allow libraries to be compressed
#compressLevel	"1"	#- Defines the compression level to use (default: 1)
convertPathToPathSeg	"false"	#- Convert Two Point Path(s) to PathSeg(s)
detectOverflow	"false"	#- Enable overflow cellview creation on reaching maximum limit
of shapes in a cellview		
dbuPerUU	""	#- DB Units Per User Units

Design Data Translators Reference

Design Translation Using XOasis Translator

#enableColoring	"false"	#- Enable Coloring Support
enableIncrementalImport	"false"	#- Enable incremental import of multiple OASIS Files
enableLocking	"false"	#- Enable File Locking
enableNegativeViaExt	"false"	#- Enable Negative Extension for Via Detection
excludeMapToVia	""	#- Via exclusion Data file
hierDepth	"32767"	#- (0-32767) - Hierarchy Depth Limit
ignoreObjectProp	"false"	#- Ignore Object Properties
ignoreZeroWidthPath	"false"	#- Ignore Zero Width Path(s)
infoToWarn	""	#- Treat the Specified Info Messages Ids as the Warnings
keepOasisCells	"false"	#- Keep Oasis File Cells
layerMap	""	#- Layer Map File Name
library	"oasisInLib"	#- Library Name
loadTechFile	""	#- ASCII Technology File Name
logFile	"oasisIn.log"	#- Log File Name
maxCellsInTargetLib	"30000"	#- Maximum Cells in Target Lib
mergeUndefPurposToDrawing	"false"	#- Merges Undefined Purposes to "drawing" Purpose
noDetectVias	"false"	#- Do Not Detect possible viaDefs from Oasis File and create vias
noInfo	""	#- Do not Print the Specified Info Messages Ids
noWarn	""	#- Do not Print the Specified Warning Messages Ids
oasisFile	"test.oasis"	#- Oasis File Name
objectMap	""	#- Object Mapping File Name
#overflowCellNameSuffix	"_xoasOverflow"	#- Cell name suffix of overflow cellviews
#overflowViewName	"layout"	#- View name of overflow cellviews
#overwriteRootCells	"false"	#- Overwrite the top cells in case write mode is rename
preservePinAtt	"false"	#- Preserve Pin Connectivity Information
propMap	""	#- Property Map File Name
propSeparator	","	#- Property Separator Character
refLibList	""	#- Reference Library List File
replaceBusBitChar	"false"	#- Replace "[]" With "<>"
runDir	(".")	#- Run Directory
scale	"1.00000"	#- Design Scaling Factor
showCompletionMsgBox	"true"	#- Show Completion Message Box
skipUndefinedLPP	"false"	#- Skip Creating Undefined Layer-Purpose Pair in Technology
snapToGrid	"false"	#- Snap XY to Grid Resolution
summaryFile	""	#- Summary File Name
techRefs	""	#- Ordered List of Technology Libraries
textCase	"preserve"	#- preserve upper lower - Text Case Sensitivity
textFont	"stick"	#- euroStyle gothic math roman script stick fixed
swedish milSpec		
textHeight	"1.00000"	#- Text Height (+ve value only)
textOrient	"R0"	#- R0 R90 R180 R270 MY , MYR90 MX MXR90 - Default
Text Orientation		
topCell	""	#- Top Cell(s) to Translate
translateUnmappedLPP	"false"	#- Allow undefined LPP to be translated by incrementally mapping
LPP using layerMap file, SKILL hook piLayerMap, and		automatic layer mapping
useExistingViaDefsForDetection	"false"	#- Enable the usage of standard viaDefs from technology database
for via detection		
userSkillFile	""	#- Users' Skill File Name
verbose	"false"	#- Generate detailed information in the log and summary files
viaMap	""	#- Via Map File Name
view	"layout"	#- View Name(s) to Translate
virtualMemory	"false"	#- Import to Virtual Memory
warnToErr	""	#- Treat the Specified Warning Messages Ids as the Errors
wildCardInCellMap	"false"	#- Wild Card in Cell Map
writeMode	"overwrite"	#- Write Mode for Existing Cells



During XOasis In and XOasis Out both the oasisFile and library options are mandatory. However, null strings (" ") can be allowed while loading the template file.

Library Cell(s) Limit

Since the OpenAccess library structure complies to 5.x architecture, the total number of cells in a library can go only upto the maximum number of sub-directories that are supported by the file system. This limits XOasis to support only those OASIS files that have less 32,767 than cells, which are to be translated in a library.

To overcome this limit, XOasis has been enhanced to keep a count of the number of cells in the target library. Whenever the number of cells in the target library gets to 20,000 and, if XOasis needs to translate more cells, a new target library *TargetLib_#* (where # can be 1,2, or 3), is created.

If a new target library already exists, XOasis creates another library instead of overwriting the existing one. For example, if a target library *TargetLib_1* already exists, XOasis creates *TargetLib_2*. The technology library for the new target library is attached to the technology library of the original target library.

Performance Guidelines for XOasis

XOasis In and XOasis Out provide many options and features to make its usage model extremely flexible in accordance with increasing customer requirements.

The following section provides information on the speed impact of some of the XOasis In and XOasis Out options. It also provides some guidelines about extracting maximum performance by avoiding those options that reduce the speed of translation in specific customer designs.

Guidelines for XOasis In

- In order to optimize the time consumed in searching for reference libraries during XOasis In, use the methods described below:

- Reduce the `cds.lib` file size

The `cds.lib` file should consist of a minimal list of target and reference libraries for XOasis In to perform faster. The minimum number of libraries listed in `cds.lib` should be the target and reference libraries. The cells of these libraries should be instantiated in the OASIS file. If there are a large number of libraries mentioned in the `cds.lib` file, XOasis In spends extra time in searching for different cellviews in those libraries and thus takes more time. Another way to resolve this issue without modifying the `cds.lib` file is by using the *Reference Library List* option.

- Use the *Reference Library List* option

Suppose you have a number of reference libraries in `cds.lib` and a design or a sub design/module is using a small fraction of those. In another case, the libraries defined in `cds.lib` may not get ordered in an optimal fashion to minimize the search for a referenced cell during XOasis In. In such cases, use the *Reference Library List* option.

The syntax of the *Reference Library List* option in the template file is as follows:

```
refLibList "lib1 lib2 lib3 lib4"
```

where, *lib1*, *lib2*, *lib3*, and *lib4* are the names of reference libraries separated by blank spaces.

Example:

Consider a situation where you are using instances from five different libraries: A, B, C, D, and E. You know that the usage density (the maximum probability of finding the referenced cell from a library) is from C and E. In such a case, in order to speed up the XOasis In process, use the `refLibList` option, as shown below:

```
refLibList "C E A B D"
```

Design Data Translators Reference

Design Translation Using XOasis Translator

If you provide a proper reference library name for a structure (cellview) in the cell name map file, XOasis does not search for that cellview in the list of libraries.

Therefore, a cell name map file can also provide speed advantage when you use the *Reference Library List* option.

- If the value of the *Convert Arrays to Scalar Instances* option is chosen as ON (a non-default value), it will explode each AREF record in an OASIS file. This will be done by translating an AREF record to the corresponding individual instances in the OpenAccess database. In some specific cases, this can reduce the translation speed and increase the output database size.
- Ensure that there is enough virtual memory in your system.
- Use the *Enable file locking* option to enable file locking during XOasis In. By default, file locking is disabled. Enabling increases the run time of XOasis In especially if the designs contain a large number of cellviews in their hierarchy or library.

Guidelines for XOasis Out

- For specific purposes, you can use a limited set of layers to be translated. This can be done by configuring the layer mapping table accordingly. Using the limited set of layers, you can speed up translation because the amount of data to be translated is reduced.
- Avoid using the *Output Paths as Polygons* option. This option converts each OpenAccess path to a polygon (boundary record in OASIS format) during XOasis Out. If there are a large number of paths in a design, this option can reduce the speed of translation due to the additional overhead of converting each path to a polygon internally.
- When the *Convert Arrays to Scalar Instances* option is enabled, all arrays explode in OpenAccess. This is because each element of the array is translated to a corresponding SREF record in the OASIS file. This significantly slows down the translation and increases the size of the output OASIS file.

For information about XOasis SKILL functions, see the *Design Data Translator's SKILL Reference*.

Chip Finishing Flow

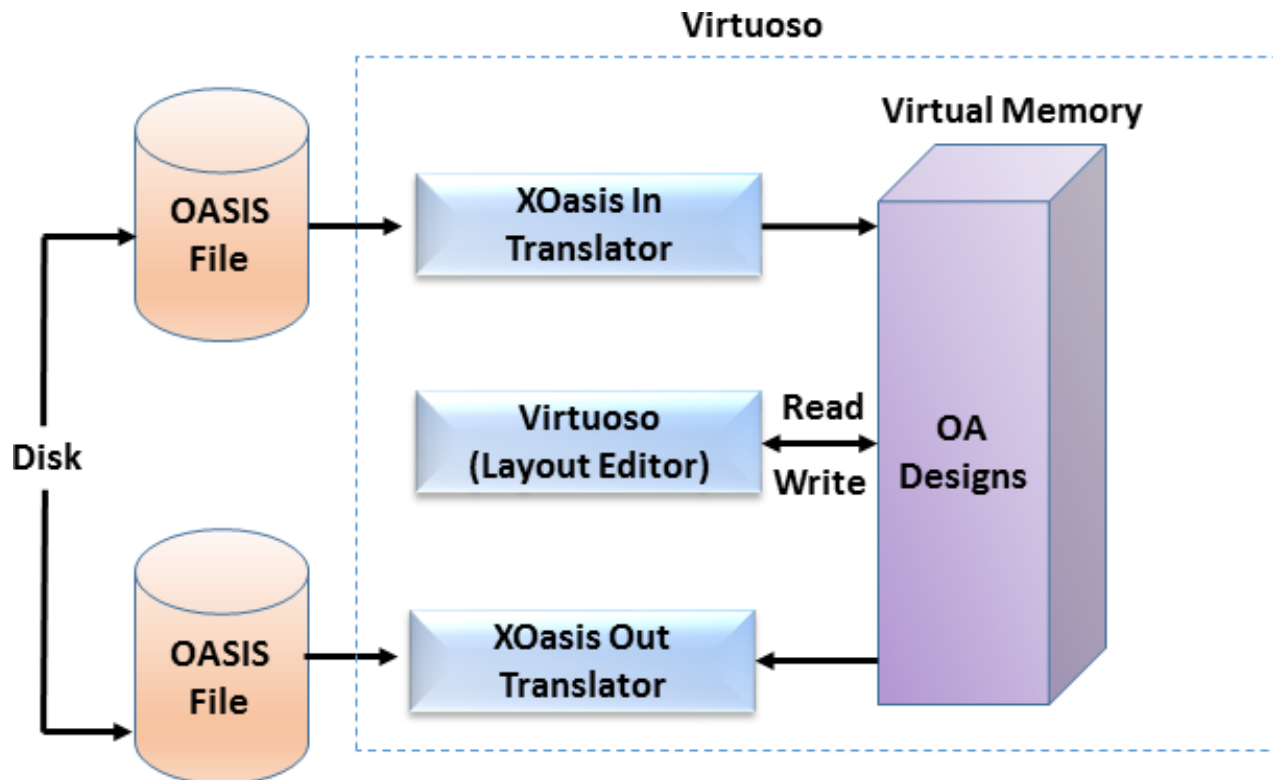
To improve the speed at which OA designs are accessed, the following features have been introduced:

- Virtuoso XOasis In translator reads a OASIS file from the disk and creates an OA design in the virtual memory.
- You can use the Virtuoso layout editor to view or edit an OA design stored in the virtual memory.

Note: You can also use DFII SKILL functions to access the OA design stored in the virtual memory.

- You can use the XOasis Out translator to export an OA design from virtual memory to an OASIS file on the disk.

The figure below illustrates how XOasis In and XOasis Out work.



Note: You do not need a special license to use XOasis In with virtual memory. XOasis In uses the same license that is used by the Virtuoso layout editor.

Design Data Translators Reference

Design Translation Using XOasis Translator

To use XOasis In with virtual memory, you need to select the Import OASIS to Virtual Memory check box in the XOasis In form. Similarly, if you want to use XOasis Out with virtual memory, you need to select the Export OASIS from Virtual Memory check box.

Important

Import OASIS to Virtual Memory requires space for storing temporary data, though the OA designs are not saved on disk.

In Linux, the temporary data is stored in `/dev/shm`. If the available space in `/dev/shm` is less than 5 MB, temporary data is stored in `/tmp`. Moreover, if the available space in `/tmp` is less than 5 MB, the temporary data is stored in `.` / (the run directory).

In Solaris and IBM, the temporary data is stored in `/tmp`. If the available space in `/tmp` is less than 5 MB, temporary data is stored in `.` / (the run directory).

The temporary data is deleted when the Virtuoso session is exited.

By default, information (INFO) and warning (WARNING) messages are not displayed during XOasis In to virtual memory.

Netlist Import Using Spice In

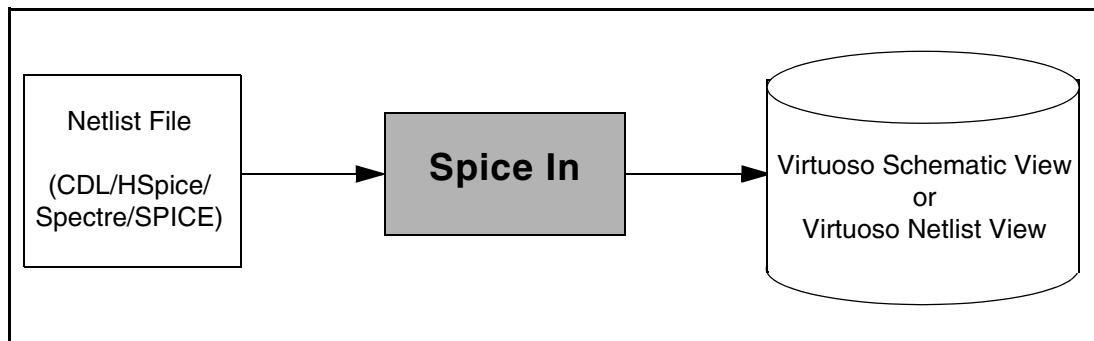
This chapter covers the following:

- [Introduction to Spice In](#) on page 450
- [Using Spice In](#) on page 458
 - [Running Spice In from GUI](#) on page 459
 - [Running Spice In from Command Line](#) on page 488
- [Inherited Connections Supported](#) on page 490
- [Creating a Spice In Parameter File](#) on page 497
- [Creating a Device-Map File](#) on page 502
- [Handling Parameters](#) on page 508
- [Handling Terminal Order for Sub-Circuits](#) on page 514

Introduction to Spice In

Spice In imports CDL, HSpice, Spectre, and SPICE netlists into the OpenAccess database by generating either Virtuoso Schematic or Virtuoso Netlist views. It also creates Virtuoso Symbol views wherever required.

The figure below shows the types of netlists that can be imported using Spice In and the types of output views generated.



Note: The term Spice In this chapter is used to refer to the SPICE2 and SPICE3 dialects of the SPICE language. The Simulation Program with Integrated Circuit Emphasis (SPICE) language was developed at the University of California, Berkeley, and is used by many EDA vendors to simulate all types of circuits.

The Spice In examples are present at the following location: <install_dir>/tools/dfll/samples/*Spice In*. For more information on examples of Spice In, refer the readme.txt file present at the above mentioned location.

Features of Spice In

General Features

- Generates a Virtuoso Schematic or a Virtuoso Netlist view.
- Imports sub-circuit level parameters of a sub-circuit as Component Description Format (CDF) Component Parameters at the cell level. For more information about CDF, see the *[Component Description Format User Guide](#)*.

For more information about how Spice In imports sub-circuit level parameters, see *[Handling Parameters](#)* on page 508.

Design Data Translators Reference

Netlist Import Using Spice In

- Uses terminal order (`termOrder`) from the CDF Simulation Information of the specified simulator for creating connections between the nets of an instance and the terminals of the master cell of the instance.

For more information about how and where Spice In creates terminal order, see [Handling Terminal Order for Sub-Circuits](#) on page 514.

- Saves the terminals of a sub-circuit as `termOrder` in the CDF Simulation Information of the specified simulator.
- Maps devices from model libraries to a user PDK.
- Enhances placed and routed schematics by using the Analog Schematic Generation feature. For more information on ASG, see [Analog Schematic Generation Tab](#).

Spice/Spectre

- Imports global parameters as design variables at the top-cell level. The design variables can be used in pre-processing of the design before simulating it using Virtuoso® Analog Design Environment (ADE). For more information about ADE, see [Virtuoso ADE L User Guide](#) or [Virtuoso ADE XL User Guide](#).

For more information about how Spice In imports the global parameters, see [Handling Parameters](#) on page 508.

Flows Using Spice In

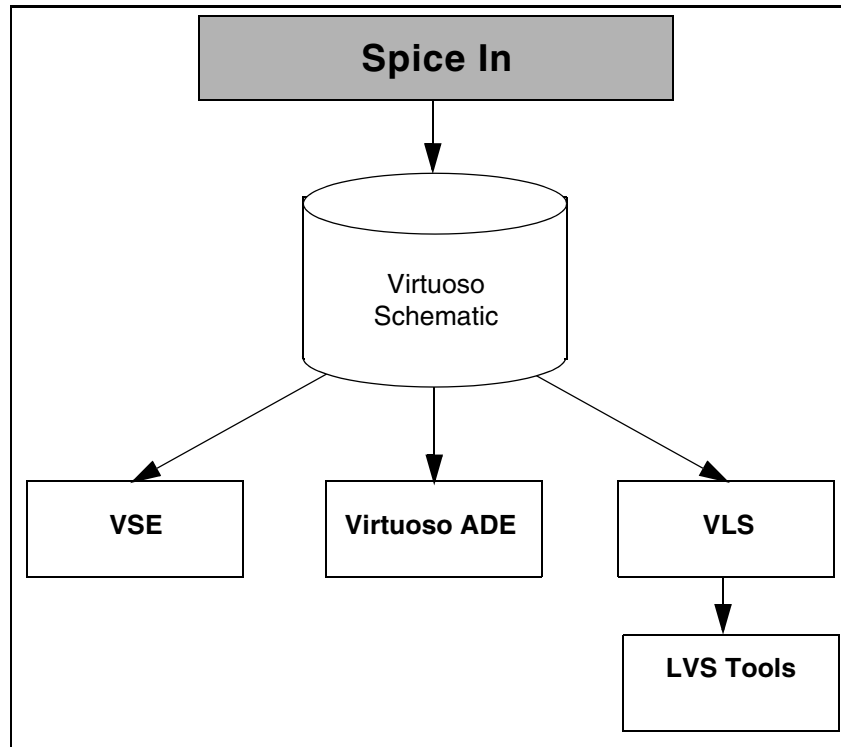
Spice In aids you in a number of design and simulation flows, such as:

- Designing and simulating analog circuits using Virtuoso® Schematic Editor (VSE) and Virtuoso ADE
- Creating Virtuoso Layout views using Virtuoso® Layout Suite (VLS), and subsequently running layout-versus-schematic (LVS) using LVS tools.

Design Data Translators Reference

Netlist Import Using Spice In

The following figure shows the flows that use Spice In.



Licensing in Spice In

Spice In searches for the following licenses in the specified order and checks out one of them:

- 1 license of Virtuoso® Schematic Editor L

For more information about the Virtuoso Schematic Editor L, see the [*Virtuoso Schematic Editor User Guide*](#)

- 1 license of Virtuoso® Schematic Editor XL

For more information about the Virtuoso Schematic Editor XL, see the [*Virtuoso Schematic Editor User Guide*](#)

- 1 license of Virtuoso® Layout Suite L

For more information about the Virtuoso Layout Suite L, see the [*Virtuoso Layout Suite L User Guide*](#)

- 1 license of Virtuoso® Layout Suite XL

Design Data Translators Reference

Netlist Import Using Spice In

For more information about the Virtuoso Layout Suite XL, see the [*Virtuoso Layout Suite XL User Guide*](#).

- 4 tokens of Virtuoso® Layout Suite GXL

For more information about the Virtuoso Layout Suite GXL, see the [*Virtuoso Layout Suite GXL Reference*](#).

Note: If the ASG option is enabled, then licenses of Virtuoso® Schematic Editor L and Virtuoso® Layout Suite L are not referred.

Important

By default, the ASG option is disabled in the GUI mode. You can enable the ASG option by using the `.cdsenv` setting:

```
* envSetVal("spiceIn" "asgMode" 'boolean t)
```

However, in the Spice In command line the ASG option is enabled by default.

The `-noasg` argument can be used to run Spice In without ASG.

Spice In searches for the following licenses in the specified order and checks out one of them:

- 1 license of Virtuoso® Schematic Editor L

For more information about the Virtuoso Schematic Editor L, see the [*Virtuoso Schematic Editor User Guide*](#)

- 1 license of Virtuoso® Schematic Editor XL

For more information about the Virtuoso Schematic Editor XL, see the [*Virtuoso Schematic Editor User Guide*](#)

- 1 license of Virtuoso® Layout Suite XL

For more information about the Virtuoso Layout Suite XL, see the [*Virtuoso Layout Suite XL User Guide*](#).

- 4 tokens of Virtuoso® Layout Suite GXL

For more information about the Virtuoso Layout Suite GXL, see the [*Virtuoso Layout Suite GXL Reference*](#).

- 1 license of Virtuoso® Layout Suite EXL

For more information about the Virtuoso Layout Suite EXL, see the [*Virtuoso Layout Suite EXL User Guide*](#).

Design Data Translators Reference

Netlist Import Using Spice In

- 1 license of Virtuoso® Layout Suite EAD

For more information about the Virtuoso Layout Suite EAD, see the *Virtuoso Electrically Aware Design Flow Guide*.

Affects of Control Commands during Import

There are various control commands, such as `*.scale`, which have an affect during import of cdl netlist.

*.scale Command

- If the `*.scale` command is present in netlist with or without the `micron` argument, the values are expanded during parsing of netlist.

Even if the `*.scale` command is not present in the netlist during parsing will have the same effect.

All these conditions are illustrated below:

- The `*.scale` command is present without an argument

```
*.BIPOLAR
*.CAPVAL
*.RESVAL
*.DIOAREA
*.DIOPERI

*.scale

.subckt topCell a b c d

C1 a b 20n M=2 $[CX]
D1 c d DI AREA=10n PJ=5p M=9
Q1 c a d nz M=11 $EA=17f $L=9n $w=8p
R1 d a 10M M=4 $.MODEL=rx $w=19p $L=11n
M1 d c a b px W=15n L=17f M=2
J1 a b c nx W=5p L=10f
.END
```

- The `*.scale` command is present with the `micron` argument

```
*.BIPOLAR
*.CAPVAL
*.RESVAL
*.DIOAREA
*.DIOPERI

*.scale micron
```

Design Data Translators Reference

Netlist Import Using Spice In

```
.subckt topCell a b c d

C1 a b 20n M=2 $[CX]
D1 c d DI AREA=10n PJ=5p M=9
Q1 c a d nz M=11 $EA=17f $L=9n $w=8p
R1 d a 10M M=4 $.MODEL=rx $w=19p $L=11n
M1 d c a b px W=15n L=17f M=2
J1 a b c nx W=5p L=10f
.END
```

- ❑ The *.scale command is not present

```
*.BIPOLAR
*.CAPVAL
*.RESVAL
*.DIOAREA
*.DIOPERI
```

```
.subckt topCell a b c d

C1 a b 20n M=2 $[CX]
D1 c d DI AREA=10n PJ=5p M=9
Q1 c a d nz M=11 $EA=17f $L=9n $w=8p
R1 d a 10M M=4 $.MODEL=rx $w=19p $L=11n
M1 d c a b px W=15n L=17f M=2
J1 a b c nx W=5p L=10f
.END
```

Based on the above conditions the parameter values are expanded as:

```
C1 a b 20n M=2 $[CX] * 2e-08 , 2
D1 c d DI AREA=10n PJ=5p M=9 * 1e-08, 5e-12, 9
Q1 c a d nz M=11 $EA=17f $L=9n $w=8p * 11, 1.7e-14, 9e-09, 8e-12
R1 d a 10M M=4 $.MODEL=rx $w=19p $L=11n * 1e+07, 4, 1.9e-11, 1.1e-08
M1 d c a b px W=15n L=17f M=2 * 1.5e-08, 1.7e-14, 2
J1 a b c nx W=5p L=10f * 5e-12, 1e-14
.END
```

- If the *.scale meter is present in netlist then only *Length* and *Width* parameters are multiplied by 1e+06. There is no impact on other parameters. For example:

```
*.BIPOLAR
```


Design Data Translators Reference

Netlist Import Using Spice In

```

*.CAPVAL
*.RESVAL
*.DIOAREA
*.DIOPERI

*.scale meter

.subckt topCell a b c d

C1 a b 20n M=2 ${CX}          * 2e-08 , 2    ;; No affect of meter
                                as L, W are affected only

D1 c d DI AREA=10n PJ=5p M=9   * 1e-08, 5e-12, 9    ;; No affect of
                                meter as L, W are affected only

Q1 c a d nz M=11 $EA=17f $L=9n 11, 1.7e-14, .009, 8e-06 ;
$w=8p

R1 d a 10M M=4 $.MODEL=rx $w=19p * 1e+07, 4, 1.9e-05, .011
$L=11n

M1 d c a b px W=15n L=17f M=2   * 0.015 , 1.7e-08, 2

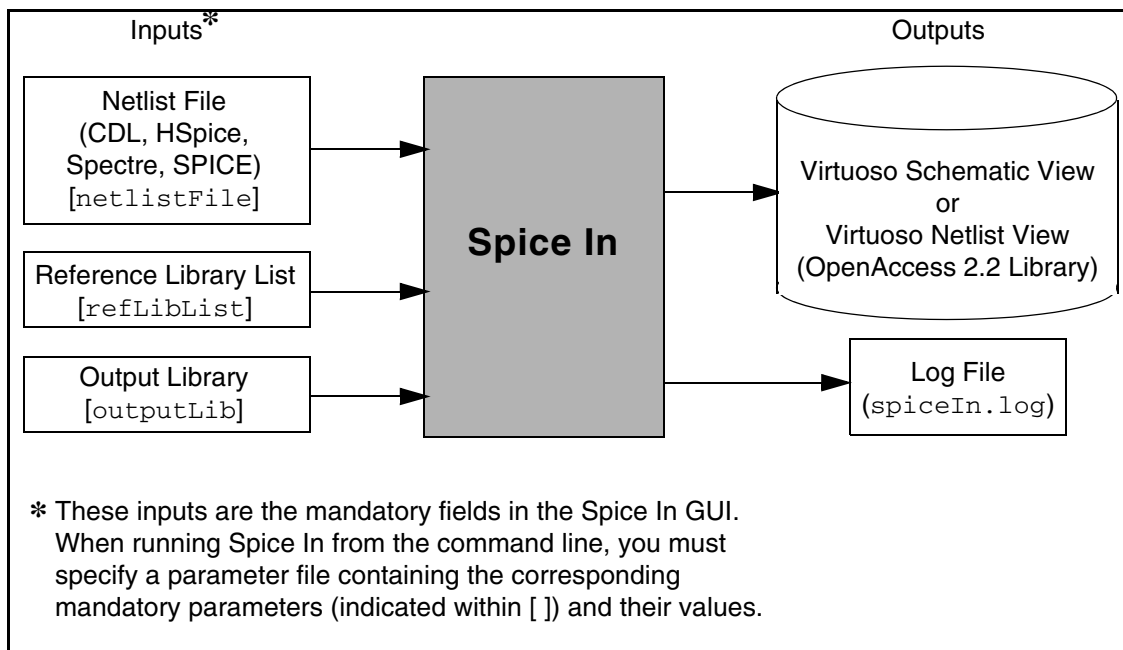
J1 a b c nx W=5p L=10f         * 5e-06, 1e-08

.END

```

Using Spice In

Spice In can be run using either the GUI or the command line interface. The following figure depicts the inputs and outputs of Spice In.



Running Spice In from GUI

Use the Spice In GUI to specify the inputs and the desired outputs for the design import. The following sections explain how to launch and use the GUI.

- [Launching the Spice In GUI](#) on page 459
- [Working with the Spice In GUI](#) on page 461
- [Customizing the Spice In Environment](#) on page 484
- [Running Spice In](#) on page 486

Launching the Spice In GUI

Spice In can be launched from Virtuoso® Design Environment workbenches: `virtuoso` and `layout`.

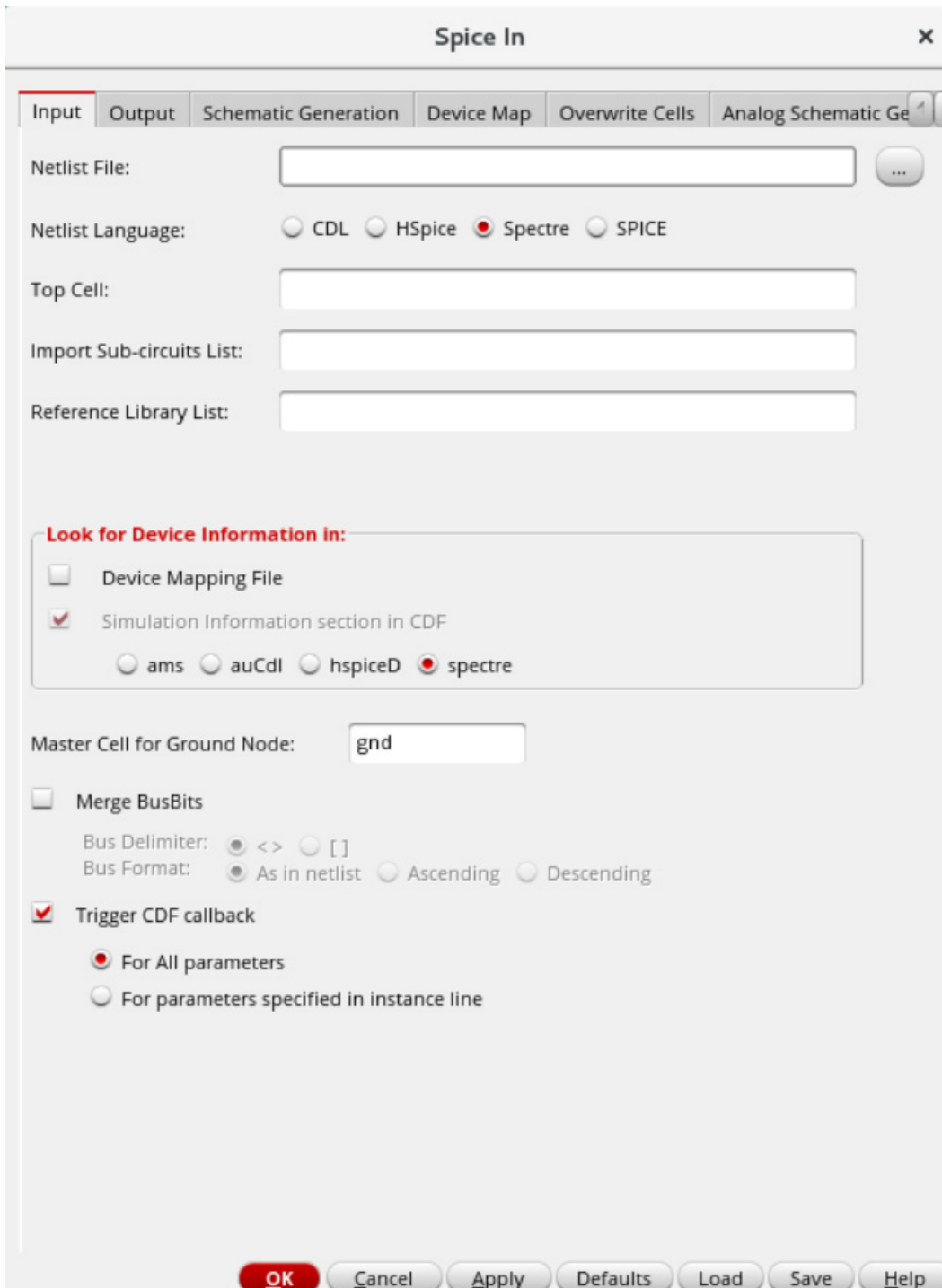
To launch the Spice In GUI, perform the following steps:

1. Open the CIW of the aforementioned Virtuoso workbenches.

Design Data Translators Reference

Netlist Import Using Spice In

2. Select *File* — *Import* — *Spice*. The Spice In form is displayed.



The image shows the 'Spice In' dialog box with the 'Input' tab selected. The dialog has a title bar with a close button (X). Below the title bar are tabs: 'Input' (selected), 'Output', 'Schematic Generation', 'Device Map', 'Overwrite Cells', and 'Analog Schematic Ge'. The 'Input' tab contains the following fields and options:

- Netlist File:** A text input field with a browse button (three dots).
- Netlist Language:** Radio buttons for CDL, HSpice, Spectre (selected), and SPICE.
- Top Cell:** A text input field.
- Import Sub-circuits List:** A text input field.
- Reference Library List:** A text input field.
- Look for Device Information in:** A section with a checkbox for 'Device Mapping File' (unchecked) and a checked checkbox for 'Simulation Information section in CDF'. Below this are radio buttons for ams, auCdI, hspiceD, and spectre (selected).
- Master Cell for Ground Node:** A text input field containing 'gnd'.
- Merge BusBits:** A checkbox (unchecked). Below it are radio buttons for 'Bus Delimiter' (<> selected, [] unselected) and 'Bus Format' (As in netlist selected, Ascending unselected, Descending unselected).
- Trigger CDF callback:** A checked checkbox. Below it are radio buttons for 'For All parameters' (selected) and 'For parameters specified in instance line' (unselected).

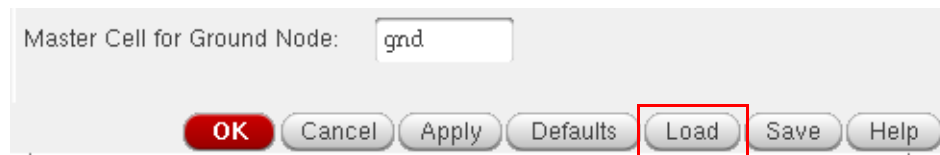
At the bottom of the dialog are buttons: OK (highlighted in red), Cancel, Apply, Defaults, Load, Save, and Help.

Working with the Spice In GUI

You can use the Spice In GUI to load an existing parameter file as well as modify a parameter file. A parameter file, written in SKILL[®] language, is a list of parameters specified as name-value pairs. The parameter file is used to save values for the various fields in the form, thereby allowing the values used in one session to be reused in another session. For more information about the parameter file, see [Creating a Spice In Parameter File](#) and [Mapping Parameter File Options with the Spice In GUI Options](#).

Loading a Parameter File

You can load an existing parameter file, by using the *Load* button at the bottom of the *Spice In* form as shown in the figure.



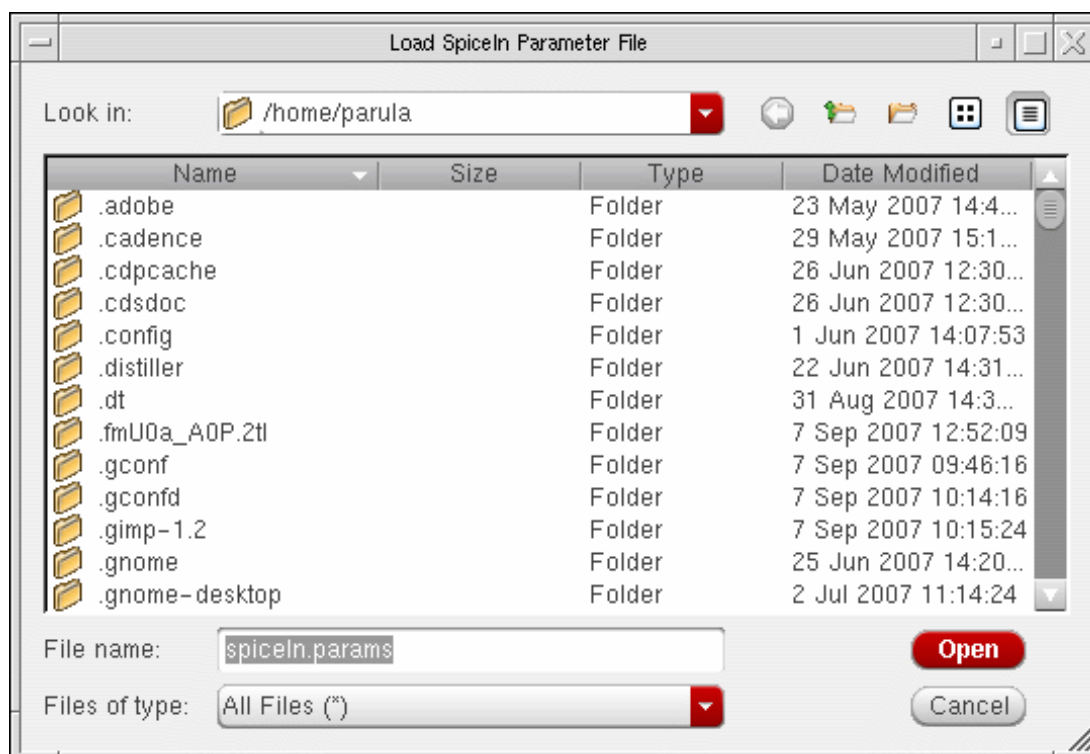
To load a parameter file, perform the following steps:

1. Click *Load*.

Design Data Translators Reference

Netlist Import Using Spice In

The *Load SpiceIn Parameter File* window is displayed.



2. Browse to the location where the parameter file is located.
3. Select the parameter file and click *Open*. By default, the parameter file is displayed as *spicein.params*.

The values of the fields on all the tabs of the *Virtuoso Spice In* form are updated with the values from the parameter file. However, if any errors or warnings are displayed in CIW, correct them before proceeding further.

In addition to the option name-value pairs, the Spice In parameter file contains the name of the *Schematic Generation* options parameter file. Therefore, when you click the *Load* button on the *Input* tab, the values of the fields in the *Schematic Generation* tab are updated from the *Schematic Generation* parameter file.

Saving an Option to a Parameter File

You can also save the values in various fields in GUI form to a parameter file. To save the values, perform the following steps:

1. Specify values for one or more GUI fields, and set the options as desired.

Design Data Translators Reference

Netlist Import Using Spice In

2. Click Save. The *Save SpiceIn Parameters* window is displayed.



3. Specify the name of a file in which you want to save the GUI settings. Spice In creates the file if it does not exist in your current working directory.
4. Click *Save*.

The values of the GUI fields for which the default values have been updated are saved to the specified parameter file. The *Schematic Generation* parameters are saved in a separate file *schOpts_<filename>*.

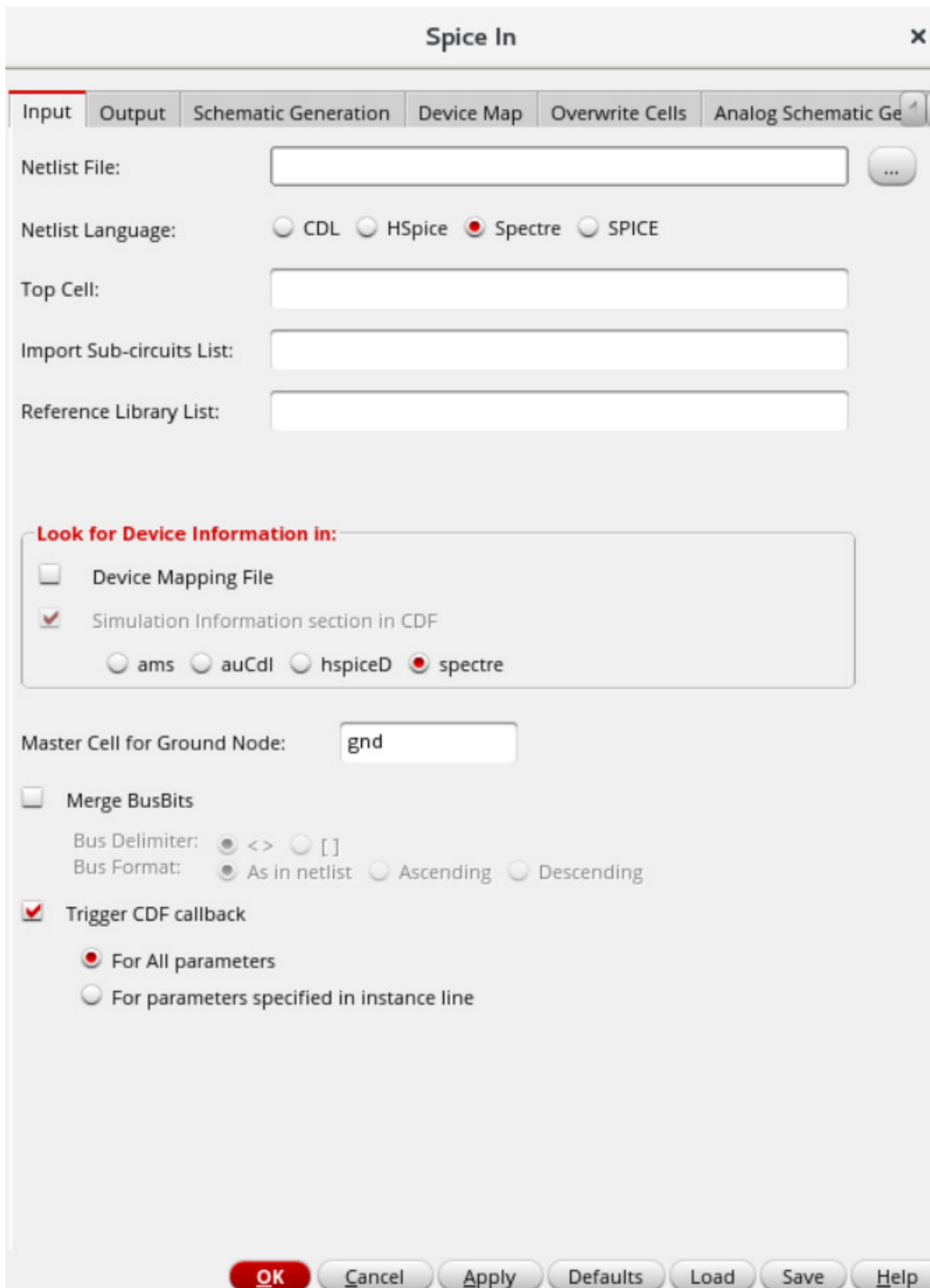
The other options in the Spice In GUI form are grouped into the following tabs:

- ☐ *Input Tab*
- ☐ *Output Tab*
- ☐ *Schematic Generation Tab*
- ☐ *Device Map Tab*
- ☐ *Overwrite Cells Tab*
- ☐ *Analog Schematic Generation Tab*

Design Data Translators Reference

Netlist Import Using Spice In

Input Tab



The image shows the 'Spice In' dialog box with the 'Input' tab selected. The dialog has a title bar 'Spice In' with a close button. Below the title bar are tabs: 'Input' (selected), 'Output', 'Schematic Generation', 'Device Map', 'Overwrite Cells', and 'Analog Schematic Generation'. The 'Input' tab contains the following fields and options:

- Netlist File:** A text field with a browse button (three dots).
- Netlist Language:** Radio buttons for CDL, HSpice, Spectre (selected), and SPICE.
- Top Cell:** A text field.
- Import Sub-circuits List:** A text field.
- Reference Library List:** A text field.
- Look for Device Information in:** A section with a checkbox for 'Device Mapping File' (unchecked) and a checked checkbox for 'Simulation Information section in CDF'. Below this are radio buttons for ams, auCdl, hspiceD, and spectre (selected).
- Master Cell for Ground Node:** A text field containing 'gnd'.
- Merge BusBits:** A checkbox (unchecked).
- Bus Delimiter:** Radio buttons for '<>' (selected) and '['].
- Bus Format:** Radio buttons for 'As in netlist' (selected), 'Ascending', and 'Descending'.
- Trigger CDF callback:** A checked checkbox with two radio button options: 'For All parameters' (selected) and 'For parameters specified in instance line'.

At the bottom of the dialog are buttons: OK (highlighted in red), Cancel, Apply, Defaults, Load, Save, and Help.

The following sections explain the tasks you can achieve using the options on the *Input* tab.

Design Data Translators Reference

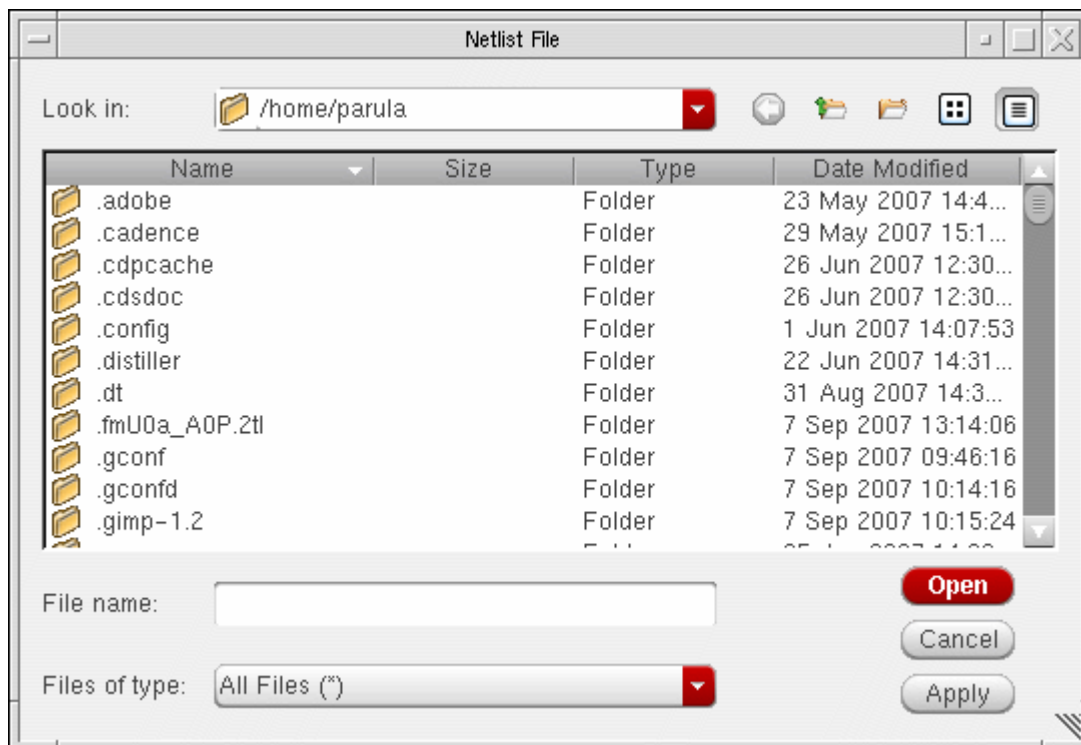
Netlist Import Using Spice In

Specifying the Netlist File

In the *Netlist File* field, specify the netlist you want to import. This is a mandatory field.

If the netlist file is not located in the current working directory, you can either manually specify the absolute or relative path of the file or search for the file by using the *Ellipses (...)* button.

- a. Click the Ellipses (...) button. The *Netlist File* window appears.



- b. Browse to the location where the netlist file is located.
- c. Select the netlist file and click either *Open* or *Apply*.

This would close the *Netlist File* window and the name of the netlist file along with its absolute path appears in the *Netlist File* field.

Specifying the Language of the Netlist File

Select the appropriate *Netlist Language* option to specify the language of the netlist. You can select from the options *CDL*, *HSpice*, *Spectre*, and *SPICE*.

Default Value: Spectre

Specifying the Top Cell

Top Cell specifies the name of the top cell of the design hierarchy to be imported. The top cell concept for a CDL netlist is different from that for HSpice, Spectre, and SPICE netlists.

- For a CDL Netlist

CDL netlists always have instances bound to a `subckt` statement. Therefore, the concept of top cell and “loose” instances, as applicable to HSpice, Spectre, and SPICE netlists, does not hold true for a CDL netlist. The *Top Cell* field will not be available in GUI for a CDL netlist.

- For HSpice, Spectre, and SPICE Netlists

Specify the top cell as the top-level sub-circuit name where all the “loose” instances will be placed. This is illustrated through the example below.

Design Data Translators Reference

Netlist Import Using Spice In

For example, consider the following snippet of the Spectre netlist:

```
simulator language = spectre
global 0 vdd! vss!
parameters CAP=.8p RES=100
```

```
//Library name: test
// Cell name: blockB
// View name: schematic
subckt blockB in out
  R0 (in out) resistor r=RES*2
  C1 (in out) capacitor c=CAP
ends blockB
// End of subcircuit definition.
```

```
// Library name: solutions
// Cell name: amplifier
// View name: schematic
subckt amplifier inh inp iref out inh_bulk_n
parameters my_res my_cap=100f
  Q0 (out net15 net10 inh_bulk_n) trnqn
  C0 (net13 out) capacitor c=CAP
  R0 (net15 net13) resistor r=2.5K
  M0 (net15 inp net30 vdd!) trpmos w=(8u)*16 l=8u
  I0 (net068 net067) blockB
ends amplifier
// End of subcircuit definition.
```

```
// Library name: solutions
// Cell name: ampTest
// View name: schematic
I8 (net5 vin net6 out 0) amplifier new=10
R1 (net5 out) resistor r=20K
I2 (net6 0) isource dc=500u type=dc
V2 (vin 0) vsource mag=1 type=sine sinedc=0 ampl=50m freq=1M
```

Top Cell
Without
'subckt-ends'
Statements

"Loose"
Instances

In this netlist, the "loose" instances, I8, R1, I2, V2, are not contained within any `subckt-ends` statement. These loose instances are ought to be present in a top cell named "ampTest". Top cell is a container cellview (OA database) for all loose instances. All loose instances are a part of a cellview. If there are no global nets, no global parameters, and no loose instances in a cellview, then no topcell will be created.

You can specify this top cell value in the *Top Cell* field if you are importing an HSpice, Spectre, or SPICE netlist.

Default Value: top

Design Data Translators Reference

Netlist Import Using Spice In

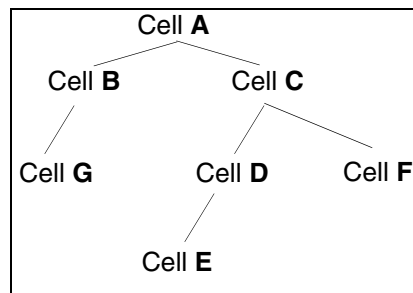
Note: If top cell is given a name which is a sub-circuit in the netlist, an error message will be displayed and import will abort.

Importing the Sub Circuits List

Import sub circuits list is a list of sub-circuits that needs to be completely imported along with their hierarchy. The sub-circuit list is separated by a white space. For example: `amplifier control vsource`.

■ For a CDL Netlist

In the case a CDL netlist, the input netlist might comprise of multiple levels of design hierarchies. In that case, specify the top cell of the part of the design hierarchy you want to import. Consider the hierarchy depicted in the following figure:



If you want to translate cells C, D, E, and F but not A and B, specify cell C as one of the entry in the import sub-circuit list. If you want to translate cells B and C but not A, specify cells B and C as import sub-circuit list entries.

Pin direction for CDL netlist can be set in the following ways:

- The `*.PININFO` control command is used to specify pin type `*.PININFO signalName:pinType` where, `pinType` specifies one of the following pin types:

i or I – input pin

o or O – output pin

b or B – input/output (bidirectional)

p or P – power

g or G – ground

Example

Below is the example of Pin direction for CDL netlist:

Design Data Translators Reference

Netlist Import Using Spice In

```
.subckt inv Y A
*.PININFO Y:O A:I
M1 VDD A Y vdd p w=10u l=2u
M0 Y A gnd gnd N w=5u l=2u
.ends inv
```

- ❑ A slash (/) character in `.subckt` statements distinguishes the output ports (before the slash) and the input ports (after the slash). In the example below, ports `a` and `b` are output ports, and ports `d` and `e` are input ports for cell `xxx`.

```
.subckt xxx a b / d e
```

■ For HSpice, Spectre, and SPICE Netlists

To import all loose instances and/or more hierarchies, enter the name of the top cell and the sub-circuit names in the *Import Sub-Circuits List* field. If the top cell name is not specified in the *Import Sub-Circuits List* field, none of the loose instances and their masters, if any, will be imported.

If you do not specify any value in the *Import Sub-circuits List* field, all the sub-circuits in the netlist will be imported. However, if the name specified in the *Import Sub-circuits List* field is not a sub-circuit in the netlist, a warning message will be displayed. Also, if all the specified names are invalid, an error message will be displayed and Spice In will abort.

Note: While importing sub-circuit Pins, if sub-circuit symbol is present in the output library then the sub-circuit pins directions are created by seeing direction of sub-circuit symbol. However, if sub-circuit pins are not present in the output directory then sub-circuit pins directions are created as bidirectional.

Specifying the Reference Libraries

In the *Reference Library List* field, specify all the reference libraries. The names of reference libraries should be separated by blank spaces. Ensure that the specified reference library names are defined in the `cds.lib` file. This is a mandatory field.

During the netlist import, Spice In searches the specified libraries sequentially for the master cellview definitions for primitive devices. In case of CDL, the sub-circuits with box definition (box elements), the master will be searched in the given reference libraries.

The criteria for match is:

- cell name should match and a symbol view must exist.

Design Data Translators Reference

Netlist Import Using Spice In

- terminal count should match the required number of terminals for a device.
- termOrder information must be present either in siminfo section or device mapping file. There is no concept of default termOrder of primitives.
- pins specified in the termOrder must exist on the corresponding cellview.

Note: If no matching criteria is met and the search fails, Spice In will abort.

Triggering CDF parameters callback

When the *Trigger CDF parameters callback* check box on the *Input* tab is selected, CDF parameters callback is triggered to update the derived or dependent parameters immediately after the Spice In import is complete. If the cellviews are already imported, the callbacks are triggered for the list of cellviews that is to be overwritten in the current Spice In run. Callbacks are triggered for both the netlist and schematic views.

The *Trigger CDF parameters callback* checkbox is selected by default.

Note: Callbacks are not triggered in batch mode. Additionally, CDF Callbacks are not triggered for data types cyclic, radio, and button due to CDBA limitation.

Specifying the Input for Device Information

In the frame *Look for Device Information in* on the *Input* tab, select an option to specify the location where Spice In should search for the device information, `termOrder` in this case. The `termOrder` exists in CDF Simulation Information and is different for different simulators. In contrast to the analog PDKs, such as the `analogLib` from Cadence, where every master cell has CDF, digital PDKs do not have any CDF associated with the master cells. In the case of digital PDKs, therefore, Spice In reads the terminal order through the `termOrder` construct of the device-map file. For more information about the `termOrder`, see [Handling Terminal Order for Sub-Circuits](#) on page 514.

So, in the *Look for Device Information in* frame, you can also select the *Device Mapping File* option. Spice In, accordingly, reads `termOrder` from a device-map file. The `termOrder` specified in the device mapping file will have precedence over `siminfo`.

- Select the appropriate simulator for which you want Spice In to read the `termOrder`. You can select one from *ams*, *auCdl*, *hspiceD*, and *spectre*.

Default value: *spectre*

- If you select the *Device Mapping File* option, specify the name of the device-map file on the *Device Map* tab. For details on creating a device map file, see the [Creating a Device-Map File](#) section of this chapter. If you have selected the *Device Mapping File*

Design Data Translators Reference

Netlist Import Using Spice In

option but not specified any device mapping file in the *Device Map* tab, an error message will be displayed asking you to specify a valid device mapping file in the Device Map tab.

Default value: Deselected

Specifying the Master Cell for Ground Node

The net 0 represents the ground node. Spice In creates connections between all occurrences of net 0 and a ground node instance. For creating the connection, Spice In first creates an instance of the ground node master cell at the sub-circuit level. Next, it connects a net 0 with the ground node instance, and proceeds to connect all occurrences of net 0 in a sub-circuit to the ground node instance.

Consider the following instance definition:

```
R0 (net5 0) resistor r=10K
```

In the above definition, the nets `net5` and `0` are connected to the two terminals of the resistor `R0`. Net `0` is the ground node. At the sub-circuit level, Spice In creates an instance of the ground node master cell. It then connects the net `0` to the ground node instance.

To create an instance of the ground node, Spice In requires the name of the master cell of the ground node. You can update this value in the *Master Cell for Ground Node* field. Spice In searches for this master cell in the reference libraries.

Default value: gnd

Specifying Merge BusBits



Bus Delimiter

Specifies the delimiter for the netlist before import, choose either angular `<>` or square `[]` delimiters. If you specify an incorrect delimiter, it leads to schematics with scalar bits. The Spice In Parameter for this option is `busDelimiter`. Valid values are `squarebracket` and `angularbracket`. The default is `<>`.

Design Data Translators Reference

Netlist Import Using Spice In

Bus Format

Specifies the bus format while merging the bus bits given in the cell definition.

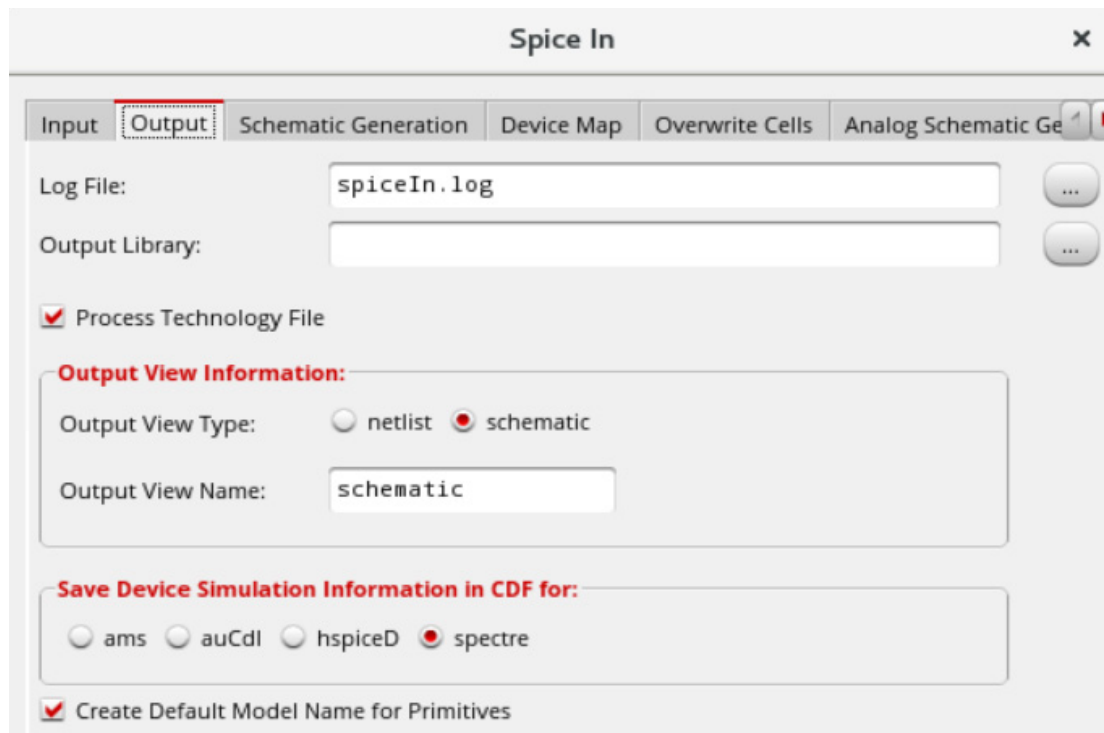
- ☐ *As in netlist*: Merges the vector pins in the same order as provided by the netlist. This is the default.
- ☐ *Ascending*: Merges the vector pins in ascending order.
- ☐ *Descending*: Merges the vector pins in descending order.

For example,

for the pins A<1> A<2> A<0> B, the bus bits are merged as follows: *As in netlist*, A<1:2> A<0> B, *Ascending*, A<0:2> B, *Descending*, A<2:0> B.

The Spice In Parameter for this option is `busFormat`. Valid values are "asinnetlist", "ascending", and "descending".

Output Tab



The following sections explain the tasks you can achieve using the options on the *Output* tab.

Specifying the Log File

In the *Log File* field, specify the name of the log file that Spice In would use for storing progress, error, warning, and information messages. Alternatively, click the ellipses (...) button to search for the log file. All the parameter values that you specify at the time of importing a netlist are also recorded in the log file. The log file also contains messages regarding the device-map feature.

Default value: spiceln.log

Specifying the Output Library

In the *Output Library* field, specify the name of the library to which the netlist file should be imported. Alternatively, click the ellipses (...) button to search for an already existing library to which you want to import the netlist file. If the library does not exist, Spice In will create a new library in the current working directory. Spice In creates the output OpenAccess 2.2 cellviews in the specified output library. This is a mandatory field.

Specifying the Processing of Technology File

The *Process Technology File* option allows you to decide whether you want to create a technology file during the Spice In translation. If you deselect this option, no `tech.db` is created during translation.

Default Value: Selected

Specifying the Output View Information

■ Output View Type

Spice In can import the netlist file as either a Virtuoso Netlist view or a Virtuoso Schematic view. A Virtuoso Netlist view has only the connectivity information, with all the sub-circuits and instances placed at the origin whereas the Virtuoso Schematic view has adequate placement and routing information.

Select the *Output View Type* as either *netlist* or *schematic*.

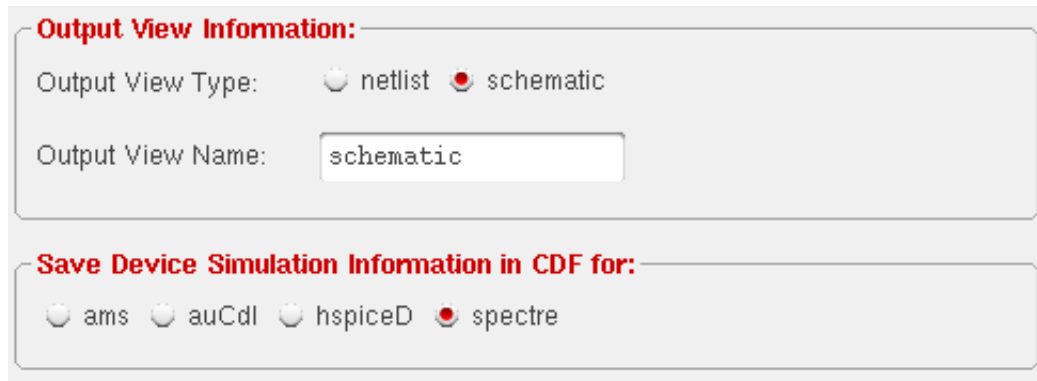
Default value: schematic

■ Output View Name

Design Data Translators Reference

Netlist Import Using Spice In

The default names used by Spice In for the Virtuoso Netlist and Virtuoso Schematic views are `netlist` and `schematic`, respectively. This is shown in the figure below.



Output View Information:

Output View Type: ☐ netlist ☒ schematic

Output View Name:

Save Device Simulation Information in CDF for:

☐ ams ☐ auCdl ☐ hspiceD ☒ spectre

You can change the view name in the *Output View Name* field. For example, you can specify `my_schematic` as the Virtuoso Schematic view name.

Default value: `schematic` (when view type is selected as schematic)

Default value: `netlist` (when view type is selected as netlist)

Specifying the Output for Device Information

Spice In saves information for parameters and `termOrder` of each sub-circuit in the CDF Simulation Information at the base-cell level. Since different Simulation Information sections exist for different simulators, Spice In requires the name of the simulator for which you want to save the `termOrder`.

In the *Save Device Simulation Information in CDF for* frame on the *Output* tab, select a simulator — *ams*, *auCdl*, *hspiceD*, and *spectre* — for which Spice In should save the device information (`termOrder`) for the sub-circuits defined in the input netlist. For more information about the `termOrder`, see [Handling Terminal Order for Sub-Circuits](#).

Default value: `spectre`

Creating Default Model Names for Primitives

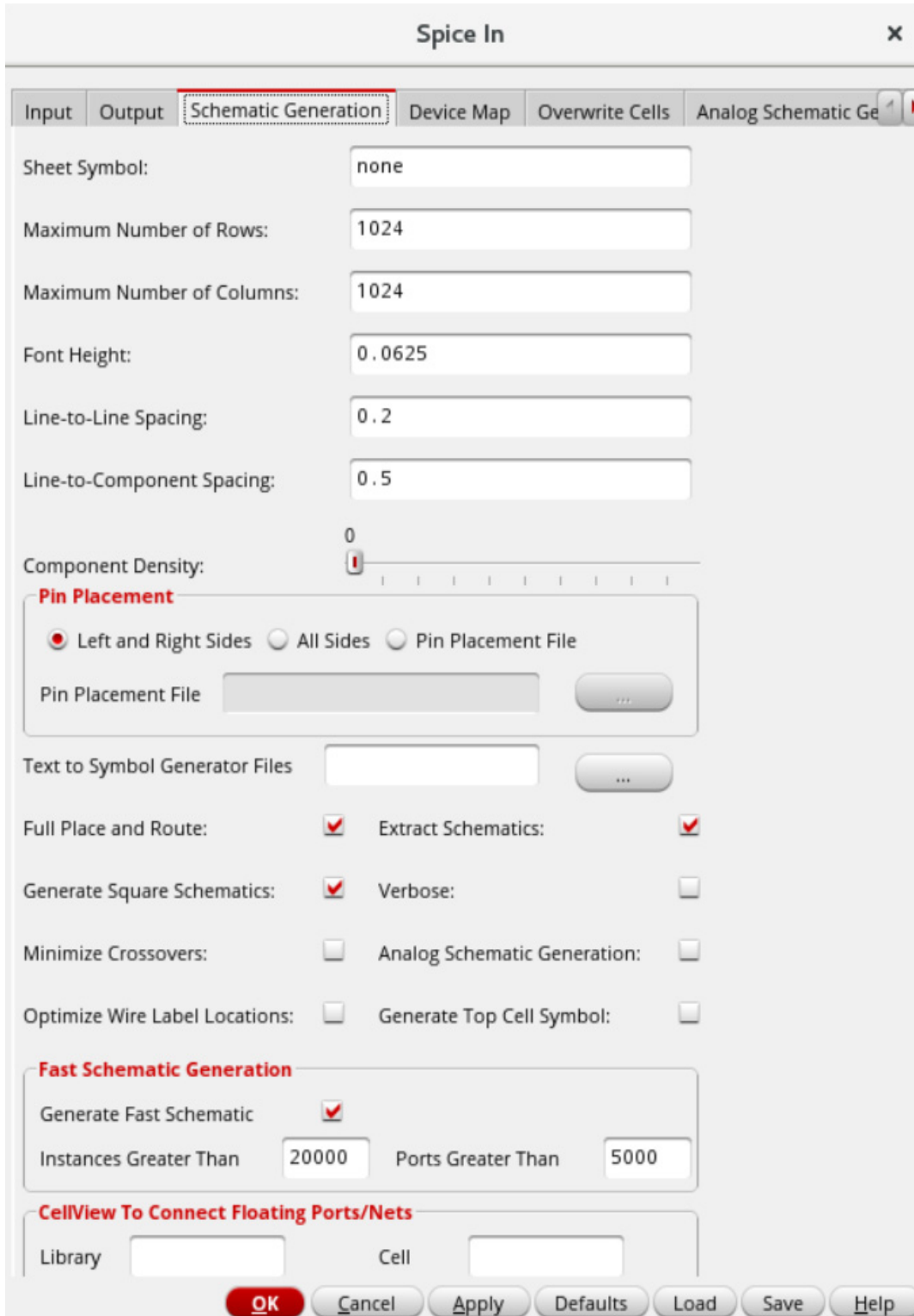
The *Create Default Model Name for Primitives* enables you to specify whether Spice In should automatically add default model names for primitive devices, such as capacitor, resistor, and inductor that have no model name.

Default Value: Selected

Design Data Translators Reference

Netlist Import Using Spice In

Schematic Generation Tab



The image shows the 'Spice In' dialog box with the 'Schematic Generation' tab selected. The dialog has a title bar with a close button (X). Below the title bar is a tabbed interface with tabs for 'Input', 'Output', 'Schematic Generation' (which is highlighted with a red dashed border), 'Device Map', 'Overwrite Cells', and 'Analog Schematic Generation'. The 'Schematic Generation' tab contains several settings:

- Sheet Symbol: none
- Maximum Number of Rows: 1024
- Maximum Number of Columns: 1024
- Font Height: 0.0625
- Line-to-Line Spacing: 0.2
- Line-to-Component Spacing: 0.5
- Component Density: A slider set to 0.
- Pin Placement** section:
 - Radio buttons: ☒ Left and Right Sides, ☐ All Sides, ☐ Pin Placement File
 - Pin Placement File: A text field with a browse button (...).
- Text to Symbol Generator Files: A text field with a browse button (...).
- Checkboxes:
 - Full Place and Route: ☒ Extract Schematics: ☒
 - Generate Square Schematics: ☒ Verbose: ☐
 - Minimize Crossovers: ☐ Analog Schematic Generation: ☐
 - Optimize Wire Label Locations: ☐ Generate Top Cell Symbol: ☐
- Fast Schematic Generation** section:
 - Generate Fast Schematic: ☒
 - Instances Greater Than: 20000 Ports Greater Than: 5000
- CellView To Connect Floating Ports/Nets** section:
 - Library: A text field
 - Cell: A text field

At the bottom of the dialog are buttons for 'OK' (highlighted in red), 'Cancel', 'Apply', 'Defaults', 'Load', 'Save', and 'Help'.

Design Data Translators Reference

Netlist Import Using Spice In

For descriptions of the options in the *Schematic Generation* tab, refer to the *Schematic Generation Options* section in the *Connectivity-to-Schematic User Guide*.

Note: The *Generate Top Cell Symbol* option is available only on the *Schematic Generation* tab of the Virtuoso Spice In form. Select this option to generate the symbol of the top cell during schematic generation.

Note: The *Analog Schematic Generation* option is deselected by default. Also, the Analog Schematic Generation tab page will be deselected.

Design Data Translators Reference

Netlist Import Using Spice In

Device Map Tab

The screenshot shows the 'Spice In' dialog box with the 'Device Map' tab selected. The interface includes a tab bar at the top with 'Input', 'Output', 'Schematic Generation', 'Device Map' (highlighted), 'Overwrite Cells', and 'Analog Schematic Generation'. Below the tabs, there is a 'Current Device Map file:' label and a 'File' button. The main area is titled 'Device Mapping Details' and contains a table with four columns: 'DevToMap', 'MappedTo', 'PropMatch', and 'TermOrder'. The table is currently empty. Below the table are three buttons: 'Add Above', 'Add Below', and 'Delete'. At the bottom, there is an 'Add / Edit' section with several input fields: 'Device to Map:' (a dropdown menu), 'Mapped to:', 'PropMatch:', 'Term Order:', 'Inherited Terminals:', 'User Prop Mapping:', and 'User Added Properties:'. At the very bottom of this section are two buttons: '<<' and '>>', and two larger buttons: 'Commit' and 'Discard'.

The various columns in the *Device Mapping Details* table are described below.

Field	Description
DevToMap	The primitive devices that need to be mapped to a user-defined master cell name. This field cannot be empty.

Design Data Translators Reference

Netlist Import Using Spice In

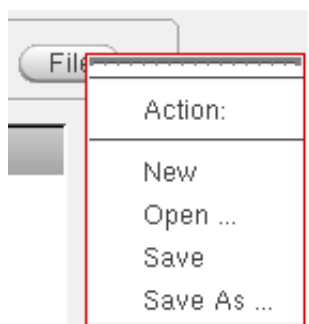
MappedTo	The name of the user-defined master cell name to which the corresponding primitive device is mapped. This field cannot be empty.
PropMatch	Specifies the device properties to match while mapping primitive devices to library cells.
TermOrder	Specifies the terminal order to use.
Inherited Terminals	Specifies the inherited terminals. For example, VDD [@VDD:%:VDD!]
PropMap	Specifies the properties to map.
AddProp	Specifies the properties to be added.

Note: Device mapping is supported only for primitive cell in Spice In.

The following section describes the task you can perform using the *Device Map* tab.

Working with Device Map Files

You can create, open, and save device map files by using the *File* button on the right-top corner as shown in the graphic.



The following table describes the various options available on clicking the *File* button.

Option	Description
New	Initializes the Device Mapping Details table and enables you to create a file.
Open	Opens an existing file.
Save	Saves current file.
Save As	Saves current file with a new name.

Design Data Translators Reference

Netlist Import Using Spice In

Editing the Device Mapping Details

You can use the *Add/Edit* section to create, modify, and delete rows in the *Device Mapping Details* table. After editing the device mapping details, click the *File -> Save* button before running Spice In.

To create a row, select one of the entries in the *Device Mapping Details* section and then click the *Add Above* or the *Add Below* button. The *Add/Edit* section is enabled.

The screenshot shows a dialog box titled "Add / Edit" in red text. It contains several input fields and buttons. The fields are arranged in a grid-like fashion. At the top, there are three fields: "Device to Map:" with a dropdown menu, "Mapped to:" with a text box, and "PropMatch:" with a text box. Below these are two more fields: "Term Order:" with a text box and "Inherited Terminals:" with a text box. Further down are "User Prop Mapping:" with a text box and "User Added Properties:" with a text box. At the bottom left are two buttons labeled "<<" and ">>". At the bottom right are two buttons labeled "Commit" and "Discard".

Specify the device to map, device mapped to, device property to match, and the device term order in the *Device to Map*, *Device Mapped to*, *PropMatch* (Device Property to Match), and *Term Order* fields, respectively and then, click the *Commit* button.

The new row is displayed in the *Device Mapping Details* table.

To modify an existing row, select the row in the *Device Mapping Details* table, modify any of the properties in the *Add/Edit* section, and then click the *Commit* button to accept or the *Discard* button to cancel the changes.

To delete a row, select the row in the *Device Mapping Details* table and then click the *Delete* button.

You can use the << and the >> buttons to traverse through the rows in the *Device Mapping Details* table. When you click either of the buttons, the values in the *Device to Map*, *Device Mapped to*, *Device Property to Match*, and *Device Term Order* fields get populated for the selected row.

Note: If you have specified a device mapping file on *Device Map* tab page but not enabled device mapping option on the *Input* tab page an error message will be displayed asking whether or not you want to enable the device mapping feature.

Design Data Translators Reference

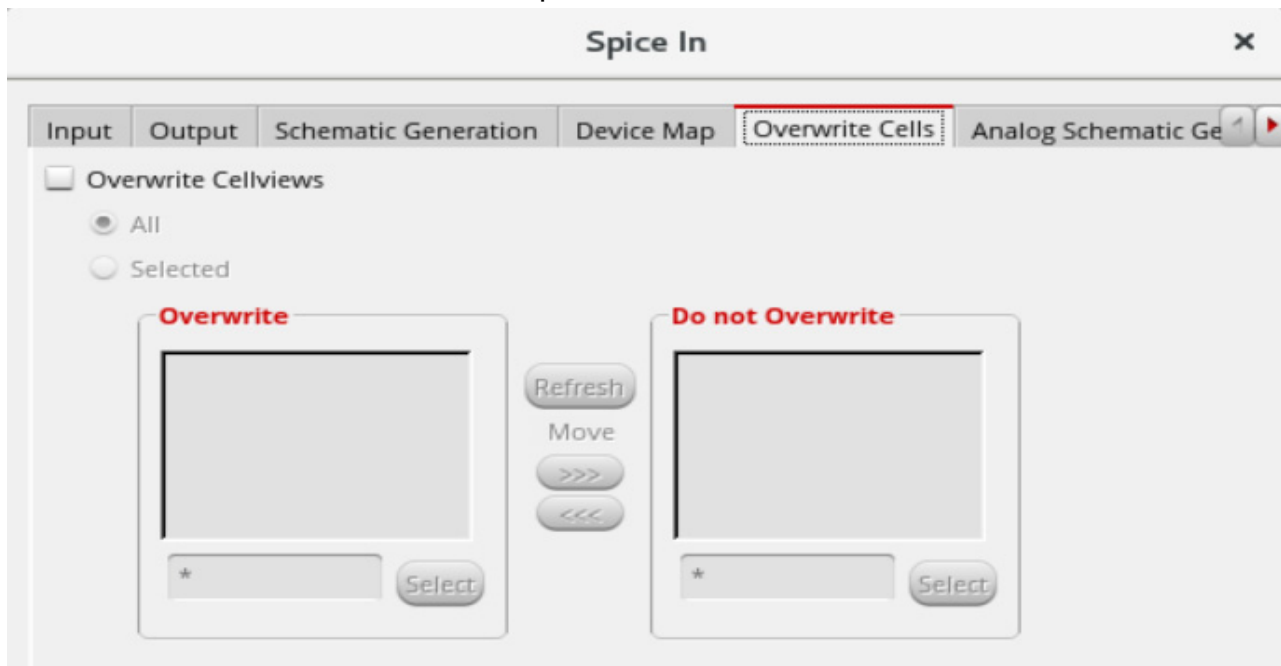
Netlist Import Using Spice In

Note: To link the device map file with Spice In, you must save the file onto the disk and select the *Device Mapping File* option in the *Input* tabbed page.

Overwrite Cells Tab

The *Overwrite Cells* tab enables you to selectively retain cellviews in a netlist that you have already imported in the design. You can also selectively overwrite cellviews that you have imported. The *Overwrite* list box is populated with the cellviews in the output library specified in the *Output* tab. You can use the *Refresh* button to explicitly refresh the list with the cellviews in the current output library.

Note: You must refresh after each import run.



When *Overwrite Cellviews* is selected, symbol gets recreated if the existing symbol view is no longer in use. Symbol gets overwritten for CDL netlist import in the following conditions:

- Mismatch in terminal count or name in symbol view and sub-circuit
- When merge bus bit is set to off and existing symbol view has bus
- When merge bus bit is set to on and existing symbol view does not have a bus and sub-circuit ports form a bus

Additionally, when *Overwrite Cellviews* is enabled and there is a mismatch in the pin direction of the subcircuit definition and existing symbol, then the pin direction of the symbol view is updated.

Design Data Translators Reference

Netlist Import Using Spice In

You can perform the following tasks using the *Overwrite Cells* tab.

■ Overwriting All Cellviews

To overwrite all cellviews, select the *Overwrite Cellviews* check box, select the *All* radio button, and then click *OK* or *Apply*. All the cellviews in the netlist that you have imported will be overwritten.

■ Overwriting Cellviews Selectively

To selectively overwrite cellviews, select the *Overwrite Cellviews* check box and then click the *Selected* radio button. The *Overwrite* and *Do not Overwrite* list boxes are enabled.

To overwrite cellviews, retain the cellviews in the *Overwrite* list box and move the others to the *Do not Overwrite* list box. To move cellviews to the *Do not Overwrite* list box, select the cellviews in the *Overwrite* list box and then click the >>> button. You can use the <<< button to move cellviews from the *Do Not Overwrite* list box to the *Overwrite* list box.

You can also filter the cellviews list by specifying a search string in the text box below the lists and then click the *Select* button.

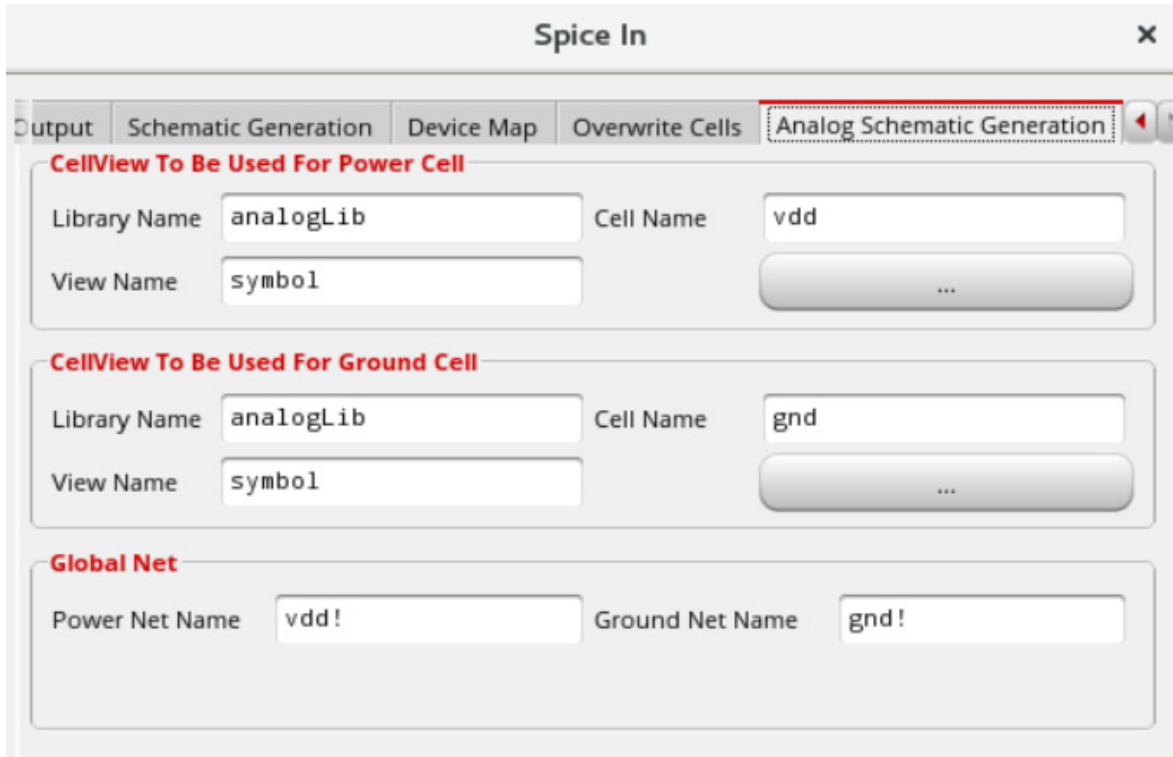
Analog Schematic Generation Tab

For description of the options in the Analog Schematic Generation tab, refer to the *Analog Schematic Generation options* section in the *Connectivity-to-Schematic User Guide*.

Design Data Translators Reference

Netlist Import Using Spice In

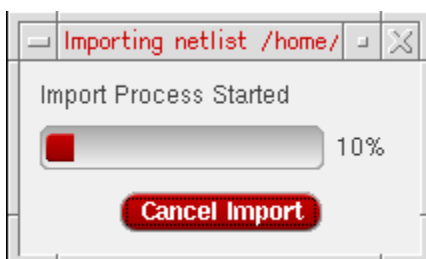
Once you select the *Analog Schematic Generation* option in the *Schematic Generation* tab the Analog Schematic Generation tab page will be deselected.



The **Spice In** dialog box features a tabbed interface with the following tabs: Output, Schematic Generation, Device Map, Overwrite Cells, and **Analog Schematic Generation** (which is currently selected and highlighted with a red border). The dialog is organized into three main sections:

- CellView To Be Used For Power Cell:** Contains input fields for Library Name (analogLib), Cell Name (vdd), and View Name (symbol). A button with three dots is located to the right of the Cell Name field.
- CellView To Be Used For Ground Cell:** Contains input fields for Library Name (analogLib), Cell Name (gnd), and View Name (symbol). A button with three dots is located to the right of the Cell Name field.
- Global Net:** Contains input fields for Power Net Name (vdd!) and Ground Net Name (gnd!).

After you add values in the required fields, you need to click the *OK* button to start the import process. Once you click the *OK* button, the progress bar is displayed.

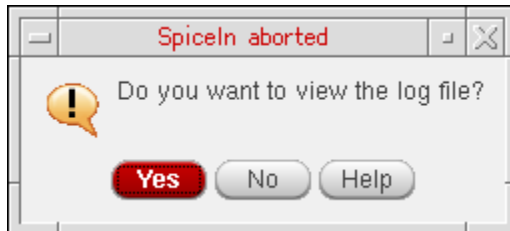


To cancel the import process, you can either click the *Cancel Import* button or the close button .

Design Data Translators Reference

Netlist Import Using Spice In

After you cancel the import process, the Spice In aborted dialog box is displayed.



Design Data Translators Reference

Netlist Import Using Spice In

Customizing the Spice In Environment

To customize the default values of the fields in the Spice In form, do the following:

1. Copy the CDS environment file (`.cdsenv`) from the following location to your current working directory:

```
install_dir/tools/dfII/etc/tools/spiceIn/
```

For more information about the `.cdsenv` file, see Chapter 9, “Specifying Environment Settings”, in the *Virtuoso Studio Design Environment User Guide*.

A sample Spice In `.cdsenv` file is shown below.

```
spiceIn topCell string "" nil
spiceIn logFile string "spiceIn.log" nil
spiceIn overwriteExistingCells string "none" nil
spiceIn language string "Spectre" nil
spiceIn simName string "spectre" nil
spiceIn outputSimName string "spectre" nil
spiceIn outputViewName string "schematic" nil
spiceIn outputViewType string "schematic" nil
spiceIn masterCellForGnd string "gnd" nil
spiceIn devInfo string "Simulation Information Section in CDF" nil
```

2. Update the file based on your requirements and save it.

The following table shows the default Spice In `.cdsenv` variables and the corresponding GUI options.

Spice In <code>.cdsenv</code> Variables	GUI Option
topCell	<u>Top Cell</u>
logFile	<u>Log File</u>
language	<u>Netlist Language</u>

Design Data Translators Reference

Netlist Import Using Spice In

Spice In .cdsenv Variables	GUI Option
simName	This variable corresponds to one of the following options of ■ <i>spice</i> ■ <i>auCdl</i> ■ <i>hspiceD</i> ■ <i>spectre</i> The default value of this variable is <i>spectre</i>. The value for the variable <code>simName</code> cannot be NULL.
outputSimName	<u><i>Save Device Simulation Information in CDF for</i></u>
outputViewName	<u><i>Output View Name</i></u>
outputViewType	<u><i>Output View Type</i></u>
masterCellForGnd	<u><i>Master Cell for Ground Node</i></u>

3. Restart Spice In to load the changed settings.

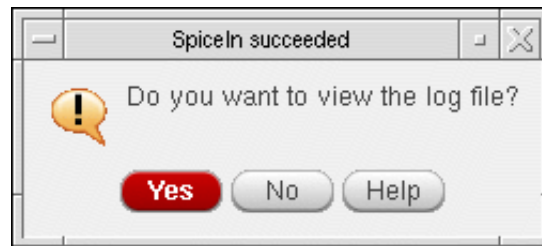
Design Data Translators Reference

Netlist Import Using Spice In

Running Spice In

1. Start the Spice In GUI. (see [Launching the Spice In GUI](#) on page 459)
2. Specify the required input information in the *Input* and *Device Map* tabs to import your netlist. You can customize the output schematic using the options on the *Schematic Generation* tab. (see [Working with the Spice In GUI](#) on page 461)
3. Click *OK* or *Apply*. Spice In begins the import.

Based on whether the execution is successful or not, the *Spice In succeeded!* or the *Spice In failed!* dialog box comes up.

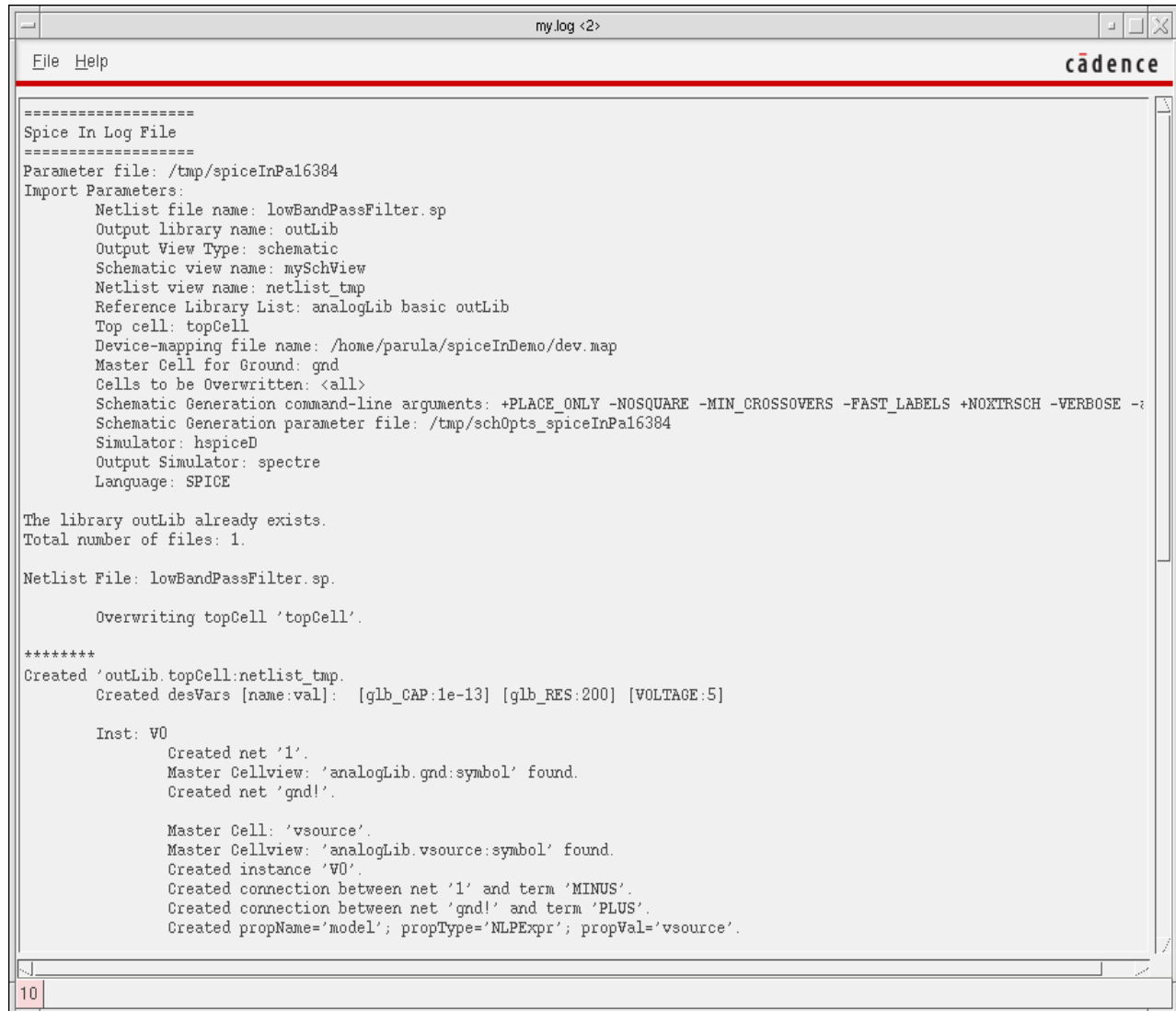


4. Click *Yes* to view the log file.

Design Data Translators Reference

Netlist Import Using Spice In

The `<logfile>` window appears.



```
=====
Spice In Log File
=====
Parameter file: /tmp/spiceInPa16384
Import Parameters:
  Netlist file name: lowBandPassFilter.sp
  Output library name: outLib
  Output View Type: schematic
  Schematic view name: mySchView
  Netlist view name: netlist_tmp
  Reference Library List: analogLib basic outLib
  Top cell: topCell
  Device-mapping file name: /home/parula/spiceInDemo/dev.map
  Master Cell for Ground: gnd
  Cells to be Overwritten: <all>
  Schematic Generation command-line arguments: +PLACE_ONLY -NOSQUARE -MIN_CROSSOVERS -FAST_LABELS +NOXTRSCH -VERBOSE -:
  Schematic Generation parameter file: /tmp/schOpts_spiceInPa16384
  Simulator: hspiceD
  Output Simulator: spectre
  Language: SPICE

The library outLib already exists.
Total number of files: 1.

Netlist File: lowBandPassFilter.sp.

  Overwriting topCell 'topCell'.

*****
Created 'outLib.topCell:netlist_tmp'.
  Created desVars [name:val]: [glb_CAP:1e-13] [glb_RES:200] [VOLTAGE:5]

  Inst: V0
    Created net '1'.
    Master Cellview: 'analogLib.gnd:symbol' found.
    Created net 'gnd!'.

    Master Cell: 'vsource'.
    Master Cellview: 'analogLib.vsource:symbol' found.
    Created instance 'V0'.
    Created connection between net '1' and term 'MINUS'.
    Created connection between net 'gnd!' and term 'PLUS'.
    Created propName='model'; propType='NLPEXPR'; propVal='vsource'.
```

You can go through this file to view the progress of Spice In as well as the intermediate and final outputs generated by Spice In.

Running Spice In from Command Line

You can run Spice In from the command line by using the command `spiceIn` and specifying a parameter file name. The parameter file must contain the mandatory import parameter options (`netlistFile`, `outputLib`, and `refLibList`) and their values. Specify the parameter file using the `param` option. For more information about the parameter file, see [Creating a Spice In Parameter File](#) and [Mapping Parameter File Options with the Spice In GUI Options](#).

Use the following syntax for running the `spiceIn` command:

Syntax

```
spiceIn -param <paramFileName>
```

The other options that can be used with the `spiceIn` command are listed below.

<code>-help</code>	Displays the descriptions of the command-line options.
<code>-V</code> or <code>-version</code>	Provides Cadence release version.
<code>-W</code>	Provides Cadence release subversion.
<code>-param paramFile</code> or <code>-p paramFile</code>	Takes SKILL parameter file as input.
<code>-netlistFile netlistFile</code>	File to be imported.
<code>-outputLib outputLibraryName</code>	Output library for imported data.
<code>-language language</code>	Netlist language.
<code>-reflibList refLibList</code>	List of reference libraries (in double quotes if multiple libraries).
<code>-devmapFile devMapFile</code>	Device mapping to be used.
<code>-noAsg</code>	Do not use Analog Schematic Generation.

Note: The parameter values specified in command line override the values in the parameter file.

Design Data Translators Reference

Netlist Import Using Spice In



The `spiceIn` command options `V`, `W`, and `version` are case-sensitive. The remaining commands are NOT case sensitive.

Inherited Connections Supported

Inherited connections enables you to selectively override global signals within your design originated in the Virtuoso® Schematic Editor. To have this override information available to other Cadence® tools across the design flow, the information is communicated to the software by means of net expressions and a class of properties called `netSet` properties.

Initially, while importing a spice family netlist in OA database using Spice In, user typically needs symbol views of primitive devices to have exact number of terminals as used on the instance line, failing which the graphical view is not created successfully.

However, now Spice In will be able to use the primitive devices symbol views with lesser number for IO terminals to create graphical views from a textual netlist without breaking the connectivity information.



There should be inherited terminals (nets with net expressions) only on primitive device views. If there are inherited net expressions at any hierarchical cellview level, the solution will not work.

Note: User will be able to write the required information in the device mapping file, which will be used by Spice In to identify the net expression used in the primitive device.

Functional Description

The Cadence Netlister generated netlist tries to resolve inherited connection by providing extra terminals at sub-circuit interface level. This extra terminal set is a union of net expressions in the cellview and net expressions defined on the instance masters of the instances used in the cellview. The generated netlist does not have any marking between the two classes of extra terminals.

Importing Sub-Circuit Terminals

Spice In iterates over all the extra terminals at the sub-circuit interface. Therefore, all extra terminals (ihn*) will be imported as schematic terminals too.

Importing Instance Connections

The instance connection in the netlist may have extra connection representing net expression used in the switch view during netlisting. The textual netlist will not have any information about

Design Data Translators Reference

Netlist Import Using Spice In

the switch view used during netlisting. Therefore, Spice In will search for inherited terminal in the following order:

- First, Spice In will search for inherited terminal in the device map file. To view the syntax of inherited terminal refer to [The inhTerms Construct](#).

Note: The terminal must be a part of `termOrder`.

- If the information is not found in the device map then it will search the user-defined `simview`. For more information, refer to [Specifying the Input for Device Information](#)

To process these extra terminals Spice In will look up the master in the device mapping file supplied by the user to identity the name of net expression used in the switch view. If such information is found in device mapping file, the instance will be created in the schematic with connections depending on how the sub-circuit terminals are created in the schematic.

Creating Extra Sub-circuit Terminals

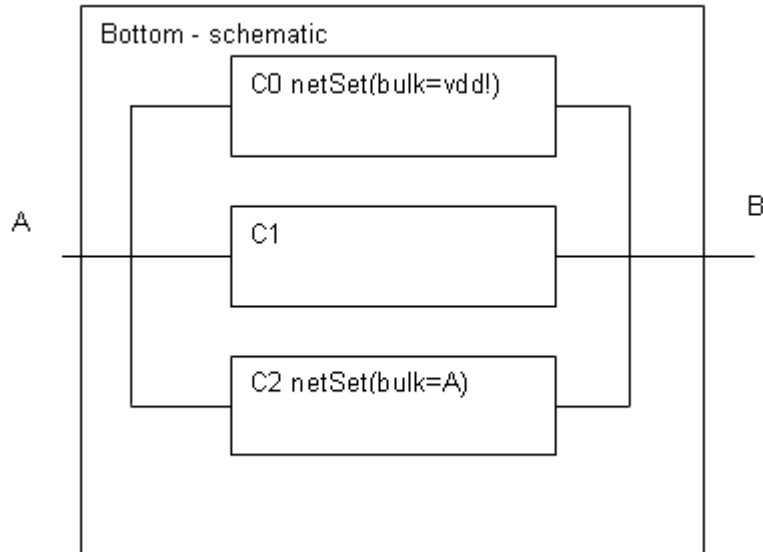
If the sub-circuit is imported with extra terminal, Spice In has the flexibility of not creating `netSet` on hierarchical instances because there are explicit terminals to pass down the value from top to bottom. However, in this case Spice In will create `netSet` on all primitive devices found in the netlist.

Scenario

There can be situations where the primitive device using capacitor with a symbol having two terminals and the switch view having three terminals with one terminal, BULK that has a `netExpression`, `[bulk:%:gnd!]`

Design Data Translators Reference

Netlist Import Using Spice In

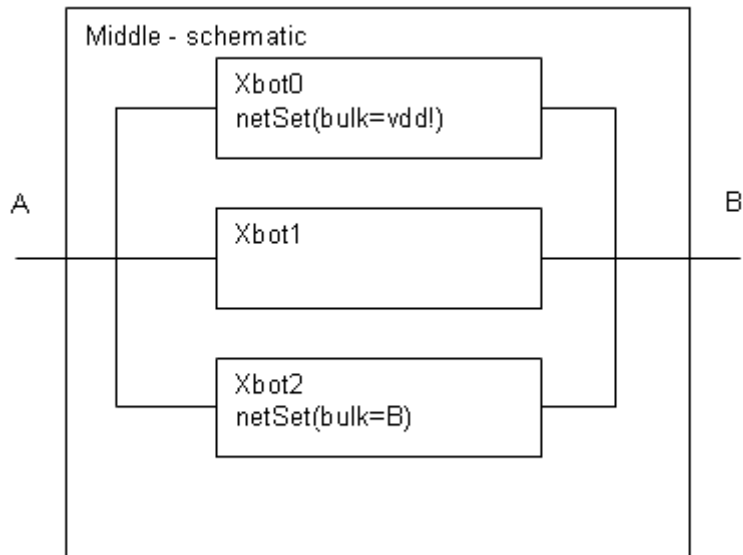


The generated netlist is:

```
.subckt bottom A B inh_bulk ## this is the extra terminal
C0 A B vdd! cap
C1 A B inh_bulk cap
C2 A B A cap
.ends
```

Design Data Translators Reference

Netlist Import Using Spice In

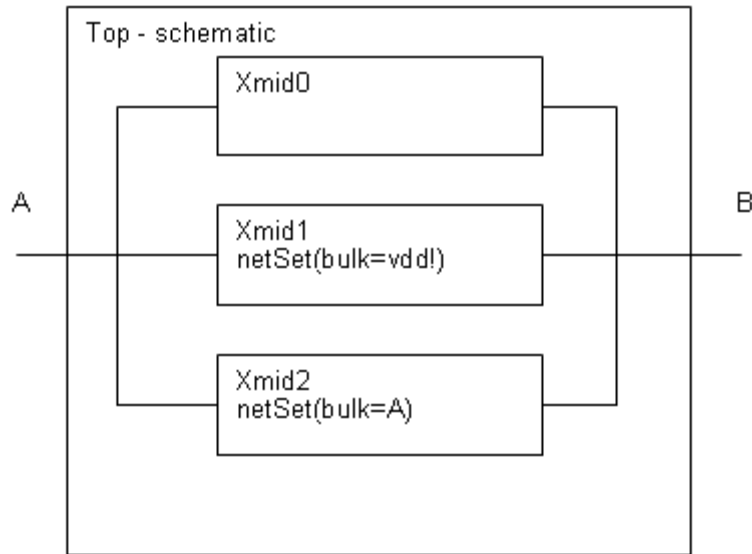


The generated netlist is:

```
.subckt middle A B inh_bulk
Xbot0 A B vdd!
Xbot1 A B inh_bulk
Xbot2 A B B
.ends
```

Design Data Translators Reference

Netlist Import Using Spice In



The generated netlist is:

```
Xmid0 A B gnd! middle
```

Note: Default value (gnd!) automatically created in netlist

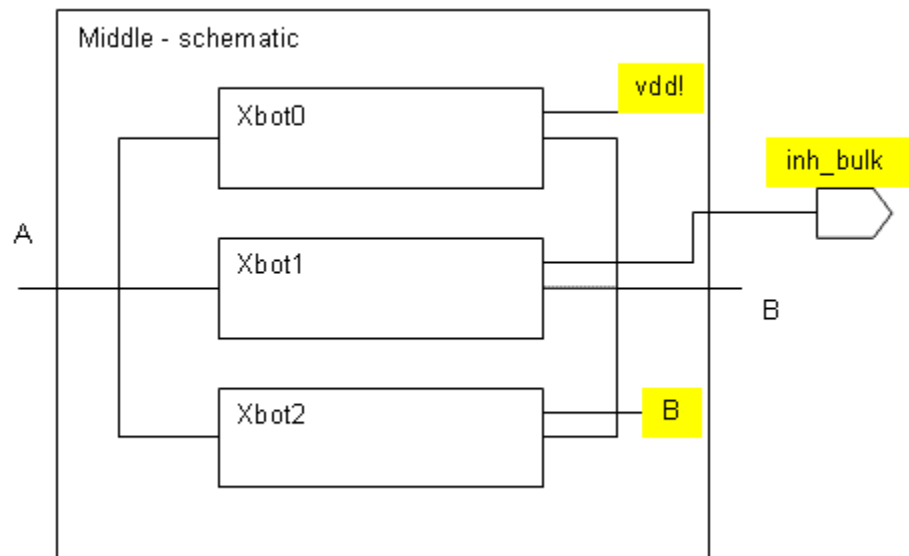
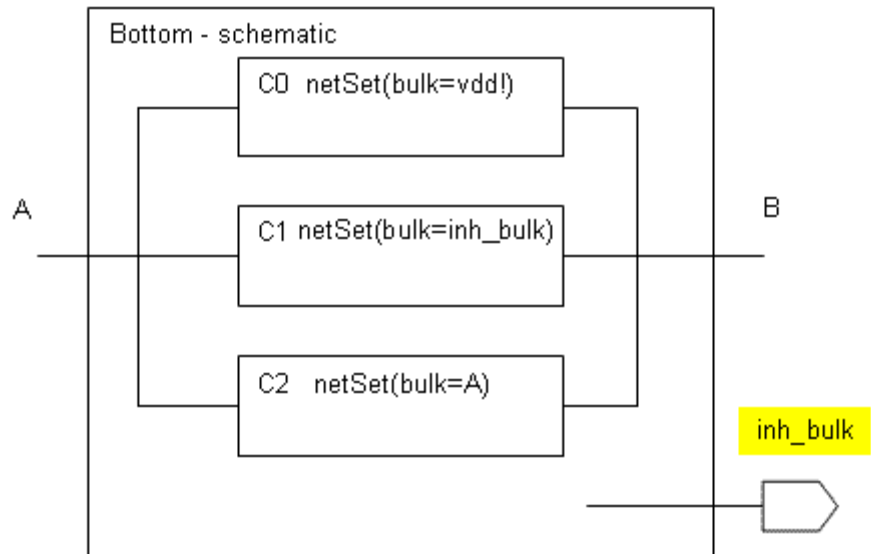
```
Xmid1 A B vdd! middle
```

```
Xmid2 A B A middle
```

Now, if you import the complete netlist in a new Spice In then there can be two schematics with extra terminals.

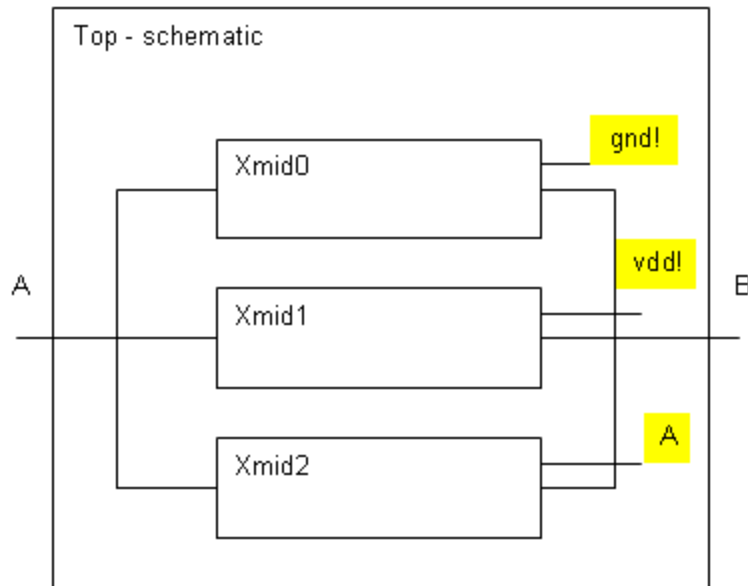
Design Data Translators Reference

Netlist Import Using Spice In



Design Data Translators Reference

Netlist Import Using Spice In



The netlist generated after netlisting the above scenario is:

```
.subckt bottom A B inh_bulk
C0 A B vdd! cap
C1 A B inh_bulk cap
C1 A B A
.ends
```

```
.subckt middle A B inh_bulk
Xbot0 A B vdd! bottom
Xbot1 A B inh_bulk bottom
Xbot2 A B B bottom
.ends
```

//loose instances

```
Xmid0 A B gnd! middle
Xmid1 A B vdd! middle
Xmid2 A B A middle
```


Creating a Spice In Parameter File

The Spice In parameter file, written in SKILL language, is a Disembodied Property List (DPL) of parameters specified as name-value pairs. These values are required to import the netlist file. The various parameters map to different fields in the GUI. The parameter file is used both to populate the fields in the GUI and also to save values from the GUI. You can create a Spice In parameter file in the following ways:

- Use the *Save* button in the Spice In form to save the information currently entered in the form. The information is saved to the file you specify as the parameter file. You can use this method to create a Schematic Generation parameter file as well.
- Copy the sample parameter file, `param.il`, from the `samples/spiceIn` directory to your current working directory and modify the contents of the file as required.

Note: You can use any of the above methods to create a *Schematic Generation* parameter file.

It is mandatory to specify the following parameters and their values in the parameter file:

- `netlistFile`: Input netlist file
- `outputLib`: Name of the output library
- `refLibList`: List of reference libraries

Note: Spice In does not support the following legacy (5141 based CDL In) parameter file options:

`cdlFile
opusLib
primaryCell
viewName
viewType
overWriteExistingCVS
overWriteExistingCVSList`

To convert these CDLIn parameter file options to a Spice In readable parameter file, run the `cdl2spcTmp1.pl` script. The `cdl2spcTmp1.pl` script converts the above CDLIn

Design Data Translators Reference

Netlist Import Using Spice In

parameter file options to the following Spice In parameter options. For more information, type `cd12spcTmp1.pl -h` on the command line. The new mapping is as follows:

<code>cdlFile</code>	<code>netlistFile</code>
<code>opusLib</code>	<code>outputLib</code>
<code>primaryCell</code>	<code>topCell</code>
<code>viewName</code>	<code>outputViewName</code>
<code>viewType</code>	<code>outputViewType</code>
<code>overWriteExistingCVS</code>	<code>overwriteCells</code>
<code>overWriteExistingCVSList</code>	<code>overwriteCellsList</code>

Sample Spice In Parameter File

```
spiceInParams = list(nil
  'simName           "spectre"
  'outputSimName     "spectre"
  'netlistFile       ""
  'outputLib         ""
  'outputViewType    "schematic"
  'outputViewName    "schematic"
  'language          "spectre"
  'devMapFile        ""
  'topCell           "top"
  'logFile           "spiceIn.log"
  'refLibList        ""
  'overwriteCells    "none"
  'overwriteCellsList ""
  'masterCellForGnd  "gnd"
  'paramCaseValue    "lower"
)
```

Note: The option names in the parameter file are case-insensitive when running through command line. However, when loading the option names in parameter file using the GUI, the option names are case sensitive.

Design Data Translators Reference

Netlist Import Using Spice In

Mapping Parameter File Options with the Spice In GUI Options

The following table provides a list of all the Spice In parameter file options and their corresponding Spice In GUI options.

Parameter File			Spice In GUI Option
Parameter Name	Valid Values	Default Value	
language	cdl, hspice, spectre, spice	spectre	<u>Netlist Language</u>
netlistFile			<u>Netlist File</u>
topCell		'top'	<u>Top Cell</u>
outputLib			<u>Output Library</u>
refLibList			<u>Reference Library List</u>
logFile		spiceIn.log	<u>Log File</u>
outputViewType	netlist, schematic	schematic	<u>Output View Type</u>
importSubList		"" (all subckts in netlist)	<u>Import Sub-circuits List</u>
outputViewName	netlist, schematic	- netlist (outputViewType = "netlist") - schematic (outputViewType = "schematic")	<u>Output View Name</u>
devMapFile			<u>Device-Map File</u>
simName	ams, aucdl, hspiced, spectre	spectre	Simulation Information Section in CDF Note: This GUI option is grayed out for this release. This is a known issue and will be taken up in the next release.
outputSimName	ams, auCdl, hspiced, spectre	spectre	<u>Save Device Simulation Information in CDF for</u>

Design Data Translators Reference

Netlist Import Using Spice In

Parameter File			Spice In GUI Option
Parameter Name	Valid Values	Default Value	
masterCellForGnd		gnd	<u>Master Cell for Ground Node</u>
overwriteCells	all, none, selected	none	<u>Overwrite Cells Tab</u>
overwriteCellsList			<u>Overwrite Cells Tab</u>
conn2schArgs			Schematic Generation Tab
+PLACE_ONLY			<i>Full Place and Route</i>
NOSQUARE			<i>Generate Square Schematics</i>
MIN_CROSSOVERS			<i>Minimize Crossovers</i>
FAST_LABELS			<i>Optimize Wire Label Locations</i>
+NOXTRSCH			<i>Extract Schematics</i>
VERBOSE			<i>Verbose</i>
ASG			<i>Enhanced Schematic Generation</i>
conn2schParamFile			No equivalent option in the GUI Used to specify a schematic generation parameter file that contains the schematic generation parameters and their values

Design Data Translators Reference

Netlist Import Using Spice In

Parameter File			Spice In GUI Option
Parameter Name	Valid Values	Default Value	
paramCaseValue	lower		If paramCase value is set to <code>lower</code> , then it will change the parameter to lower case. However, if this option is not specified in the parameter file, then it will take to default param case.

Creating a Device-Map File

A device-map file is a list of mapping between model names and master cells. The model name is present in the definition of an instance in the input netlist.

The tool uses the following model names for different devices:

- `nfet` and `pfet` for a MOSFET device
- `resistor` for a resistor device
- `capacitor` for a capacitor device
- `diode` for a diode device
- `npn` and `pnp` for a BJT device
- `inductor` for an inductor device
- `phyres` for three terminal resistor device

Spice In needs the master cell name of an instance to create the instance in the schematic. You can specify the device-map file name in the *Device Map* form or in the parameter file by using the ``devMapFile` option.

The Spice In device-map file includes six constructs — `devSelect`, `termOrder`, `propMatch`, `inhTerms`, `propMap`, and `addProp`. These are further explained in the sections below.

The devSelect Construct

This construct is used for specifying the device maps in a device-map file. It allows you to replace the name of the master cell of an instance with the user-defined master cell name.

Syntax

```
devSelect := <primitiveDeviceName> <userDefinedMasterCellName>
```

where,

primitiveDeviceName

Master cell of an instance as specified
in the input netlist file.

Design Data Translators Reference

Netlist Import Using Spice In

userDefinedMasterCellName Master cell name that you want Spice In to use for that instance. This master cell is likely present in any of the reference libraries (user's PDKs).

There should be a whitespace character on either side of the construct separator (`:=`). Spice In is case-insensitive to construct names. Therefore, `devSelect` may be written as `DevSeLect` or with any other case variation.

Example 1

In the following example, `trnmos` is mapped to `nmos4` and `trpnp` to `pmos4`.

```
devselect := trnmos nmos4
devselect := trpnp pmos4
```

Example 2

If no device-mapping is used, Spice In assumes the master cell name to be the same as the model name used in the instance definition. It then uses this master cell name to search for its symbol view in the specified reference libraries.

Consider the following instance definition:

```
R0 (net5 0) resistor r=10K
```

In this instance definition, `resistor` is the model name for the instance `R0`. If no device-mapping is used, it can be deduced that the master cell for `R0` is also `resistor`. In that case, Spice In searches for the symbol view of the master cell `resistor` in the reference libraries.

The termorder Construct

The device-map file allows another construct definition, that of the `termOrder`. In the case of digital PDKs, CDF is not present at the cell level. In that case, you can specify the names

Design Data Translators Reference

Netlist Import Using Spice In

and order of the terminals at the master-cell level through the `termOrder` construct of device map.

Syntax

```
termOrder := <list of terminals separated by a whitespace character>
```


The propMatch Construct

Each specified property name must be followed by the property value. The property names must exist in the specified primitive devices. Also, the values of each specified property must match else mapping will not be done for the device. Spice In supports all valid properties of an instance while matching.

The propMatch construct is used in both Spice and CDL context to match on basis of properties to different library cells.

Example:

Consider two voltage devices in a SPICE netlist with different types:

```
V0 M N PULSE DC=1.0
```

```
V1 A B SIN DC=1.0
```

To map each type to a different cell, you need to create device mapping as follows:

```
devSelect := vsource vpulse
```

```
propMatch := type pulse
```

and

```
devSelect := vsource vsin
```

```
propMatch := type sine
```

Example:

Consider the following CDL netlist:

```
.SUBCKT spiceprob A B C D
```

```
M1 A B C D NM1 w=3m
```

```
M2 A B C D NM2
```

```
M3 A B C D NM3 w=1 l=2
```

```
.ENDS
```

To map each type to a different cell, you need to create device mapping as follows:

Design Data Translators Reference

Netlist Import Using Spice In

```
devSelect := nfet nmos4
```

```
propMatch := subtype NM2
```

Note: The CDL property, `subtype` is an internal property that was considered during spice import.

```
devSelect := nfet nmos4
```

```
propMatch := 1 2
```

```
devSelect := nfet nmos
```

```
propMatch := w 0.003
```

The `inhTerms` Construct

This construct is used for specifying the inherited terminals in the device-map file.

Syntax

```
inhTerms := <terminal name> [<NetExp>]
```

Example:

```
inhTerms := B [@bulk_n:%:gnd!]
```

The propMap Construct

This construct is used if properties of the mapped device are different from the primitive device.

Syntax

```
propMap := <list of properties to map>
```

Example:

If the primitive device, `pfet` has properties `w` and `l` and the corresponding property in the mapped device, `pmos` is width and length, then the `propMap` syntax is as follows:

```
propMap := w width l length
```

If both user-defined propmapping (device map propmapping) and `cdfPropmapping` is given then priority is given to user-defined propmapping.

For example, if in `cdfPropmapping` `w` may be mapped to width and in device map file, `w` is mapped to `wid`, then while creating a netlist, `wid` property will be created.

Note: If you do not want to create a property or overwrite an existing property, then you can map this property to another property by using the `propMap` construct in device mapping file.

The addProp Construct

This construct is used to specify additional properties and their values if the mapped device has user-defined string properties. Both property name and property value must be specified. While adding properties, propmapping is not considered.

Note: The property value is of `string` type.

Syntax

```
addProp := <list of additional properties to map>
```

Example:

If the primitive device, `pfet` has an additional property `Name` and the corresponding property value is `Pm01`, then the `addProp` syntax will be as follows:

```
addProp := Name Pm01
```

Handling Parameters

This section explains how Spice In handles global and sub-circuit level parameters.

Global Parameters

- For Spice, Spectre Netlist

Global parameters are defined at the beginning of a netlist and outside the sub-circuit definitions. It is optional to provide a value to a global parameter. The keywords `.PARAM` and `parameters` are used to define a global parameter in HSpice and Spectre netlists, respectively.

Spice In imports global parameters as design variables at the top-cell level. It creates the property `designVarList` at the top-cell level. The value of `designVarList` is a list of property name-value pairs. When you open ADE from VSE, the design variables can be viewed in ADE. You can change the values of the design variables and do pre-processing before the simulation. This feature of Spice In provides a stronger integration channel between VSE and ADE.

Design Data Translators Reference

Netlist Import Using Spice In

The example of amplifier mentioned previously, consider the following snippet of the Spectre netlist:

```
simulator language = spectre
global 0 vdd! vss!
parameters CAP=.8p RES=100
```

Global
Parameter

```
//Library name: test
// Cell name: blockB
// View name: schematic
subckt blockB in out
  R0 (in out) resistor r=RES*2
  C1 (in out) capacitor c=CAP
ends blockB
// End of subcircuit definition.
```

```
// Library name: solutions
// Cell name: amplifier
// View name: schematic
subckt amplifier inh inp iref out inh_bulk_n
parameters my_res my_cap=100f
  Q0 (out net15 net10 inh_bulk_n) trnpn
  C0 (net13 out) capacitor c=CAP
  R0 (net15 net13) resistor r=2.5K
  M0 (net15 inp net30 vdd!) trpmos w=(8u)*16 l=8u
  I0 (net068 net067) blockB
ends amplifier
// End of subcircuit definition.
```

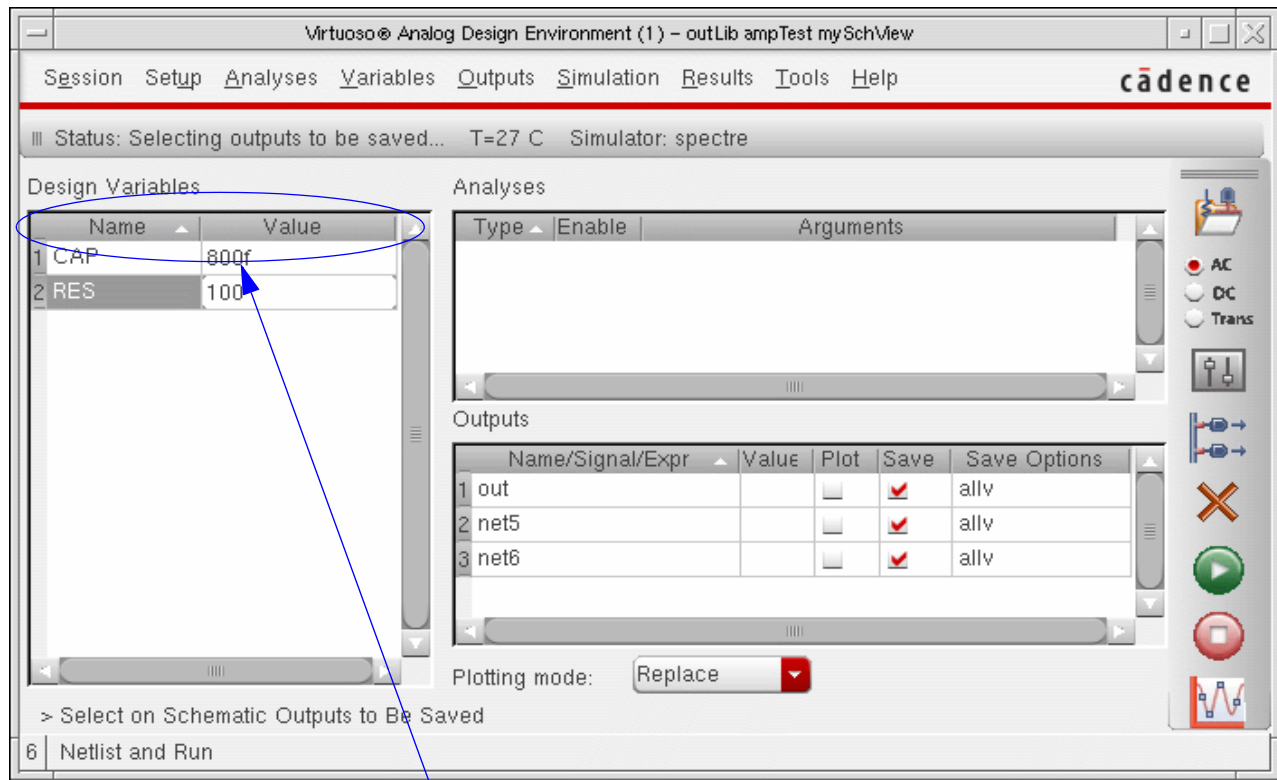
```
// Library name: solutions
// Cell name: ampTest
// View name: schematic
I8 (net5 vin net6 out 0) amplifier new=10
R1 (net5 out) resistor r=20K
I2 (net6 0) isource dc=500u type=dc
V2 (vin 0) vsource mag=1 type=sine sinedc=0 ampl=50m freq=1M
```

This netlist contains the global parameters `CAP` and `RES` with the default value of `.8p` and `100ohms`, respectively. When Spice In imports this netlist, `CAP` and `RES` are created as

Design Data Translators Reference

Netlist Import Using Spice In

design variables on `ampTest`, which is the top cell. The following figure of ADE-L opened from VSE shows that the design variable along with its value has been imported.



Global Parameter CAP

■ For a CDLNetlist

Global parameters are defined usually at the top of netlist using `.PARAM` control statement. The global parameters are added to each sub-circuit that uses the global parameters in its instance line. These global parameters are stored as CDF properties at base level and the default values are assigned as given in the netlist. The data type of these properties is also determined from the netlist.

For example, consider the following snippet of a CDL netlist

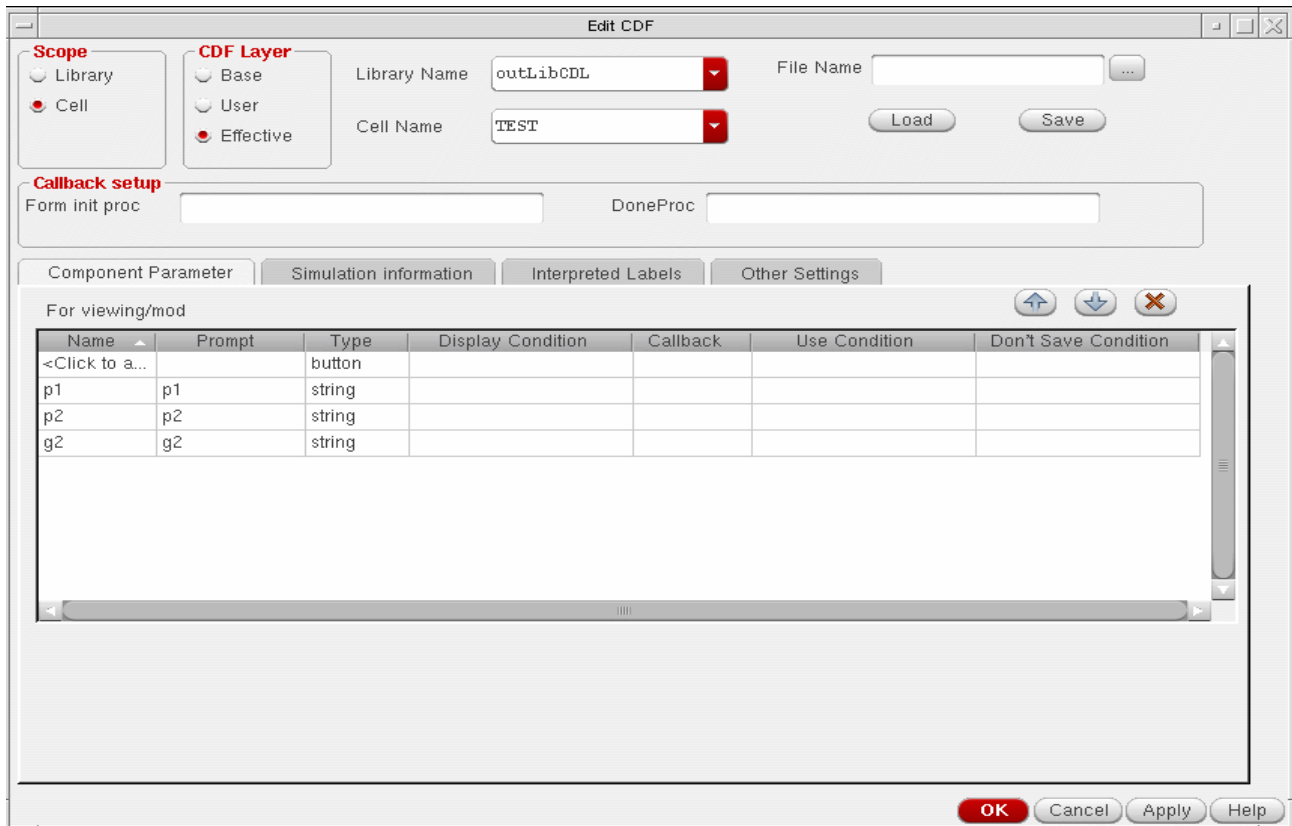
```
.GLOBAL VDD:P GND:P
.SUBCKT TEST Y A p1=10p p2=g1*10e-3
M0 Y A VDD GND NN l=g2 w=p1
.ENDS
```

Design Data Translators Reference

Netlist Import Using Spice In

After importing, the CDF for sub-circuit TEST will be p1(default value 10p) , p2(default value 10u) and g2(default value 20u) .

Note: g2 is appearing in the CDF of TEST because its a global parameter and is used at instance M0.



Sub-Circuit Level Parameters

Sub-circuit level parameters are defined at the sub-circuit level. It is optional to provide a value to a sub-circuit level parameter. Just like global parameters, the keywords `.PARAM` and `parameters` are used to define a local parameter in HSpice and Spectre netlists, respectively. In case of CDL these parameters are specified after then IO pin declaration at `SUBCKT` statement.

Spice In imports sub-circuit level parameters as CDF Component Parameters at the base-cell level for that sub-circuit. The default value assigned to a sub-circuit level parameter is saved as the default value of the corresponding CDF Component Parameter.

Consider the following sub-circuit definition used in the illustrative netlist:

```
// Library name: solutions
// Cell name: amplifier
// View name: schematic
subckt amplifier inm inp iref out inh_bulk_n
parameters my_res my_cap=100f
  Q0 (out net15 net10 inh_bulk_n) trnpn
  C0 (net13 out) capacitor c=CAP
  R0 (net15 net13) resistor r=2.5K
  M0 (net15 inp net30 vdd!) trpmos w=(8u)*16 l=8u
  I0 (net068 net067) blockB
ends amplifier
// End of subcircuit definition.
```


Design Data Translators Reference

Netlist Import Using Spice In

The sub-circuit `amplifier` has the sub-circuit level parameter `my_res` and `my_cap`. No default value is assigned to `my_res`. However, `my_cap` has a default value of `100f`. The following figure shows the CDF Component Parameter after Spice In runs on the given netlist.

Edit CDF

Scope
☐ Library
☒ Cell

CDF Layer
☐ Base
☐ User
☒ Effective

Library Name: File Name:
Cell Name:

Callback setup
Form init proc: DoneProc:

Component Parameter | Simulation information | Interpreted Labels | Other Settings

For viewing/mod

Name	Prompt	Type	Display Condition	Callback	Use Condition	Don't Save Condition
<Click to a...		button				
my_res	my_res	string				
my_cap	my_cap	string				

Default Value: Store Default: Parse as CEL:
Editable Condition: Parse as number:
Units:

Handling Terminal Order for Sub-Circuits

This section explains how Spice In uses CDF Simulation Information for creating net-terminal connections.

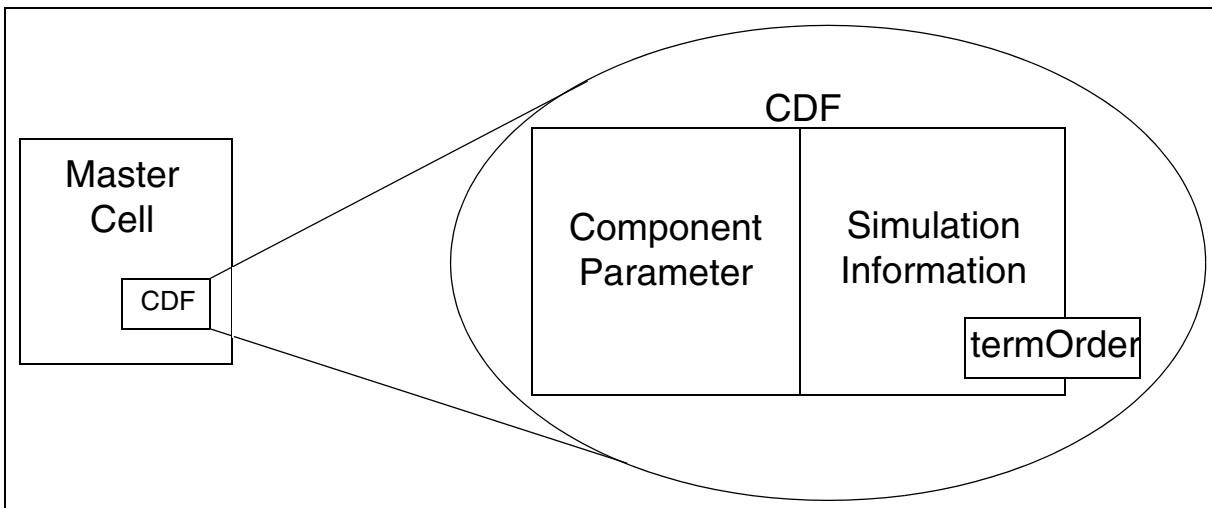
The `termOrder` provides the following information:

- Names of the terminals of the master cell
- Order of the terminals of the master cell during connection

Spice In needs to read `termOrder` to establish connections between the nets of an instance and the terminals of the master cell of the instance.

Source of the Terminal Order Information

In analog PDKs, such as the `analogLib` from Cadence, every master cell has CDF. CDF comprises of two parts — Component Parameters and Simulation Information. The `termOrder` exists in the Simulation Information, as shown in the figure below.



A separate CDF Simulation Information section exists for each simulator, such as *ams*, *auCdl*, *hspiceD*, and *spectre*, and therefore, the `termOrder` is different for every simulator.

Creation of Terminal Order

While importing an input netlist, Spice In does the following:

1. Creates the base cell CDF for each cell
2. Imports any sub-circuit level parameters as the base cell CDF Component Parameters
3. Saves the names and order of the terminals as `termOrder` in the CDF Simulation Information for the specified simulator

You select a simulator in the *Save Device Simulation Information in CDF for* frame on the *Import* tab or specify a simulator name for the parameter `outputSimName` in the parameter file. Spice In creates the `termOrder` in the CDF Simulation Information section for the specified simulator. The device-map file also provides a construct for specifying the `termOrder`. For more information, see [Creating a Device-Map File](#) on page 502.

Consider the following sub-circuit definition from the netlist:

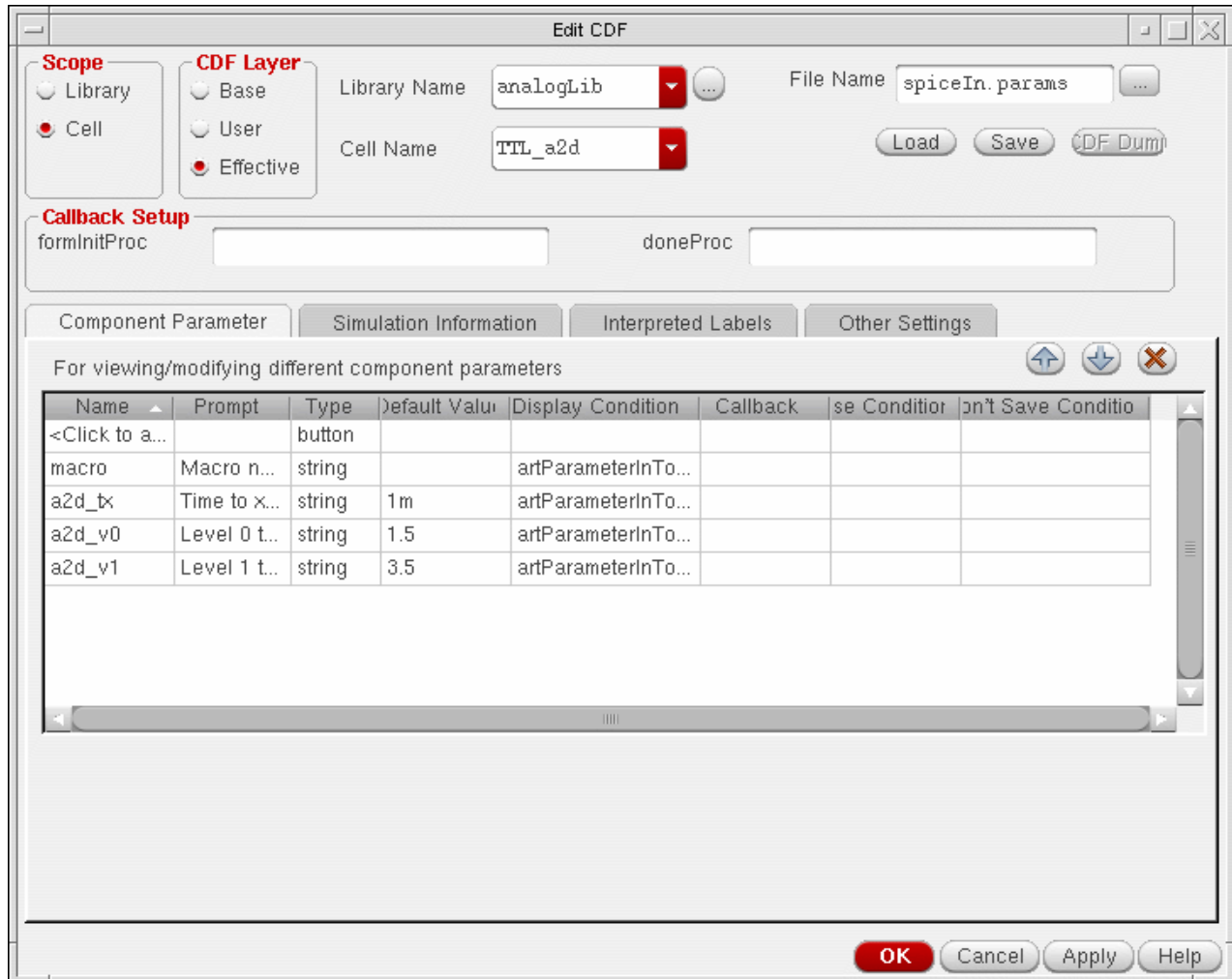
```
// Library name: solutions
// Cell name: amplifier
// View name: schematic
subckt amplifier inh inp iref out inh_bulk_n
parameters my_res my_cap=100f
  Q0 (out net15 net10 inh_bulk_n) trnpn
  C0 (net13 out) capacitor c=CAP
  R0 (net15 net13) resistor r=2.5K
  M0 (net15 inp net30 vdd!) trpmos w=(8u)*16 l=8u
  I0 (net068 net067) blockB
ends amplifier
// End of subcircuit definition.
```

In this scenario, assume that the simulator for which the device information is to be saved is specified as *spectre*. Therefore, Spice In saves the `termOrder` of the sub-circuit `amplifier` in the CDF Simulation Information for *spectre*.

Design Data Translators Reference

Netlist Import Using Spice In

The following figure shows the creation of this `termOrder` for *spectre*:



For information about Spice In SKILL functions, see the [Design Data Translator's SKILL Reference](#).

Design Translation Using CDL Out

This chapter contains the following sections:

- [Overview](#) on page 518
- [Using CDL Out](#) on page 518
 - [Running CDL Out from the GUI](#) on page 522
 - [Running CDL Out from the Command Line](#) on page 540
- [How CDL Out Translates Data](#) on page 548
- [CDL Out Formats](#) on page 559
- [How to Customize CDL Netlist](#) on page 563

Overview

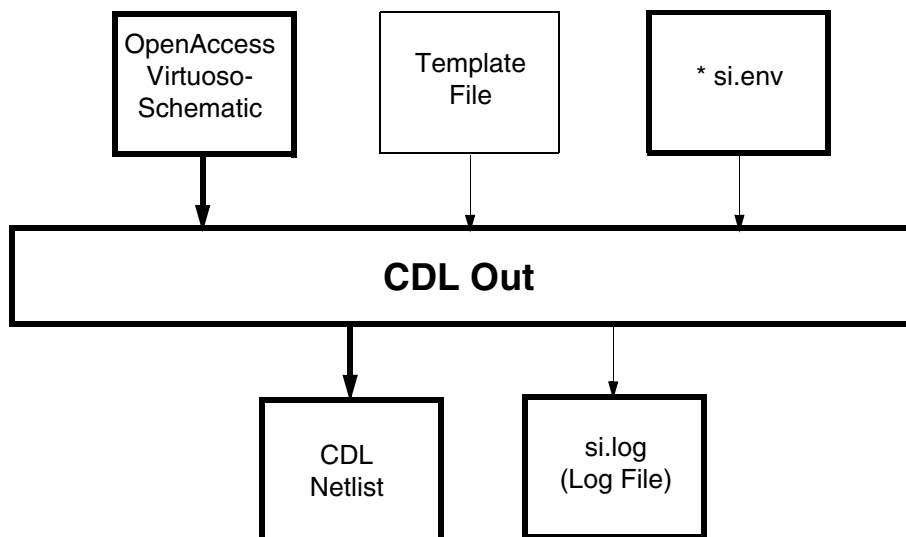
Circuit Description Language (CDL) is a subset of Simulation Program Integrated Circuit Emphasis (SPICE) language.

Since CDL Out is an OSS based netlister, it obeys the OSS licensing scheme. For more information on OSS licensing scheme, refer to the [Open Simulation Systems User Guide](#).

Using CDL Out

Note: This CDL Out documentation is meant for the digital CDL netlister. For information about the analog CDL netlister (auCdl), refer to Appendix C, “auCdl Netlisting”, in the [Virtuoso ADE L User Guide](#).

CDL Out translates an OpenAccess Virtuoso-Schematic view into a CDL netlist. The following figure shows the inputs and outputs of CDL Out.



————— Mandatory
————— Optional

* You must specify an si.env file when you use the `si` command to run CDL Out. The `si` command is the noninteractive, batch mode version of CDL Out.



CDL Out is a 32 bit product and currently runs perfectly under the 64 bit environment. However, it has not been tested thoroughly to validate its use under the 64 bit environment.

Preparing the Virtuoso-Schematic

The lowest levels of the schematic must contain primitives that CDL Out recognizes; for example, transistors, resistors, and capacitors. You must extract and save all levels of the schematic hierarchy before you netlist your design using CDL Out.

If you work on a circuit-level schematic and you create a circuit cell, which uses different parameter values for different instances of that cell, “parameter inheritance is important. For example, you might generate an inverter in your library and use it with 1X drive strength in one cell and with 2X drive strength in another. To ensure that the correct parameter values are passed to the appropriate instances, pass them through the hierarchy.

When you use hierarchical netlisting and parameter inheritance at the same time, special limitations apply. For information about these limitations, see the section [How CDL Out Translates Parameters](#) on page 550.

Creating an Instance Array

CDL Out lets you make an array from an instance. The specific instance is expanded with the name and property type. You can choose the number of times you want CDL Out to expand the instance.

Name	Property Type
CDLMultiplier	integer

For example, if CDLMultiplier = 3 on instance MP8, the netlist appears as follows:

MN8.1	VDD	IN	OUT	C21PLUS	P
MN8.2	VDD	IN	OUT	C21PLUS	P
MN8.3	VDD	IN	OUT	C21PLUS	P

Creating a Black Box

Using black box properties, you can direct CDL Out to not netlist various blocks in a design. You add the black box properties directly to the property list of the symbol view.

The black box feature blocks out a particular cell by adding properties from the symbol view.

Design Data Translators Reference

Design Translation Using CDL Out

You can add the following properties to the symbol view of the instance master:

Name	Property Type
CDLBlackBox	t / nil
CDLGenerateSubcircuitCard	t / nil
CDLPinList	string
CDLParameterNameList	string
CDLParameterValueList	string

CDLBlackBox

If `CDLBlackBox` does not exist or is equal to `nil`, CDL processing occurs as normal for that block. If the property exists and is equal to `t`, the CDL netlister checks the following properties:

Name	Property Type
CDLGenerateSubcircuitCard	
CDLPinList	I/O list of block
CDLParameterNameList	parameters of block
CDLParameterValueList	values of parameters of block

CDLGenerateSubcircuitCard

If you set the property `CDLGenerateSubcircuitCard` to `nil`, the CDL netlister does not generate a subcircuit. The CDL netlister orders the connections for all instances of a block according to the `CDLPinList` value. If you do not set the `CDLPinList` property, or set it to a null string, the CDL netlister orders the connections for all instances of a block alphabetically, using outputs, followed by bidirectionals, followed by inputs.

If you set the property `CDLGenerateSubcircuitCard` to `t`, the CDL netlister generates an empty subcircuit model card in the following form:

```
.SUBCKT  AX  PIN1  PIN2...PINN
.ENDS
```


CDLPinList

The property `CDLPinList` lists the order of connections for all instances in the block. If you do not set the `CDLPinList` property or set it to a null string, the CDL netlister makes the connections for all instances in the block alphabetically, using outputs, followed by bidirectionals, followed by inputs.

An example of the `CDLPinList` property format is as follows:

```
pin1 pin2 ...
```

The format is a string data type. You do not need to use double quotes. Separate each pin name by one or more blanks. If you do not need the `CDLPinList`, do not specify a value.

Note: For the definition of an instance, CDL Out looks for the `CDLPinList` order in the CDL view of the master cell.

CDLParameterNameList

The property `CDLParameterNameList` defines the parameters of the block with values specified by `CDLParameterValueList`. These values are overridden by explicit *Name=Value* assignments in each instance of the block. The netlister does not assign parameters to instances unless the parameters are defined in `CDLParameterNameList`. See the example in the section [“Parameter-Value Inheritance and Schematic Levels”](#).

CDLParameterValueList

The property `CDLParameterValueList` defines the value of the parameters listed by `CDLParameterNameList`. `CDLParameterNameList` is optional. If you do not list the values, CDL Out uses default values.

Design Data Translators Reference

Design Translation Using CDL Out

Running CDL Out from the GUI

In the CIW of a Virtuoso workbench, select *File – Export – CDL*. This displays the *CDL Out* form.

The screenshot shows the "CDL Out" dialog box with the following sections and fields:

- Template File**: A text field with a "Browse..." button, a "Load..." button, and a "Save..." button.
- Design to be Netlisted**:
 - Library Name**: A text field with a "Library Browser..." button.
 - Top Cell Name**: A text field.
 - View Name**: A text field containing "schematic".
 - Switch View List**: A text field containing "auCd1 schematic".
 - Stop View List**: A text field containing "auCd1".
- Output**:
 - Output CDL Netlist File**: A text field containing "netlist" with a "View..." button.
 - Run Directory**: A text field containing "." with a "Browse..." button.
- Netlisting Mode**: Radio buttons for "Digital" and "Analog" (selected).
- Run in Background**: A checked checkbox.
- Renetlist**: An unchecked checkbox.
- Other Inputs**:
 - Analog Netlisting Type**: Radio buttons for "Connection By Order" (selected) and "Connection By Name".
 - Resistor Threshold Value**: A text field containing "2000".
 - Resistor Model Name**: A text field.
 - Equivalents**: A text field.
 - Include File**: A text field with a "Browse..." button.

Design Data Translators Reference

Design Translation Using CDL Out

Check Resistors ☒ value ☐ size ☐ none

Check Capacitors ☒ value ☐ area ☐ perimeter ☐ both ☐ none

Check Diodes ☐ area ☐ perimeter ☒ both ☐ topology ☐ none

Scale ☒ meter ☐ micron ☐ none

Shrink Factor for Width and Length

Check LDD ☐

Display Pin Information ☒

Map Bus Name from < > to [] ☐ Preserve \' in the Netlist ☐

Global Power Signals

Global Ground Signals

Print Inherited Connections ☐

OK Cancel Defaults Apply Help

Template File

Specify the name of the template file in the *Template File* field. If the file is not located in the current working directory, specify the absolute path of the file. The template file is used to save values for the various fields in the form, thereby allowing the values used in one session to be reused in another session.

Browse You can search for the template file in the *Template file* window that opens when you click *Browse*. In the *Template file* window, search for the file, select it, and click either *Open* or *Apply*. The name of the netlist file along with its absolute path appears in the *Template File* field.

Load Specify a template file in the *Template File* field and click *Load*. The values of fields in the form are updated with the values from the template file.

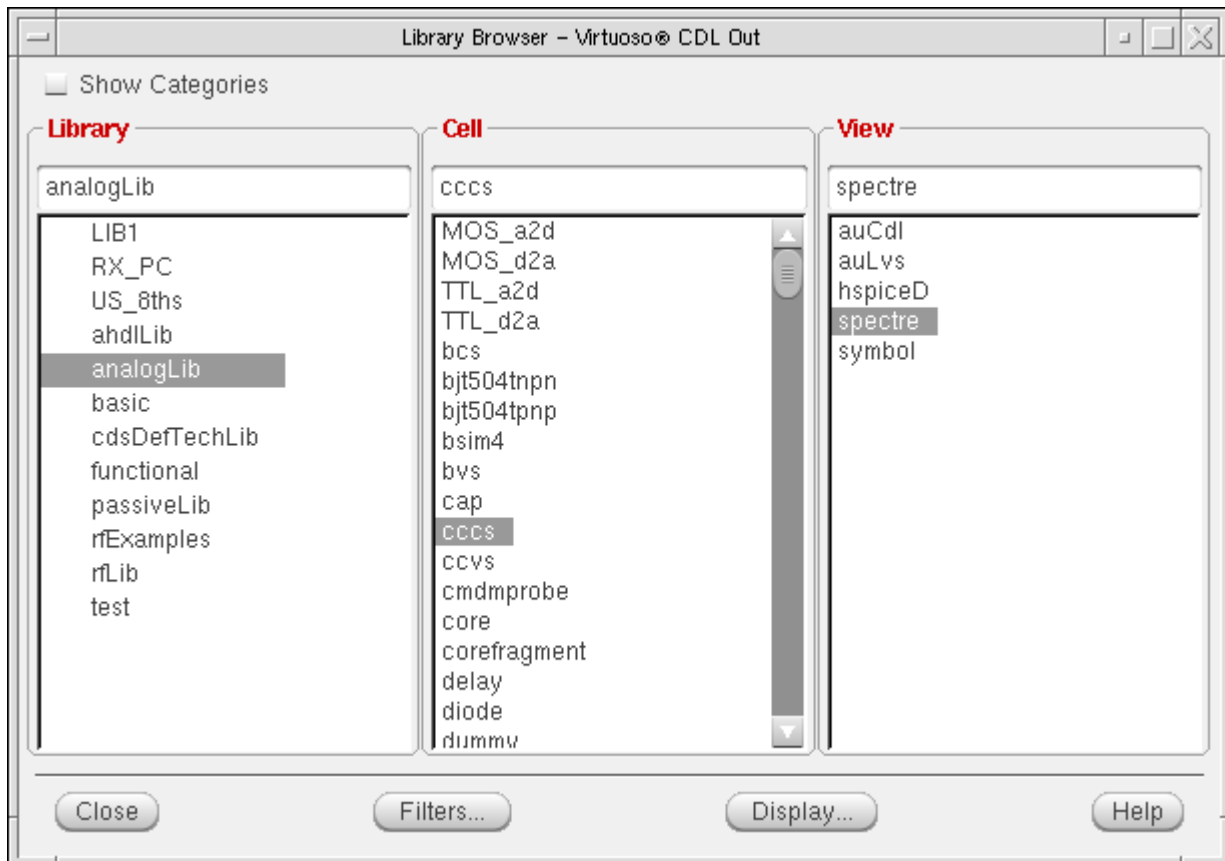
Save After updating one or more fields, click *Save*. The values of the fields are saved to the template file specified in the *Template File* field.

Design Data Translators Reference

Design Translation Using CDL Out

Library Name

Specify the name of the library that contains the view you want to translate from OpenAccess database into CDL format in the *Library Name* field. You can also select a library from a list in the *Library Browser* window that opens when you click the *Library Browser* window.



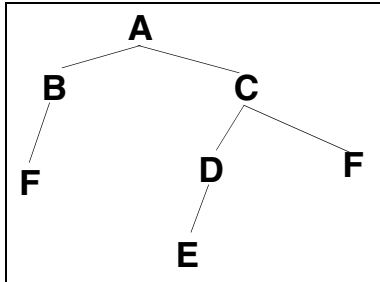
Select the library name, top cell name, and the view name in the *Library Browser* window, and click the *Close* button. The selected values appear in the *Library Name*, *Top Cell Name*, and *View Name* fields in the *CDL Out* form, respectively.

Design Data Translators Reference

Design Translation Using CDL Out

Top Cell Name

Specify the name of the cell at the top of the hierarchy that you want to netlist in the *Top Cell Name* field. For example, assume the cell hierarchy is as follows:



If you want to translate all cells to the output file, enter **A** in this field. If you want to translate just cells C, D, E, and F, but not A and B, type **C** in this field.

You can also populate this field by using the [Library Browser](#) window.

View Name

Specify the view name of the design you want to translate in the *View Name* field. You can also populate this field by using the [Library Browser](#) window.

Default value: **schematic**

Switch View List

Specify a list of the views that the software switches into when searching for design variables.

Stop View List

Specify a list of views that identify the stopping view to be netlisted.

Output CDL Netlist File

Specify the name of the output file in the Output CDL NetList File. CDL Out creates the output file in the run directory you specify in the *Run Directory* field.

Default value: **netlist**

Run Directory

Specify the path of the directory in which you want CDL Out to place the intermediate files in the *Run Directory* field. Alternatively, you can also search for the directory in the *Run Directory* window that opens when you click the *Browse* button. In the *Run Directory* window,

Design Data Translators Reference

Design Translation Using CDL Out

search for the directory in which you want CDL Out to place the intermediate files, select it, and click either *Open* or *Apply*. The name of the directory along with its absolute path appears in the *Run Directory* field.

Default value: current working directory (.)

Netlisting Mode

By default, CDL Out supports two netlisting modes, *Digital* and *Analog*. The *Digital* mode is selected by default if `CDS_Netlisting_Mode` environment variable is not set. If the `CDS_Netlisting_Mode` environment variable is set, the value of `CDS_Netlisting_Mode` is considered.

Valid Values:

<i>Digital</i>	Selects CDL Out.
<i>Analog</i>	Selects auCdl.

Default value: ***Analog***

Run In Background

Runs CDL Out in background mode. When this option is selected, the settings specified in the CDL Out form gets copied in the `si.env` file and then these settings are used to run the netlister, the options specified by variables in CIW gets ignored. It is selected by default. This mode is useful when you run CDL Out on large designs.

Renetlist

Select the *Renetlist* option to set the value of the parameter `simReNetlistAll` in the `si.env` file to `t`. By selecting the Renetlist option, incremental netlisting is turned off and complete design is netlisted even if some of the cells in the design have not been modified since last netlisting.

Default value: Deselected

Analog Netlisting Type

If the *Netlisting Mode* is *Analog*, select either of the two analog netlisting types. The supported types are *Connection By Order* and *Connection by Name*.

Design Data Translators Reference

Design Translation Using CDL Out

Valid Values:

Connection By Order

Nets connected to instance terminals, separated by white spaces, appear in the same order as the order in which master subcircuit or primitive terminals are printed. This is also referred to as an implicit connection.

Note: If this option is selected, then the netlist procedure mentioned in the CDF will be used. If the netlist procedure is not mentioned in the CDF, then `ansCdlSubcktCall` will be used.

Connection By Name

Prints the terminal name on primitive or subcircuit and net name on the corresponding instance terminal in `<termName>=<netName>` format. This is also referred to as explicit connection.

Note: If this option is selected, the netlist procedure `ansCdlHnlPrintInst` is used whether the CDF contains a netlist procedure or not.

Default value: *Connection By Order*

Important

By default, explicit netlisting is done for hierarchical blocks only.

If the `auCdlOverrideNetlistProcForAll` .simrc variable is set to `t`, and *Connection by Name* is selected, then the stopping cells are also netlisted explicitly.

If a cell CDF contains netlist procedure `ansCdlHnlPrintInst`, then this procedure will be used irrespective of the type of netlisting selected, implicit or explicit.

For command mode, the value of the variable `auCdlDefNetlistProc` should be set to `ansCdlHnlPrintInst` for explicit netlisting and `ansCdlSubcktCall` for implicit netlisting.

Resistor Threshold Value

Specify the threshold value at which the resistors should short in the *Resistor Threshold Value* field. You can change the default value by declaring a variable in the `.cdsinit` file. In a situation where you require a *Resistor Threshold Value* of 500 as the default, enter the following line in the `.cdsinit` file:

```
CdlDefaultShortResi = 500.00
```

Default value: **2000 (ohms)**

Resistor Model Name

Specify the resistor model that you want to check in the *Resistor Model Name* field.

Equivalents

Specifying the equivalent item informs the layout versus schematic comparison (LVS) tool that some nets or device models are electrically equivalent. List any equivalent items in the *Equivalents* field. Everything you type in this field appears in the netlist following the keyword `*.EQUIV`. For example, if you type `a=b c=d` in the *Equivalents* field, the line in the netlist shows:

```
*.EQUIV a=b c=d
```

Include File

List the file names to be included during the run in the *Include File* field. Each file name appears on a separate line following the keyword `.INCLUDE`. For example, if you enter the two file names `abc def` in the *Include File* field, the netlist shows:

```
.INCLUDE abc  
.INCLUDE def
```

Alternatively, you can also search for a file in the *Include File* window that opens when you click the *Browse* button. In the *Include File* window, search for the file you want to include during the run and click either *Open* or *Apply*. The name of the file is displayed in the *Include File* field.

Check Resistors

Specify whether CDL Out should print a CDL control command related to resistors in the output CDL netlist file.

Valid Values:

- value** Prints the CDL control command `*.RESVAL` in the output CDL netlist file.
- size** Prints the CDL control command `*.RESIZE` in the output CDL netlist file.
- none** Prints neither `*.RESVAL` nor `*.RESIZE` in the output CDL netlist file.

Default value: **value**

Design Data Translators Reference

Design Translation Using CDL Out

Check Capacitors

Specify whether CDL Out should print a CDL control command related to capacitors in the output CDL netlist file.

Valid Values:

value	Prints the CDL control command <code>*.CAPVAL</code> in the output CDL netlist file.
area	Prints the CDL control command <code>*.CAPAREA</code> in the output CDL netlist file.
perimeter	Prints the CDL control command <code>*.CAPPERI</code> in the output CDL netlist file.
both	Prints both the CDL control commands <code>*.CAPAREA</code> and <code>*.CAPPERI</code> in the output CDL netlist file.
none	Prints neither <code>*.CAPAREA</code> nor <code>*.CAPPERI</code> in the output CDL netlist file.

Default value: **value**

Check Diodes

Specify whether CDL Out should print a CDL control command related to diodes in the output CDL netlist file.

Valid Values:

area	Prints the CDL control command <code>*.DIOAREA</code> in the output CDL netlist file.
perimeter	Prints the CDL control command <code>*.DIOPERI</code> in the output CDL netlist file.
both	Prints both the CDL control commands <code>*.DIOAREA</code> and <code>*.DIOPERI</code> in the output CDL netlist file.
topology	Checks diode topology. Prints No diode statement in the output CDL Netlist file.
none	Prints neither <code>*.DIOAREA</code> nor <code>*.DIOPERI</code> in the output CDL netlist file. However, prints the <code>*.DIODE</code> statement in the output CDL netlist file.

Default value: **both**

During translation, the use of *Check Resistors*, *Check Capacitors*, and *Check Diodes* options determines which of the following statements are written to the output netlist.

`*.BIPOLAR`
`*.RESI`
`*.RESSIZE`

Design Data Translators Reference

Design Translation Using CDL Out

*.RESVAL
 *.CAPA
 *.CAPVAL
 *.CAPAREA
 *.CAPPERI
 *.DIODE
 *.DIOAREA
 *.DIOPERI

These statements appear in different combinations depending on the values selected for the *Check Resistors*, *Check Capacitors*, and *Check Diodes* options. This is shown in the following table.

Check Resistors	Check Capacitors	Check Diodes	Output Netlist Statements
<i>none</i>	<i>none</i>	<i>none</i>	None
<i>size</i>	<i>none</i>	<i>none</i>	*.BIPOLAR *.RESI = 2000 *.RESSIZE *.CAPA *.DIODE
<i>value</i>	<i>none</i>	<i>none</i>	*.BIPOLAR *.RESI = 2000 *.RESVAL *.CAPA *.DIODE
<i>none</i>	<i>value</i>	<i>none</i>	*.BIPOLAR *.CAPVAL *.DIODE
<i>none</i>	<i>area/perimeter</i>	<i>none</i>	*.BIPOLAR *.CAPAREA/ *.CAPPERI *.DIODE
<i>none</i>	<i>none</i>	<i>area/perimeter</i>	*.BIPOLAR *.CAPA *.DIOAREA/ *.DIOPERI
<i>none</i>	<i>none</i>	<i>topology</i>	*.BIPOLAR *.CAPA

Design Data Translators Reference

Design Translation Using CDL Out

Check Resistors	Check Capacitors	Check Diodes	Output Netlist Statements
<i>value</i>	<i>none</i>	<i>topology</i>	*.BIPOLAR *.RESI = 2000 *.RESVAL *.CAPA
<i>size</i>	<i>none</i>	<i>topology</i>	*.BIPOLAR *.RESI = 2000 *.RESSIZE *.CAPA
<i>none</i>	<i>value</i>	<i>topology</i>	*.BIPOLAR *.CAPVAL
<i>none</i>	<i>area/perimeter</i>	<i>topology</i>	*.BIPOLAR *.CAPAREA/ *.CAPPERI

Scale

Specify the scale to be used in the generated netlist by CDL Out.

Valid Values:

- meter*** Prints *.SCALE METER in the netlist.
- micron*** Prints *.SCALE MICRON in the netlist.
- none*** Does not print *.SCALE in the netlist.

Default value: ***meter***

Shrink Factor for width and length

Specify the percentage by which you want the design to shrink in the *Shrink Factor for width and length* field. This field takes a real number. For example, if the transistor width is 10 and length is 1 and you require to shrink the transistor by 80%, then in the generated netlist, the width is 2 and the length is 0.2.

Default value: 0

Check LDD

Select the *Check LDD* option to enable CDL Out to print the statement `* .LDD` in the netlist.

Default value: ***Deselected***

Display Pin Information

Select the *Display Pin Information* option to enable CDL Out to display the pin type information.

Default value: ***Selected***

Map Bus Name From <> To []

Select the *Map Bus Name from <> to []* option to set the mapping of bus pins to []. Once set, the netlister will generate the netlist with that bus mapping only. To change the mapping, you have to exit and restart the application and set the required bus mapping again.

Default value: ***Deselected***

Preserve "!" in the Netlist

Select the *Preserve "!" in the Netlist* option to retain the bang character “!” in the netlist. If you do not select this option, “!” will be removed when creating the netlist.”

Default value: ***Deselected***

Global Power Signals

Specify a list of global power signals in the *Global Power Signals* field. The global power signals specified here will have a designator `:P` attached to them in the netlist.

Global Ground Signals

Specify a list of global ground signals in the *Global Ground Signals* field. The global ground signals specified here will have a designator `:G` attached to them in the netlist.

Note: If the fields are not null, the signals specified in the list are assumed to be final global signals. No attempt is made to search in OpenAccess and then to append/prepend global signals found in OpenAccess (if different).

Print Inherited Connections

Select the *Print Inherited Connections* to print the constructs associated with inherited connections in the output CDL netlist file. This option is the GUI equivalent of the variable `simPrintInhConnAttributes`, which can be set in the `.simrc` file.

Default value: Deselected

To run CDL Out, perform the following steps by using the *CDL Out* form:

1. Select *File – Export – CDL*.

The *CDL Out* form appears.

2. In the *Top Cell Name* field, type the name of the top cell.
3. In the *Library Name* field, type the name of the OpenAccess library containing the view you want to netlist.
4. Specify values for the optional fields you want.
5. Click *Apply* or *OK*.

CDL Out begins the translation.

Note: When you click *OK*, all settings you made in the CDL Out form are saved into the `si.env` file, which is created inside the run directory. The information in the `si.env` file is overwritten every time you change the field value. However, if a `.simrc` file exists, then `.simrc` overrides the current `si.env` values from the CDL Out form. For more information on `si.env` and `.simrc` files, see [How SE Locates Customization Files](#).

Related topic

[How SE Locates Customization Files](#)

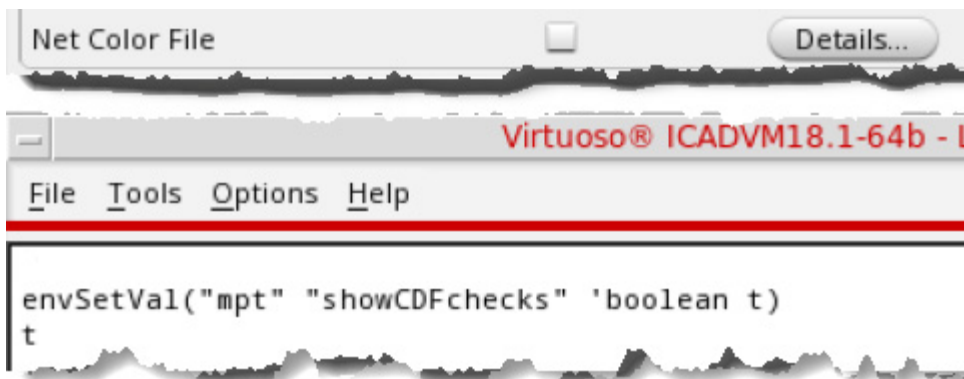
Net Color File

Select the *Net Color File* option to generate the color constraint file to perform Layout Versus Schematic (LVS) coloring checks.

This option is displayed when the `showCDFchecks` environment variable is set to `t`.

Design Data Translators Reference

Design Translation Using CDL Out

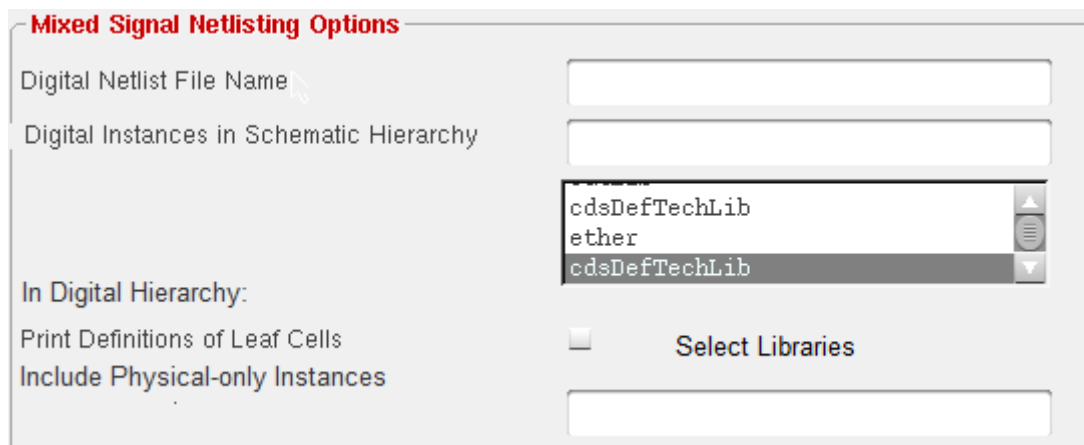


Details

Click *Details* to display the [Export Color Constraint File](#) form.

Support for Virtuoso Schematic and Verilog Driven Mixed-Signal Flow

Using the *Mixed Signal* netlisting mode, included to support the VMS flow, the Embedded Module Hierarchy (EMH) CDL netlister generates the digital and analog CDL netlists, simultaneously.



Digital Netlist File Name

When the *Mixed Signal* netlisting mode is used in the *CDL Out* form, two netlist files are created. One netlist file is for the top-level mixed signal schematic, and the other netlist file is

Design Data Translators Reference

Design Translation Using CDL Out

for the digital module in the Embedded Module Hierarchy (EMH). The digital netlist file is included in the top-level netlist using the `.INCLUDE` statement.

For example, if `top` is the *Top Cell Name* for which the netlist is being generated, the *Digital Netlist File Name* text box will be populated with `block.cdl`, by default, as shown in the following figure. In addition, the *Output CDL Netlist File* text box will be populated with the default filename, `netlist`.

Design to be Netlisted

Library Name: DemoLib Library Browser...

Top Cell Name: top

View Name: schematic

Switch View List: auCd1 schematic

Stop View List: auCd1

Output

Output CDL Netlist File: netlist View...

Run Directory: . Browse...

Netlisting Mode: ☐ Digital ☐ Analog ☒ Mixed Signal

Digital Netlist File Name: block.cdl

Default value: `<top_cell_name>.cdl`

List of Digital Instances in Schematic Hierarchy

This text box contains a list of digital instances that exist in the schematic hierarchy. The list is automatically populated when you specify a top-level design and select the *Netlisting Mode* as *Mixed Signal*.

This field is automatically populated if the top-level design contains instances with the `lxEMHStart` property. You will need to update the mapped name to ensure the mapping is correct. For example, (`"DemoLib/Top/schematic " "I0" "DemoLib/Top/Layout"`) should be replaced with (`"DemoLib/Top/schematic" "I0" "DemoLib/Top/Layout" "| I0"`) if names are mapped with a pipe (`|`) prefix in the Layout. This happens when the following environment variable is used during Schematic-to-Layout generation:

```
envSetVal("layoutXL" "prefixLayoutInstNamesWithPipe" 'boolean t)
```

Design Data Translators Reference

Design Translation Using CDL Out

If the top-level design does not contain any instance having the `lxEMHStart` Boolean property, the *List of digital instances in schematic hierarchy* text box is automatically populated without the instance names. For example, if (`"DemoLib/top/schematic" " " "DemoLib/top/layout"`), you need to update the instance names in this text box.

It is recommended that before generating a netlist, you review the list to verify that it contains only valid SKILL forms and update accordingly. You can chose to generate the netlist in any one of the following way:

- Generate the netlist for the entire EMH contained in the specified layout view.
- Generate the netlist for the module hierarchy corresponding to the digital instances listed in this text box.

Sample inputs:

- `I0` and `I1` are two digital instances in the schematic hierarchy

```
(( "DemoLib/top/schematic" "I1" "DemoLib/block/layout") ("DemoLib/
mid/schematic" "I0" "DemoLib/top/layout"))
```

- `I0` is a digital instance in schematic, but has a mapped name `I0` in the target layout cellview. This means that `I0` is bound to the master module of the instance `I0` in the layout module hierarchy.

```
(( "DemoLib/top/schematic" "I0" "DemoLib/block/layout" "|I0"))
```

- `I0` is a digital instance in top-level schematic, and is bound to the top-level module in the layout-embedded module hierarchy.

```
(( "DemoLib/block/schematic" "I0" "DemoLib/block/layout" t))
```

Netlist Macro Instances from These Libraries

Lists all the libraries from the `cds.lib` file.

You can select multiple libraries from the list box. By default, no library is selected.

The digital module hierarchy in layout can contain some instances of analog schematic cells. By default, these instances are considered as stopping instances, and the mixed signal netlister does not print subcircuit definition for master cells of these instances. However, if the *Print Definition of Leaf Cells in Digital Hierarchy* option is selected, an empty leaf cell definition is printed for the master cells of these instances.

However, if one or more libraries are specified in the *Netlist macro instances from these libraries* field, the instances of analog schematic cells from these libraries are considered as

hierarchical instances. The mixed signal netlister descends into these schematic cells and netlists them hierarchically.

Print Definition of Leaf Cells in Digital Hierarchy

The digital module hierarchy in the layout contains instances of standard cells. By default, these instances are considered as stopping instances, and the mixed signal netlister does not print the subcircuit definition for standard cells. If you have the standard cell definitions available from the foundry, you can directly include these definitions during LVS or SVS.

However, if the *Print Definition of Leaf Cells in Digital Hierarchy* option is selected, the mixed signal netlister prints the empty standard cell definitions in the digital netlist.

Default value: Deselected

Only Include Instances of These Physical Cells

The digital module hierarchy in layout may contain physical-only instances, such as filler cell instances. By default, the mixed signal netlister prints all the physical-only instances in the digital hierarchy. If no value is specified in the *Only Include Instances of These Physical Cells* field, all physical instances are printed in the netlist.

You can also use * wild card as a suffix when specifying cell names, such as `FILL*`, `MFILL*`. The physical instances of cells matching the specified pattern are printed in the netlist. However, the other physical-only instances are ignored.

The following table describes the variables that need to be defined in `.simrc` file to control the EMH netlisting behavior:

Table 5-1 Variables that Control Mixed Signal Netlisting Behavior

.simrc variable controlling netlisting behavior	Description	Default value for Mixed Signal CDL Netlisting
<code>hn1EMH.digitalInsts</code>	Specifies the instances in the schematic hierarchy that will be considered as digital instances. Corresponds to <i>List of digital instances in schematic hierarchy</i> field in CDLOut form.	None Note: This field is populated automatically based on the top cell name and the <code>hn1EMH.targetView</code> variable.

Design Data Translators Reference

Design Translation Using CDL Out

.simrc variable controlling netlisting behavior	Description	Default value for Mixed Signal CDL Netlisting
<code>hn1EMH.netlistFile</code>	Specifies the name of the netlist file for the digital hierarchy. A file with this name will be created in <code><simRunDir></code> and is included in the top level netlist with <code>.INCLUDE</code> statement. Corresponds to <i>Digital Netlist File Name</i> field in the <i>CDL Out</i> form.	None Note: This field is populated automatically based on top cell name.
<code>hn1EMH.printEmptyLeafCellDef</code>	Specifies whether the empty subcircuit definitions for standard cells should be printed or not. Note: This variable should be used only if the standard CDL model file is not available. Corresponds to <i>Print definitions of leaf cells in digital hierarchy</i> field in the <i>CDL Out</i> form.	<code>nil</code>
<code>hn1EMH.printPGConns</code>	Specifies whether to create ports for power and ground connectivity across the digital module hierarchy or not. Note: This does not impact top level netlist.	<code>t</code>
<code>hn1EMH.descendLibs</code>	Specifies the names of schematic libraries. If a digital module hierarchy contains an instance of one of these libraries, the netlister creates a non-empty subcircuit definition for its master. Corresponds to <i>Netlist macro instances from these libraries</i> field in the <i>CDL Out</i> form.	None

Design Data Translators Reference

Design Translation Using CDL Out

.simrc variable controlling netlisting behavior	Description	Default value for Mixed Signal CDL Netlisting
<code>hn1EMH.printPhysOnly</code>	Specifies whether the physical only instances in the digital module hierarchy should be printed or not.	<code>t</code>
<code>hn1EMH.physicalCellFilter</code>	Specifies the physical only devices that should be printed in the netlist for digital hierarchy. You can use wildcards, for example, <code>FILL*</code> . Corresponds to <i>Only include instances of these physical cells</i> in the <i>CDL Out</i> form.	None (Print all physical only instances)
<code>hn1EMH.targetView</code>	Only applicable to standalone EMH netlisting (netlisting an entire module hierarchy inside a layout cellview starting with top module). This is mutually exclusive with <code>hn1EMH.digitalInsts</code> variable.	None

Table 5-2 Common Options in the CDL Out Form

Option Name	Description
<i>OK</i>	Used to accept all the changes on the form. All options present on the respective forms are saved to the LEF/DEF file.
<i>Cancel</i>	Used to reject all the changes on the form. Use the <i>Cancel</i> button to exit the <i>Virtuoso CDL Out</i> form.
<i>Defaults</i>	Used to restore the default settings of the form.
<i>Apply</i>	Used to incorporate all the option settings in the form.
<i>Help</i>	Used to invoke online help.

Running CDL Out from the Command Line

1. Prepare the `si.env` file. Among other information, the `si.env` file carries the name of the design to be netlisted.

For more information, see the section [Preparing the si.env File](#) on page 544.

2. Type the following at the command line:

```
si -batch -command netlist
```

Automatic Netlist Inclusion

This feature allows you to include the CDL netlist of the blocks, which do not have a schematic view, in the final top level CDL netlist. During CDL Out, it is helpful if some of the blocks in the design do not have a schematic view but their CDL netlist is available. For such blocks, perform the following steps:

1. Create a blank schematic view.
2. In the schematic view, create dummy pins. The pin names should match with the pins present in the corresponding symbol view.
3. Create the string property `nlAction` on the schematic and symbol view and set its value to `stop`.
4. Create a boolean property `CDL_INCLUDE_FILE` on the schematic view and set its value to `TRUE`.
5. Copy the CDL netlist file `cdlNetlist` of this cell in the schematic view directory.

Now, if you run CDL out on the top level cell, then it will not create the `.subckt` in the netlist for the cellviews which have property `nlAction` set as `stop`. For such cellviews, if the `CDL_INCLUDE_FILE` property is set to `true`, then its netlist file `cdlNetlist` is copied as:

```
./includeFiles/libName_cellName_viewName
```

All these files are included at the bottom of the netlist with the help of `INCLUDE` statement.

Preparing a Template File

A template file is a collection of file names and option values that you can load into the *CDL Out* form. You can create a CDL Out template file in one of two ways:

- Enter values in the *CDL Out* form and click the *Save* command to save the option values to the file you specify as the template file.
- Create a text file. You can copy the sample template file `cdlOut.il` in the `samples/transUI` directory and modify the file.

The required values in the CDL Out template file are the name of the OpenAccess library containing the top level cellview and the top cell name to translate. The other values are optional.

Sample of a Template File

The following is the sample CDL Out template file Cadence supplies in `samples/transUI/CDLOut.il`:

```
cdlOutKeys = list(nil
  'simLibName           "opus"
  'simCellName          "latch.cdl"
  'simViewName          "schematic"
  'hnlNetlistFileName   "netlist"
  'simRunDir            "."
  'shortRES             2000.000000
  'resistorCheck        "value"
  'capacitorCheck       "value"
  'diodeCheck           "both"
  'displayPININFO       t
)
```

Pin Order for Top Cells

In analog mode, CDL Out prints subcircuit pins in alphabetical order. You can add the following statement in the `.simrc` file to enable CDL Out to use termOrder information specified in the CDF section of the `auCdl` netlist.

```
auCdlCDFPinCntrl=t
```

In digital mode, CDL Out by default, prints the subcircuit pins in the following order:

Output	Bidirectional	Input

The pin order shown is applicable for all the subcircuits in the design including the top cell. You can override the default pin order for the top cell by defining the SKILL variable `cdlTopCellPinOrder` in the `.simrc` file. The `cdlTopCellPinOrder` variable must have a string that fulfills the following format requirements:

Design Data Translators Reference

Design Translation Using CDL Out

- The string must consist of O, B, I, and / characters.
- The characters O, B, and I must appear once in a string.
- The characters O, B, and I can appear multiple number of times.
- The character / can be used between other characters.

Example: Assume that you want to specify the pin order for top cells as: input pins, followed by bidirectional pins, followed by /, and then by bidirectional pins and output pins. To do this, set the `cdlTopCellPinOrder` variable in the `.simrc` file as follows:

```
cdlTopCellPinOrder = "IB/BO"
```

Mapping Template File Options with the CDL Out GUI Options

The following table provides a list of all the CDL Out template file options and their corresponding CDL Out GUI options.

Option in the Template File and their Valid and Default Values		CDL Out GUI Option
<code>capacitorCheck</code>	Default: <code>value</code>	<u><i>Check Capacitors</i></u>
<code>cdsNetlistingMode</code>	Default: <code>Digital</code> assuming that the environment variable <code>CDS_Netlisting_Mode</code> is also set to <code>Digital</code>	<u><i>Netlisting Mode</i></u>
<code>checkLDD</code>	Default: <code>nil</code>	<u><i>Check LDD</i></u>
<code>checkScale</code>	Default: <code>meter</code>	<u><i>Scale</i></u>
<code>diodeCheck</code>	Default: <code>both</code>	<u><i>Check Diodes</i></u>
<code>displayPININFO</code>	Default: <code>t</code>	<u><i>Display Pin Information</i></u>
<code>globalGndSig</code>	No default value	<u><i>Global Ground Signals</i></u>
<code>globalPowerSig</code>	No default value	<u><i>Global Power Signals</i></u>
<code>hnlNetlistFileName</code>	Default: <code>netlist</code>	<u><i>Output CDL Netlist File</i></u>
<code>incFILE</code>	No default value	<u><i>Include File</i></u>
<code>preserveBangInNetlist</code>	Default: <code>nil</code>	<u><i>Preserve "!" in the Netlist</i></u>

Design Data Translators Reference

Design Translation Using CDL Out

Option in the Template File and their Valid and Default Values		CDL Out GUI Option
pinMAP	Default: nil	<u>Map Bus Name From <> To []</u>
resistorCheck	Default: value	<u>Check Resistors</u>
resistorModel	No default value	<u>Resistor Model Name</u>
runInBackground	Default: t assuming that cdsNetlistingMode is set to Digital	<u>Run In Background</u>
setEQUIV	No default value	<u>Equivalents</u>
shortRES	Default: 2000 (ohms) assuming that the environment variable CdlDefaultShortRes i is not set in si.env or .simrc	<u>Resistor Threshold Value</u>
shrinkFACTOR	Default: 0.000000	<u>Shrink Factor for width and length</u>
simCellName	No default value	<u>Top Cell Name</u>
simLibName	No default value	<u>Library Name</u>
simRunDir	Default: .	<u>Run Directory</u>
simViewName	Default: schematic	<u>View Name</u>
No equivalent option in the template file.	Can be set using the environment variable simPrintInhConn Attributes in si.env or .simrc.	<u>Print Inherited Connections</u>
No equivalent option in the template file.	Can be set using the environment variable simReNetlistAll in si.env or .simrc.	<u>Renetlist</u>

Preparing the si.env File

To run CDL Out from command line, you must prepare an `si.env` file. The `si.env` file contains few mandatory and few optional variables. If you run CDL Out from the CIW, CDL Out creates an `si.env` file automatically in the run directory. For more information about the `si.env` file, see the [Open Simulation System \(OSS\) Reference](#).



The `.simrc` control file, which can be searched using the CSF mechanism, always overwrites the values in the `si.env` file, set either manually or by using the CDL Out form, in the run directory. For more information on, see [How SE Locates Customization Files](#).

In case, the netlister is invoked from the command line by using the `si -batch -command netlist` command, and `checkScale` is set to `nil` in `si.env` then CDL Out does not print `*.SCALE` statement in the netlist.

Default value: ***meter***

Sample of an si.env File

```
simLibName = "opus"
simCellName = "latch.cdl"
simViewName = "schematic"
hnlNetlistFileName = "netlist"
simRunDir = "/cds/1.0/test/translator/cdlout/paramCase/"
simSimulator = "cdl"
simViewList = '("cdl" "schematic" "gate.sch" "symbol")
simPrintInhConnAttributes = 'nil
simNetNamePrefix          = N
simInstNamePrefix         = X
simModelNamePrefix        = M
hnlMaxLineLength          = 79

preserveALL = t
preserveRES = t [XX]
shortRES    = 250.000000
checkRESSIZE = t
preserveCAP = t
checkCAPAREA = t
preserveDIO = t
checkDIOAREA = t
retainBusses = t
CDLUsePortOrderForPinList = 'nil
```


Variables in the si.env File

A few of these variables are the same as the template file options. Some others are described below.

preserveALL

If the property `preserveALL` is set to `t`, instances of resistors, capacitors, and diodes in the input schematic are netlisted. However, if `preserveALL` is set to `nil`, CDL Out does not print instance definitions of resistors, capacitors, and diodes in the output CDL netlist.

Default: `nil`

preserveRES

If you set the property `preserveRES` to `t`, resistors are preserved for checking in LVS. You can define the variables `shortRES` and `checkRESSIZE` if `preserveRES = t`. Using the optional variable `[XX]`, you can specify a model name that preserves only the specified type of resistor.

Default: `nil`

shortRES

The value you set for `shortRES` defines the threshold of the resistors that are shorted.

checkRESSIZE

The value you set for `checkRESSIZE` is the resistor size. If you set a value for `checkRESSIZE`, LVS will check the resistor size when you run it.

Default: `nil`

preserveCAP

If you set the property `preserveCAP` to `t`, CDL Out preserves capacitors for checking in LVS. You can define the variable `checkCAPAREA` if `preserveCAP = t`.

Default: `nil`

checkCAPAREA

If you set the property `checkCAPAREA` to `t`, LVS will check the capacitor area when you run it.

Default: `nil`

preserveDIO

If you set the property `preserveDIO` to `t`, CDL Out preserves the diodes for checking in LVS. You can define the variable `checkDIOAREA` if `preserveDIO = t`.

Default: `nil`

checkDIOAREA

If you set the property `checkDIOAREA` to `t`, LVS will check the diode area when you run it.

Default: `nil`

retainBusses

If you set the property `retainBusses` to `t`, busses are printed in the same way in the output CDL netlist file as they are in the input schematic. When set to `nil`, CDL Out expands the busses in the schematic to individual scalars.

The following example shows how the display of the busses differs in the output CDL netlist file with the use of this property:

When `retainBusses = t`,

```
BUS<0:3>
```

When `retainBusses = nil`,

```
BUS<0> BUS<1> BUS<2> BUS<3>
```

Default: `nil`

CDLUsePortOrderForPinList

You can set the SKILL variable `CDLUsePortOrderForPinList` in `si.env` or `.simrc`. If you set `CDLUsePortOrderForPinList` to `t`, then the netlist pin order is determined from the `portOrder` property present on the schematic view of a cell, otherwise, the

Design Data Translators Reference

Design Translation Using CDL Out

`portOrder` property is ignored. The `CDLUsePortOrderForPinList` option is applicable to all the cells, including blackboxes.

If the `CDLPinList` property is set on a cellview, then for the corresponding subcircuit in the CDL netlist, pin ordering will be done by the order in which `CDLPinList` has been specified. This overrides the `portOrder` property. If the `CDLUsePortOrderForPinList` is not set to `t` and the `CDLPinList` property has not been specified on a cellview, then the pin order is printed as is available in the database: the output pin first, followed by the input-output and the input pins.

For example, consider the `portOrder` of the cellview `testLib/cell/schematic` to be ("A" "H" "Y") and assume that the property `CDLUsePortOrderForPinList` is either not set or set to `nil`. In this case, the list of terminals as retrieved from the database would be ("H" "A" "Y"), with the directions "inputOutput", "input", and "output" respectively. The pin list in the output CDL netlist would be "Y H A", with the output terminal appearing first.

Default: `nil`



The variables `simSimulator`, `simViewList`, `simNetNamePrefix`, `simInstNamePrefix`, `simModelNamePrefix`, and `hnlMaxLineLength` are passed by CDL to OSS.

How CDL Out Translates Data

There are important differences between CDL format and the OpenAccess Virtuoso-Schematic view. CDL Out generates a netlist hierarchy that duplicates the hierarchy of your design. Each cell in the schematic becomes a separate subcircuit in the netlist. The hierarchical netlister automatically prefixes each instance name with the proper character for its element type; for example, “M” for MOSFET and “R” for resistor. This minimizes mapping and name translation.

CDL Out names instances and nets differently than a flat netlister. A flat netlister maps all the names to unique names. This avoids naming conflicts if you use identical names for instances in different cells of your schematic.

Unlike a flat netlister, CDL Out maps only illegal names to new names in the hierarchical netlist. CDL Out considers a name illegal if it contains illegal characters or is more than 60 characters. CDL Out searches first for illegal characters, which it maps, as shown in the following table.

First Character	Net	Instance
\$; : = \t <space>	N	X
0 to 9	N0 to N9	X0 to X9
A to Z		XA to XZ
a to z		X_A to X_Z

For example, if \$333 is a net name, CDL Out maps it to N333. If \$333 is an instance name, CDL Out maps it to X333.



To avoid conflicting instance names, if a name uses less-than and greater-than signs (< and >), CDL Out strips the greater-than sign from the name and replaces the less-than sign with an underscore. For example, the instance named x1<0> would be mapped to x1_0.

If a name has more than 60 characters, CDL Out maps it to a unique number preceded by one of the following letters:

N for net names

I for instance names

Design Data Translators Reference

Design Translation Using CDL Out

M for macro names

CDL Out strips the following middle illegal characters from the instance names and net names:

semicolon	;
colon	:
equals sign	=
dollar sign	\$
<space>	“ “
tab	\t

Important

If Map Bus Name From <> To [] check box is selected then CDL Out strips middle illegal characters from net names.

CDL Out strips the following middle illegal characters from the net names:

period	.
open parentheses	(
close parentheses	)
open bracket	[
close bracket	]
dollar sign	\$
exclamation point	!
plus sign	+

This section describes how CDL Out translates the following:

- Parameters
- Inherited Connections
- Primitive Components
- Global Signals

How CDL Out Translates Parameters

This section explains the rules for using different types of parameters and parameter inheritance.

Circuit-level simulations frequently involve element and model parameters. When combining parameter inheritance and hierarchical netlisting, Design Framework II parameter inheritance mechanisms are more general than those that CDL supports. The graphic display of the schematic might show different parameter values than those printed in the netlist.

To avoid this problem, when you create your schematics follow the guidelines in this section for assigned parameters, inherited parameters, and inheritance of inherited parameters.

Fixed-Value Inheritance

Most parameters have a fixed value for simple circuits. For example, if you want the instance *m10* transistor in your schematic to have a width of 20 microns, add

```
w = 20u
```

to the property list of the instance representing the *m10* transistor.

In more complex circuits, parameters can be inherited through the schematic hierarchy. Instead of assigning a fixed value to a parameter, use an `nlpExpr` value enclosed in brackets and separated by colons.

Now if you set the *w* parameter of the *m10* transistor to the following:

```
w = [@wn:5u]
```

the *w* property is an `nlpExpr` type, indicating to CDL Out that *w* is an inherited parameter. CDL Out assigns the *wn* value to *w*, if it exists, or assigns the default value of 5u if *w* does not exist.

The arguments in the parameter assignment are as follows:

<i>wn</i>	Name of the property whose value to assign to <i>w</i> .
%	Format argument that causes CDL Out to set the first argument to the default value specified in the third argument, if the first argument does not have a specified value.
5u	Default value for all unspecified elements of this property.

Separate these arguments with colons (:).

Design Data Translators Reference

Design Translation Using CDL Out

The following table shows the parameters to use for sizing the width of a pmos device that is used in building a hierarchy. CDL Out requires the parameters w and l at the bottom level. The names at higher levels can be different as long as they are legal names.

Level	Symbol	Instance
1 (device)	[@w]	w=[@pw1]
2 (gate)	[@pw1]	pw1=[@pw2]
3 (block)	[@pw2]	pw2=[@pw3]
.		
.		
N (block)	[@pw(N-1)]	pw(N-1)=VALUE

For further information about inherited property syntax, refer to the *Open Simulation System Reference Manual*.

Assigned Parameters

You can also assign default values to parameters. If you want all transistors to have a length of 1.2 microns, add the $l = 1.2\mu$ property to the CDL view of the transistor master. CDL Out uses this default length unless you add the l property to a particular transistor to override it for that instance.

To assign default properties for netlisting, add them to the *CDL* view of the element. To assign default parameters for graphic display, add them to the *symbol* view of the element.

CDL Out prints the assigned parameters for instances, nets, and macro elements on the corresponding lines in the netlist.

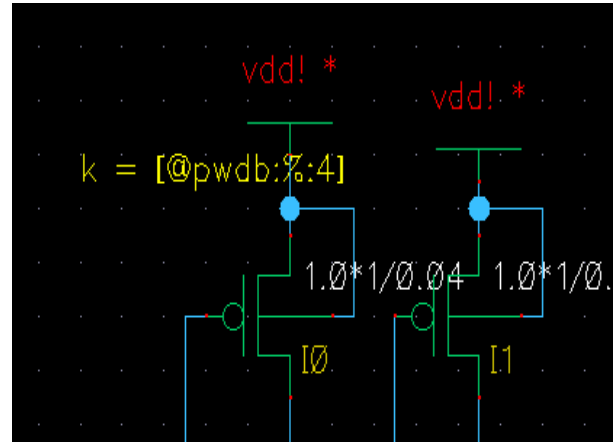
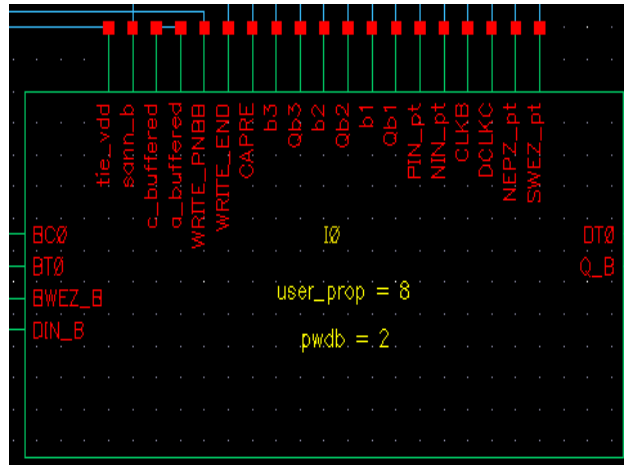
Any user property defined on an instance in the schematic is printed along with its value in the instance definition of that subcircuit in the output CDL netlist file. This is true for instances of both primitive and non-primitive devices.

If an instance has some inherited properties (defined by NLP expressions) from its switch master, CDL Out used to print only those inherited properties. However, in the 5.1.41.500.5.122 and above releases all the user properties, where the value is other than NIL, will be printed on the instance.

Design Data Translators Reference

Design Translation Using CDL Out

For example, we have an instance I0 with property `user_prop = 8`.



This instance has inherited the `pwdb` property from its switch master. Prior to 5.1.41.500.5.122 release, the property was not printed on the instance, as shown below:

```
XI0 net95 net19 net91 net18 QNR_B_k vss! dclkb_m8 vss! vss! vss! vss!
CAPRE_k +WRITE_PNB_k buffered sann_b_m8 vdd! vdd! vss! net98 net17
net96 net16 +SUEZ_B_k DINNR_B_k io_bot pwdb = 2
```

However, in 5.1.41.500.5.122 and above release all the user properties are printed on the instance as shown below:

```
XI0 net95 net19 net91 net18 QNR_B_k vss! dclkb_m8 vss! vss! vss! vss!
CAPRE_k +WRITE_PNB_k buffered sann_b_m8 vdd! vdd! vss! net98 net17
net96 net16 +SUEZ_B_k DINNR_B_k io_bot user_prop = 8 pwdb = 2
```

Also, if you do not want to print certain properties, you can list the properties in the `cdlOutSkipInstProps` SKILL variable in the `.simrc` file. For example, including the following line in the `.simrc` file does not print the `partName` and `vendorName` properties in the output CDL netlist file.

```
cdlOutSkipInstProps=list("partName" "vendorName")
```

In addition, if you do not want to print any property, you can include the following line in the `.simrc` file.

```
cdlOutSkipInstProps= "All"
```


Parameter-Value Inheritance

You assign an inherited parameter on the element or model line, the same way you would with fixed parameter assignments. However, instead of assigning a fixed value to the parameter, assign the value of another parameter. For example, if the hierarchical netlist represents *m10* as

```
m10 d g s b nmos w = wn...
```

CDL Out sets the width of transistor *m10* equal to the value of the *wn* parameter.

For subcircuit-level cells, place the default value of the *w* parameter's *nlpExpr* property on the *.subckt* definition line for subcircuit-level *m10* cells. If the cell containing *m10* is in the top-level schematic, place the value in a *.param* statement. For example, if the *nlpExpr* property specifies a default value of 5u, and the *m10* cell is part of a subcircuit, write the subcircuit definition line in the hierarchical netlist as follows:

```
.subckt xxx node1 node2 ... wn = 5u ...  
m10 d g s b nmos w = wn
```

If *m10* is in the top-level schematic, place a *.param* line in the hierarchical netlist:

```
.param ... wn = 5u ...
```

Both of these lines assign a default value of 5u to *wn*.

Parameter-Value Inheritance and Schematic Levels

If you want an element to inherit a parameter value from another element, that parameter must come from a higher schematic level.

For example, if you have an *inv* inverter in your schematic that contains transistor *w10*, and you place an instance of that inverter named *inv1*, transistor *w10* in *inv1* inherits the width specified in the property list of the original *inv* inverter. If you want transistor *w10* of inverter *inv1* to have a width of 25u, you must add the following property to the *inv* property list:

```
wn=25u
```

The hierarchical netlist reflects this specification as follows:

```
xinv1 node1 node2 ... inv ... wn = 25u ...
```

The width assignment in the *inv* property list overrides the default value for width on the *.subckt* definition of the *inv1* schematic. You are not limited to making these assignments within individual instances of *inv*. You can make the *wn* assignment at any level above where *inv1* is used because parameter assignments apply down through the hierarchy.

To assign a default value to a parameter that another element inherits, you must add it to the *nlpExpr* of that parameter or it is ignored by CDL Out.

Inheritance of Inherited Parameters

You can assign the value of an inherited parameter to the value of another inherited parameter if you make the assignments at different levels and if the second assignment is at a higher level in the schematic. For example, you cannot assign `w = [@wn:5u]` on the instance representing transistor `m10`, and also assign `wn = 25u` on the same instance, because they are on the same level.

Enhanced Support for Inherited Connections

When CDL Out netlists a design that contains inherited connections, it creates dummy ports (or pseudo ports) to maintain connectivity across modules and their instantiations. These dummy ports introduce unwanted nets in the hierarchy and also result in the loss of data when design information is shared across various EDA tools in the flow.

The switch is a SKILL environment variable named.

You can turn off the creation of the dummy ports by using the *CDL Out* GUI check box option Print Inherited Connections. The corresponding switch in the `si.env` file is `simPrintInhConnAttributes`. It is a boolean variable with the default value `nil`. This means that in the default state CDL Out will continue to create the dummy ports for inherited connections in a design. You can turn off the creation of the dummy ports by setting this variable to `t`. You can set this environment variable in the `.simrc` file.

Design Data Translators Reference

Design Translation Using CDL Out

The two statements to be used in conjunction with the `simPrintInhConnAttributes` variable are `*.NETEXPR` and `$netSet`.

Statement	Syntax	Example
<code>*.NETEXPR</code>	<code>*.NETEXPR <i>prop_name netName default_signal</i></code>	<code>*.NETEXPR vdd vdd! vdd!</code>
<code>\$netSet</code>	<code>\$netSet <i>prop_name</i> = <i>value</i></code>	<code>\$netSet vdd = VDD!</code>

Support for Inheritance and Transfer of Supply Sensitivity Data

CDL Out supports the `SUPPLYSENSITIVITY` constructs that enable you to combine logical and physical data in the flow. The use of these constructs ensures that there is no loss of supply sensitivity data when information is passed across various EDA tools in the design flow from the design entry to the verification stage.

The `SUPPLYSENSITIVITY` constructs include `*.SUPPLYSENSITIVITY` and `*.GROUNDSENSITIVITY`. These constructs have the following syntax:

Syntax of `*.SUPPLYSENSITIVITY`

```
*.SUPPLYSENSITIVITY netName value
```

Syntax of `*.GROUNDSENSITIVITY`

```
*.GROUNDSENSITIVITY netName value
```

Note: These property values are printed in the netlist corresponding to the nets to which these terminals are connected.

Since CDL Out does not distinguish between terminals and the nets that connect to these terminals, CDL Out uses the `*.PINMAP` statement that defines the mapping between a net and the cell terminal to which it is connected.

Syntax of `*.PINMAP`

```
*.PINMAP net1: cell_terminal1 net2: cell_terminal 2...
```

Example

```
*.PINMAP GND1:gnd! VDD1:vdd!
```

How CDL Out Translates Instances of Primitive Devices

This section describes how you must define an instance of a primitive device in the schematic so that CDL Out netlists the instance. CDL Out recognizes instances of only those cells that have both a symbol view and a CDL view.

You can store default values of parameters in the symbol view so that they are displayed in the schematic. In the CDL view, store the default values so that they are printed in the output CDL netlist.

To create CDL views, use the Symbol/Simulation Library Generator (S/SLG) described in the *Virtuoso Schematic Editor User Guide*.

Given below is a complete list of formatting functions (`hnlCDLFormatInst`) and parameter list variables (`hnlCDLParamList`) for all the primitive cells supported by CDL Out:

Component	<code>hnlCDLFormatInst</code>	<code>hnlCDLParamList</code>
BJT	<code>hnlCDLPrintBJTElement</code>	<code>hnlCDLBJTParamList</code>
Capacitor	<code>hnlCDLPrintCapacitorElement</code>	<code>hnlCDLCapacitorParamL</code>
Diode	<code>hnlCDLPrintDiodeElement</code>	<code>hnlCDLDiodeParamList</code>
ICIsrc	<code>hnlCDLPrintICIsrcElement</code>	<code>hnlCDLICIsrcParamList</code>
ICVsrc	<code>hnlCDLPrintICVsrcElement</code>	<code>hnlCDLICVsrcParamList</code>
Inductor	<code>hnlCDLPrintInductorElement</code>	<code>hnlCDLInductorParamList</code>
Isrc	<code>hnlCDLPrintIsrcElement</code>	<code>hnlCDLIsrcParamList</code>
Jfet	<code>hnlCDLPrintJfetElement</code>	<code>hnlCDLJfetParamList</code>
MOSfet	<code>hnlCDLPrintMOSfetElement</code>	<code>hnlCDLMOSfetParamList</code>
NMOSfet	<code>hnlCDLPrintNMOSfetElement</code>	<code>hnlCDLMOSfetParamList</code>
PMOSfet	<code>hnlCDLPrintPMOSfetElement</code>	<code>hnlCDLMOSfetParamList</code>
Resistor	<code>hnlCDLPrintResistorElement</code>	<code>hnlCDLResistorParamList</code>
Tline	<code>hnlCDLPrintTlineElement</code>	<code>hnlCDLTlineParamList</code>
VCIsrc	<code>hnlCDLPrintVCIsrcElement</code>	<code>hnlCDLVCIsrcParamList</code>
VCVsrc	<code>hnlCDLPrintVCVsrcElement</code>	<code>hnlCDLVCVsrcParamList</code>
Vsrc	<code>hnlCDLPrintVsrcElement</code>	<code>hnlCDLVsrcParamList</code>

Design Data Translators Reference

Design Translation Using CDL Out

For more information on SKILL functions mentioned above, see [How to Customize CDL Netlist](#).

hnlCDLFormatInst	Specifies what formatting procedure to call to print the line of that type of element. For example, the value of this property for the nmos components in cdslib is hnlCDLPrintNMOSfetElement: <pre>hnlCDLFormatInst="hnlCDLPrintNMOSfetElement"</pre>
hnlCDLParamList	Specifies which parameters can be inherited. You must set the value of this parameter to the name of a SKILL variable that is itself a list of the parameters that can be inherited. Any parameter not in this list cannot inherit a value and can only be assigned a fixed value. For example, the value of this property for the nmos components in cdslib is hnlCDLMOSfetParamList. <pre>hnlCDLParamList="hnlCDLMOSfetParamList"</pre>
hnlCDLElementSubType	Specifies the element type. For example, the value of this property for the nmos components in cdslib is n. <pre>hnlCDLElementSubType= "n"</pre>

The following example shows part of a file that creates CDL views with values of these three properties for different components.

```
lmDefViewProp(cap cdl
  hnlCDLParamList = "hnlCDLCapParamList"
  hnlCDLFormatInst = "hnlCDLPrintCapElement()"
  hnlCDLElementSubType = "c1"
)

lmDefViewProp(capacitor cdl
  hnlCDLParamList = "hnlCDLCapacitorParamList"
  hnlCDLFormatInst = "hnlCDLPrintCapacitorElement()"
  hnlCDLElementSubType = "c2"
)

lmDefViewProp(diode cdl
  hnlCDLParamList = "hnlCDLDiodeParamList"
  hnlCDLFormatInst = "hnlCDLPrintDiodeElement()"
  hnlCDLElementSubType = "di"
)
```

The complete parameter list that can be present for each component is given in the [CDL Out Formats](#) section.

Note: For a four terminal MOSfet in which the bulk terminal is connected to the global net, use the formatting function corresponding to NMOSfet or PMOSfet and not MOSfet.

How CDL Out Translates Global Signals

CDL Out interprets a signal that ends with a `!` as a global signal. The signal name appears in the `.global` and `.pin` statements. If the global signal is associated with a component, the signal does not appear in the I/O signal list of a subcircuit definition.

Do not use global signals with a hierarchical input connector unless the schematic is the top-level block of the design. If you use the global signal with a hierarchical input connector, the signal appears in the I/O signal list of a subcircuit definition and creates warnings when you run LOGLVS. Assign wire or label connections directly to instances in the schematic (for example, when you use globals with components.)

For `nmos`, `pmos`, and `cap` devices in the Cadence Library, by default a substrate connection to power or ground is made in the netlist. To override the VDD and GND defaults for these connections, include the following declarations in the `.simrc` file:

```
hnlCDLNMOSBulkNetName="cvss!"  
hnlCDLPMOSBulkNetName="cvdd!"  
hnlCDLCAPBulkNetName="dvss!"
```

In this example, `cvss` is the bulk connection for the `nmos` device, `cvdd` is the bulk connection for the `pmos` device, and `dvss` is the bulk connection for the `cap` device. The signals appear as `CVSS!`, `CVDD!`, and `DVSS!` in the netlist.

Just like in the schematic where the global signal names are suffixed with `!`, the global signal names in the output CDL netlist file also end with `!`.

CDL Out Formats

The CDL view for primitive cells must contain `hnlCDLFormatInst`, `hnlCDLParamList` and `hnlCDLElementSubType` in its property list. This section documents the CDL Out formats for the primitive components generated by CDL Out and the `cdslib` components that use them.

BJT Element Format

Used by `npns` and `pnps` component types.

```
QInstanceName C B E [SUB] cellName $EA=@area $L=@l $W=@w {$SUB=@sub} @off ic=@ic  
m=@m
```

BSIM3SOI Element Format

Used by BSIM3SOI component types.

```
M<name> <D> <G> <S> <E> <model> l=@l w=@w ad=@ad as=@as pd=@pd ps=@ps nrs=@nrs  
nrd=@nrd nrb=@nrb @off bjtoff=@bjtoff ic=@ic rtho=@rtho ctho=@ctho debug=@debug  
nbc=@nbc nseg=@nseg pdbc=@pdbc psbcp=@psbcp agbcp=@agbcp aebcp=@aebcp  
vbsusr=@vbsusr tnodeout=@tnodeout
```

Cap Element Format

Used by `cap` component types.

```
CInstanceName Y global_gnd @cm=@m $[cellName] {$SUB=@sub} @ns @tc1 @tc2 @scale @cj  
ic=@ic
```

Capacitor Element Format

Used by `capacitor` and `pcapacitor` component types.

```
CInstanceName PLUS MINUS @c $[cellName] {$SUB=@sub} @ns @tc1 @tc2 @scale @cj  
ic=@ic m=@m {$SUB=@sub}
```

Diode Element Format

Used by `diode` and `pdiode` component types.

```
DInstanceName PLUS MINUS cellName @area PJ=@pj w=@w l=@l wp=@wp lp=@lp wm=@wm  
@off ic=@ic {@area} {@periphery} {$SUB=@sub}
```

Design Data Translators Reference

Design Translation Using CDL Out

Inductor Element Format

Used by `inductor` component types.

```
LInstanceName PLUS MINUS @l @tc1 @tc2 @nt ic=@ic
```

JFET Element Format

Used by `njfet` and `pjfet` component types.

```
JInstanceName D G S cellName w=@w l=@l @off ic=@ic m=@m
```

MOSFET Element Format

Used by `sdalib ndepl`, `nfet`, `nsfntn`, `pdepl`, `pfet`, and `psfntn` component types.

```
MInstanceName D G S B cellName w=@w l=@l ad=@ad as=@as pd=@pd ps=@ps nrd=@nrd  
nrs=@nrs @off ic=@ic m=@m $LDD[@LDD] {$NONSWAP}
```

NMOSFET Element Format

Used by `nxfr`, `Nmos`, `nmosd`, and `nmose` component types.

```
MInstanceName D G S global gnd, cellName w=@w l=@l ad=@ad as=@as pd=@pd ps=@ps  
nrd=@nrd nrs=@nrs @off ic=@ic m=@m $LDD[@LDD]
```

NPN Element Format

Used by `npn` component types.

```
QInstanceName C B E cellName M=@m $EA=@area
```

PMOSFET Element Format

Used by `pxfr`, `Pmos`, `pmosd`, and `pmose` component types.

```
MInstanceName D G S global_Vdd cellName w=@w l=@l ad=@ad as=@as pd=@pd ps=@ps  
nrd=@nrd nrs=@nrs @off ic=@ic m=@m $LDD[@LDD]
```

PNP Element Format

Used by `pnp` component types.

```
QInstanceName C B E cellName M=@m $EA=@area
```


Design Data Translators Reference

Design Translation Using CDL Out

Res Element Format

Used by `res` component types.

```
R<InstanceName> A Y @ns | @r $[cellName] $SUB=@sub $w=@w $l=@l @ns @tc1 @tc2 @scale  
@rsh ac=@ac m=@m
```

Resistor Element Format

Used by `resistor` component types.

```
R<InstanceName> PLUS MINUS @ns | @r $[cellName] m=@m {$SUB=@sub} $w=@w $l=@l @ns @tc1  
@tc2 @scale @rsh ac=@ac {$SUB=@sub}
```

Transmission Line Element Format

Used by `tline` component types.

```
T<InstanceName> N1 N2 N3 N4 z0=@z0 td=@td f=@f nl=@nl ic=@ic
```

Voltage Source Element Format

Used by `Vsrc` element.

```
V<InstanceName> N+ N- @DCValue @TRANValue @ACMag @ACPhase
```

Voltage Controlled Current Source Element Format

Used by `VCIsrc` element.

```
G<InstanceName> N+ N- NC+ NC- @value
```

Voltage Controlled Voltage Source Element Format

Used by `VCVsrc` element.

```
E<InstanceName> N+ N- NC+ NC- @value
```

Current Controlled Current Source Element Format

Used by `ICIsrc` element.

```
F<InstanceName> N+ N- V<controlNum> @value
```

Specify the `controlNum` property at the instance or the `cdl` view of the master. The instance property takes higher precedence.

Current Controlled Voltage Source Element Format

Used by `ICVsrc` element.

`H<InstanceName> N+ N- V<controlNum> @value`

Specify the `controlNum` property at the instance or the `cdl` view of the master. The instance property takes higher precedence.

Current Source Element Format

Used by `Isrc` element.

`I<InstanceName> N+ N- @DCValue @TRANValue @ACMag @ACPhase`

CDL Out Output Files

Examples earlier in this chapter show what the output Design Framework II file looks like after translation. Depending on what output files you specified in the *Virtuoso*® *CDL Out* form or template file, CDL Out produces one or more of the following files. CDL Out writes all messages to the `si.log` file.

- Error messages, preceded by “*Error*”, indicate serious, unrecoverable conditions. The converted file is incorrect. For example, if a cell instance is called but its master cell does not exist, CDL Out writes an error message to the `si.log` file.
- Warning messages preceded by “*Warning*” indicate unexpected but recoverable conditions. The converted file is usable. For example, if a string is too long and has been truncated, CDL Out writes a warning message to the `si.log` file.
- Information messages, preceded by “*Info*”, indicate the status of a process that is running or the results of a completed process. Information messages are written to the file `si.log`.
- Statistical messages that CDL Out generates as it runs are also written to the `si.log` file. You can use these messages to estimate the size of the file based on information such as the number of cells and terminals.

How to Customize CDL Netlist

CDL Out defines various `hnl` functions and variables that control the CDL Out netlist format. If these functions and variables are already defined, CDL Out does not overwrite them. You can override the value of variables or functions used in CDL Out by defining them before invoking CDL Out. This can be done by loading a SKILL file before running CDL Out or by defining the variables in `.simrc` or `.cdsinit` file. To see details about the various variables and functions that can be overridden, refer to [*Chapter 5, Open Simulation System Reference*](#).

Example - Customizing CDL Netlist

The variable `hnlCommentStr` defines the string that is placed at the beginning of a comment in the output netlist file. By default, '*' is used by CDL Out to represent the comments. If you put the value of `hnlCommentStr` variable to '>' in the `.simrc` file, then the comments start with '>'.

For information about CDL Out SKILL functions, see the [*Design Data Translator's SKILL Reference*](#).

Design Data Translators Reference

Design Translation Using CDL Out

Design Translation Using LEF/DEF Translator

This chapter contains the following sections:

- Overview
- Methods of Design Translation
 - Running LEF In and LEF Out from Command Line
 - LEF In Command Line Syntax
 - LEF Out Command Line Syntax
 - Running DEF In and DEF Out from Command Line
 - DEF In Command Line Syntax
 - DEF Out Command Line Syntax
 - Running LEF In and LEF Out from GUI
 - File – Import – LEF
 - LEF In Form
 - File – Export – LEF
 - LEF Out Form
 - Running DEF In and DEF Out from GUI
 - File – Import – DEF
 - DEF In Form
 - File – Export – DEF
 - DEF Out Form
- Preparing a Template File

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

- Setting the Environment Variables in LEF In and LEF Out
- Setting the Environment Variables in DEF In and DEF Out
- Viewing Native OpenAccess Libraries in the Virtuoso Studio Design Environment

Overview

Library Exchange Format (LEF) defines the elements of IC process technology and the associated library of cell models. Design Exchange Format (DEF) defines the elements of an IC design relevant to physical layout, including the netlist and design constraints. The LEF and DEF files allow the exchange of placement and routing information with various databases by using the LEF/DEF translators. For more information on LEF and DEF, see [LEF/DEF Language Reference](#).

The placement and routing tools on the OpenAccess database read and write LEF and DEF files by using the core OpenAccess LEF/DEF translator. The CIC customized OpenAccess LEF/DEF translator has been written as a Virtuoso Studio design environment-specific application on these translators.

The extended functionality of the CIC customized OpenAccess LEF/DEF translator includes the creation of `display.drf` file, creation of layer-purpose pairs (LPPs), creation of pin labels, and so on.

The process of reading and writing the LEF and DEF files into OpenAccess is enabled by four executables: `lefin`, `lefout`, `defin`, and `defout`. These executables are described below.

Name	Description
LEF In	Translates a LEF file to an OpenAccess technology library and macro information in the OpenAccess database.
LEF Out	Translates an OpenAccess technology library and macro information to a LEF file.
DEF In	Reads a DEF file and creates an OpenAccess cellview.
DEF Out	Writes a DEF file from an OpenAccess cellview. However, Def Out is not designed to work with schematic database/views.

This chapter describes the methods by which you can exchange design information between the LEF/DEF files and the OpenAccess database using the CIC customized OpenAccess LEF/DEF translators.

Methods of Design Translation

You can translate designs by using the LEF/DEF translators either from the command line or from the Graphical User Interface (GUI).

This section is divided into two parts. The first part describes the command line syntax using which you can run LEF In, LEF Out, DEF In, and DEF Out. The second part describes the GUI options using which you import and export LEF and DEF files.

Running LEF In and LEF Out from Command Line

You can import and export LEF files in the batch mode by running the `lefin` and `lefout` commands. Use the following syntax to run LEF In and LEF Out:

LEF In Command Line Syntax

```
lefin {-h | -lef lefFileNames
      -lib libName
      [-libPath libPath]
      [-tech techLibName]
      [-techPath techLibPath]
      [-techRefs libList]
      [-overwrite]
      [-shared]
      [-view viewName]
      [-log logfileName]
      [-layerMap layerMapFiles]
      [-commentChar char]
      [-templateFile templatefileName]
      [-pinPurp pinPurposeName]}
      [-textHeight height]
      [-textLayer layerName]
      [-compress [-compressLevel compressionLevel]]
      [-mapConflicts]
      [-pnrLibDataOnly]
      [-useFoundryInnovus]
      [-keepOriginalDisplayDrf]
      [-verilogStubFile verilogStubFiles]
      [-createFixedViaDefs]
      [-lockColors]
      [-suffix suffix ]
      [-antennaDiffAreaPropValueIsWidthAndRatio]
}
```


Design Data Translators Reference

Design Translation Using LEF/DEF Translator

where,

<code>-h</code>	Prints help message
<code>-lef <i>lefFileName</i></code>	Input names of LEF files, seperated by space. For example, <code>lefin -lef "1.lef 2.lef"</code>
<code>-lib <i>libName</i></code>	Output library name. If the output library does not exist, a new library is created in the current directory and contains the translated technology database and macros. If the output library exists, it must contain a technology database or refer to the technology database of another library. In addition, any existing macros, if redefined in the LEF file, are overwritten and new cells are created for the new macros.
<code>-libPath <i>libPath</i></code>	Complete path where library is to be created.
<code>-tech <i>techLibName</i></code>	Name of the technology library that is to be created.
<code>-techPath <i>techLibPath</i></code>	Complete path where the technology library is to be created.
<code>-techRefs <i>libList</i></code>	Lists the parent technology databases that are referenced in the incremental technology database.
<code>-overwrite</code>	Overwrite existing designs.
<code>-shared</code>	Allow other applications to access the output library while the current application is running.
<code>-view <i>viewName</i></code>	Output view name for the translated macros. The default view name is <code>abstract</code> .
<code>-log <i>logfileName</i></code>	

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

Specifies the filename where the log messages would be stored. By default, the log messages are printed at `stderr`.

`-layerMap layerMapFile(s)`

Specifies the layer map file. You can specify multiple layer map file names separated by spaces during `lef in`. While creating layers, the layer numbers specified in these files are used.

For example, `lefin -lef in.lef -layerMap "layerMap1 layerMap2" -lib LIB`

A layer map file contains a list of LEF layer names that maps to the corresponding OpenAccess layer numbers. The LEF layer names and OpenAccess layer numbers must be separated by a space.

A sample layer map file is shown below:

#LEFLayerName	OpenAccessLayerNumber
POLYS	0
CUT01	1
RX	2
CUT12	3
M1	4

`-commentChar char`

Specifies the comment character.

`-templateFile templateFileName`

A file that contains command options. The template file contains the list of LEF options and values separated by a space.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

A sample lefin template file is shown below:

```
#OptionName OptionValue
lef ./new.lef
lib macrolibt
libPath ./MACRO_T
tech techlibt
techPath ./TECH_T
view abstract
log lefin_t.log
pinPurp gui
layerMap ./layermap
commentChar #
overwrite
shared
```

`-pinPurp pinPurpose`

Pin purpose to draw the pin geometries.
Default is `drawing`.

`-textHeight height`

Defines the text height to be used for pin labels. The height is specified in database units (DBU). If pin label text height is not specified, DBU value in the technology database for maskLayout views is considered as the default pin label text height.

`-textLayer layerName`

Defines the text layer name to be used for pin labels.

`-compress [-compressLevel compressionLevel]`

Allows libraries to be compressed and defines the compression level. Default compression level is 1.

`-mapConflicts`

Maps conflicts between the incremental and base technologies to constraints and constraint parameter definitions.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

<code>-pnrLibDataOnly</code>	Filters the Place and Route information. The information can be stored only as incremental technology database on the top of existing referenced technology database. Therefore, the <code>-techRefs</code> option needs to be specified. For more information, see General Processing Rules of -pnrLibDataOnly. This option cannot be specified along with <code>-useFoundryInnovus</code> option. Note: The <code>-pnrLibDataOnly</code> option works for 20 nm and larger designs. Cadence recommends using the <code>-useFoundryInnovus</code> option, instead.
<code>-useFoundryInnovus</code>	Enables Innovus to see the same design rules that are described in LEF without impacting the Virtuoso tools. For more information, see General Processing Rules of useFoundryInnovus. This option cannot be specified along with <code>-pnrLibDataOnly</code> option.
<code>-keepOriginalDisplayDrf</code>	Retains the original <code>display.drf</code> file.
<code>-verilogStubFile <i>verilogStubFile</i></code>	Specifies list of input Verilog files. The specified Verilog files are sent to the <code>verilogAnnotate</code> utility as a post process to set the <code>isInterface</code> property and specify the terminal order of the bus terminals. An additional log file <code>verilogAnnotate.log</code> is also generated.
<code>-createFixedViaDefs</code>	Allows creation of fixed via definitions.
<code>-t_lockColors</code>	Locks color data of all shapes created using the PORT and PIN definitions and are associated with a MASK construct.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

`-suffix suffix`

Appends the specified suffix to the names of the standard constraint groups, LEFDefaultRouteSpec, LEFSpecialRouteSpec, and foundry_innovus. These suffixed constraint groups are then used during LEF In translation, instead of the standard constraint groups.

Note: When you use this option, you can import the LEF file into different constraint group of existing technology without using `-techRefs` option.

`-antennaDiffAreaPropValueIsWidthAndRatio`

Translates values specified for the LEF MACRO pin property LEF58_ANTENNADIFFAREA as width and ratio.

- `lefin` creates layer-purpose pairs and pin labels on pins. In addition, it creates the `display.drf` file that consists of packet display information. If `display.drf` already exists in the current library, then a copy of the existing `display.drf` is saved as `display.drf.orig` and new packet display information is appended to the existing `display.drf` file in Virtuoso Studio design environment on OpenAccess 2.2. The `display.drf.orig` is created once and is never overwritten.

In OpenAccess 2.0, if `display.drf` already existed in the current library, a new file, `display.drf.lefin` was created to store the packet display information. You had to manually merge the files if required.

- You can specify the `-cdslib` option to access the `cds.lib` file from any other directory.

`-lefin -cdslib <cds.lib_file_path>`

- `lefin` creates new layer-purpose pairs in the technology library and display packets to support the display of blockages and track patterns. For each layer defined in the `lef` file, `lefin` will create layer-purpose pairs only for the following purposes: `drawing`, `fill`, `slot`, `gapfill`, `fillOPC`, `blockage`, `track`, `boundary`, `net`, and `pin`. For example, if a layer L1 is defined in the `lef` file, following LPP's will be created in technology:

L1 `drawing`

L1 `fill`

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

L1 slot
L1 gapFill
L1 fillOPC
L1 blockage
L1 track
L1 boundary
L1 pin
L1 net

This information is available in the `techLayerPurposePriorities` section of the `layerDefinitions` section in the technology library. `Lefin` also creates display packets for all these LPP's in the technology library. This is available in the `techDisplays` section of `layerDefinitions` section of the technology library. In addition, `lefin` creates a `display.drf` file, which contains the complete information.

Note: LPPs for track purpose will not be created for non routing layers.

- During LEF In, via layers in the technology file now get populated.

Note: For a library that does not have a tech database, the attached technology data is used. In native OpenAccess, the `tech.db` in the library overrides an attached technology library. In Virtuoso Studio design environment on OpenAccess 2.2, however, the attached technology library overrides the `tech.db` in the library and `lefout` uses the technology information from the attached technology data.

- When any workbench is run, if `display.drf` does not exist in the directory from which it is run, the default `display.drf` file is loaded. During LEF In, if `display.drf` does not exist in the directory from which it is run, a new `display.drf` is created. On subsequent LEF In runs, the new information is added to the existing `display.drf`. When you exit a session, you are prompted to save the layer display information. On saving it, some extra sections, such as `drDefineColor` and `drDefineStipple`, are added to the `display.drf` file. If you do not save `display.drf`, the extra sections `drDefineColor` and `drDefineStipple`, are not added to the file.

Note: The `-appendTech` option has been deleted. You can now import LEF files incrementally. Doing this does not change the definition of existing technology data, but if new data is defined in the LEF file, it is appended to the technology library.

- LEF in throws an error and skips the MACRO translation when macro name and foreign name are different and view specified is `layout` to avoid conflict. To workaround this issue, use same FOREIGN name or specify view as `abstract`.

General Processing Rules of -pnrLibDataOnly

Data processing rules for LEF LAYER statements:

- If the layer already exists in referenced technology database:
 - Data related to `LEFDefaultRouteSpec` constraints group constraints such as `WIDTH`, `LENGTH`, and `PITCH` are stored.
 - Data related to `Foundry` constraint group, such as `MINWIDTH` and `MINSTEP` are ignored.
- If the layer does not exist in the referenced technology database, new layer is not created and all information is skipped.

Processing of LEF VIAS, MACROS, VIAS, and VIARULES:

- If the object with the same name does not exist in the referenced technology database, it is created in the referenced technology database.
- If the object with the same name exists in referenced technology database, it cannot be modified and all LEF information is skipped.

Note: The `-pnrLibDataOnly` option works for 20 nm and larger designs. Cadence recommends using the `-useFoundryInnovus` option, instead.

General Processing Rules of useFoundryInnovus

Virtuoso and Innovus treat LEF files data differently but the `-useFoundryInnovus` option allows merging the Virtuoso and Innovus data. Here, Innovus-specific LEF file can be placed on the top of Virtuoso-specific referenced technology database. Most of the incompatibilities relate to the treatment of the `Foundry` group constraints. Therefore, the translator creates special constraint group `foundry_innovus` and uses it instead of `Foundry` constraints group to store the Innovus-specific information. Innovus also uses Innovus-specific layers, so unlike the `-pnrLibDataOnly` option, this option allows creation of new layers and modification of some existing layers properties. However, all changes are stored in the incremental technology database.

General data processing rules for LEF LAYER statement:

- Data related to `LEFDefaultRouteSpec` constraint group constraints such as `WIDTH`, `LENGTH`, and `PITCH` is placed in the `LEFDefaultRouteSpec` constraints group.
- Data related to `Foundry` constraint group constraints such as `MINWIDTH` and `MINSTEP` is placed in the `foundry_innovus` constraint group.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

- LEF58_PROPERTY type properties information is stored in the `foundry_innovus` constraint group `layerProp` constraints values.
- New REGION layers are ignored and info messages are displayed.
- Layer-type information is stored in the `foundry_innovus` constraint group in the `lefType` constraint value.
- Order of the LEF layers is preserved and is stored in the `foundry_innovus` constraint group in the `lefLayerOrder` constraint value.

Processing of LEF VIAS, MACROS, VIAS, and VIARULES:

- If the object with the same name does not exist in the referenced technology database, it is created in referenced technology database.

If the object with the same name exists in referenced technology database, it cannot be modified and all LEF information is skipped.

Use of Suffix with useFoundryInnovus or pnrLibDataOnly

When using the `useFoundryInnovus` or `pnrLibDataOnly` options, you can add the information to the existing library by adding a suffix to the file names.

Example

Consider that:

- Base pdk is `base_pdk`
- Two technology LEF files compatible with the `base_pdk` are `techLef1.lef` and `techLef2.lef`

You can create a multi-technology rapid MSOA PDK in the following way:

```
lefin -lef techLef1.lef -suffix _techLef1 -lib rmsoa_pdk -techRefs base_pdk -
useFoundryInnovus
lefin -lef techLef2.lef -suffix _techLef2 -lib rmsoa_pdk -useFoundryInnovus
```

The first `lefin` command creates the `rmsoa_pdk` library and adds the information from `techLef1`. The second `lefin` command adds the information from `techLef2`. The suffix prevents collisions between the two technology files.

The suffix affects the naming of the constraint group `foundry_innovus` (and any member constraint groups created with it), `LEFDefaultRouteSpec`, and `LEFSpecialRouteSpec`.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

LEF Out Command Line Syntax

```
lefout {-h | -lef fileName
        -lib libName
        [-cells cellName [cellName...]]
        [-cellList cellListFile]
        [-views viewName [viewName...]]
        [-log logfileName]
        [-noTech]
        [-ver <5.4|5.5|5.6|5.7|5.8|6.0>]
        [-techOnly]
        [-useFoundryInnovus]
        [-templateFile templateFileName]
        [-lockedColorsOnly]
        [-suffix suffix ]
        [-antennaDiffAreaPropValueIsWidthAndRatio]
}
```

where,

-h	Prints help message
-lef <i>fileName</i>	Output LEF file name
-lib <i>libName</i>	Input library name
-cells <i>cellName</i> [<i>cellName</i> ...]	Input cell name to be translated as macros. There can be more than one cell name.
-cellList <i>cellListFile</i>	Specifies a file containing a list of cell names that are to be translated as macros. The cell list file contains the list of library names and cell names separated by a space. In a cell list file for Virtuoso Studio design environment on OpenAccess 2.2 you do not need to specify the view name.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

The cell names in a cell list file appear in the following format:

`<libName> <cellName>`

A sample cell list file is shown below:

#libName	cellName
designLib1	ACCSHCINX2
designLib1	ACCSHCINX4
designLib1	ACCSHCONX2
designLib1	ACCSHCONX4

Note: If you use both the `-cells` and the `-cellList` options, the `-cells` option is ignored and a warning message is displayed.

`-views viewName [viewName...]`

View name for the input cell name. There can be more than one view name and input cell name.

The default is `abstract`. When you specify other view names, ensure that the content of these views is the same as the abstract views.

`-log logfileName`

Specifies the filename where the log messages are to be stored. By default, the log messages are printed at `stderr`.

`-noTech`

Does not output the technology information.

`-ver <5.4|5.5|5.6|5.7|5.8|6.0>`

LEF version (Default version: 5.8)

`-techOnly`

Outputs the technology information only.

`-useFoundryInnovus`

Outputs the `foundry_innovus` group constraint as a LEF LAYER LEF58 property.

`-templateFile templateFileName`

File that contains the command options.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

`-lockedColorsOnly`

Outputs colors for only those shapes that have their locked state set to true.

This includes pathSegs in nets, rects and polygons, and fill shapes and vias in specialnets.

`-suffix suffix`

Appends the specified suffix to the names of the standard constraint groups, LEFDefaultRouteSpec, LEFSpecialRouteSpec, and foundry_innovus when searching for the standard constraint groups.

`-antennaDiffAreaPropValueIsWidthAndRatio`

Creates the MACRO pin property LEF58_ANTENNADIFFAREA from the internal OpenAccess representation.

- The `-lib` option outputs the information, which is present in the technology file in the LEF format. To do so, it follows certain set of rules:
 - Any library will give precedence to its own technology.
 - All libraries present in the cell list file must have the same technology as provided with the `-lib` option.
- You can specify the `-cdslib` option to access the `cds.lib` file from any other directory.

`-lefout -cdslib <cds.lib_file_path>`



Caution

LEF Out in OA2.2 does not print a layer whose type it cannot recognize. All unknown layer types, such as layers with type `li` are ignored.

Running DEF In and DEF Out from Command Line

You can import and export DEF files in the batch mode by running the `defin` and `defout` commands. The syntax used to run DEF In and DEF Out is shown below:

DEF In Command Line Syntax

```
defin {-h | -def defFileName
      -lib libName
      [ -libpath libPath ]
      [ -cell cellName ]
      [ -view viewName ]
      [ -masterLibs libList ]
      [ -viewNameList viewNamesForComponents ]
      [ -tech techLibName ]
      [ -techRefs libList ]
      [ -log logfileName ]
      [ -useCustomVias ]
      [ -overwrite ]
      [ -createModHier ]
      [ -commentChar char ]
      [ -shared ]
      [ -noRouting ]
      [ -templateFile templateFileName ]
      [ -pinPurp pinPurpose ]
      [ -layerMap layerMapFile ]
      [ -userSkillFile skillFile]
      [ -compress [-compressLevel compressionLevel ]
      [ -ignoreDrcFill ]
      [ -lockColors ]
      [ -oaMapFile mapfile ]
      [ -alternateFoundry constraintGroupName ]
      }
```

where,

<code>-h</code>	Prints help message
<code>-def <i>defFileName</i></code>	Input DEF file name

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

<code>-lib <i>libName</i></code>	Output library name. If the output library does not exist, a new library is created in the current directory. In addition, the <code>-tech</code> option must be used to specify a name for the technology library. If the output library exists, it must contain a technology database, or refer to the technology database of another library.
<code>-libPath <i>libPath</i></code>	Complete path where a library is created.
<code>-cell <i>cellName</i></code>	Output cell name
<code>-view <i>viewName</i></code>	Output view name
<code>-masterLibs <i>libList</i></code>	List of reference libraries that contain master cells. You can specify multiple reference library names separated either by a comma, space, or a tab. The reference libraries are searched for the master cells in the order in which you list the libraries.
<code>-viewNameList <i>viewNamesForComponents</i></code>	Specifies view names for the master cells that are searched for a DEF COMPONENTS construct record. The view names must be separated by a comma, space, or a tab. The view names are searched in the order you specify. For example, if the <i>Component View List</i> field contains <code>layout abstract abstractNew</code> as the view names, then <code>layout</code> gets first priority.
<code>-tech <i>techLibName</i></code>	Technology library that is to be attached to the output library. This option is useful only if the output library does not exist.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

`-techRefs libList`

Lists the parent technology databases that are referenced in the incremental technology database.

`-log logfileName`

Specifies the filename where the log messages are to be stored. By default, the log messages are printed at `stderr`.

`-useCustomVias`

Creates custom vias for all DEF vias

`-overwrite`

Overwrite existing designs.

`-createModHier`

Creates module hierarchy from hierarchical names. Using this option can decrease DEF In speed.

`-commentChar char`

Specifies the comment character.

`-shared`

Allows other applications to access the output library while the application is running.

`-norouting`

Does not create any routing data.

`-templateFile templateFileName`

File that contains the command options.

`-pinPurp pinPurpose`

Pin Purpose to draw the pin geometries. Default is Drawing.

`-layerMap layerMapFile`

Specifies the layer map file. You can specify multiple layer map file names separated by spaces during def in. While creating layers, the layer numbers specified in these files are used.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

For example, `lefin -lef in.lef -layerMap "layerMap1 layerMap2" -lib LIB`

A layer map file contains a list of DEF layer names that maps to the corresponding OpenAccess layer numbers. The DEF layer names and OpenAccess layer numbers must be separated by a space.

A sample layer map file is shown below:

#DEFLayerName	OpenAccessLayerNumber
POLYS	0
CUT01	1
RX	2
CUT12	3
M1	4

`-techPath techLibPath`

Complete path where the technology library is to be created.

`-compress [-compressLevel compressionLevel]`

Allows libraries to be compressed and defines the compression level. Default compression level is 1.

`-userSkillFile skillFile`

Specifies a file that consists of user-defined SKILL routines.

`-ignoreDrcFill`

Maps all shapes specified with the +SHAPE DRCFILL tag in the SPECIALNETS section to the drawing purpose. If this option is not specified, such shapes are mapped to the gapFill purpose.

`-lockColors`

Locks color data of all shapes created using the PORT and PIN definitions and are associated with a MASK construct.

`-oaMapFile mapfile`

Uses the specified mapping file to post-process the design before it is saved.

For more information, see [Using the oaMapFile option for different OpenAccess Database Formats](#).

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

`-alternateFoundry constraintGroupName`

Uses the specified constraint group instead of the default foundry constraint group. Constraint group specified using this option, overwrites the name specified in the .cdsenv file using the AlternateFoundryCG environment variable.

During `lefin`, the `layerMap` option creates a layer along with the corresponding layer number in the technology file. The layer entry created in the technology file maps with the entry in the layer map file. Unlike `lefin`, `defin` does not create a layer in the technology file. It only searches for the layer in the technology file using the layer number corresponding to the layer name in the layer map file.

Note: You can specify the `-cdslib` option to access the `cds.lib` file from any other directory.

`-defin -cdslib <cds.lib_file_path>`

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

DEF Out Command Line Syntax

```
defout {-h | [-def] defFileName
      -lib libName
      -cell cellName
      -view viewName
      [ -log logfileName ]
      [ -skipNoneOrViaCellType ]
      [ -mapDividerChar mapChar ]
      [ -dividerChar dividerChar ]
      [ -busbitChars busbitChars ]
      [ -userSkillFile skillFile ]
      [ -ver <5.4|5.5|5.6|5.7|5.8|6.0> ]
      [ -templateFile templateFileName ]
      [ -outputFloatingShapes ]
      [ -warnOnNotPlacedInsts ]
      [ -skipPhysOnlyInsts ]
      [ -gdsCompatible ]
      [ -noDefOnError ]
      [ -lockedColorsOnly ]
      [ -errorOnGrayShapes ]
      [ -errorOnUnlockedShapes ]
      [ -skipTrimShapes ]
      [ -skipTrimProductShapes ]
      [ -skipTrimmedShapes ]
      [ -outputTrimmetalGapFill ]
      [ -outputTrimSegsAsNets ]
      [ -maskShiftLayers layerNames ]
      [ -oaMapFile mapfile ]
}
```

where,

All features of DEF version 5.7 are not supported in the 6.1.2 release.

-h	Prints help message
-def <i>defFileName</i>	Output DEF file name
-lib <i>libName</i>	Input library name
-cell <i>cellName</i>	Input cell name
-view <i>viewName</i>	Input view name
-log <i>logfileName</i>	Specifies the filename where the log messages are to be stored. By default, the log messages are printed at <code>stderr</code> .
-skipNoneOrViaCellType	Specifies that the cells of type 'none' or 'via' are not written in the output def file.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

<code>-mapDividerChar <i>mapChar</i></code>	Specifies a single character to be replaced in instance and net names by the <i>DEF</i> <i>DIVIDERCHAR</i> .
<code>-dividerChar <i>dividerChar</i></code>	Specifies a new divider character to be used during output.
<code>-busbitChars <i>busbitChars</i></code>	Specifies new busbit character pair to be used during output.
<code>-userSkillFile <i>skillFile</i></code>	Specifies a file that consists of user-defined SKILL routines.
<code>[-ver <5.4 5.5 5.6 5.7 5.8 6.0>]</code>	DEF version (Default version: 5.8)
<code>-templateFile <i>templateFileName</i></code>	File that contains the command options.
<code>-outputFloatingShapes</code>	Outputs floating shapes into the SPECIALNETS section. Floating shapes are shapes that do not have any connectivity information. All shapes on <i>tsvMetal</i> and <i>padMetal</i> layers that do not have any connectivity are also considered as floating shapes. When specified during DEF OUT translation, any drawing shape on a layer, which is not a member of a net is exported on a net in the SPECIALNETS section named <i>_FLOATING_DRAWING_SHAPES_RESERVED</i>.
<code>-warnOnNotPlhacedInsts</code>	Adds warnings in the log file for the instances in the DEF Out output that have no placement status.
<code>-skipPhysOnlyInsts</code>	Prevents output of physical instances. If <code>-skipPhysOnlyInsts</code> is not specified, physical instances are placed in the DEF COMPONENTS section and have the <code>+ SOURCE DIST</code> statement added to them.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

<code>-gdsCompatible</code>	Generates a DEF file with limits that make it compatible with GDS.
<code>-noDefOnError</code>	Prevents the DEF output file from being generated if there are errors during translation.
<code>-lockedColorsOnly</code>	Outputs colors for only those shapes that have their locked state set to true. This includes pathSegs in nets, rects and polygons, and fill shapes and vias in specialnets.
<code>-errorOnGrayShapes</code>	Displays errors if gray or uncolored shapes or vias are found during translation. Note: When the technology library contains <u>preColoredLayers</u> constraint in the VirtuosoMPTSetup constraint group, the layer-purpose pairs (LPPs), or, the layers list, or both from the first <u>preColoredLayers</u> constraint are not considered for error checking. Additionally, <u>colorless layers and colorless LPPs</u> are not considered for error checking. (Virtuoso Advanced Node for Layout Only)
<code>-errorOnUnlockedShapes</code>	Displays errors if any colored shapes or vias with unlocked status are found during translation. Note: When the technology library contains <u>preColoredLayers</u> constraint in the VirtuosoMPTSetup constraint group, the LPPs, or the layers list, or both from the first <u>preColoredLayers</u> constraint are not considered for error checking. Additionally, <u>colorless layers and colorless LPPs</u> are not considered for error checking. (Virtuoso Advanced Node for Layout Only)
<code>-skipTrimShapes</code>	Prevents output of trim layers shapes into the FILLS section.
<code>-skipTrimProductShapes</code>	Prevents output of trim product shapes into the SPECIALNETS section.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

<code>-skipTrimmedShapes</code>	Prevents output of trim layers shapes into the SPECIALNETS section. Note: This option works only when the <code>-outputFloatingShapes</code> option is specified.
<code>-outputTrimmetalGapFill</code>	Output the trim gap fill shapes (bridge metal shapes) into the SPECIALNETS section.
<code>-outputTrimSegsAsNets</code>	Output trim product shapes into NETS section. The following options cannot not be specified together: <code>-skipTrimProductShapes</code> <code>-outputTrimSegsAsNets</code>
<code>-maskShiftLayers <i>layerNames</i></code>	Specifies list of layers used in COMPONENTMASKSHIFT statement.
<code>-oaMapFile <i>mapfile</i></code>	Uses the specified mapping file to pre-process the design before it is exported to DEF. For more information, see Using the oaMapFile option for different OpenAccess Database Formats .

Note: You can specify the `-cdslib` option to access the `cds.lib` file from any other directory.

`-defout -cdslib <cds.lib_file_path>`

Using the oaMapFile option for different OpenAccess Database Formats

Some technologies use alternative OpenAccess database formats that differ from the conventional OpenAccess formats because they use non-standard OpenAccess layers, purposes, and color masks to store technology components properties.

The DEF In and DEF Out translators are designed to work only with conventional OpenAccess databases and cannot be customized to work with alternative representations. You can use the DEF translator for alternative technologies only after converting such databases into the conventional format.

The `oaMapFile` option controls the conversion between alternative and conventional OpenAccess database formats. When this option is specified:

- The DEF In translator creates the OpenAccess database in conventional format for translation.
- After the translation is complete, the DEF Out translator converts the conventional database to an alternative database and then generates the DEF output file.
- The changes in the database format are saved in memory only. No updates are made to the source OpenAccess alternative format database after translation.
- Properties for this conversion are defined in a configuration text file called `map`.

Note: This option only supports modifications of the `oaShape` properties.

The conversion operation searches for `oaShape` objects with specific properties, such as assigned layers, purposes, or color masks and changes these properties according to the configuration stored in the configuration map file.

Format of the OpenAccess mapping configuration file

The OpenAccess mapping configuration file is in text format and can contain the following types of lines:

- **Title line.** This is a mandatory line and it containing information regarding the file type and version as shown below:

```
## OA2OA_CONFIG <config_version>
```

When DEF translation reads the configuration file, it controls its type and version. Translation cannot continue if the mapping file version is newer than the version supported by the translator.

- **Empty lines.**

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

- **Comments**, which start with a #.
- **Commands**, which start with a mapping file command name. These commands are in upper case. Currently, commands `CHANGE_SHAPE_LPP` and `CREATE_PURPOSE` are supported.

Mapping file commands

`CHANGE_SHAPE_LPP fromLayer fromPurpose fromColorMask TO toLayer toPurpose toColorMask`

This command transforms all shapes on the layer *fromLayer* with purpose *fromPurpose* and color mask *fromColorMask* to layer *toLayer* and sets the purpose to *toPurpose* purpose, and the color mask *toColorMask*.

In mapping file commands:

- The mapping format is reversible. Direct mapping is used during DEF In translation and is reversed during DEF Out translation. In reversed transformation, information in the *from* argument is treated as the *to* argument and the *to* argument is treated as the *from* argument.
- You can use the wildcard * instead of specifying all layer, purpose, and color mask names to be mapped. The mapping statement depends on where the wildcard is used:
 - ❑ You can use multiple `CHANGE_SHAPE_LPP` lines in the configuration map file. The mapping commands are run sequentially, which in some configurations can result in several transformations. This is why, if `CHANGE_SHAPE_LPP` uses the * wildcard for a property in the *from* argument of the statement, it also has to have * for the same property in the *to* argument of the statement. Otherwise, the information will be lost during transformation and the mapping cannot be used for format conversion operations.
 - ❑ If the *from* argument is replaced with *, it specifies that all applicable layers, purposes, and color masks are mapped.
 - ❑ If the *to* argument is replaced with *, it specifies that the property for the shape should be preserved during transformation from one database format to another.
 - ❑ The *fromColorMask* and *toColorMask* parameters should have IOTA mask color names `grayColor`, `mask1Color`, `mask2Color`, and `mask3Color`.

`CREATE_PURPOSE <purposeName> <purposeNumber>`

This command lets you create an `oaPurpose` property. Alternative database formats depend on custom purposes that do not exist in conventional OpenAccess technology databases. The `CREATE_PURPOSE` command creates an `oaPurpose` property with a name,

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

purposeName, and assigns to it a number, *purposeNumber*. The new *oaPurpose* is placed in the technology library, which must be editable. The command is active only in direct DEF In translation and is skipped during DEF Out translation because custom purposes are already defined in the alternative technology databases. The `CREATE_PURPOSE` command is mostly used for testing.

Examples of mapping configuration files

Example 1: Simple mapping

```
## OA2OA_CONFIG 1.0
CHANGE_SHAPE_LPP M1 drawing grayColor TO M2 fill grayColor
```

Converts all non-colored shapes on M1 layer with *oaPurpose* `drawing` to M2 layer with *oaPurpose* `fill`.

Example 2: Mapping shapes with different color masks to different layers

```
## OA2OA_CONFIG 1.0
CHANGE_SHAPE_LPP M1 drawing mask1Color TO M1_1 * grayColor
CHANGE_SHAPE_LPP M1 drawing mask2Color TO M1_2 * grayColor
CHANGE_SHAPE_LPP M1 drawing mask3Color TO M1_3 * grayColor
```

Converts colored shapes on M1 layer with *oaPurpose* `drawing` to non-colored shapes on M1_1, M1_2, and M1_3 layers.

The shapes are placed according their color masks. For instance, shapes with `mask3Color` are placed on M1_3. All the resulting shapes are specified as non-colored shapes and the purpose `drawing` is preserved.

Example 3. oaPurpose transformation of all database shapes

```
## OA2OA_CONFIG 1.0
CREATE_PURPOSE dummy 1
CHANGE_SHAPE_LPP * gapFill * TO * dummy *
```

Converts all shapes with purpose `gapFill`, to shapes with purpose as `dummy`. All color information is preserved. If *oaPurpose* `dummy` does not exist in the technology database, it is created by the `CREATE_PURPOSE` command.

Example 4. General conversion of color masks to purposes.

```
## OA2OA_CONFIG 1.0
CREATE_PURPOSE dummy 1
CREATE_PURPOSE dummy_a 2
CREATE_PURPOSE dummy_b 3
CREATE_PURPOSE dummy_c 4
```

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

```
CHANGE_SHAPE_LPP * gapFill mask1Color TO * dummy_a grayColor
CHANGE_SHAPE_LPP * gapFill mask2Color TO * dummy_b grayColor
CHANGE_SHAPE_LPP * gapFill mask3Color TO * dummy_c grayColor
```

Converts all colored shapes with purpose `gapFill` to `dummy_a`, `dummy_b`, and `dummy_c` non-colored shapes. Purposes `dummy_a`, `dummy_b`, and `dummy_c` are created if required.

If you use `*` only in the `to` argument of the command:

```
## OA2OA_CONFIG 1.0
CHANGE_SHAPE_LPP M1 drawing grayColor TO M2 * *
```

Then the command moves all non-colored drawing purpose shapes from M1 layer to M2 layer, preserving their color state (`grayColor`) and `oaPurpose` (`drawing`). This command is equivalent to:

```
## OA2OA_CONFIG 1.0
CHANGE_SHAPE_LPP M1 drawing grayColor TO M2 drawing grayColor
```


Colorless Layers and Colorless LPPs (Virtuoso Advanced Node for Layout Only)

Colorless layers are those colorable layers that are in the technology file but are not present in the `validLayers` constraint list of the constraint `virtuosoMPTSetup`.

Color checking for the `errorOnGrayShapes` and `errorOnUnlockedShapes` options is performed for all shapes on the colored layers and layer-purpose pairs (LPP). However, it is skipped for the shapes on colorless and pre-colored layers or LPPs.

Note: Colorable layers have more than one mask and pre-colored layers are the layers colored by the foundry.

DEF Out considers the following constraints to identify colorless layers:

- **validLayers:** Lets you specify a set of only those layers, LPPs, or both that can be colored, will be colored, and should be color-checked.
- **validPurposes** (when `'include` is specified): Lets you specify a set of only those purposes that should be included for color-checking on the LPPs.
- **preColoredLayers:** Lets you specify a set of layers that can be colored but need not have every shape colored. Coloring on these layers or LPPs should not be checked.

Following examples illustrate the use of these constraints while color-checking in DEF Out.

Example 1

Consider that the technology has the following colorable layers: M0, M1, M2, M3, M4, M5, M6, VIA0, VIA1, VIA2

If the `virtuosoMPTSetup` constraint group in the technology defined as:

```
;( group [override]           [definition]           [operator] )
;( ----- )
( "virtuosoDefaultMPTSetup"   nil      "virtuosoMPTSetup"
  interconnect(
    ( validLayers  (M0 VIA0 M1 VIA1 M2 VIA2 M3 M4 ) )
    ( validPurposes 'include ("drawing") )
  ) ;interconnect
  spacings(
    ( preColoredLayers "VIA0" "VIA1" "VIA2")
  ) ;spacings
) ;virtuosoDefaultMPTSetup
```

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

Here:

- **Colorable layers** are (M0, M1, M2, M3, M4, M5, M6, VIA0, VIA1, VIA2)
- **Valid layers** are (M0, VIA0, M1, VIA1, M2, VIA2, M3, M4).
- **Colorless layers** are all colorable layers that are not valid layers. These layers are (M5, M6).
- **Precolored layers** are (VIA0, VIA1, VIA2).
- **Exclusion layers**, which are layers excluded from DEF Out error-checking, include both precolored layers and colorless layers. These layers are (VIA0, VIA1, VIA2, M5, M6).

Example 2: When validPurposes with the include list and preColoredLayers with layers and LPPs is specified

Consider that the technology has the following colorable layers: M1, M2, M3

LPPs on these colorable layers are: ((M1 pin) (M1 drawing) (M1 net) (M2 pin) (M2 drawing) (M2 net) (M3 pin) (M3 drawing) (M3 net))

If the virtuosoMPTSetup constraint group in the technology defined as:

```
;( group [override]           [definition]           [operator] )
;( ----- )
( "virtuosoDefaultMPTSetup"    nil      "virtuosoMPTSetup"
  interconnect(
    ( validLayers (M1 M2) )
    ( validPurposes 'include      ("pin" "drawing") )
  ) ;interconnect
  spacings(
    ( preColoredLayers "M1" ("M3" "drawing") )
  ) ;spacings
) ;virtuosoDefaultMPTSetup
```

In this case, although color-checking should be performed for all shapes on the colored layers, M1, M2 and M3, it is restricted to only certain layers and LPPs. This is because considering the above constraints, some layers and LPPs are excluded from color-checking.

Here,

- **Colorless layers or LPPs** are ((M1 net) (M2 net) M3)
- **Pre-colored layers or LPPs** are (M1 (M3 drawing))

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

- **Exclusion layers and LPPs**, which are layers and LPPs that `errorOnGrayShapes` and `errorOnUnlockedShapes` options will exclude from DEF Out error-checking, include both precolored and colorless layers and LPPs. These are (M1 (M2 net) M3)
- Color-checking will be done only for layers and LPPs ((M2 pin) (M2 drawing))

Example 3: `validLayers` with LPPs and `preColoredLayers` with layers and LPPs:

Consider that the technology has the following colorable layers: M1, M2, M3

LPPs on these colorable layers are: ((M1 pin) (M1 drawing) (M1 net) (M2 pin) (M2 drawing) (M2 net) (M3 pin) (M3 drawing) (M3 net))

If the `virtuosoMPTSetup` constraint group in the technology is specified as:

```
; ( group [override]           [definition]           [operator] )
; ( ----- )
( "virtuosoDefaultMPTSetup"    nil      "virtuosoMPTSetup"
  interconnect (
    ( validLayers ( ( M1 pin) ( M1 drawing) ( M2 pin) ( M2 drawing) ) )
  ) ;interconnect
  spacings (
    ( preColoredLayers "M1" ("M3" "drawing") )
  ) ;spacings
) ;virtuosoDefaultMPTSetup
```

In this case, although color-checking should be performed for all shapes on the colored layers, M1, M2 and M3, it is restricted to only certain layers and LPPs. This is because considering the above constraints, some layers and LPPs are excluded from color-checking.

Here,

- **Colorless layers and LPPs** are ((M1 net) (M2 net) M3)
- **Pre-colored layers and LPPs** are (M1 (M3 drawing))
- **Exclusion layers and LPPs**, which are layers and LPPs that `errorOnGrayShapes` and `errorOnUnlockedShapes` options will exclude from DEF Out error-checking, include both precolored and colorless layers and LPPs. These are (M1 (M2 net) M3)
- Color-checking will be done only for layers and LPPs ((M2 drawing) (M2 pin))

Running LEF In and LEF Out from GUI

In addition to using the LEF/DEF translator from the command line, you can also run it from the GUI. This can be done by choosing the *File – Import – LEF* and *File – Export – LEF* commands from CIW.

This section provides description on the interactive graphical user interface forms used for running LEF In and LEF Out.

File – Import – LEF

Reads a LEF file and stores information in the OpenAccess library. When you run this command from the CIW, the *LEF In* form is displayed.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

LEF In Form

LEF In

×

LEF File Name

...

Target Library Name

▼

☐ Overwrite

☐ Share Library

Target Library Path

...

Target Tech Library Name

▼

Target Tech Library Path

...

Ref. Technology Libraries

Macro Target View Name

abstract

▼

Log File Name

Layer Map File Name

☐ Use Template File

☒ Use GUI Fields

Template File Name

Save Template File Name

...

Save

Comment Char

Pin Purpose

☐ Use Text Layer same as Pin Layer

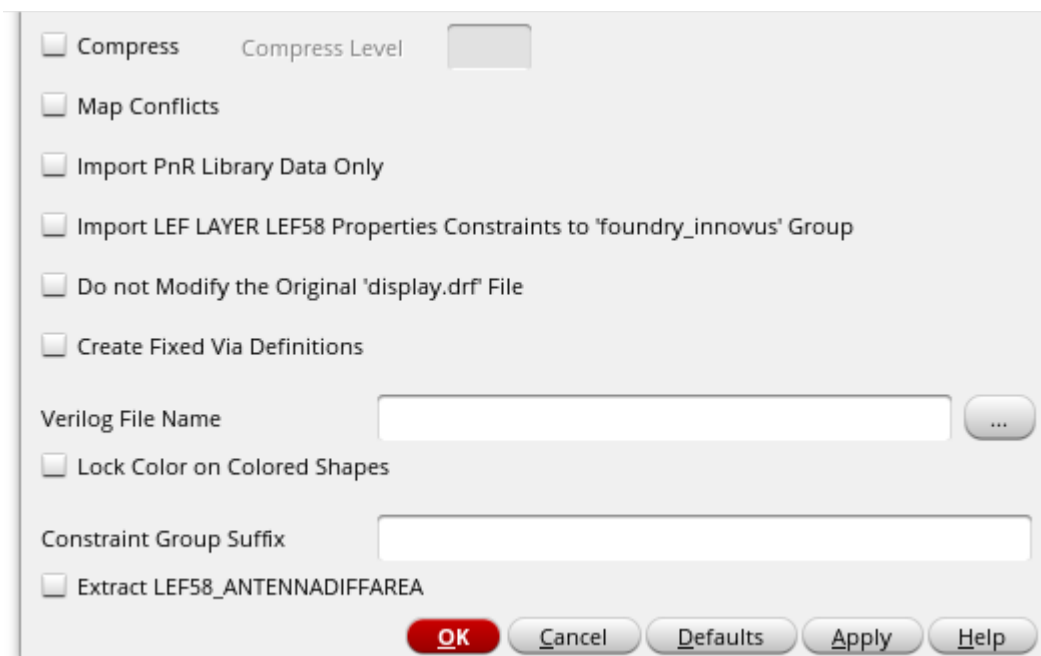
Text Layer Name

Text Purpose

Text Height

Design Data Translators Reference

Design Translation Using LEF/DEF Translator



LEF File Name

Use the *LEF File Name* field to specify a name for the input LEF file. Alternatively, click the ellipses (. . .) button to locate the LEF file on your machine.

Target Library Name

Use the *Target Library Name* field to specify a name for the target library.

If a target library does not exist, it is created in the current directory and contains the translated technology database. If the target library exists, it must have a technology database or refer to the technology database of another library.

In addition, existing macros, if redefined in the LEF file, are overwritten and new cells are created for the new macros.

Overwrite

Use the *overwrite* option to overwrite an existing design in the target library with the translated data. By default, this option is deselected. If this option is deselected, the default behavior is to update the existing library.

Share Library

Use the *Share Library* option to share the input-output library with other applications while the current application is running.

Target Library Path

Use the *Target Library Path* field for specifying the complete path where the target library is to be created. The path must be valid because the translator does not interpret the path and stores it in the `cds.lib` file. Alternatively, click the ellipses (. . .) button to locate the target library on your machine.

Target Tech Library Name

Use the *Target Tech Library Name* field to specify a name of the technology library to be used when creating a new cell library. If the specified technology library does not exist, the library is created using the technology information in the LEF file.

Target Tech Library Path

Use the *Target Tech Library Path* field to specify the path where the technology library is to be created. The path must be valid because the translator does not interpret the path and stores it in the `cds.lib` file. Alternatively, click the ellipses (. . .) button to locate the target technology library on your machine.

Ref. Technology Libraries

Use the *Ref. Technology Libraries* field to specify the parent technology databases that will be referenced by the incremental technology database. You can specify multiple parent technology databases in this field. If there are conflicts in the parent technology databases that you specify in this field, the tool generates the following error and stops.

ERROR: (OALEFDEF-90033): LefIn created a technology file that has conflicts. The created technology file is corrupt. Remove the file manually and run LefIn after resolving the conflicts.

Macro Target View Name

Use the *Macro Target View Name* field to select a view name for the translated macros. This option stores LEF macros with the specified view name. The two possible values for this

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

field are `abstract` and `layout`. If you do not provide any name in this field, then `abstract` is accepted as the default view name.

Log File Name

Use the *Log File Name* field to specify the filename where the log messages are to be stored. By default, the log messages are printed at `stderr`.

Layer Map File Name

Use the *Layer Map File Name* field to specify a layer map file. You can specify multiple layer map file names separated by spaces in this field. While creating layers, the layer numbers specified in these files are used.

A layer map file contains a list of LEF layer names that map to the corresponding OpenAccess layer numbers. The LEF layer names and OpenAccess layer numbers must be separated by a space.

A sample layer map file is shown below:

#LEFLayerName	OpenAccessLayerNumber
POLYS	0
CUT01	1
RX	2
CUT12	3
M1	4

In the above file, LEF layer `POLYS` is mapped to OpenAccess layer number 0. Similarly, `CUT01` is mapped to OpenAccess layer number 1, and so on.

Use Template File

You can choose to specify the values for LEF In using either a template file or through the GUI fields. When you select the *Use Template File* option, you are expected to provide a template file in the *Template File Name* field as the other GUI fields become disabled.

Template File Name

Use the *Template File Name* field to specify the template file that contains all the command options. This field is enabled only if you select the *Use Template File* option.

Save Template File Name

Use the *Save Template File Name* field to specify the name of the template file you want to save. You can either specify a name for the template file or click the ellipses (...) button to locate the template file on your machine. To save the file, click the *Save* button.

Save

Click the *Save* button, to save the file.

Use GUI Fields

Use this option to specify the LEF In options through the GUI form. This is the default option.

Comment Char

Use the *Comment Char* field to specify the special character that identifies comments.

Pin Purpose

Use the *Pin Purpose* field to specify the purpose for pin geometries.

Use Text Layer same as Pin Layer

Use the *Use text Layer same as Pin Layer* option to specify the text layer name from pin.

Text Layer Name

Use the *Text Layer Name* field to specify the layer on which the text for pin label should be created. If the specified layer does not exist in the technology database, it is added. However, if the text layer name is not specified, *text* is considered as the default pin label text layer.

Text Purpose

Use the *Text Purpose* option to specify the text purpose name.

Text Height

Use the *Text Height* field to specify an integer value for the height on which the text for pin label should be created. The height is specified in database units (DBU). If pin label text height is not specified, DBU value in the technology database for maskLayout views is considered as the default pin label text height.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

For example, your techlib defines 1 micron as 2000 DBU, and the `textHeight` value should be 0.1 micron, which is equal to 200 DBU. Since `textHeight` must be in DBU, specify the `textHeight` as 200.

Note: The Data model option was introduced in the IC 612 release. However, this option no longer exists in the current release. Lef In would always create the OpenAccess data with the highest data model, which is presently 4.

Compress

Use the *Compress* option to enable compression during LEF In.

Default value: Deselected

Compress Level

Use the *Compress Level* option to specify the level of compression to be applied during LEF In. You can specify a value from 1 to 9 for this option. Value 1 signifies the lowest level of compression and 9 signifies the highest level of compression. If you do not specify any value in the *Compress Level* option, value 1 is considered as the compression level during LEF In. This option is enabled only if the *Compress* option is selected.

Map Conflicts

Use the *Map Conflicts* field to map conflicts between the incremental and base technologies to constraints and constraint parameter definitions.

Import PnR Library Data Only

Use the *Import PnR Library Data Only* option to filter the Place and Route information. The information can be stored only as incremental technology database on the top of existing referenced technology database. Therefore, the Ref. Technology Libraries option needs to be specified. For more information, see General Processing Rules of - pnrLibDataOnly.

This option cannot be specified along with Import LEF LAYER LEF58 Properties Constraints to 'foundry_innovus' Group option.

Note: The *Import PnR Library Data Only* option works for 20 nm and larger designs. Cadence recommends using the *Import LEF LAYER LEF58 Properties Constraints to 'foundry_innovus' Group* option, instead.

Import LEF LAYER LEF58 Properties Constraints to 'foundry_innovus' Group

Use the *Import LEF LAYER LEF58 Properties Constraints to 'foundry_innovus' Group* option to enable Innovus to see the same design rules that are described in LEF without impacting the Virtuoso tools. For more information, see [General Processing Rules of useFoundryInnovus](#).

This option cannot be specified along with *Import PnR Library Data Only* option.

Do Not Modify the Original 'display.drf' File

Use the *Do Not Modify the Original 'display.drf' File* option to specify whether the original `display.drf` file is retained.

Create Fixed Via Definitions

Use the *Create Fixed Via Definitions* option to enable creation of fixed via definitions.

Verilog File Name

Use *Verilog File Name* field to specify list of input Verilog files. The specified Verilog files are sent to the `verilogAnnotate` utility as a post process to set the `isInterface` property and specify the terminal order of the bus terminals. An additional log file `verilogAnnotate.log` is also generated..

Lock Color on Colored Shapes

Use the *Lock Color on Colored Shapes* option to enable locking color on colored shapes during LEF In.

Constraint Group Suffix

Use the *Constraint Group Suffix* option to append the specified suffix to the names of the standard constraint groups, `LEFDefaultRouteSpec`, `LEFSpecialRouteSpec`, and `foundry_innovus`. These suffixed constraint groups are then used during LEF In translation, instead of the standard constraint groups.

Note: When you use this option, you can import the LEF file into different constraint group of existing technology without using [Ref. Technology Libraries](#) option.

Extract LEF58_ANTENNADIFFAREA

Use the *Extract LEF58_ANTENNADIFFAREA* option to specify whether the values specified for the LEF MACRO pin property LEF58_ANTENNADIFFAREA should be translated as width and ratio.

Default value: Deselected

File – Export – LEF

Creates a LEF format library description file from an OpenAccess library. When you run this command from the CIW, the *LEF Out* form is displayed.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

LEF Out Form

LEF Out

LEF File Name ...

Library Name ▼

☒ cells ☐ cellList File

Output Cell(s) ...

Cell List File ...

Output View(s) ...

Log File Name

LEF Version ▼ No Technology ☐

Output Technology Information Only ☐

Output 'foundry_innovus' Group Constraint as LEF LAYER LEF58 Properties ☐

Generate Cell List File By

☐ Use Template File ☒ Use GUI Fields

Template File Name

Save Template File Name ...

Output Color of Locked Color Shape Only ☐

Constraint Group Suffix

Create LEF58_ANTENNADIFFERAREA from Constraint ☐

LEF File Name

Use the *LEF File Name* field to specify a name for the output LEF file. Alternatively, click the ellipses (. . .) button to locate the LEF file on your machine.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

Library Name

Use the *Library Name* option to select the OpenAccess database library name. The OpenAccess database library contains the technology database and the designs represented in the technology. The library must be defined in the library definition file.

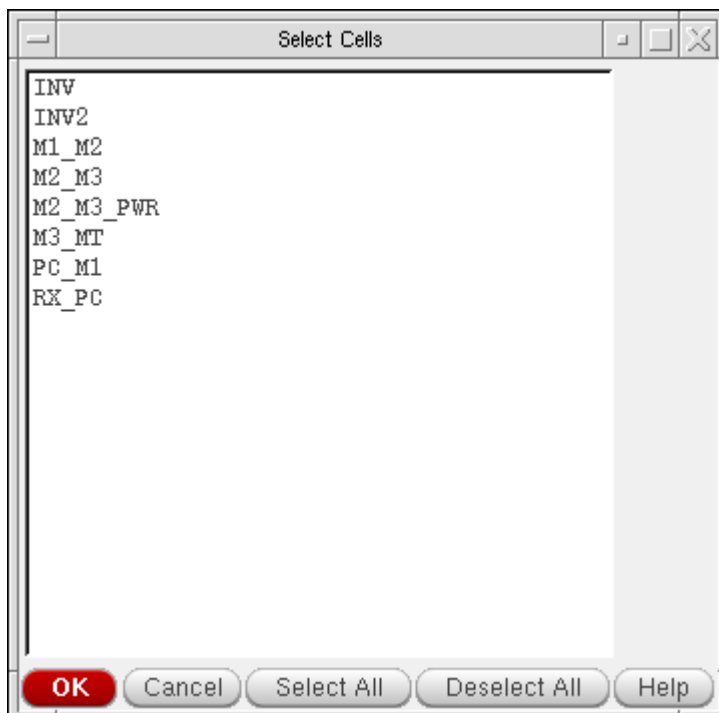
Note: The list of library names is sorted in the alphabetical order.

cells

Use the *cells* option to specify a cell name that is to be translated as a macro. You can specify more than one cell name.

Output Cell(s)

Use the *Output Cell(s)* field to specify the cell names. This option becomes disabled if you select the *cellListFile* option. You can also select the cells from the browser by clicking the ellipses button (. . .).



This browser does not display hidden cells.

Note: To hide cells, in Library Manager, right-click on a cell name and then select *Hide Cell*.

cellListFile

Select the *cellListFile* option if you want to specify a file containing a list of cell names that are to be translated as macros.

Cell List File

Use the *Cell List File* field to specify a file containing a list of cell names that are to be translated as macros. This option is enabled only if you select the *cellList File* option. You can also select a file containing a list of cell names from the browser by clicking the ellipses button (. . .).

Note: The cell names should be specified in the following format in the cell list file.
<libName> <cellName>

Output View(s)

Use the *Output View(s)* field to specify a view name for the input cell name. Alternatively, click the ellipses button (. . .) to select the view name from the browser. There can be more than one view name and input cell name.

The default is `abstract`. When you specify other view names, ensure that the content of these views is the same as the abstract views.

Log File Name

Use the *Log File Name* field to specify the filename where the log messages are to be stored. By default, the log messages are printed at `stderr`.

LEF Version

Use the *LEF Version* option to select the version of LEF. The default version is 5.8 while other supported versions are 5.4, 5.5, 5.6, 5.7, and 6.0.

No Technology

Use the *No Technology* option if you do not want to output the technology information. If you select this option the *Output Technology Information Only* option is disabled.

Output Technology Information Only

The *Output Technology Information Only* option outputs the technology information only. If you select this option, then you cannot specify cell, cell lists, and cell views for output. Also, if you select this option, *No Technology* option is disabled.

Output 'foundry_innovus' Group Constraints as LEF Layer Properties

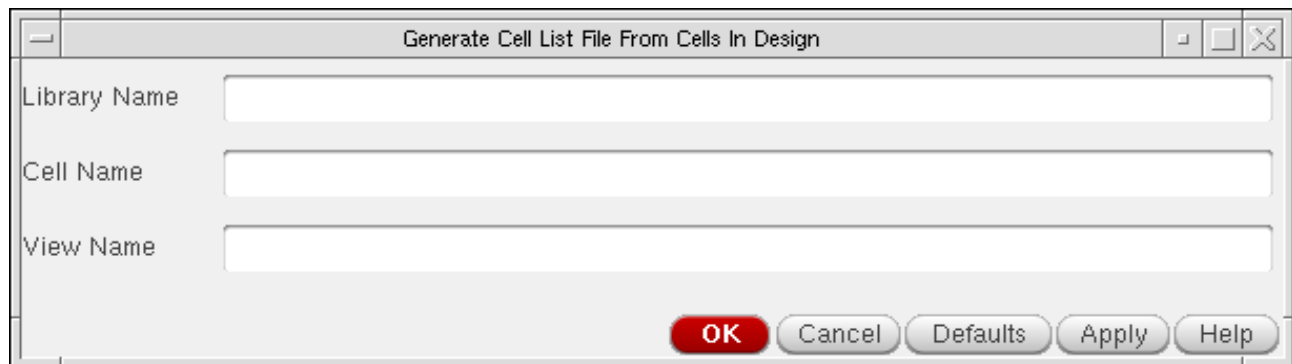
Use the *Output 'foundry_innovus' Group Constraints as LEF Layer Properties* option to export the `foundry_innovus` group constraints as a LEF LAYER LEF58 property.

Generate Cell List File By

The *Generate Cell List File By* option specifies the cell list file generation utility for building the cell list.

Cells in Design

The *Cells in Design* button lets you generate a cell list from all the master cells used in the design specified in the *Generate Cell List File From Cells in Design* form.



When you click *OK*, a cell list file with the name `lefout.list` is generated by default. However, if a cell list name file is specified in the *LEF Out* form, then the cell list file is created with the specified name.

A cell list file has the following format:

```
libName      cellName
```

If more than one view of the cell exists, then you can specify the view name in the parent *LEF Out* form.

Use Template File

You can choose to specify the values for LEF Out using either a template file or through the GUI fields. When you select the *Use Template File* option, you are expected to provide a template file in the *Template File Name* field.

Template File Name

Use the *Template File Name* field to specify the template file that contains all the command options. This option is enabled only if you select the *Use Template File* option.

Save Template File Name

Use the *Save Template File Name* field to specify the name of the template file you want to save. You can either specify a name for the template file or click the ellipses (...) button to locate the template file on your machine. To save the file, click the *Save* button.

Use GUI Fields

The *Use GUI Fields* option allows you to specify the LEF Out options in the GUI form. This is the default option.

Output Color of Locked Shape Only

Use the *Output Color of Locked Shape Only* option to output colors for only those shapes that have their locked state set to true. This includes pathSegs in nets, rects and polygons, and fill shapes and vias in specialnets.

Constraint Group Suffix

Use the *Constraint Group Suffix* option to append the specified suffix to the names of the standard constraint groups, LEFDefaultRouteSpec, LEFSpecialRouteSpec, and foundry_innovus when searching for the standard constraint groups.

Create LEF58_ANTENNADIFFAREA from Constraint

Use the *Create Extract LEF58_ANTENNADIFFAREA from Constraint* option to specify whether values specified for the LEF MACRO pin property LEF58_ANTENNADIFFAREA should be created as width and ratio from constraint.

Default value: Deselected

OpenAccess LEF Translators

LEF In and LEF Out translators support 32 nm constraint data interoperability between Cadence applications. It contains the functionality to parse LEF58 property strings that refers to the 32nm constraint and create the corresponding constraints on the associated objects in a technology database.

- The LEF In supports recognizing and parsing LEF58 property strings and mapping them to the corresponding constraint data.
- The LEF Out supports recognizing the 32nm constraint or 32nm parameters on a constraint and generate the corresponding LEF58 property string.

Running DEF In and DEF Out from GUI

You can run DEF In and DEF Out from the GUI by choosing the *File – Import – DEF* and *File – Export – DEF* commands from CIW. This section provides description on the interactive GUI forms used for running DEF In and DEF Out.

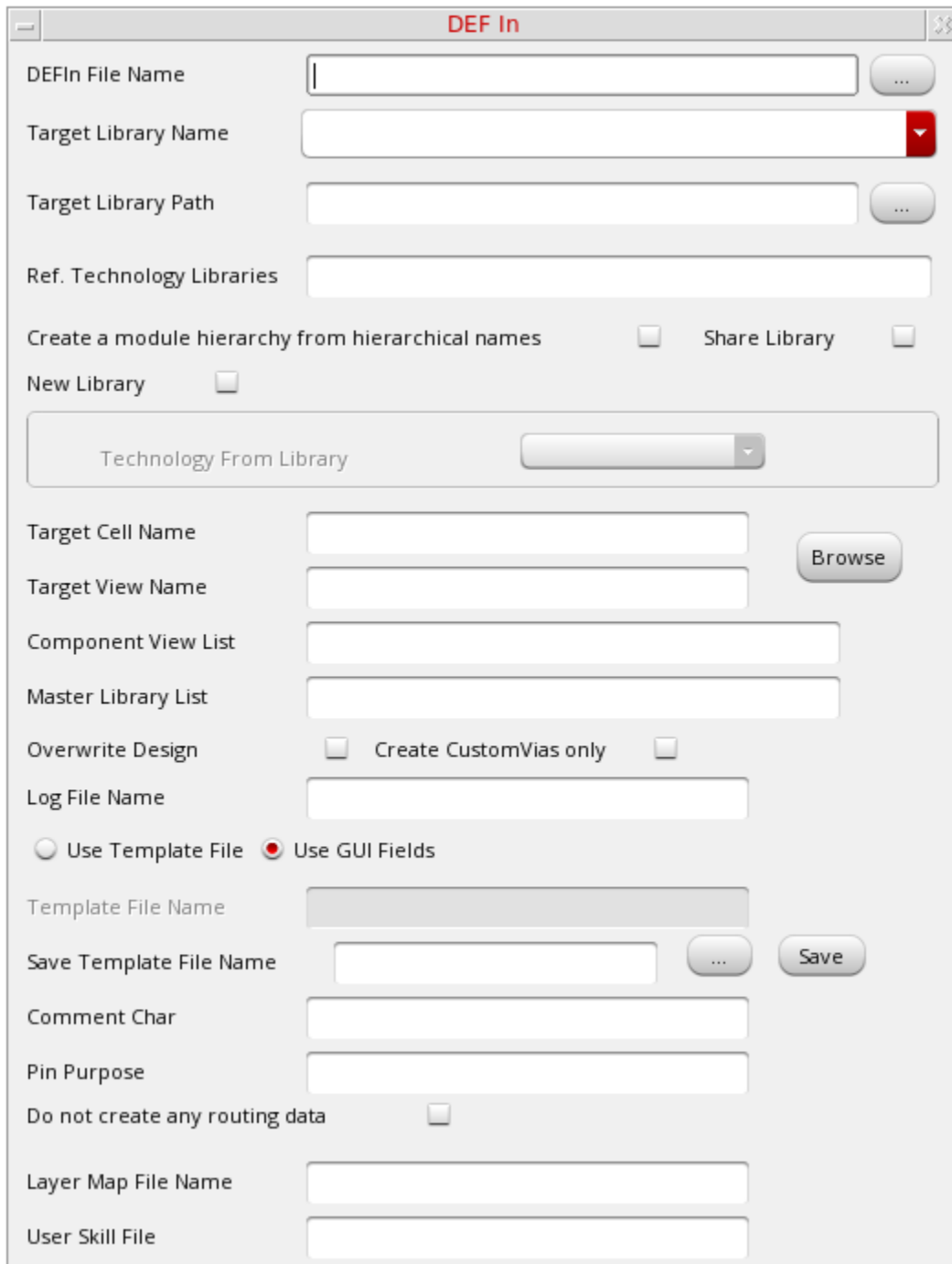
File – Import – DEF

Reads a DEF file into the OpenAccess database. When you run this command from the CIW, the *DEF In* form is displayed.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

DEF In Form



The image shows a software window titled "DEF In" with a standard macOS-style title bar. The window contains a series of input fields and checkboxes for configuring a design translation. The fields are arranged in a vertical stack, with some having associated buttons like "Browse" or "Save".

DEF In

DEFIn File Name

Target Library Name

Target Library Path

Ref. Technology Libraries

Create a module hierarchy from hierarchical names ☐ Share Library ☐

New Library ☐

Technology From Library

Target Cell Name

Target View Name

Component View List

Master Library List

Overwrite Design ☐ Create CustomVias only ☐

Log File Name

☐ Use Template File ☒ Use GUI Fields

Template File Name

Save Template File Name

Comment Char

Pin Purpose

Do not create any routing data ☐

Layer Map File Name

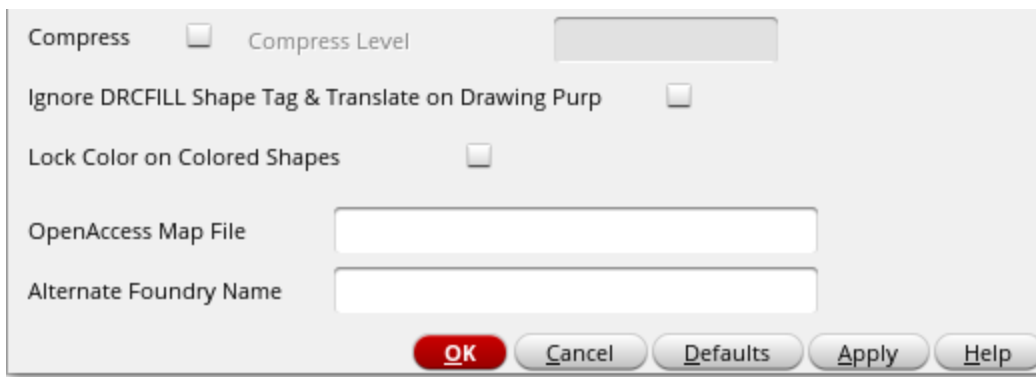
User Skill File

Browse

Save

Design Data Translators Reference

Design Translation Using LEF/DEF Translator



DEFIn File Name

Use the *DEFIn File Name* field to specify a name for the input DEF file. Alternatively, click the ellipses (. . .) button to locate the DEF file on your machine.

Target Library Name

Use the *Target Library Name* field to specify a name for the target library.

If a target library does not exist, it is created in the current directory and contains the translated technology database. If the target library exists, it must have a technology database or refer to the technology database of another library.

In addition, existing macros, if redefined in the DEF file, are overwritten and new cells are created for the new macros.

Target Library Path

Use the *Target Library Path* field for specifying the complete path where the target library is to be created. The path must be valid because the translator does not interpret the path and stores it in the `cds.lib` file. Alternatively, click the ellipses (. . .) button to locate the target library on your machine.

Ref. Technology Libraries

Use the *Ref. Technology Libraries* field to specify the parent technology databases that will be referenced by the incremental technology database. You can specify multiple parent

technology databases in this field. If there are conflicts in the parent technology databases that you specify in this field, the tool generates the following error and stops:

ERROR: (OALEFDEF-90032): DefIn has detected a conflict in the technology database graph. The specified reference technology libraries have conflict among themselves or with the technology of the target library. The conflicts need to be resolved manually before running the DefIn application.

Create a module hierarchy from hierarchical names

Use the *Create a module hierarchy from hierarchical names* option to create a module hierarchy based on hierarchical names of instances and nets. By default, this option is deselected. When this option is not selected, only the physical data is created and no module hierarchy is derived from the DEF hierarchical names. Also, the translator escapes hierarchy delimiters in the names of instances and nets in the OpenAccess database.

Note: Using this option may decrease the DEF In speed and enhance the interoperability with tools that expect an embedded module hierarchy (EMH).

Share Library

Use the *Share Library* option to share the input-output library with other applications while the current application is running.

New Library

Use the *New Library* option if you want to create a new target library.

Technology from Library

Use the *Technology from Library* field to specify the technology library name to attach to the output library. You can use this option only when you create a new target library by selecting the *New Library* option.

Target Cell Name

Use the *Target Cell Name* field to specify a name for the destination cell.

Target View Name

Use the *Target View Name* field to specify a name for the destination view.

Browse

Use the *Browse* button to select a library, cell, and view name from the library browser.

Component View List

Use the *Component View List* field to specify view names for the master cells that are searched for a DEF COMPONENTS construct record. The view names must be separated by a comma, space, or a tab. The view names are searched in the order you specify. For example, if the *Component View List* field contains `layout abstract abstractNew` as the view names, then `layout` gets first priority.

If you do not specify any view name in the *Component View List* field, DEF In searches for the `abstract` and `autoAbstract` views. If none of these views are found, an error message is displayed.

Master Library List

Use the *Master Library List field* to specify all the reference libraries that contain master cells.

Overwrite Design

Use the *Overwrite Design* option to overwrite an existing design in the target library with the translated data. By default, this option is deselected. If this option is deselected, the default behavior is to update the existing library

Create CustomVias only

Use the *Create CustomVias only* option if you want to create vias using custom viaDefs instead of existing standard viaDefs for all vias in the def file.

Log File Name

Use the *Log File Name* field to specify the filename where the log messages are to be stored. By default, the log messages are printed at `stderr`.

Use Template File

You can choose to specify the values for DEF In using either a template file or through the GUI fields. When you select the *Use Template File* option, you are expected to provide a template file in the *Template File Name* field as the other GUI fields become disabled.

Template File Name

Use the *Template File Name* field to specify the template file that contains all the command options. This field is enabled only if you select the *Use Template File* option.

Save Template File Name

Use the *Save Template File Name* field to specify the name of the template file you want to save. You can either specify a name for the template file or click the ellipses (...) button to locate the template file on your machine. To save the file, click the *Save* button.

Use GUI Fields

Use this option to specify the DEF In options through the GUI form. This is the default option.

Comment Char

Use the *Comment Char* field to specify the special character that identifies comments.

Pin Purpose

Use the *Pin Purpose* field to specify the purpose for pin geometries.

Do not create any routing data

Use the *Do not create any routing data* option if you want DEF In to ignore any routing data during translation including the vias section. Using this option can save processing time and space.

Layer Map File Name

Use the *Layer Map File Name* field to specify a layer map file. You can specify multiple layer map file names separated by spaces in this field. While creating layers, the layer numbers specified in these files are used.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

A layer map file contains a list of LEF layer names that map to the corresponding OpenAccess layer numbers. The LEF layer names and OpenAccess layer numbers must be separated by a space.

A sample layer map file is shown below:

#LEFLayerName	OpenAccessLayerNumber
POLYS	0
CUT01	1
RX	2
CUT12	3
M1	4

In the above file, LEF layer POLYS is mapped to OpenAccess layer number 0. Similarly, CUT01 is mapped to OpenAccess layer number 1, and so on.

Note: The Data model option was introduced in the IC 612 release. However, this option no longer exists in the current release. Def In would always create the OpenAccess data with the highest data model, which is presently 4.

User Skill File

Use the *User SKILL File* field to specify a file that consists of user-defined SKILL routines. You can specify mapping mechanism for cell names, layers, and fonts through the use of programming routines written in SKILL in the SKILL file.

In order to load all the required SKILL procedures in a single file, use only one SKILL file during translation. The SKILL file is searched with respect to the run directory if the complete path of the SKILL file is not specified.

UserSkillFile will always be loaded before the libInit.il file. The libInit.il file is usually present in the technology file.

Compress

Use the *Compress* option to enable compression during DEF In.

Default value: Deselected

Compress Level

Use the *Compress Level* option to specify the level of compression to be applied during DEF In. You can specify a value from 1 to 9 for this option. Value 1 signifies the lowest level of compression and 9 signifies the highest level of compression. If you do not specify any value

in the *Compress Level* option, value 1 is considered as the compression level during DEF In. This option is enabled only if the *Compress* option is selected.

Ignore DRCFILL Shape Tag & Translate on Drawing Purp

Use *Ignore DRCFILL Shape Tag & Translate on Drawing Purp* to map all shapes specified with the +SHAPE DRCFILL tag in the SPECIALNETS section to the drawing purpose. If this option is not specified, these shapes are mapped to the gapFill purpose.

Lock Color on Colored Shapes

Use the *Lock Color on Colored Shapes* option to enable locking color on colored shapes during DEF In.

Default value: Deselected

OpenAccess Map File

Use the *OpenAccess Map File* option to use the specified mapping file to post process the design before it is saved.

For more information, see [Using the oaMapFile option for different OpenAccess Database Formats.](#)

Alternate Foundry Name

Use the *Alternate Foundry Name* option to use the specified constraint group instead of the default foundry constraint group. Constraint group specified using this option, overwrites the name specified in the .cdsenv file using the [AlternateFoundryCG](#) environment variable.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

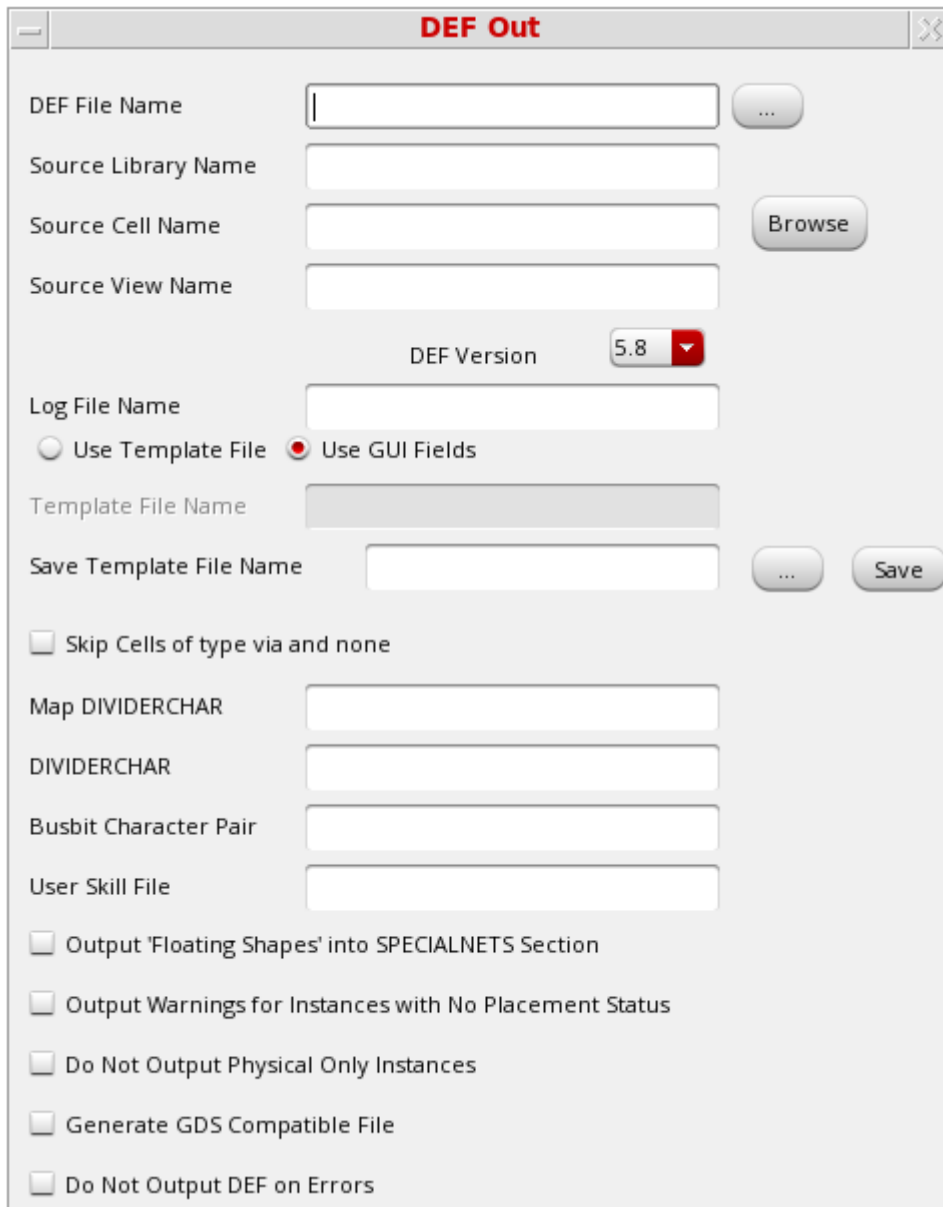
File – Export – DEF

Creates a DEF file from an OpenAccess design cellview. When you run this command from the CIW, the *Virtuoso* is displayed.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

DEF Out Form



The image shows a software dialog box titled "DEF Out" in red text. It contains various input fields and checkboxes for configuring the DEF output process. The fields include DEF File Name, Source Library Name, Source Cell Name, Source View Name, DEF Version (a dropdown menu set to 5.8), Log File Name, Template File Name, and Save Template File Name. There are also checkboxes for "Use Template File", "Skip Cells of type via and none", "Output 'Floating Shapes' into SPECIALNETS Section", "Output Warnings for Instances with No Placement Status", "Do Not Output Physical Only Instances", "Generate GDS Compatible File", and "Do Not Output DEF on Errors". Buttons for "Browse", "Save", and "..." are also present.

DEF Out

DEF File Name ...

Source Library Name

Source Cell Name Browse

Source View Name

DEF Version 5.8 ▼

Log File Name

☐ Use Template File ☒ Use GUI Fields

Template File Name

Save Template File Name ... Save

☐ Skip Cells of type via and none

Map DIVIDERCHAR

DIVIDERCHAR

Busbit Character Pair

User Skill File

☐ Output 'Floating Shapes' into SPECIALNETS Section

☐ Output Warnings for Instances with No Placement Status

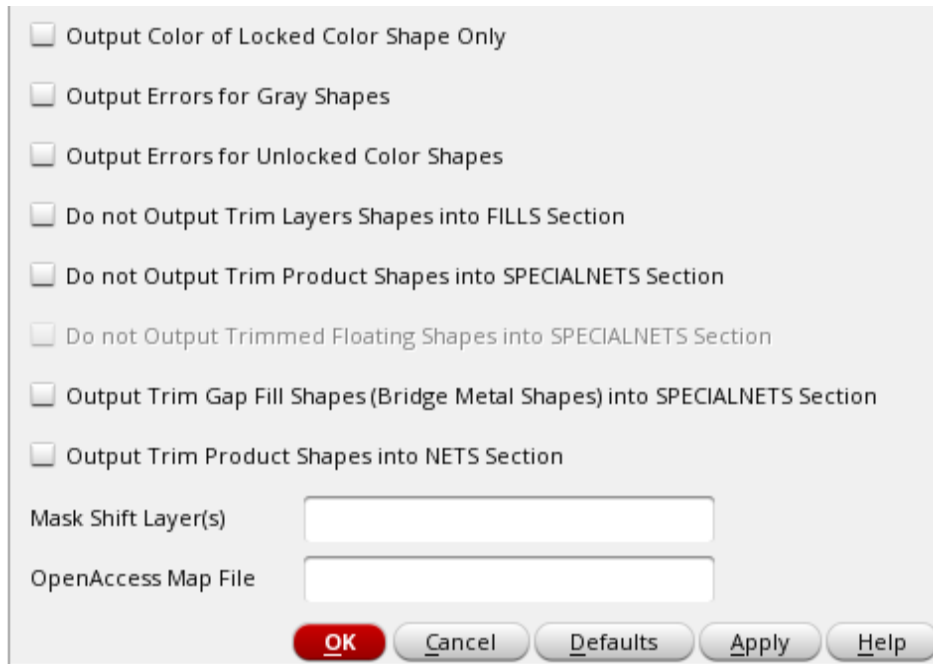
☐ Do Not Output Physical Only Instances

☐ Generate GDS Compatible File

☐ Do Not Output DEF on Errors

Design Data Translators Reference

Design Translation Using LEF/DEF Translator



The screenshot shows a dialog box with the following options and fields:

- ☐ Output Color of Locked Color Shape Only
- ☐ Output Errors for Gray Shapes
- ☐ Output Errors for Unlocked Color Shapes
- ☐ Do not Output Trim Layers Shapes into FILLS Section
- ☐ Do not Output Trim Product Shapes into SPECIALNETS Section
- ☐ Do not Output Trimmed Floating Shapes into SPECIALNETS Section
- ☐ Output Trim Gap Fill Shapes (Bridge Metal Shapes) into SPECIALNETS Section
- ☐ Output Trim Product Shapes into NETS Section

Mask Shift Layer(s):

OpenAccess Map File:

Buttons: **OK** (red), Cancel, Defaults, Apply, Help

DEF File Name

Use the *DEF File Name* field to specify a name for the output DEF file. Alternatively, click the ellipses (. . .) button to locate the DEF file on your machine.

Source Library Name

Use the *Source Library Name* field to specify a name for the library that is to be used as a source library.

Source Cell Name

Use the *Source Cell Name* field to specify a name for the cell that is to be translated.

Source View Name

Use the *Source View Name* field to specify a name for the view of the cell that is to be translated.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

Browse

Use the *Browse* button to select the source library, cell, and view name from the library browser.

Note: If a cell is open in Virtuoso and view type is `maskLayout`, source library, cell, and view names are automatically filled in the DEF Out form.

DEF Version

Use the *DEF Version* option to select the version of DEF. The default version is 5.8 while other supported versions are 5.4, 5.5, 5.6, 5.7, and 6.0.

Log File Name

Use the *Log File Name* field to specify the filename where the log messages are to be stored. By default, the log messages are printed at `stderr`.

Use Template File

You can choose to specify the values for DEF Out using either a template file or through the GUI fields. When you select the *Use Template File* option you are expected to provide a template file in the *Template File Name* field as the other GUI fields become disabled.

Template File Name

Use the *Template File Name* field to specify the template file that contains all the command options. This field is enabled only if you select the *Use Template File* option.

Save Template File Name

Use the *Save Template File Name* field to specify the name of the template file you want to save. You can either specify a name for the template file or click the ellipses (...) button to locate the template file on your machine. To save the file, click the *Save* button.

Use GUI Fields

Use the *Use GUI Fields* option to specify the DEF Out options through the GUI form. This is the default option.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

Skip Cells of type via and none

Use the *Skip Cells of type via and none* option to prevent the cells with type none and via from being written to the output def file. By default, this option is deselected.

Table 6-1 Common Options in the LEF/DEF Forms

Option Name	Description
<i>OK</i>	Used to accept all the changes on the form. All options present on the respective forms are saved to the LEF/DEF file.
<i>Cancel</i>	Used to reject all the changes on the form. Use the <i>Cancel</i> button to exit the <i>Virtuoso LEF In, LEF Out, DEF In, and DEF Out</i> form.
<i>Defaults</i>	Used to restore the default settings of the form.
<i>Apply</i>	Used to incorporate all the option settings in the form.
<i>Help</i>	Used to invoke online help.

Map DIVIDERCHAR

Use the *Map DIVIDERCHAR* option to specify a single character to be replaced in instance and net names by the DEF.

DIVIDERCHAR

Use the *DIVIDERCHAR* option specifies a new divider character to be used during output.

Busbit Character Pair

Use the *Busbit Character Pair* option to specify new busbit character pair to be used during output.

User Skill File

The *User SKILL File* field lets you specify a file that consists of user-defined SKILL routines. Using this file, you can specify a mapping mechanism for cell names, layers, and fonts through the use of programming routines written in SKILL.

In order to load all the required SKILL procedures in a single file, use only one SKILL file during translation. The SKILL file is searched with respect to the run directory if complete path of the SKILL file is not specified.

Output 'Floating Shapes' into SPECIALNETS Section

Use the *Output 'Floating Shapes' into SPECIALNETS Section* to allow the output of trim layer shapes into the `SPECIALNETS` section. When specified during DEF OUT translation, any drawing shape on a layer that is not a member of a net is exported on a net in the `SPECIALNETS` section named `_FLOATING_DRAWING_SHAPES_RESERVED`.

Note: The trim layer material is used to specify a layer that removes material from another layer. For example, a cutPoly layer that removes material from a Poly layer. The trim layer is the preferred method used by Virtuoso Space-based Router to determine the usage of cut poly operations.

Output Warnings for Instances with No Placement Status

Use the *Output Warnings for Instances with no Placement Status* option to add warnings in the log file for the instances in the DEF Out output that have no placement status.

Do Not Output Physical Only Instances

Use the *Do Not Output Physical Only Instances* option to prevent the output of physical instances. If this option is not specified, physical instances are placed in the `DEF COMPONENTS` section and have the `+ SOURCE DIST` statement added to them.

Generate GDS Compatible File

Use the *Generate GDS Compatible File* option to generate a DEF file with limits that make it compatible with GDS. When this option is specified DEF Out automatically chops all polygons and blockages so that a valid GDS file is generated from the DEF file. Chopping of these polygons and blockages occurs when the number of vertexes for the respective polygon or blockage reach a specified limit value.

Do Not Output DEF on Errors

Use the *Do Not Output DEF on Errors* option to prevent the DEF output file from being generated if there are errors during translation.

Output Color of Locked Shape Only

Use the *Output Color of Locked Shape Only* option to output colors for only those shapes that have their locked state set to true. This includes pathSegs in nets, rects and polygons, and fill shapes and vias in specialnets.

Output Errors for Gray Shapes

Use the *Output Errors for Gray Shapes* option to display errors if gray or uncolored shapes or vias are found during translation.

Note: When the technology library contains preColoredLayers constraint and in the VirtuosoMPTSetup constraint group, the layers list from the first preColoredLayers constraint is not considered for error checking. Additionally, colorless layers and colorless LPPs are not considered for error checking. (Virtuoso Advanced Node for Layout Only)

Output Errors for Unlocked Color Shapes

Use the *Output Errors for Unlocked Color Shapes* option to display errors if any colored shapes or vias with unlocked status are found during translation.

Note: When the technology library contains preColoredLayers constraint and in the VirtuosoMPTSetup constraint group, the layers list from the first preColoredLayers constraint is not considered for error checking. Additionally, colorless layers and colorless LPPs are not considered for error checking. (Virtuoso Advanced Node for Layout Only)

Do Not Output Trim Layers Shapes into FILLS Section

Use the *Do Not Output Trim Layers Shapes into FILLS Section* to prevent the output of trim layer shapes into the FILLS section.

Do Not Output Trim Product Shapes into SPECIALNETS Section

Use the *Do Not Output Trim Product Shapes into SPECIALNETS Section* to prevent the output of trim layer shapes into the SPECIALNETS section.

Do not Output Trimmed Floating Shapes into SPECIALNETS Section

Use the *Do not Output Trimmed Floating Shapes into SPECIALNETS Section* to prevent trimmed floating shapes from being imported to the SPECIALNETS section. This

option is valid only when the Output 'Floating Shapes' into SPECIALNETS Section option is specified.

Output Trim Gap Fill Shapes (Bridge Metal Shapes) into SPECIALNETS Section

Use the *Output Trim Gap Fill Shapes (Bridge Metal Shapes) into SPECIALNETS Section* to allow output of the trim gap fill shapes for the bridge metal shapes into the SPECIALNETS section.

Output Trim Product Shapes into NETS Section

Use the *Output Trim Product Shapes into NETS Section* to allow output of the trim product shapes into the NETS section.



Options *Do Not Output Trim Product Shapes into SPECIALNETS Section* and *Output Trim Product Shapes into NETS Section* cannot be specified together.

Mask Shift Layer(s)

Use the *Mask Shift Layer(s)* option to specify list of layers used in COMPONENTMASKSHIFT statement. The list should be separated by a white space.

OpenAccess Map File

Use the ***OpenAccess Map File option*** to use the specified mapping file to pre-process the design before exported to DEF.

For more information, see Using the oaMapFile option for different OpenAccess Database Formats.

Preparing a Template File

A template file is a collection of file names and option values that reflects the same information given in the LEF-DEF forms. You use the template file to load previously saved information to the LEF-DEF forms or to load the arguments that are required in the forms. You create a template file in the following ways:

- Use the *Template File Name* command in the LEF In, LEF Out, DEF In or DEF Out forms to save the information currently entered in the form. The information is saved to the file you specify as the template file.
- Create a text file. You can specify the command line option names and value pairs on every newline.

Following is a sample LEF In template file:

```
#lefin test.lef
lib mylib
tech mytechLib
overwrite
shared
log lefinlog.txt
pinPurp net
```

Similarly you can create template files for LEF Out, DEF In, and DEF Out.

Setting the Environment Variables in LEF In and LEF Out

You can set the default values of the LEF Out and LEF In GUI options by using the `LEF_OUT` and `LEF_IN` environment variables, respectively. These variables contain the path to the LEF Out or LEF In template files that consist the values of the LEF Out and LEF In GUI options.

Therefore, each time you run LEF Out or LEF In from the GUI, the following happens:

1. First, the path of the respective template file is taken from the `LEF_OUT` or `LEF_IN` environment variable.
2. Next, the values of the GUI options are populated from the template file.

Note: You can override these values by specifying different values explicitly.

For example, set `LEF_IN` and invoke LEF In by using the following command:

```
setenv LEF_IN /home/user1/templateIn.txt
```

Here, LEF In is invoked in the interactive mode and the values of LEF In options are taken from the `templateIn.txt` template file specified in the `LEF_IN` environment variable.

Setting the Environment Variables in DEF In and DEF Out

You can set the default values of the DEF Out and DEF In GUI options by using the `DEF_OUT` and `DEF_IN` environment variables, respectively. These variables contain the path to the DEF Out or DEF In template files that consist the values of the DEF Out and DEF In GUI options.

Therefore, each time you run DEF Out or DEF In from the GUI, the following happens:

1. First, the path of the respective template file is taken from the `DEF_OUT` or `DEF_IN` environment variable.
2. Next, the values of the GUI options are populated from the template file.

Note: You can override these values by specifying different values explicitly.

For example, set `DEF_IN` and invoke DEF In by using the following command:

```
setenv DEF_IN /home/user1/templateIn.txt
```

Here, DEF In is invoked in the interactive mode and the values of DEF In options are taken from the `templateIn.txt` template file specified in the `DEF_IN` environment variable.

Viewing Native OpenAccess Libraries in the Virtuoso Studio Design Environment

Native OpenAccess libraries need the LPP information in the technology file and the `Display.drf` file to work in the Virtuoso Studio design environment. The Virtuoso Studio design environment `oa2dfII` utility takes a native library as input to create the LPP information in the technology file and the `Display.drf` file.

oa2dfII Command Line Syntax

```
oa2dfII {-h | -lib <libName> [-force]}
```

where,

<code>-h</code>	Prints help message
<code>-lib libName</code>	Input library name
<code>-force</code>	Writes the display packets even though corresponding LPPs already exist.

Note: Display packets are not generated properly if LPP is already present in the technology. In that case, you can use the `-force` option to forcefully write the display packets.

For example, to run the `oa2dfII` utility by using `myNativeOALib` as input library, use the following command:

```
oa2dfII -lib myNativeOALib
```

Before running the utility, ensure that the `myNativeOALib` input library must be writable for the updation.

The `oa2dfII` utility creates the required `Display.drf` file which contains the display information for viewing the data in Virtuoso Studio design environment applications. The utility does the tech uprev and automatically creates the system layers/purposes and LPPs. Next, the tool sets the required default display attributes such as validity, visibility, selectability, and dragability for the LPPs.

Note: The utility updates the technology of the input native OpenAccess library. Therefore after you run the `oa2dfII` utility, the input native OpenAccess library gets updated with additional information for the Virtuoso Studio design environment and might not work as expected in the native OpenAccess environment.

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

The `oa2dfll` utility creates new layer-purpose pairs in the technology library and display packets to support the display of blockages and track patterns. For each layer defined in the OpenAccess database, `oa2dfll` will create layer-purpose pairs only for the following purposes: `drawing`, `fill`, `slot`, `gapfill`, `fillOPC`, `blockage`, `track`, `boundary`, `net`, and `pin`. For example, if a layer `L1` is defined in the OpenAccess database, following LPP's will be created in technology:

```
L1 drawing
L1 fill
L1 slot
L1 gapFill
L1 fillOPC
L1 blockage
L1 track
L1 boundary
L1 pin
L1 net
```

For information about LEF/DEF SKILL functions, see the [Design Data Translator's SKILL Reference](#).

Design Data Translators Reference

Design Translation Using LEF/DEF Translator

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

This chapter contains the following sections:

- [Introduction to Allegro Translator](#)
- [Using the Allegro Translator](#)
- [Methods Used to Perform Package Design Translation](#)
- [Import Allegro File Form](#)
- [Export Allegro File Form](#)
- [Design Object Mapping](#)
- [Translating Parametrized Tline Cells](#)
- [Data Model Mapping](#)
- [Layer Mapping](#)
- [Troubleshooting Allegro File Translation Issues](#)

Introduction to Allegro Translator

VRF is a schematic-driven layout flow in Virtuoso. The technology lets you use Virtuoso to create and edit package schematics and layouts. Package design and implementation are done in Allegro, which supports objects such as SMD (Surface Mounted Devices), padstacks, dies, bondwires, and bondwire fingers.

If the package implementation is initiated in Allegro and continued in Virtuoso, the Allegro Import translator maps these Allegro concepts into instances of Pcells. The supermasters are provided in a generic library and are also used when the package implementation is initiated

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

in Virtuoso. During instance creation, CDF parameters are used to bind instance parameters to appropriate values.

Since Virtuoso does not have all the features to finish the package, implementation needs to be finished in Allegro. Layout data is transferred from Virtuoso to Allegro using the Allegro Export translator. The Allegro Export translator maps the parameters on the Pcell instances back to the Allegro objects.

Using the Allegro Translator

Allegro translator enables you to translate the following Allegro files:

- Cadence SiP Layout (.sip)
- Allegro Package Designer (.mcm)
- Technology file with layer stack up and constraint information
- Catalog of DRA
- A board or package file created in Allegro PCB Layout (.brd)

Where,

- A single SiP (sip, mcm, .brd) file or technology file is converted into an OpenAccess database.
- A catalog of DRA are converted into base cellviews.

Note: This tool works with cdnsip version 17.20.052 or later.

Allegro Import involves the following two phases:

1. In the first phase, the SiP file is read in the cdsnip tool that creates an intermediate file containing all the necessary information.
2. The Allegro Import translator reads this file and creates the OpenAccess database in Virtuoso.

User can now edit this database in Virtuoso.

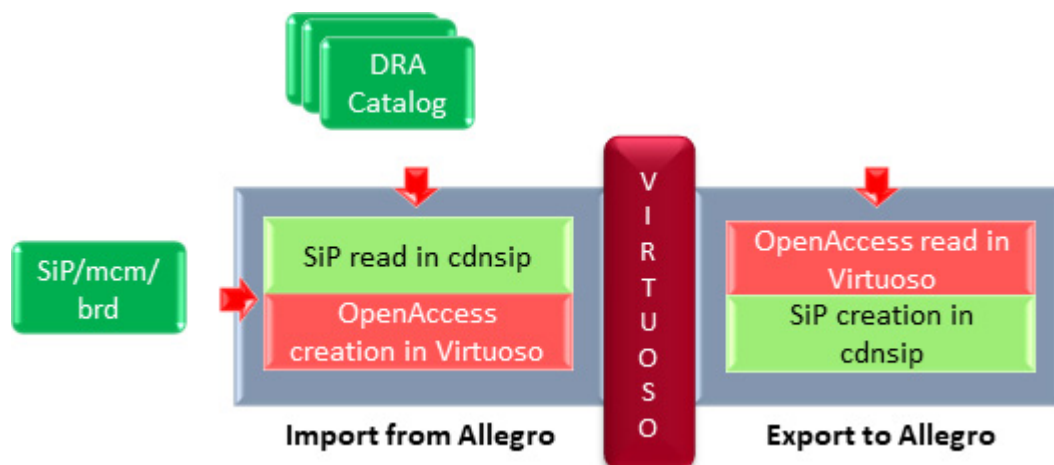
Allegro Export also involves two phases, which are the following:

1. The updated OpenAccess database is read in Virtuoso and an intermediate file is created.

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

2. The cdnsip tool is invoked to read this intermediate file and create the Allegro file or a DRA catalog.



Methods Used to Perform Package Design Translation

You can translate package design files by using the Allegro translators accessible through a Graphical User Interface (GUI).

To translate an Allegro package file to Virtuoso, launch:

- [Import Allegro File Form](#)

To export a file from Virtuoso to an Allegro package file, launch:

- [Export Allegro File Form](#)

Import Allegro File Form

To launch Allegro Import translator and display the Import Allegro File form, select the *File – Import – From Allegro* option from the CIW.

This section describes the various options on the Import Allegro File form.

Import Mode

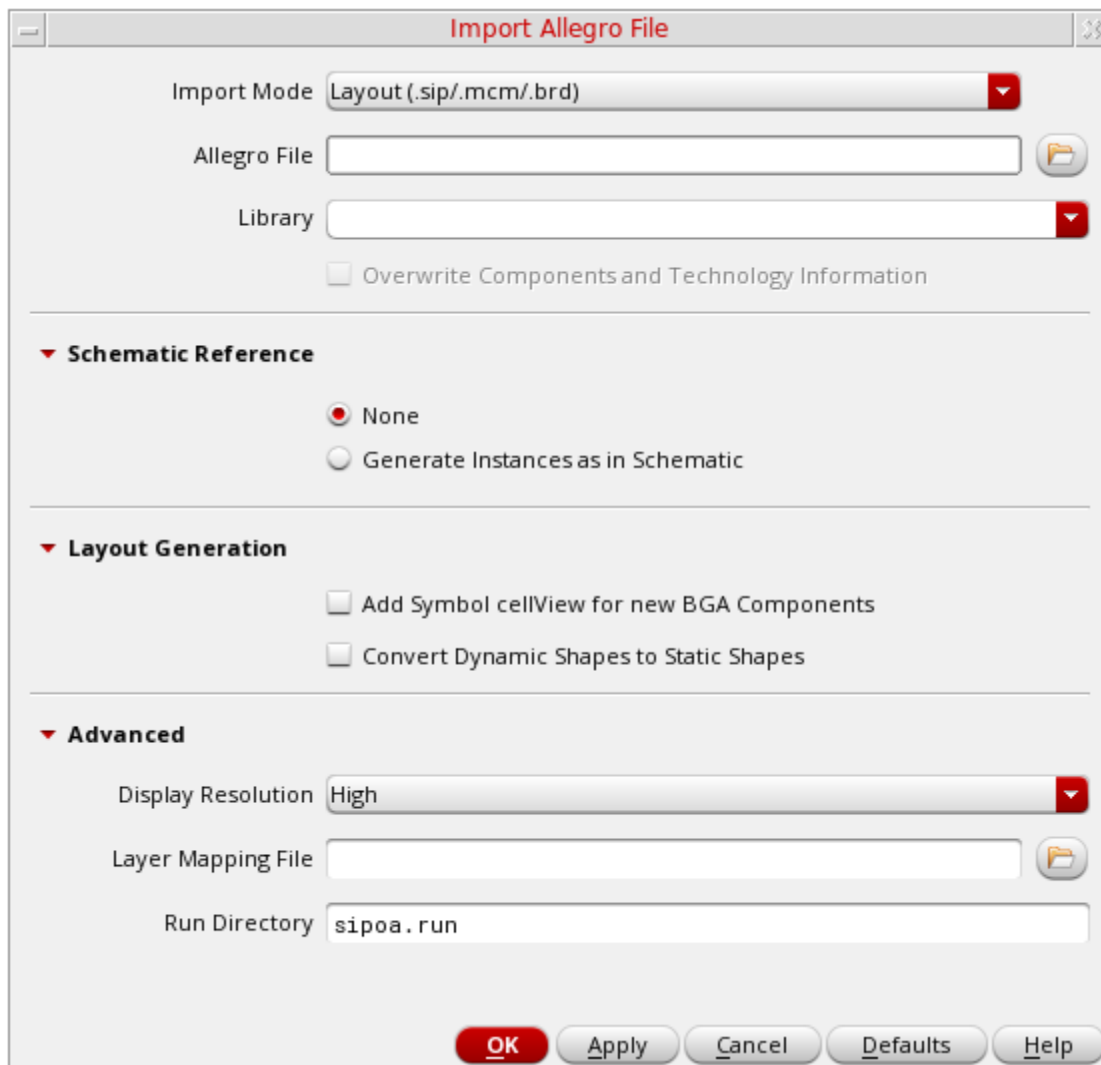
Use the *Import Mode* option to specify the mode that you want to use to import package design files from Allegro:

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

- [Layout\(sip/mcm/brd\)](#)
- [Technology Information \(sip/mcm/brd\)](#)
- [DRA Files](#)

Layout(sip/mcm/brd)



The *Layout(sip/mcm/brd)* option lets you import the design from Allegro into Virtuoso. This is the default option.

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

Allegro File

Specify the *Allegro File* to be imported to Virtuoso. For example, `./sip_files/PKG_TOP4.sip`. You can also use the *File Browser* button to select the Allegro file.

A typical Allegro file contains BGA cells, the vias to be used for package routing, and the technology file used in SiP. The technology file is compiled into the Virtuoso technology database.

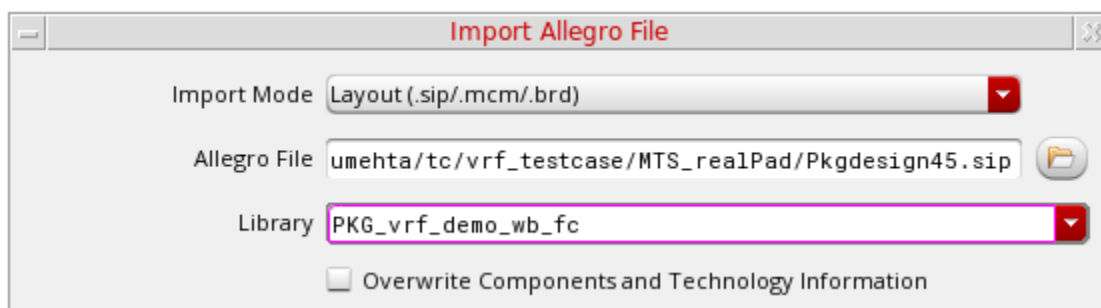
Note: The package technology file is not the same as the Virtuoso technology file.

Library

Use the *Library* field to specify the target library where data is to be translated. The library name should be a logical name and follow the naming conventions supported by OpenAccess. For example, `VRF_PKG_TOP`. You can also select an existing library as the target library from the drop-down list.

Overwrite Components and Technology Information

Select the *Overwrite Components and Technology Information* option to overwrite all the existing information in the target library during translation. This option is disabled by default and is enabled only when you select an existing library file, as shown below:



This option enables you to import files from Allegro in the following two modes:

- **Incremental mode** when the *Overwrite Components and Technology Information* option is deselected.

As stated earlier, not all updates related to a package can be done in Virtuoso. Therefore, you need to frequently import and export package designs between Allegro and Virtuoso. The default incremental mode enables you to update the existing library file every time you import a package design from Allegro.

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

- **Non-incremental mode** when the *Overwrite Components and Technology* is selected.

In this mode, the contents of the selected library file are overwritten during import.

Schematics Reference

Expand the *Schematics Reference* section to specify whether you want to import the schematics-related information in the layout.

- **None**
Specify the None option if you do not want to add schematics references. This is the default value.
- **Generate Instances as in Schematic**
Select the *Generate Instances as in Schematic* option to use an existing package schematic to create layout instances with the layout master of the corresponding schematic instance in Virtuoso.

When this option is specified, only those components from Allegro layout are imported that do not already exist in Virtuoso package schematic.

For example, consider a package schematic with die, D1, and SMD cellviews, R1 and R2, and that D1, R1, and R2 already exist in the schematic in Virtuoso. Now, when you specify the *Generate Instances as in Schematic* option, then D1, R1, and R2 are not imported. Instead, components from the schematic in Virtuoso are instantiated in the layout.

This option is useful in reimporting an Allegro layout that was previously exported from Virtuoso.

Layout Generation

Expand the Layout Generation section to specify how you want to handle some of the layout components during import.

- **Add Symbol cellview for new BGA Components**
Specify the *Add Symbol cellview for new BGA Components* option to create a `symbol` view for the BGA that is being imported. The BGA is detected as the cell with the following configuration:

```
COMPONENTCLASS=IO
```

The `symbol` view of the BGA components can be instantiated in the Package schematic.

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

■ Convert Dynamic Shapes to Static Shapes

Specify the Convert Dynamic Shapes to Static Shapes option to convert dynamic shapes to statics shapes during import. No dynamic layer-purpose pair is created during this process.

Advanced

Expand the *Advanced* section to specify the following additional options.

■ Display Resolution

Use the Display Resolution drop-down list to specify the resolution for the top level cellview. Valid values are *Very High*, *High*, *Medium*, and *Low*. These values can be updated using [Display Options](#) in Virtuoso Layout Suite documentation.

■ Layer Mapping File

Use the *Layer Mapping File* field to specify an existing layer map file. You can also click the *File Browser* button to select the layer map file.

■ Run Directory

Use the *Run Directory* field to specify the default directory for all the files you specify in the Import Allegro File form. The run directory is created automatically (provided you have the appropriate permissions, if it does not exist during Allegro translation).

Default value: Current working directory

Technology Information (sip/mcm/brd)

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

Use the *Technology Information (sip/mcm/brd)* option if you want to import only the technology information, such as stack up and constraints and vias. When you select this option the design information is not updated.

Allegro File

Specify the *Allegro File* to be imported to Virtuoso. For example, `./sip_files/PKG_TOP4.sip`. You can also use the *File Browser* button to select the Allegro file.

A typical Allegro file contains the BGA cells, the vias to be used for package routing, and the technology file used in SiP. The technology file is compiled into the Virtuoso technology database. In this mode, only the technology information and vias are imported.

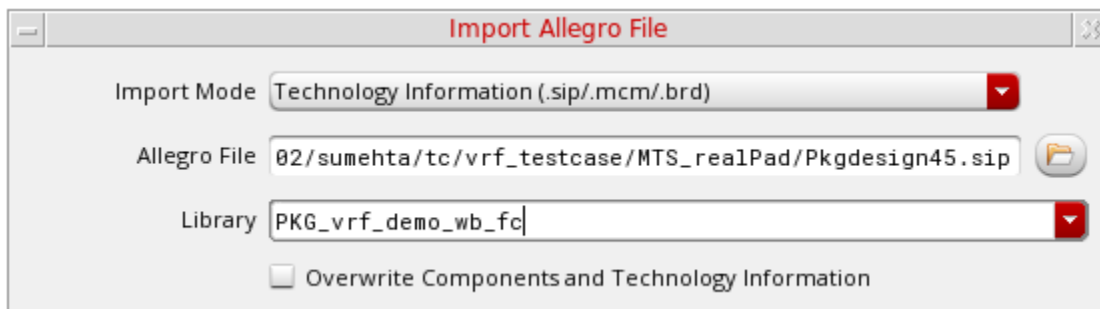
Note: The package technology file is not the same as the Virtuoso technology file.

Library

Use the *Library* field to specify the target library where data is to be translated. The library name should be a logical name and follow the naming conventions supported by OpenAccess. For example, `VRF_PKG_TOP`. You can also select an existing library as the target library from the drop-down list.

Overwrite Components and Technology Information

Select the *Overwrite Components and Technology Information* option to overwrite all the existing information in the target library during translation. This option is disabled by default and is enabled only when you select an existing library file, as shown below:



This option enables you to import files from Allegro in following two modes:

- **Incremental mode** when the *Overwrite Components and Technology Information* option is deselected

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

As stated earlier, not all updates related to a package can be done in Virtuoso. Therefore, you need to frequently import and export package designs between Allegro and Virtuoso. The default incremental mode enables you to update the existing library file every time you import a package design from Allegro.

- **Non-incremental mode** when the *Overwrite Components and Technology* is selected

In this mode, the contents of the selected library file are overwritten during import.

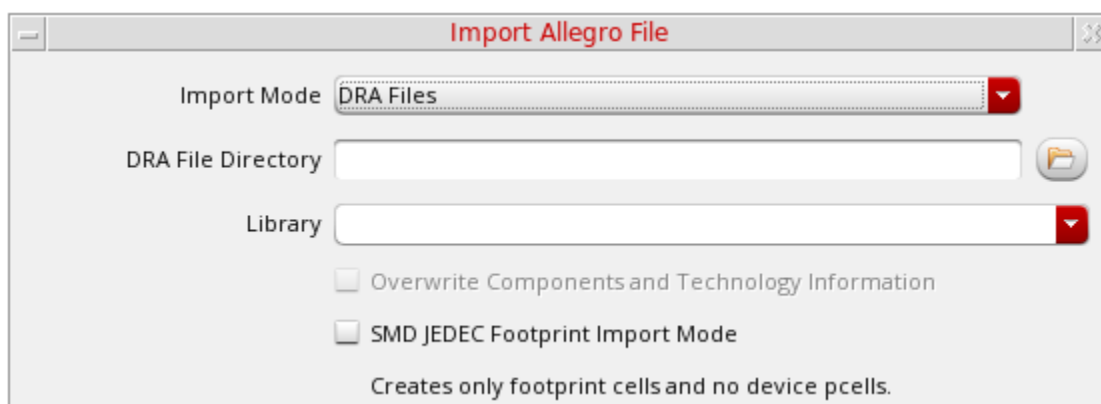
Advanced

Expand the Advanced options to specify the following:

- **Layer Mapping File**
Use the *Layer Mapping File* field to specify an existing layer map file. You can also click the *File Browser* button to select the layer map file.
- **Run Directory**
Use the *Run Directory* field to specify the default directory for all the files you specify in the *Import from Allegro* form. The run directory is created automatically (provided you have the appropriate permissions) if it does not exist during Allegro translation.

Default value: Current working directory

DRA Files



Specify *Import Mode* as *DRA Files* to import multiple DRA files from Allegro. In this mode, a technology independent layout Pcell (TILP) and the base symbol definition (symdef_base) are created in Virtuoso.

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

DRA File Directory

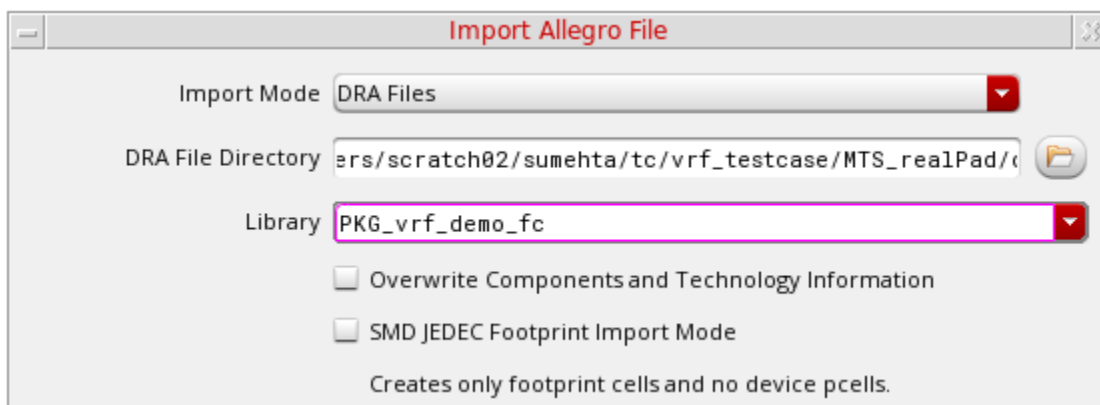
Use DRA File Directory to specify the path to the directory containing the DRA files and padstack definitions in the DRA. You can also click the *Directory Browser* button to select the directory.

Library

Use the *Library* field to specify the target library where a catalog of DRA cellviews will be saved after import. The library name should be a logical name and follow the naming conventions supported by OpenAccess. For example, `VRF_PKG_TOP`. You can also select an existing library as the target library from the drop-down list.

Overwrite Components and Technology Information

Select the *Overwrite Components and Technology Information* option to overwrite all the existing information in the target library during translation. This option is disabled by default and is enabled only when you select an existing library file, as shown below:



This option enables you to import files from the Allegro in following two modes:

- **Incremental mode** when the *Overwrite Components and Technology Information* option is deselected.

As stated earlier, not all updates related to a package can be done in Virtuoso. Therefore, you need to frequently import and export package designs between Allegro and Virtuoso. The default incremental mode enables you to update the existing library file every time you import a package design from Allegro.

- **Non-incremental mode** when the *Overwrite Components and Technology* is selected.

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

In this mode, the contents of the selected library file are overwritten during import.

SMD JEDEC Footprint Import Mode

When the *SMD JEDEC Footprint Import Mode* option is selected, only the base abstract cellview `layout` is created.

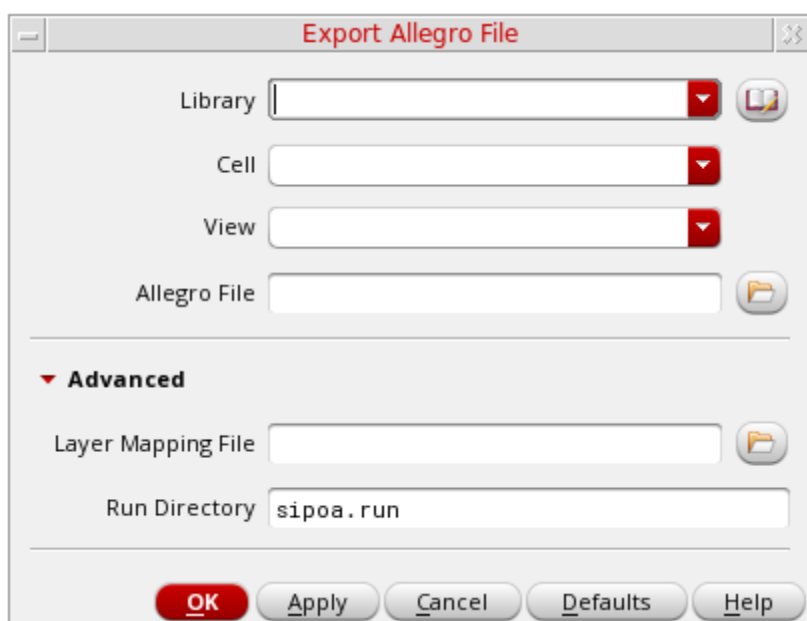
Export Allegro File Form

You can export SiP or MCM files from Virtuoso by using the Allegro Export translator.

After export, a new package file is created from the specified cellview and saved in the Allegro file.

Note: The Allegro Export translator, only exports full files. It does not support incremental export.

To launch Allegro Export translator and display the Export Allegro File form, select the *File – Export – To Allegro* option from the CIW.



The screenshot shows the 'Export Allegro File' dialog box. It has a title bar with the text 'Export Allegro File'. Inside, there are four input fields: 'Library' (with a dropdown arrow and a file icon button), 'Cell' (with a dropdown arrow), 'View' (with a dropdown arrow), and 'Allegro File' (with a file icon button). Below these is a section titled 'Advanced' with a dropdown arrow. Inside the 'Advanced' section, there are two more input fields: 'Layer Mapping File' (with a file icon button) and 'Run Directory' (with the text 'sipoa.run'). At the bottom of the dialog are five buttons: 'OK' (highlighted in red), 'Apply', 'Cancel', 'Defaults', and 'Help'.

This section describes the various options on the Export Allegro File form.

Library

Use the *Library* option to specify a library from which a design is to be translated to the Allegro format. This is a mandatory field. You can also click the *Library Browser* button to select the library from the Library Browser form.

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

Cell

Use the *Cell* option to specify the name of the cell at the top of the hierarchy that you want to translate to the Allegro package file.

View

Use the *View* field to specify the view names that are created in the target library during translation.

Allegro File

Specify the *Allegro File* to which the exported data will be saved. You can also use the *File Browser* button to select the Allegro file.

Allegro Export proceeds only when you select a file with the correct file extension. The type of file depends on the fabric type in the technology database. It is:

- .sip for a package
- .mcm for a module
- .brd for a board

Allegro Export can automatically detect the design type and propose the correct file extension.

Advanced

Expand the *Advanced* option to specify the following additional options:

Layer Mapping File

Use the *Layer Map* field to specify an existing layer map file. You can also click the *File Browser* button to select the layer map file.

Run Directory

The *Run Directory* field accepts the default directory for all the files you specify in the Export Allegro File form. If you want to read from or write to a file that is not present in the run directory, type the full path in the field for the file.

Design Data Translators Reference

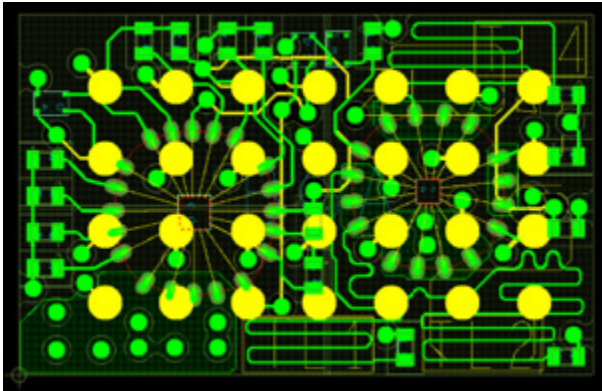
Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

Use the directory browser to select the path of the run directory.

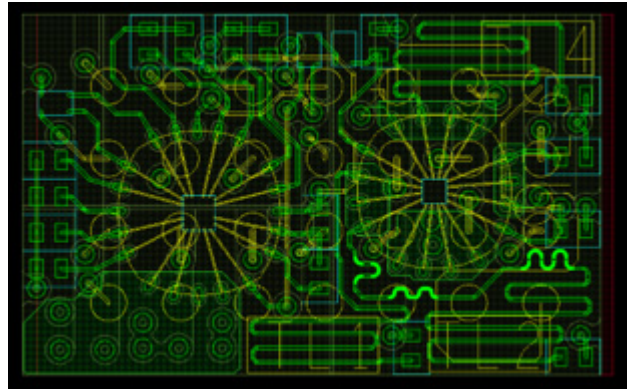
Default value: Current working directory

Design Object Mapping

The following images show a design in Allegro and how it looks when it is translated to Virtuoso:



Design In Allegro

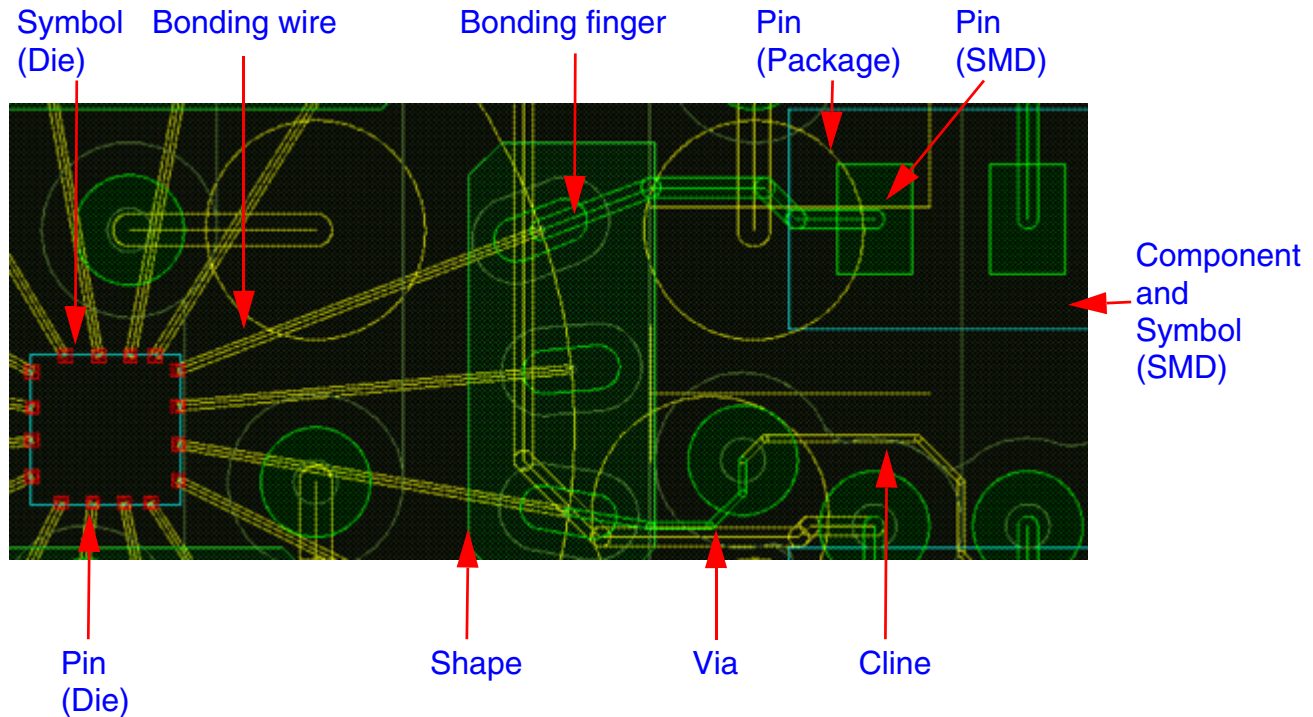


Design in Virtuoso

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

The following image shows some of the Allegro design objects that have been translated to Virtuoso using the Allegro Import translator.



For more information on Allegro package design objects, see [Packaging-Related Terminology](#) in *Virtuoso RF Solution Guide*.

Here, *die* is a wire-bound die component which has an outline and bounding pads.

Attached to the die are bounding wires that are three-dimensional objects and are represented as Pcells in Virtuoso and clines in Allegro.

Bonding fingers are the attachment points on the substrate for the wire bonds. They are pieces of metal, which are represented as separate objects in Allegro and finger instances in Virtuoso.

Next, you can see a *pin of a package*, where a package is a ball grid array (BGA) component, which is the container for the pins that have connections to the board.

Next, a component in the design is the surface mounted device (SMD), which has two pins in the design. At the bottom of the design, you can see some other components, such as pins of the die, shapes, vias, and clines.

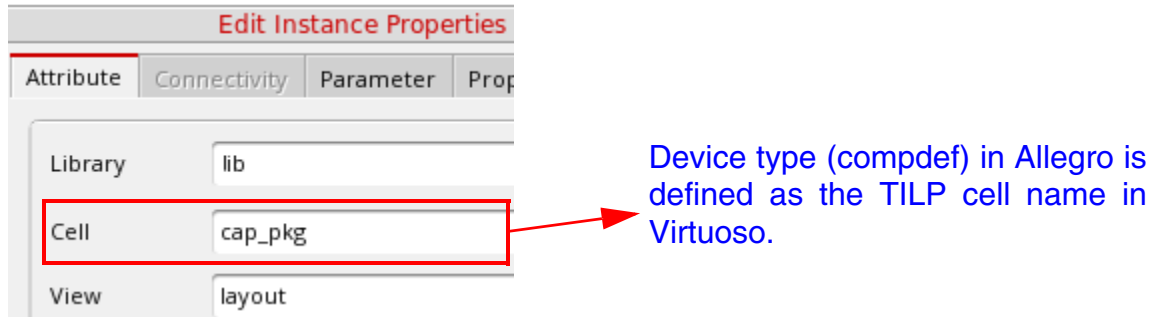
Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

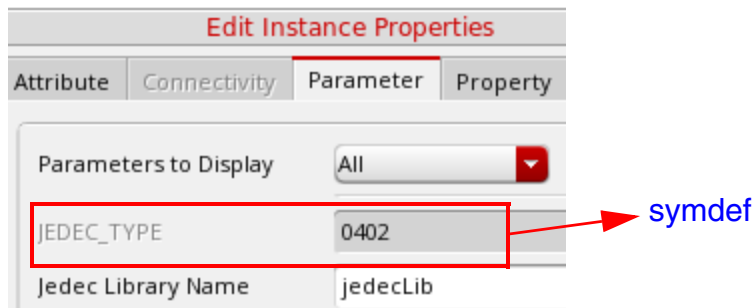
Components and Symbols in SMD

During Allegro Import translation, we translate only the device type, which is the component definition (compdef) of the device. It is specified as the name of the TILPs, which is instantiated in Virtuoso.

In the following screenshot, the name of the cell is the name of the TILP



When a TILP is instantiated it takes several parameters. In case of SMD, one of the parameters is *JEDEC_TYPE*. This represents the name of the package based on its size. For example, in the screenshot below, 0402 represents package of a certain size. In Allegro, this is a symbol, which is a physical device, and in Virtuoso it is the symbol definition or symdef.



In Virtuoso, there is a hierarchy of a TILP, which is the device type, and this Pcell refers to the base cell, which is the symdef.

Translating Parametrized Tline Cells

The Allegro translator preserves the instances of the parametrized `Tline` cells from the `rfTlineLib` library when they are exported to SiP from Virtuoso and then imported back.

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

When the design is exported to SiP, the library/cell/view and the Pcell parameter values of the Tline Pcell are stored in the SIPOA_TLINE property on the Tline symbol in SiP. At the same time, the Tline symbol in SiP is locked to prevent this property from being modified.

When the design is imported from Allegro back to Virtuoso, the value of this property is used to recreate the Tline Pcell as the instance of the cell from rfTlineLib.

Data Model Mapping

Data models in Allegro and Virtuoso have significant differences. The Allegro Import and Export translators bridge these modular differences.

The following table shows how package design objects in Allegro are mapped to objects in Virtuoso:

Package Design Objects in Allegro	Objects created in Virtuoso	Description
design	cellview	In Allegro, design is the top-level view. It is represented by the top cellview in Virtuoso.
symdef	Base cellview	The <i>sipObjectType</i> of the TILP is set as package, SMD, or die.
padstack	Base cellview	Padstacks can be instantiated as pins or vias. The <i>sipObjectType</i> property for a padstack is padstack. The <i>sipPadStackType</i> property is set to via or other, where other corresponds to pins.
compdef	TILP	This is a logical object, where a device type in Allegro is defined as the TILP cell name in Virtuoso. Therefore, the TILP name is the same as the name of the device name in Allegro. Note: Other components, such as functions and function pins are not modeled in Virtuoso.

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

Package Design Objects in Allegro	Objects created in Virtuoso	Description
symbol	TILP instance	Instance of a physical device in Allegro. In SMD, the symdef cell name is defined as <i>JEDEC_TYPE</i>, which will have the name of the actual base cell defining the symbol. If this is not an SMD and is a package or a die, then the <i>baseCellViewName</i> property is defined, which gives the name of the symbol definition.
component	TILP instance	Same as the symbol TILP instance
pin	TILP instance pin	Pins are instances of a padstack in Allegro and are allowed only inside symbols. In Virtuoso they are instance pins with terminals, instantiating a padstack. The definition for pins is added to the symbol definitions of cellviews representing symdef. Padstack view is defined in the <i>baseCellViewName</i> property.
via	TILP via instance	Vias are instances of a padstack Allegro and can be placed anywhere in the design. Padstack view is defined in the <i>baseCellViewName</i> property. Vias also need a customViaDef in the technology database.
Bonding Finger	pkgLib/finger instance	Bonding fingers are instances of padstack Allegro. In Virtuoso, they are Pcells with several parameters such as shape type and dimension. They can also refer to an existing Pcell.

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

Package Design Objects in Allegro	Objects created in Virtuoso	Description
Bonding Wire	pkgLib/wirebond instance	Bonding wires are represented as Pcells in Virtuoso with two attachments as the parameters.
Cline	path	Clines are mapped to a curved path in Virtuoso.
Shape	curved polygon	Shapes, including rectangles and circles, are specified as curved polygons in Virtuoso.
Text	Label/Curved Paths	Text in Allegro can be translated as either labels or geometries in Virtuoso. Geometries are created by converting text to curved paths in figGroups.
Line	Path/Line	Specified as non-metal layers with non-zero or zero width

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

Layer Mapping

Cross-section layers in Allegro are mapped to technology layers with functions: metal, dielectric, and diestack.

Non-cross-section layers are mapped according to the definitions in the technology library, cdsDefTechLib.

Layer type	Type in Allegro	Type in Virtuoso
Solder Mask	SOLDERMASK_TOP SOLDERMASK_BOTTOM	solderMaskTop, solderMaskBottom
Place bound top	PLACE_BOUND_TOP PLACE_BOUND_BOTTOM	prBoundary
DFA bound top	DFA_BOUND_TOP DFA_BOUND_BOTTOM	assemblyBoundary
Substrate geometry	SUBSTRATE_GEOMETRY/ WB_GUIDE_LINE	wirebondFingerGuide
Customer-defined layers	COMPONENT_GEOMETRY SILKSCREEN_TOP	COMPONENT_GEOMETRY SILKSCREEN_TOP In this case, Class name is mapped to purpose name Subclass is mapped to layer name

Troubleshooting Allegro File Translation Issues

This section discusses issues that you might face during translation of Allegro files and shows how to either resolve or workaround them.

Allegro Import translation of an SIP file fails with an error in dbCreateVia

This issue occurs when layers of type `conductor` are defined outside the substrate. This causes issues because layers outside the substrate can be only of type `diestack` or `dielectric`.

In this case, if you open the Layer Stack Editor, Allegro Import translator will issue another warning indicating that something is wrong with the layer stack definition.

To resolve this issue, fix the SiP design using the `config substrate layers` command at the SiP command prompt.

Missing instances after mirrored rotation

In Allegro, a user is allowed to change elements in the instantiation of a symbol definition. In contrast, it is not possible to edit the instance of a master in Virtuoso.

For example consider a text element that is rotated on some or all of the instances by 180 degrees. The rotation parameter of the instance is correctly translated by Allegro Import translator and all the other parameters (like mirrored) are also correctly propagated. However, the imported instance does not match the component definition any longer because the instance was edited earlier.

In most cases, the translator detects these cases and creates a variant. Examples of such changes are symbol rotation, mirroring, and flipping. Other changes like changes to specific objects inside the symbol instance cannot be detected and result in mismatches between the Allegro view of the geometry within the edited symbol and the Virtuoso view of the underlying master.

To resolve this issue, the variants must be manually created in Allegro by associating the symbols with new symbol definitions containing the edited geometries.

Finger instances in Virtuoso are exported to Allegro as padstacks

When you export instances of bond fingers from Virtuoso to Allegro, they are always created as padstack instances in the SIP layout. The shape type in Virtuoso is ignored. When such bond fingers are imported from Allegro back to Virtuoso, their layout is restored, but their

Design Data Translators Reference

Package Design Translation Using Allegro Translators (Virtuoso MultiTech Framework)

shape type becomes padstack regardless of their previous shape type. The exported bond finger shape type can only be restored to its original shape type manually by defining all the required parameters, such as `shapeType`, `width`, `length`, `layer`, `diameter`, and `soldermask` for each bond finger instance. Additionally, it needs to be ensured that the `shapeType` and other parameters stored on the finger match the definition of the padstack the finger instance is referencing. If the change of shape type when exporting and importing bond fingers to and from Allegro is not desired, it can be avoided by defining their `shapeType` as padstack.

Priority of dynamic layers are changes after Allegro Export and Allegro Import round-trip

If you change priority of a dynamic shape in Virtuoso and perform Allegro Import and Allegro Export round-trip, the changed priority number of the dynamic shape is not retained.

This is an expected behavior that happens because the priority number is relative and not absolute. Allegro PCB Editor adjusts the numbers, as necessary. You should only use the priority number for comparison with priority numbers of other dynamic shape.

Translating Shapes with Fillet Properties

Shapes with additional fillet properties can be automatically deleted and recreated by the fillet generation functions either in Virtuoso or in Allegro. These fillet properties are preserved during Allegro Import and Export round trip.

Stream Format

This chapter contains the following sections:

- [Overview](#) on page 656
- [Stream File Description](#) on page 657
- [Differences in Stream Format Versions](#) on page 675
- [Stream Syntax](#) on page 675
- [Example of a Stream Format File](#) on page 677
- [How to read, write and print Stream dates to avoid Y2K problems](#) on page 686

Overview

When you want to transfer Virtuoso data to other systems, the best format to translate your data to is Stream format. This appendix describes the Stream format used in the files you can produce and read using the *Stream In* and *Stream Out* commands from the *Translators* menu in the CIW.

You can transfer libraries preserved in Stream format to other systems for processing. In addition, Stream format is upward compatible, which means newer releases of the Stream translators can read libraries produced with an older release.

For complete descriptions of the options and arguments for *LEF In*, *Stream Out*, refer to [Chapter 2, “Design Translation Using XStream Translator”](#).

Although you can use the `Stream In` and `Stream Out` commands to translate your Stream data, you might want to directly edit your Stream file, or write programs or scripts that manipulate your Stream data.

Note: Portions of this appendix describe features and data types that were applicable to some older tools (like GDSII Custom Plus and Construct systems). These features might not apply to the Virtuoso system.

About GDSII

History

The GDSII format was invented at Calma Co. in the early 1970s, which is now a part of Cadence Design Systems, Inc. Therefore, GDSII is today a registered trademark of Cadence.

GDSII Format

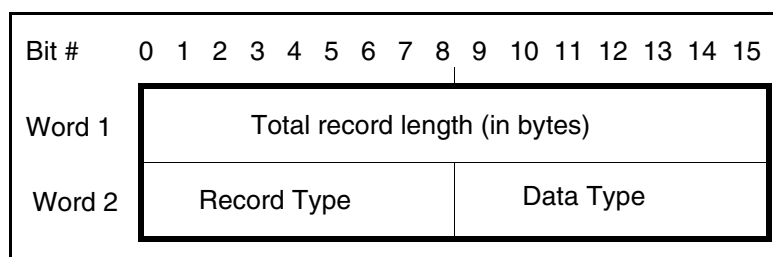
GDSII is the open-standard Stream file format for transferring or archiving two-dimensional graphical design data. It is a binary, platform-independent format. You will read more about the Stream format in the subsequent sections. For a sample of the GDSII file, see [Example of a Stream Format File](#) on page 677.

Stream File Description

The information stored in a Stream file is coded in variable-length records. The length of a record is measured in eight-bit bytes. The minimum record length is four bytes. There is no limit on record length. Within the record, two bytes (16 bits) is a *word*. The 16 bits in a word are numbered 0 to 15, left to right.

The first four bytes of a record are the *header*. The first two bytes of the header specify how many eight-bit bytes the record contains. The third byte of the header specifies the record type. The fourth byte of the header specifies the type of data contained within the record. The fifth through last bytes of the record are data. The next record begins immediately after the last byte in the record.

The following figure shows a typical record header:



If the Stream file is on a magnetic tape, the records of the library are usually divided in 2048-byte physical blocks. Records can overlap physical block boundaries; a record is not required to be wholly contained in a single physical block.

Two consecutive zero bytes are a *null word*. You can use null words to fill the space between the last record of a library and the end of its physical block.

Stream records are always an even number of bytes. If a record contains ASCII string data and the ASCII string is an odd number of bytes, the last character is a null character.

Stream Data Types

The following table lists the Stream data types and their values. The data type is specified by the fourth byte of the record.

Data Type	Value
No data present	0

Design Data Translators Reference

Stream Format

Data Type	Value
Bit array	1
Two-byte signed integer	2
Four-byte signed integer	3
Four-byte real	4 (not used)
Eight-byte real	5
ASCII string	6

The following paragraphs describe these Stream data types. As a reminder, a word consists of 16 bits, numbered 0 to 15, left to right.

■ Bit array (1)

A bit array is a word that contains bits, or a group of bits, that represent data. A bit array allows one word to contain more than one piece of information.

■ Two-byte signed integer (2)

2-byte integer = 1 word 2s-complement representation

The range of two-byte signed integers is -32,768 to 32,767.

The following is a representation of a two-byte integer, where *S* is the sign and *M* is the magnitude.

SMMMMMM MMMMMMMM

The following are examples of two-byte integers:

```
00000000 00000001 = 1
00000000 00000010 = 2
00000000 10001001 = 137
11111111 11111111 = -1
11111111 11111110 = -2
11111111 01110111 = -137
```

■ Four-byte signed integer (3)

4-byte integer = 2 word 2s-complement representation

The range of four-byte signed integers is -2,147,483,648 to 2,147,483,647.

The following is a representation of a four-byte integer, where *S* is the sign and *M* is the magnitude.

SMMMMMM MMMMMMMM MMMMMMMM MMMMMMMM

The following are examples of four-byte integers:

Design Data Translators Reference

Stream Format

```
00000000 00000000 00000000 00000001 = 1
00000000 00000000 00000000 00000010 = 2
00000000 00000000 00000000 10001001 = 137
11111111 11111111 11111111 11111111 = -1
11111111 11111111 11111111 11111110 = -2
11111111 11111111 11111111 01110111 = -137
```

■ Four-byte real (4) and eight-byte real (5)

4-byte real = 2-word floating point representation

8-byte real = 4-word floating point representation

For all nonzero values:

A floating point number has three parts: the sign, the exponent, and the mantissa.

- ❑ The value of a floating point number is defined as
(Mantissa) x (16 raised to the true value of exponent field).
- ❑ The exponent field (bits 1-7) is in Excess-64 representation. The field shows a number 64 greater than the exponent.
- ❑ The mantissa is always a positive fraction greater than or equal to 1/16 and less than 1. For a 4-byte real, the mantissa is bits 8 through 31. For an 8-byte real, the mantissa is bits 8 through 63. The binary point is just to the left of bit 8. Bit 8 represents the value 1/2, bit 9 represents 1/4, and so on.

To keep the mantissa in the range of 1/16 to 1, the results of floating point arithmetic are *normalized*. Normalization is a process that shifts the mantissa left one hex digit at a time until its left FOUR bits represent a non-zero quantity. For every hex digit shifted, the exponent is decreased by one. Since the mantissa is shifted four bits at a time, it is possible for the left three bits of the normalized mantissa to be zero. A zero value, also called *true zero*, is represented by a number with all bits zero.

The following are representations of 4-byte and 8-byte reals, where *S* is the sign, *E* is the exponent, and *M* is the magnitude. Examples of 4-byte reals are included on the following pages, although 4-byte reals are not used currently. The representation of the negative values of real numbers is the same as the positive, except that the highest order bit is 1, not 0.

In the eight-byte real representation, the first four bytes are the same as in the four-byte real representation. The last four bytes contain additional binary places for higher resolution.

4-byte real:

SEEEEEEE MMMMMMMM MMMMMMMM MMMMMMMM

8-byte real:

Design Data Translators Reference

Stream Format

SEEEEEEE MMMMMMMM MMMMMMMM MMMMMMMM
MMMMMMMM MMMMMMMM MMMMMMMM MMMMMMMM

Examples of 4-byte real:

In the first six lines of the following example, the 7-bit exponent field is 65. The exponent is $65-64=1$.

01000001	00010000	00000000	00000000	= 1
01000001	00100000	00000000	00000000	= 2
01000001	00110000	00000000	00000000	= 3
11000001	00010000	00000000	00000000	= -1
11000001	00100000	00000000	00000000	= -2
11000001	00110000	00000000	00000000	= -3
01000000	10000000	00000000	00000000	= 0.5
01000000	10011001	10011001	10011001	= 0.6
01000000	10110011	00110011	00110011	= 0.7
01000001	00011000	00000000	00000000	= 1.5
01000001	00011001	10011001	10011001	= 1.6
01000001	00011011	00110011	00110011	= 1.7
00000000	00000000	00000000	00000000	= 0
01000001	00010000	00000000	00000000	= 1
01000001	10100000	00000000	00000000	= 10
01000010	01100100	00000000	00000000	= 100
01000011	00111110	10000000	00000000	= 1000
01000100	00100111	00010000	00000000	= 10000
01000101	00011000	01101010	00000000	= 100000

■ ASCII string (6)

A collection of bytes representing ASCII characters. All odd-length strings are padded with a null character (the number zero), and the byte count for the record containing the ASCII string includes this null character. If you write a program to read Stream data, the program must check for the null character and, if present, decrease the length of the string by one.

Stream Records

This section describes the records that make up a Stream file. The descriptions include the following:

- Record number
- Record name
- Data type
- The four digits that make up the second word in the record, divided as follows:
 - The first two digits are the hexadecimal value of the record number.
 - The second two digits are the numeric value for the data type.

Design Data Translators Reference

Stream Format

- A description of the information contained in the record.

0	HEADER	Two-Byte Signed Integer.
[0002]		Contains the Stream version number.
1	BGNLIB	Two-Byte Signed Integer.
[0102]		This record type marks the beginning of a library. It contains the last modification time of a library (one word each for years since 1900, month, day, hour, minute, and second), the time of last access (same format). The time fields correspond with the contents of a struct tm, except for the month, which is (tm->tm_mon + 1). See the table for the BGNLIB description in Example of a Stream Format File on page 677, and How to read, write and print Stream dates to avoid Y2K problems on page 686.

Design Data Translators Reference

Stream Format

The following figure shows the meaning of each word.

Bit #	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Word 1	1C (hex) # of bytes in record															
Word 2	01 (hex)								02 (hex)							
Word 3	year (last modification time)															
Word 4	month															
Word 5	day															
Word 6	hour															
Word 7	minute															
Word 8	second															
Word 9	year (last access time)															
Word 10	month															
Word 11	day															
Word 12	hour															
Word 13	minute															
Word 14	second															

2 LIBNAME

[0206]

ASCII String

Contains the library name. The library name must follow UNIX filename conventions for length and valid characters. The library name can include the file extension (.sf or.db in most cases).

3 UNITS

[0305]

Eight-Byte Real.

Contains two numbers indicating the number of user units in a database unit and the size of a database unit in meters. Typically, the number of user units in a database unit is less than 1 because you use more than 1 database unit per user unit. To calculate the size of a user unit in meters, divide the second number by the first.

Design Data Translators Reference

Stream Format

For example, if you create a library using the default unit (user unit = 1 micron and 1000 database units per user unit), the first number is .001 and the second number is $1E-9$.

4	ENDLIB	No Data Present.
	[0400]	Marks the end of a library.
5	BGNSTR	Two-Byte Signed Integer.
	[0502]	Contains the creation time and last modification time of a structure (in the same format as the BGNLIB record), and marks the beginning of a structure.
6	STRNAME	ASCII String.
	[0606]	Contains the structure name. The limit of size of string in STRNAME record in Stream format is may be up to 32 characters. This limit has been removed. Legal structure name characters are: A through Z a through z 0 through 9 Underscore (_) Question mark (?) Dollar sign (\$)
7	ENDSTR	No Data Present
	[0700]	Marks the end of a structure.
8	BOUNDARY	No Data Present
	[0800]	Marks the beginning of a boundary element.
9	PATH	No Data Present
	[0900]	Marks the beginning of a path element.
10	SREF	No Data Present
	[0A00]	Marks the beginning of an SREF (structure reference) element.
11	AREF	No Data Present
	[0B00]	Marks the beginning of an AREF (array reference) element.
12	TEXT	No Data Present

Design Data Translators Reference

Stream Format

[0C00]	Marks the beginning of a text element.
13 LAYER	Two-Byte Signed Integer
[0D02]	Specifies the layer number. The value of the layer must be in the range of 0 to 32767.
14 DATATYPE	Two-Byte Signed Integer
[0E02]	Specifies the data type. The value of the datatype must be in the range of 0 to 32767.
15 WIDTH	Four-Byte Signed Integer
[0F03]	Specifies the width of a path in database units. If the value for the width is negative, the width is absolute and is not affected by the magnification factor of any parent reference. If this record type is omitted, the default is zero.
16 XY	Four-Byte Signed Integer
[1003]	Contains an array of the XY coordinates for path, boundary, text, contact, SREF, node, box, and AREF elements. The coordinates are in database units. Each X or Y coordinate is four bytes long.

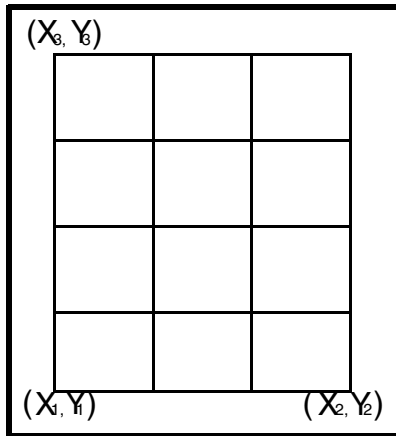
The elements have the following number of coordinates:

- Path elements have a minimum of 2 and a maximum of 8000 coordinates.
- Boundary elements can have a minimum of 4 and a maximum of 8000 coordinates. The first and last coordinates must coincide.
- A text, contact, or SREF element can have only one coordinate.
- A node can have from 1 to 50 coordinates.
- A box must have five coordinates, with the first and last coordinates coinciding.
- An AREF has three coordinates. In an AREF, the first coordinate is the array reference point (origin point). The second coordinate locates a position that is displaced from the reference point by the inter-column spacing times the number of columns. The third

Design Data Translators Reference

Stream Format

coordinate locates a position that is displaced from the reference point by the inter-row spacing times the number of rows. The following is an example of an array lattice.



17	ENDEL	No Data Present
[1100]		Marks the end of an element.
18	SNAME	ASCII String
[1206]		Contains the name of a referenced structure. See also STRNAME.
19	COLROW	Two-Byte Signed Integer
[1302]		Contains four bytes. The first two bytes contain the number of columns in the array. The third and fourth bytes contain the number of rows. The number of columns and the number of rows must be in the range 1 to 32,767 (decimal).
20	TEXTNODE	No Data Present (Not currently used)
[1400]		Marks the beginning of a text node.
21	NODE	No Data Present
[1500]		Marks the beginning of a node.
22	TEXTTYPE	Two-Byte Signed Integer
[1602]		Specifies the text type. The value of the text type must be in the range 0 to 255.
23	PRESENTATION	Bit Array
[1701]		Specifies how text is presented.

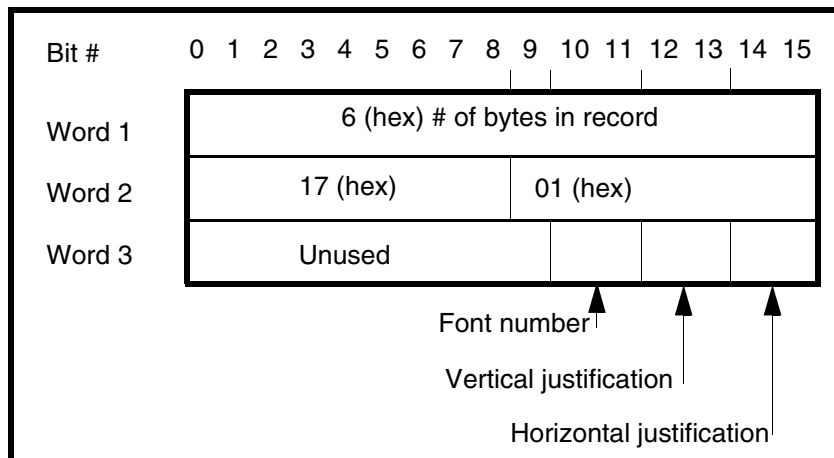
Design Data Translators Reference

Stream Format

The bits in the bit array have the following values:

- Bits 10 and 11, used together as a binary number, specify the font (00 is font 0, 01 is font 1, 10 is font 2, and 11 is font 3).
- Bits 12 and 13 specify the vertical justification (00 means top, 01 means middle, and 10 means bottom).
- Bits 14 and 15 specify the horizontal justification (00 means left, 01 means center, and 10 means right).
- Bits 0 through 9 are reserved for future use and must be cleared.

If this record is omitted, top-left justification and font 0 are the default values. The following shows a PRESENTATION record.



24	SPACING	Not currently used
25	STRING	ASCII String
[1906]		Contains up to 512 characters of text to present.
26	STRANS	Bit Array
[1A01]		Contains two bytes of bit flags for SREF, AREF, and text transformation. Bit 0 (the leftmost bit) specifies reflection.

The bits in the bit array have the following values:

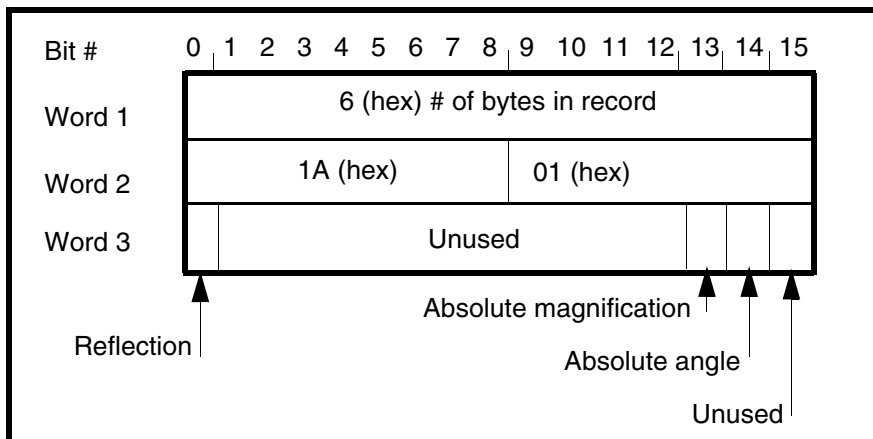
- If bit 0 is set, the element is reflected about the X-axis before angular rotation. For an AREF, the entire array is reflected with the individual array members rigidly attached.
- Bit 13 flags absolute magnification.

Design Data Translators Reference

Stream Format

- Bit 14 flags absolute angle.
- Bits 1 to 12 and 15 are reserved for future use and must be clear.

If this record is omitted, the defaults for the element are no reflection, non-absolute magnification, and non-absolute angle. The following shows an STRANS record.



27 MAG

[1B05]

Eight-Byte Real

Contains the magnification factor. If this record is omitted, the default magnification factor is one.

28 ANGLE

[1C05]

Eight-Byte Real

Contains the angular rotation factor. The angle of rotation is measured in degrees and in the counterclockwise direction. For an AREF, the ANGLE rotates the entire array (with the individual array members rigidly attached) about the array reference point. If this record is omitted, the default angle is zero degrees.

29 UINTEGER

Not currently used. User Integer data was used in GDSII Release 2.0 only.

30 USTRING

Not currently used. User String data, formerly called character string data (CSD), was used in GDSII Releases 1.0 and 2.0.

31 REFLIBS

ASCII String

Design Data Translators Reference

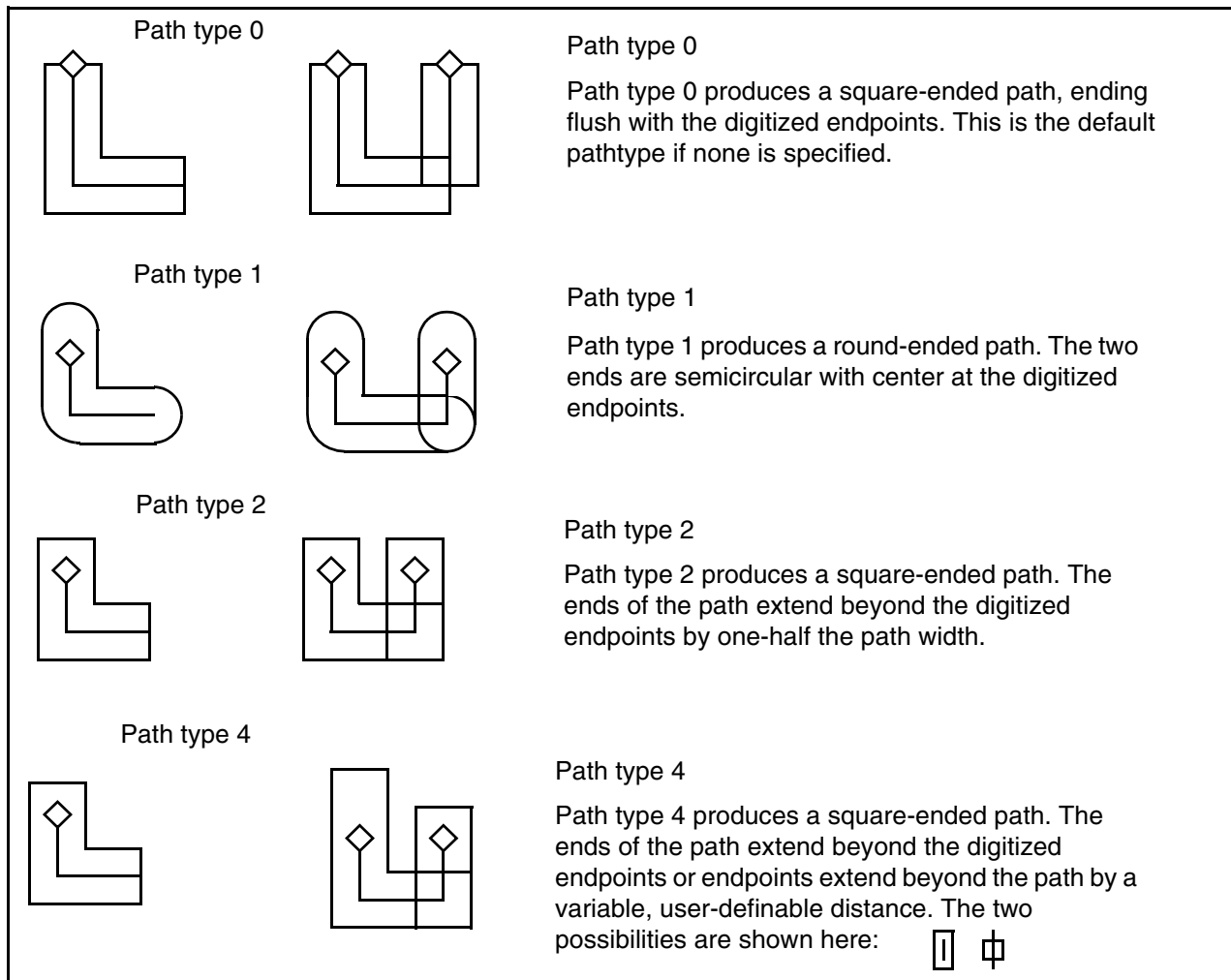
Stream Format

[1F06]	Contains the names of the reference libraries. This record must be present if any reference libraries are bound to the working library. The name of the first reference library starts at byte 5 (immediately following the record header) and continues for 44 bytes. The next 44 bytes contain the name of the second library. The record is extended by 44 bytes for each additional library (up to 15) which is bound for reference. The reference library names may include directory specifiers (separated with "/") and an extension (separated with "."). If either the first or second library is not named, its place is filled with nulls.
32 FONTS	ASCII String
[2006]	Contains the names of the text font definition files. This record must be present if any of the four fonts have a corresponding text font definition file. This record must not be present if none of the fonts have a text font definition file. The filename for text font 0 starts the record, followed by the filenames for the remaining three fonts. Each filename is 44 bytes long. The filename is padded with nulls if the name is shorter than 44 bytes. The filename is null if no filename corresponds to the font. The filenames may include directory specifiers (separated with "/") and an extension (separated with ".").
33 PATHTYPE	Two-Byte Signed Integer
[2102]	Contains a value indicating the type of path endpoints. The value is 0 for square-ended paths that end flush with their endpoints, 1 for round-ended paths, 2 for square-ended paths that extend a half-width beyond their endpoints, and 4 for paths with variable square-ended extensions (see records 48 and 49). If not specified, assumes a type of 0.

Design Data Translators Reference

Stream Format

The following illustration shows the path types.



34 GENERATIONS

[2202]

Two-Byte Signed Integer

Contains the number of copies of deleted or back-up structures to retain. This number must be at least 2 and not more than 99. If the GENERATIONS record is omitted, the default value is 3.

35 ATTRTABLE

[2306]

ASCII String

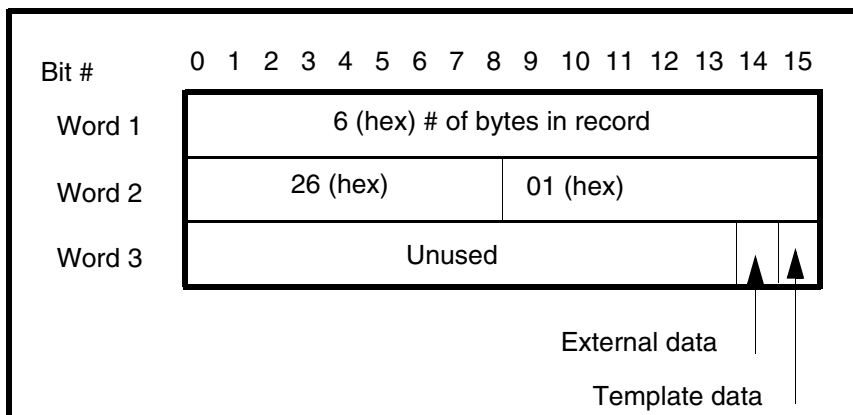
Contains the name of the attribute definition file. This record is present only if an attribute definition file is bound to the library. The attribute definition filename can include directory specifiers (separated with "/") and an extension (separated with "."). The maximum record size is 44 bytes.

Design Data Translators Reference

Stream Format

36 STYPTABLE	ASCII String (Unreleased feature)
[2406]	
37 STRTYPE	Two-Byte Signed Integer (Unreleased feature)
[2502]	
38 ELFLAGS	Bit Array
[2601]	Contains two bytes of bit flags. Bit 15 (the rightmost bit) specifies Template data. Bit 14 specifies External data (also referred to as Exterior data). All other bits are currently unused and must be cleared to 0. If this record is omitted, the default value for all bits is 0.

The following illustration shows an ELFLAGS record.



39 ELKEY	Four-Byte Signed Integer (Unreleased feature)
[2703]	
40 LINKTYPE	Two-Byte Signed Integer (Unreleased feature)
[28]	
41 LINKKEYS	Four-Byte Signed Integer (Unreleased feature)
[29]	
42 NODETYPE	Two-Byte Signed Integer
[2A02]	Contains a value indicating the node type. The value of the node type must be in the range of 0 to 255.

Design Data Translators Reference

Stream Format

43 PROPATTR [2B02]	Two-Byte Signed Integer Contains the attribute number. The attribute number is an integer from 1 to 127. Attribute numbers 126 and 127 are reserved for the user integer and user string (CSD) properties which were used prior to Release 3.0.
44 PROPVALUE [2C06]	ASCII String Contains the string value associated with the attribute named in the preceding PROPATTR record. The maximum string length is 126 characters. The attribute-value pairs associated with any one element must all have distinct attribute numbers. Also, the total amount of property data that can be associated with any one element is limited: the total length of all the strings, plus twice the number of attribute-value pairs, must not exceed 128. (Or, if the element is an SREF, AREF, contact, nodeport, or node, the length must not exceed 512.) For example, if a boundary element uses property attribute 2 with property value “metal,” and property attribute 10 with property value “property,” the total amount of property data is 18 bytes. This is 6 bytes for “metal” (odd-length strings must be padded with a null) plus 8 for “property” plus 2 times the 2 attributes (4), which equals 18.
45 BOX [2D00]	No Data Present Marks the beginning of a box element.
46 BOXTYPE [2E02]	Two-Byte Signed Integer Contains a value indicating the box type. The value of the box type must be in the range of 0 to 255.
47 PLEX [2F03]	Four-Byte Signed Integer A unique positive number that is common to all elements of the plex to which this element belongs. The head of the plex is flagged by setting the seventh bit; therefore, plex numbers should be small enough to occupy only the rightmost 24 bits. If this record is not present, the element is not a plex member.
48 BGNEXTN [3003]	Four-Byte Signed Integer Applies to Pathtype 4. Contains four bytes that specify in database units the distance a path outline begins before or after the first point of the path. This value can be negative.

Design Data Translators Reference

Stream Format

49 ENDEXTN	Four-Byte Signed Integer
[3103]	Applies to Pathtype 4. Contains four bytes that specify in database units the distance a path outline begins before or after the last point of the path. This value can be negative.
50 TAPENUM	Two-Byte Signed Integer
[3202]	Contains the number of the current reel of tape for a multi-reel Stream file. For the first tape, the tape number is 1; for the second tape, the tape number is 2. For each additional tape, increment the tape number by one.
51 TAPECODE	Two-Byte Signed Integer
[3302]	Contains a unique, 6-integer code common to all reels of a multi-reel Stream file. It verifies that the correct reels are being read.
52 STRCLASS	Two-Byte Bit Array (Cadence internal use only)
[3401]	If Stream tapes are produced by non-Cadence programs, this record should either be omitted or cleared to zero.
53 RESERVED	Not currently used
[3503]	This record type was used for NUMTYPES but was not required.
54 FORMAT	Two-Byte Signed Integer

Design Data Translators Reference

Stream Format

[3602]	This optional record defines the format of a Stream tape in two bytes. The possible values are: 0 for GDSII Archive format 1 for GDSII Filtered format 2 for EDSIII Archive format 3 for EDSIII Filtered format An Archive Stream file contains elements for all the layers and data types. In an Archive Stream file, the FORMAT record is followed immediately by the UNITS record. A file that does not have the FORMAT record is assumed to be an Archive file. A Filtered Stream file contains only the elements on the layers and with the datatypes you specify during creation of the Stream file. The list of layers and datatypes specified appear in MASK records. At least one MASK record must immediately follow the FORMAT record. The MASK records are terminated with the ENDMASKS record.
55 MASK	ASCII String
[3706]	This record is required for and present only in Filtered Stream files. It contains the list of layers and data types specified when the file was created. At least one MASK record must immediately follow the FORMAT record. More than one MASK record can occur. The last MASK record is followed by the ENDMASK record. In the MASK list, datatypes are separated from the layers with a semicolon. Individual layers or datatypes are separated with a space. A range of layers or datatypes is specified with a dash. An example MASK list looks like this: 1 5-7 10 ; 0-255
56 ENDMASKS	No Data Present
[3800]	This record is required for and present only in Filtered Stream files. It marks the end of the MASK records. The ENDMASKS record must follow the last MASK record. ENDMASKS is immediately followed by the UNITS record.
57 LIBDIRSIZE	Two-Byte Signed Integer
[3902]	Contains the number of pages in the library directory.

Design Data Translators Reference

Stream Format

58 SRFNAME	ASCII String
[3A06]	Contains the name of the spacing rules file, if one is bound to the library.
59 LIBSECUR	Two-Byte Signed Integer
[3B02]	Contains an array of Access Control List (ACL) data. Each ACL entry consists of a group number, a user number, and access rights. Up to 32 ACL entries can be present.
60 BORDER	No Data Present
[3C00]	Marks the beginning of a border element.
61 SOFTFENCE	No Data Present
[3D00]	Marks the beginning of a soft fence element.
62 HARDFENCE	No Data Present
[3E00]	Marks the beginning of a hard fence element.
63 SOFTWARE	No Data Present
[3F00]	Marks the beginning of a soft wire element.
64 HARDWARE	No Data Present
[4000]	Marks the beginning of a hard wire element.
65 PATHPORT	No Data Present
[4100]	Marks the beginning of a path port element.
66 NODEPORT	No Data Present
[4200]	Marks the beginning of a node port element.
67 USERCONSTRAINT	No Data Present
[4300]	Marks the beginning of a user constraint.
68 SPACER ERROR	No Data Present
[4400]	Marks the beginning of a spacer error.
69 CONTACT	No Data Present
[4500]	Marks the beginning of a contact element.

Note: The record types 60-69 were used by the GDSII application called Custom Plus (aka Sticks) earlier. They are not used by the Virtuoso system.

Differences in Stream Format Versions

There are five available versions of Stream format, version 3, 4, 5, 6 and 7. Stream versions 4 and 5 are the same. The records listed in the above section are a combined list of all the records supported by the various Stream formats. The various versions provide support for a subset of records present in this list.

The Stream format is usually broken down into 4 versions:

- Stream format version 3 supports basic geometric information. From the list of supported Stream records, Stream 3.0 only supports up to record 44. See [Stream Records](#) on page 660.
- Stream format version 4 or 5 provides support for geometric information along with supporting new types in gdsII 4, such as nodes, ports, template, plex data. From the list of supported Stream records, version 5.2 supports up to record 56.
- Stream format 6, has some additional information for a product called “Custom Plus” that includes things like “attach file” and “lib access control lists”. From the list of supported Stream records, version 6.0 supports up to record 59.
- The last format version is generically called 7 which has lots of the limits removed. In this version, polygons can have more than 200 points, more than 64 layers and datatypes are supported, and more than 10 levels of hierarchy are supported. From the list of supported Stream records, version7 supports all the records.

Stream 6 was GDSII-specific so not many non-Calma reader/writer applications handle it. Simple applications deal with Stream 3 (basic geometries), more advanced deal with 4/5 and general purpose remove limitations and can handle upto version 7.

Stream Syntax

This section contains a Bachus Naur representation of the Stream syntax. Bachus Naur uses ALL CAPS to represent the name of a record type and lower case for names that can be further broken down into a set of record types. The following table provides descriptions of the Bachus Naur symbols.

Name of Symbol	Symbol	Meaning
Double colon	: :	“Is composed of”
Square brackets	[]	An element that can be absent or occur one time.

Design Data Translators Reference

Stream Format

Name of Symbol	Symbol	Meaning
Braces	{ }	One of the elements within the braces can occur.
Braces with an asterisk	{ }*	The elements within the braces can be absent or occur one or more times.
Braces with a plus	{ }+	The elements within braces must occur one or more times.
Angle brackets	< >	These elements are further defined in the Stream syntax list.
Vertical bar		“Or”

```

<stream format>::=
    HEADER BGNLIB [LIBDIRSIZE] [SRFNAME] [LIBSECUR] LIBNAME [REFLIBS] [FONTS]
    [ATTRTABLE] [GENERATIONS] [<FormatType>] UNITS {<structure>}* ENDLIB

<FormatType>::= FORMAT | FORMAT {MASK}+ ENDMASKS

<structure>::= BGNSTR STRNAME [STRCLASS] {<element>}* ENDSTR

<element>::= {<boundary> | <path> | <SREF> | <AREF>
    | <text> | <node> | <box> } {<property>}*
    ENDEL

<boundary>::= BOUNDARY [ELFLAGS] [PLEX] LAYER DATATYPE XY

<path>::= PATH [ELFLAGS] [PLEX] LAYER DATATYPE [PATHTYPE] [WIDTH] [BGNEXTN]
    [ENDEXTN] XY

<SREF>::= SREF [ELFLAGS] [PLEX] SNAME [<strans>] XY

<AREF>::= AREF [ELFLAGS] [PLEX] SNAME [<strans>] COLROW XY

<text>::= TEXT [ELFLAGS] [PLEX] LAYER <textbody>

<node>::= NODE [ELFLAGS] [PLEX] LAYER NODETYPE XY

<box>::= BOX [ELFLAGS] [PLEX] LAYER BOXTYPE XY

<textbody>::=
    TEXTYPE [PRESENTATION] [PATHTYPE] [WIDTH] [<strans>] XY STRING

<strans>::= STRANS [MAG] [ANGLE]

<property> ::= PROPATTR PROPVALUE

```

Note: The WIDTH record (inside the TEXT) is not used by the Virtuoso system. The one within PATH is used.

Design Data Translators Reference

Stream Format

Example of a Stream Format File

The following is an example of a Stream format file. An explanation follows the example.

```
% od -h example.sf
000 0006 0002 0258 001C 0102 0067 0009 0003.....X....
008 0000 0000 0000 0067 0009 0003 000D 0010 .....X.....
010 0000 0006 3902 0028 000A 3B02 0003 0005 ....9...(..;.....
018 0007 0010 0206 6578 616D 706C 652E 6368 .....example.ch
020 7000 005C 1F06 7265 6631 2E63 6870 0000 p.....refl.chp..
028 0000 0000 0000 0000 0000 0000 0000 0000 .....
030 0000 0000 0000 0000 0000 0000 0000 0000 .....
****
048 0000 0000 0000 0000 0000 0000 0000 00B4 .....
050 2006 6361 6C6D 6166 6F6E 742E 666E 7400 .calmafont.fnt.
058 0000 0000 0000 0000 0000 0000 0000 0000 .....
060 0000 0000 0000 0000 0000 0000 0000 7465 .....te
068 7874 2E66 6E74 0000 0000 0000 0000 0000 xt.fnt.....
070 0000 0000 0000 0000 0000 0000 0000 0000 .....
078 0000 0000 0000 0000 0000 666F 6E74 2E66 .....font.f
080 6E74 0000 0000 0000 0000 0000 0000 0000 nt.....
088 0000 0000 0000 0000 0000 0000 0000 0000 0000 .....
090 0000 0000 0000 7067 666F 6E74 2E66 6E74 .....pgfont.fnt
098 0000 0000 0000 0000 0000 0000 0000 0000 .....
0A0 0000 0000 0000 0000 0000 0000 0000 0000 .....
0A8 0000 000C 2300 6174 7472 732E 6174 0006 ....#.attrs.at..
0B0 2202 0003 0014 0305 3E41 8937 4BC6 A7EF ".....>A.7K...
0B8 3944 B82F A09B 5A5D 001C 0502 0067 0007 9D./..ZQ.....X..
0C0 000C 0011 001D 000A 0067 0007 0011 0011 .....X.....
0C8 003A 0014 000C 0606 6578 616D 706C 6532 .|.....example2
0D0 0004 0B00 000C 1206 6578 616D 706C 6531 .....example1
0D8 0006 1A01 8000 000C 1C05 425A 0000 0000 .....BZ....
0E0 0000 0008 1302 0002 0002 001C 1003 0000 .....
0E8 4E20 0000 4E20 0000 4E20 0001 4FF0 0001 N ..N ..N ..0...
0F0 3880 0000 4E20 0004 1100 0004 0700 001C 8...N .....
0F8 0502 0067 0007 000C 000B 001C 0009 0067 ...X.....X
100 0008 001C 000F 0039 003A 000C 0606 6578 .....9.|....ex
108 616D 706C 6531 0004 0C00 0006 0D02 0000 ample1.....
110 0006 1602 0000 0006 1701 0005 0006 1A01 .....
118 8006 000C 1B05 4120 0000 0000 0000 000C .....A .....
120 1003 0000 4E20 0000 4E20 000E 1906 4920 ....N ..N ....I.
128 414D 2048 4552 450D 0004 1100 0004 0800 AM HERE.....
130 0006 2601 0001 0006 0D02 0002 0006 0E02 ..&.....
138 0003 0024 1003 0000 1388 0000 6D60 0000 ...$......m'..
140 2EE0 0000 6D60 0000 1F40 0000 84D0 0000 ....m'...._.....
148 1388 0000 6D60 0004 1100 0004 0900 0006 ....m'.....
150 0D02 0004 0006 0E02 003F 0006 2102 0001 .....?..!...
158 0008 0F03 0000 03E8 0024 1003 0000 3A98 .....$....|.
160 0000 36B0 0000 6590 0000 36B0 0000 84D0 ..6....e....6....
168 0000 2328 0000 55F0 0000 1770 0006 2B02 ..#(..U....p...+.
170 0002 000A 2C06 4D45 5441 4C00 0006 2B02 ...., .METAL...+.
178 000A 000C 2C06 5052 4F50 4552 5459 0004 ...., .PROPERTY..
180 1100 0004 0700 0004 0400 .....
```

The database that produced this Stream format output has only two structures. They are called *example1* and *example2*. *Example1* contains a boundary that is template data, a

Design Data Translators Reference

Stream Format

path with two properties, and a middle-center justified text element containing the string, I AM HERE. *Example2* contains only one element, a 2 by 2 AREF of *example1*.

The following are explanations of the records contained in the example Stream file. As a reminder, the first two words (four bytes) of a record are the record header. The first word shows the record length in bytes, and the second word identifies the record type and the data type.

```
0006 0002 0258
```

The first word reports that this record is 6 bytes long. The second word indicates that this is the HEADER (00 hex) record and that the data type is a two-byte signed integer (02 hex). The information in the third word is the Stream version number, which is version 600 (258 hex).

```
001C 0102 0067 0009 0003 0000 0000 0000 0067 0009
0003 000D 0010 0000
```

This record is 28 (1C hex) bytes. It is the BGNLIB (01 hex) record. The data type is a two-byte signed integer (02). The remaining 24 bytes of information contain the date and time the library was last modified and the date and time of last access. For example, the last six words of information contain:

Time Component	Hexadecimal Representation	Decimal Representation	Real Date Value
year since 1900	0067	00103	2003
month	0009	00009	9 (September)
day	0003	00003	3
hour	000D	00013	1 pm.
minute	0010	00016	16
second	0000	00000	0

This record indicates that this library was last accessed on September 3, 2003, at 1:16:00 p.m. This is local time and no time zone or daylight savings time information is stored.

```
0006 3902 0028
```

This record is 6 bytes. It is the LIBDIRSIZE (39 hex) record. The data type is a two-byte signed integer (02). In this example, the directory size is 40 (28 hex) pages.

```
000A 3B02 0003 0005 0007
```

Design Data Translators Reference

Stream Format

This record is 10 (A hex) bytes. It is a LIBSECUR (3B hex) record. The data type is a two-byte signed integer (02). This example has only one ACL entry. The entry has a group number of 3, a user number of 5, and access rights of 7. This means that the only one with any access rights to this library is user number 5 in group number 3. The access code (007) means this user has read and write access and is also the owner of the library.

```
0010 0206 6578 616D 706C 652E 6368 7000
```

This record is 16 (10 hex) bytes. It is the LIBNAME (02 hex) record. The data type is an ASCII string (06). The six words of information contain the library name, `example.chp`.

```
005C 1F06 7265 6631 2E63 6870 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000
```

This record is 92 (5C hex) bytes. It is the REFLIB (1F hex) record. The data type is an ASCII string (06). In this example, the library `ref1.chp` is the bound reference library. The library is padded with nulls to equal 44 bytes. At least 92 bytes of this record must be present if any reference libraries are bound to the working library. No other reference library is bound, so the last 44 bytes are filled with nulls. If more than two reference libraries are bound, the record is extended by 44 bytes for each additional library.

```
00B4 2006 6361 6C6D 6166 6F6E 742E 666E 7400 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 7465 7874 2E66 6E74 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 666F 6E74 2E66 6E74
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 7067 666F
6E74 2E66 6E74 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
```

This record is 180 (B4 hex) bytes. It is the FONTS (20 hex) record. The data type is an ASCII string (06). All 180 bytes of this record must be present if any text font files are bound to this library. In this example, four text font files (the maximum possible) are bound to the library. The files are `calmafont.fnt`, `text.fnt`, `font.fnt`, and `pgfont.fnt`. Each string is padded with nulls out to 44 bytes.

```
000C 2306 6174 7472 732E 6174
```

Design Data Translators Reference

Stream Format

This record is 12 (C Hex) bytes. It is the ATTRTABLE (23 hex) record. The data type is an ASCII string (06). This record is only present if an attribute table is bound to the library. The name of the attribute table is `attrs.at`.

```
0006 2202 0003
```

This record is 6 bytes. It is the GENERATIONS (22 hex) record. The data type is a two-byte signed integer (02). In this example, three generations of a structure are retained in the library.

```
0014 0305 3E41 8937 4BC6 A7EF 3944 B82F A09B 5A51
```

This record is 20 (14 hex) bytes. It is the UNITS (03 hex) record. The data type is an eight-byte real (05). In this example, 3E41 8937 4BC6 A7EF is 1E-3. This implies that a database unit is .001 of a user unit. The record 3944 B82F A09B 5A51 is 1E-9. This implies that a database unit is 1E-9 meters (1E-3 microns).

```
001C 0502 0067 0007 000C 0011 001D 000A 0067 0007  
0011 0011 003A 0014
```

This record is 28 (1C hex) bytes. It is the BGNSTR (05 hex) record. The data type is a two-byte signed integer (02). The information in this record is the creation time and last modification time of the structure. The information is in the same format as the BGNLIB record. This structure was created July 12, 2003, at 5:29:10 p.m. and last modified July 17, 2003, at 5:58:20 p.m.

```
000C 0606 6578 616D 706C 6532
```

This record is 12 (C hex) bytes. It is the STRNAME (06 hex) record. The data type is an ASCII string (06). The structure name is `example2`.

```
0004 0B00
```

This record is 4 bytes. It is the AREF (0B hex) record. It contains no data (00). It marks the start of an AREF.

```
000C 1206 6578 616D 706C 6531
```

This record is 12 (C hex) bytes. It is the SNAME (12 hex) record. The data type is an ASCII string (06). This record contains the name of referenced structure `example1`.

Design Data Translators Reference

Stream Format

0006 1A01 8000

This record is 6 bytes. It is the STRANS (1A hex) record. The data type is a bit array (01). In this example, only bit 0 is set, which implies that this AREF is reflected. Since bit 13 and 14 are not set, this structure's magnification and angle, respectively, are not absolute.

000C 1C05 425A 0000 0000 0000

This record is 12 (C hex) bytes. It is the ANGLE (1C hex) record. The data type is eight-byte real data (05). The data 425A 0000 0000 0000 represents 90.0, which implies that this AREF is placed at an angle of 90 degrees.

0008 1302 0002 0002

This record is 8 bytes. It is the COLROW (13 hex) record. The data type is a two-byte signed integer (02). This example contains a 2 x 2 AREF.

001C 1003 0000 4E20 0000 4E20 0000 4E20 0001 4FF0
0001 3880 0000 4E20

This record is 28 (1C hex) bytes. It is the XY (10 hex) record. The data type is a four-byte signed integer (03). The data, taken two words at a time, can be translated to decimal as 20000, 20000, 20000, 86000, 80000, 20000. Multiply these by .001 (because a data base unit is .001 of a user unit). The results are the coordinates (20, 20), (20, 86), and (80, 20). The first coordinate is the array reference point. The second coordinate is a point that is displaced from the array reference point in the Y-direction by the number of columns times the inter-column spacing. In this example, the second point was displaced 66 (86 - 20) units from the array reference point. Since the array has two columns, this implies that the inter-column spacing is 33 units. A similar calculation can be carried out to verify that the inter-row spacing is 30 units.

0004 1100

This record is 4 bytes. It is the ENDEL (11 hex) record. It contains no data (00). ENDEL marks the end of an element.

0004 0700

Design Data Translators Reference

Stream Format

This record is 4 bytes. It is the ENDSTR (07 hex) record. It contains no data (00). ENDSTR marks the end of a structure.

```
001C 0502 0067 0007 000C 000B 001C 0009 0067 0008 001C 000F 0039 003A
```

This is another BGNSTR record. This structure was created July 12, 2003, at 11:28:09 a.m., and last modified August 28, 2003, at 3:57:58 p.m.

```
000C 0606 6578 616D 706C 6531
```

This is another STRNAME record. It contains the string `example1`.

```
0004 0C00
```

This record is 4 bytes. It is the TEXT (0C hex) record. It contains no data (00). Text marks the start of a text element.

```
0006 0D02 0000
```

This record is 6 bytes. It is the LAYER (0D hex) record. The data type is a two-byte signed integer (02). This text element is on layer 0.

```
0006 1602 0000
```

This record is 6 bytes. It is the TEXTTYPE (16 hex) record. The data type is a two-byte signed integer (02). This text element is `texttype 0`.

```
0006 1701 0005
```

This record is 6 bytes. It is the PRESENTATION (17 hex) record. The data type is a bit array (01). The hex number 0005 in binary has all bits set to zero except bits 13 and 15. Since bits 10 and 11 are 00, the text element is font 0. Since bits 12 and 13 are 01, the text has a middle vertical position. Since bits 14 and 15 are 01, the text has a center horizontal presentation.

```
0006 1A01 8006
```

This is another STRANS record. This text is reflected and has an absolute magnification and absolute angle.

Design Data Translators Reference

Stream Format

000C 1B05 4120 0000 0000 0000

This record is 12 (C hex) bytes. It is the MAG (1B hex) record. The data type is eight-byte real (05). The data in this record represents 2.0, meaning that this text is magnified two times.

000C 1003 0000 4E20 0000 4E20

This is another XY record. The text is placed at coordinate (20, 20).

000E 1906 4920 414D 2048 4552 450D

This record is 14 (E hex) bytes. It is the STRING (19 hex) record. The data type is an ASCII string (06). The text string is I AM HERE.

0004 1100

This is another ENDEL record.

0004 0800

This record is 4 bytes. It is the BOUNDARY (08 hex) record. It contains no data (00). BOUNDARY marks the start of a boundary element.

0006 2601 0001

This record is 6 bytes. It is the ELFLAGS (26 hex) record. The data type is a bit array (01). Since bit 15 is set, this element is template data. Since bit 14 is not set, the element is not external data.

0006 0D02 0002

This is another LAYER record. The boundary is on layer 2.

0006 0E02 0003

This record is 6 bytes. It is the DATATYPE (0E hex) record. The data type is a two-byte signed integer (02). This boundary is of datatype 3.

Design Data Translators Reference

Stream Format

0024 1003 0000 1388 0000 6D60 0000 2EE0 0000 6D60 0000 1F40 0000 84D0 0000 1388
0000 6D60

This is another XY record. The coordinates are (5, 28), (12, 28), (8, 34), and (5, 28).

0004 1100

This is another ENDEL record.

0004 0900

This record is 4 bytes. It is the PATH (09 hex) record. It contains no data (00). PATH marks the start of a path element.

0006 0D02 0004

This is another LAYER record. The path is on layer 4.

0006 0E02 003F

This is another DATATYPE record. The path is datatype 63 (3F hex).

0006 2102 0001

This record is 6 bytes. It is the PATHTYPE (21 hex) record. The data type is a two-byte signed integer (02). This path is pathtype 1.

0008 0F03 0000 03E8

This record is 8 bytes. It is the WIDTH (0F hex) record. The data type is a four-byte signed integer (03). The number 03E8 hex is 1000 in decimal. Multiply this by .001 (because a database unit is .001 of a user unit). The result is 1; therefore, the width of this path is 1.

0024 1003 0000 3A98 0000 36B0 0000 6590 0000 36B0 0000 84D0 0000 2328 0000 55F0
0000 1770

This is another XY record. This path's coordinates are (15, 14), (26, 14), (34, 9), and (22, 6).

Design Data Translators Reference

Stream Format

0006 2B02 0002

This record is 6 bytes. It is the PROPATTR (2B hex) record. The data type is a two-byte signed integer (02). This path has a property with attribute number 2.

000A 2C06 4D45 5441 4C00

This record is 10 (A hex) bytes. It is the PROPVALUE (2C hex) record. The data type is an ASCII string (06). The property value for property attribute 2 (above) is METAL. The odd length string (five characters) is padded with a null.

0006 2B02 000A

This is another PROPATTR record. This path has another property associated with it. The property has attribute number 10 (A hex).

000C 2C06 5052 4F50 4552 5459

This is another PROPVALUE record. Property attribute 10 (above) has the value PROPERTY.

0004 1100

This is another ENDEL record.

0004 0700

This is another ENDSTR record.

0004 0400

This record is 4 bytes. This record is the ENDLIB (04 hex) record. It contains no data (00). ENDLIB marks the end of a Stream format file.

How to read, write and print Stream dates to avoid Y2K problems

The year fields in Stream dates represent the number of years since 1900, and this has led to some misunderstandings, because dates before 2000 appeared to be abbreviated rather than offset by 1900. For example, the year 1999 is represented by decimal 00099 in Stream. This can lead to errors after 2000 if the Stream year fields are assumed to be only tens and units, such as in:

```
printf("Modification date: %d/%d/19%d\n", stream[2], stream[1], stream[0]);
```

The above code will print "Modification date: 1/1/19103" in the year 2003.

The format of the year fields in Stream corresponds with the year field in a struct tm, and this structure is a convenient way of getting the necessary 6 integers to store in a Stream date. The month is the only field which needs to be modified, and it needs to be incremented when writing, and decremented when reading Stream. It is recommended that the methods used in the following example be used when writing and reading Stream, and that the date not be printed directly from the Stream contents.

Code Examples

The following code examples are the recommended methods for writing dates to a Stream file, reading and printing dates from a Stream file, and comparing Stream dates. In each case, the date is encoded and decoded to/from a time_t time stamp.

```
int strmTime[6]; /* buffer of integers read from or to be written to Stream file */
int lowYears, highYears; /* counters for bad year content */

/*****
 * Encode a time stamp into a time buffer to write to a Stream file.
 *****/

void timeStampToStreamBuffer(int *strmTime, time_t timeStamp)
{
    struct    tm *ptm;

    ptm = localtime (&timeStamp);
    strmTime[0]= ptm->tm_year;
    strmTime[1]= ptm->tm_mon + 1; /* Make month 1-based */
    strmTime[2]= ptm->tm_mday;
    strmTime[3]= ptm->tm_hour;
    strmTime[4]= ptm->tm_min;
```

Design Data Translators Reference

Stream Format

```
    strmTime[5]= ptm->tm_sec;
}

/*****
 * Extract a time_t from a Stream time buffer.
 *****/
static time_t convStrmDateToTime_t(strmTime)
int *strmTime ;
{
    struct tm ptm ;
    time_t ttime ;

    if (strmTime[0] <= 69) {
        numLowYears++ ;
        ptm.tm_year = strmTime[0] + 100 ;
    }
    else if (strmTime[0] >= 1970) {
        numHighYears++ ;
        ptm.tm_year = strmTime[0] - 1900 ;
    }
    else {
        ptm.tm_year = strmTime[0] ;
    }

    ptm.tm_mon = strmTime[1] - 1 ;
    ptm.tm_mday = strmTime[2] ;
    ptm.tm_hour = strmTime[3] ;
    ptm.tm_min = strmTime[4] ;
    ptm.tm_sec = strmTime[5] ;
    /* initialize these values for mktime. */
    ptm.tm_wday = 0;
    ptm.tm_yday = 0;
    ptm.tm_isdst = -1;
    ttime = mktime(&ptm) ;
    return(ttime) ;
}

/*****
 * Example use of the above functions.
 * The following code gets the current system time in a time_t, prints
 * the date from that time_t, encodes it in an integer buffer, which
 * could be written to a Stream file, prints the contents of that
 * buffer, decodes the buffer back into a time_t, prints the date from
 * that time_t, and then compares the before and after time_ts.
 *****/
void warnBadDates()
{
    if (lowYears > 0 ) {
        printf("Warning: years < 70 encountered in Stream file\n");
        printf("Warning: Stream year fields should be number of years
        since 1900\n");
    }
    if (highYears > 0) {
```

Design Data Translators Reference

Stream Format

```
printf("Warning: years > 1970 encountered in Stream file\n");
printf("Warning: Stream year fields should be number of years
since 1900\n");
}
}

main(int argc, char **argv)
{
    time_t timeStamp, newTimeStamp;
    short strmTime[6];
    char timeBuf[80];

    lowYears = highYears = 0;
    time(&timeStamp);
    printf("Input time is %s\n", ctime(&timeStamp));
    timeStampToStreamBuffer(strmTime, timeStamp);

    printf("Contents of Stream time buffer: %0.4x %0.4x %0.4x %0.4x
%0.4x %0.4x\n",
        strmTime[0], strmTime[1],strmTime[2],
        strmTime[3],strmTime[4],strmTime[5]);
    newTimeStamp = convStrmDateToTime_t(strmTime);
    printf("Time returned from time buffer: %s\n\n",
        ctime(&newTimeStamp));
    if (timeStamp != newTimeStamp) {
        printf("Error: timestamps differ: %d %d\n", timeStamp,
            newTimeStamp);
        printf("old: %s\n", ctime(&timeStamp));
        printf("new: %s\n", ctime(&newTimeStamp));
    }
    else
        printf("Timestamps matched\n");
    warnBadDates();
}
```

Migrating from CDL In to Spice In

CDL In has been replaced with Spice In on OpenAccess to import CDL netlists into Virtuoso. In addition to CDL, Spice In can import HSpice, Spectre, and SPICE netlists into Virtuoso.

Spice In Advantages over CDL In

Spice In offers the following advantages over CDL In:

- Using **Spice In**, you can import both kinds of CDL netlists — those created from an analog component-based schematic using an analog CDL netlister and those created from a digital component-based schematic using a digital CDL netlister.

CDL In is capable of importing CDL netlists created only from a digital component-based schematic using a digital CDL netlister.

- **Spice In** saves CDF and Simulation Information data, which helps integration of ADE with VSE and VLE.

CDL In does not save CDF and Simulation Information data.

If you still want to use CDL In, you can run it from the command line in the batch mode by using the following command:

```
nino cdlin templateFile
```

You must specify a template file when you use the `nino` command to run CDL In. The `nino` command is the non-interactive, batch mode version of CDL In.

Differences Between Spice In and CDL In

This section describe how the Spice In parameter and device-map files are different from those of CDL In.

Differences in Parameter File

The following table provides a sample CDL In parameter file and a sample Spice In parameter file for comparison.

Sample CDL In Parameter file	Sample Spice In Parameter File
<pre>cdlInKeys = list(nil 'searchPath "." 'cdlFile "" 'opusLib "" 'usrSkillFile "" 'primaryCell "" 'caseSensitivity "preserve" 'hierarchy "full" 'cellTable "" 'viewName "netlist" 'viewType "" 'refLibList "" 'pr nil 'skipDevice nil 'globalNodeExpand "full" 'schGenOption t 'schGenParamFile "conn2sch_param" 'devMapFile "/usr1/home/devMap" 'overwriteExistingCVs "selected" 'overwriteExistingCVsList "RSLATCH ")</pre>	<pre>spiceInParams = list(nil 'simName "spectre" 'netlistFile "" 'netlistViewName "netlist" 'schematicViewName "schematic" 'language "auto" 'devMapFile "" 'targetLib "" 'topCell "top" 'logFile "spiceIn.log" 'importDesignAs "schematic" 'errMsgFile "spiceIn.err" 'refLibList "" 'overwriteCells "all" 'overwriteCellsList "")</pre>

Differences in Device-Map File

The format of the device-map file for Spice In is different from that required by CDL In. In CDL In, among other properties you can also specify `propMatch`, `termOrder`, `propMap`, and `addProp` properties to correctly map a device to a primitive device. Spice In requires only the `devSelect` statement. See [Creating a Device-Map File](#) on page 502 in [Chapter 4, “Netlist Import Using Spice In”](#) for details on device-map file format that Spice In expects.

Mapping LEF to the Technology File

LEF Data Map

A Library Exchange Format (LEF) file contains library information for a class of designs. Library data includes layer, via, placement site type, and macro cell definitions. Ensemble uses LEF as the library implementation language to build the technology and macrocell library for a family of designs. The *File – Import - LEF* and *File – Export – LEF* commands provide a mechanism for moving library data between the two systems. You can create an OpenAccess library from one or more LEF files.

This chapter maps the LEF file constructs and statements with the translated constructs and statements in the technology file in Virtuoso Studio design environment on OpenAccess.

The LEF file is divided into the following sections. You can specify statements in any order; however, data must be defined before it is used. The LEF statements in each of these sections are mapped to their equivalent technology file statements.

- Version of LEF
- Units
- Property Definitions
- Layers
 - Routing Layer
 - Cut Layer
 - Masterslice Layer
 - Implant Layer
- BEGINEXT Statement
- BUSBITCHARS Statement
- CLEARANCEMEASURE Statement

Design Data Translators Reference

Mapping LEF to the Technology File

- [MACRO Section](#)
- [DIVIDERCHAR Statement](#)
- [MANUFACTURINGGRID Statement](#)
- [Maximum Stacked Via](#)
- [NAMESCASESENSITIVE Statement](#)
- [SITE Section](#)
- [SPACING Section](#)
- [USEMINSPACING Statement](#)
- [VIA Statement](#)
- [VIARULE Statement](#)
- [VIARULE GENERATE](#)

Note: *File – Import – LEF* and *File – Export – LEF* do not support the `PINPROPERTIES`, `MIN FEATURE`, `CROSS TALK`, `DIELECTRIC`, `IRDROP`, or `BEGINEXT` statements. Many of these constructs are no longer part of LEF syntax. LEF 5.4 onwards `MIN FEATURE`, `CROSS TALK`, `DIELECTRIC`, `IRDROP` statements are no longer exist.

For details on each LEF statement, see [LEF/DEF 5.6 Language Reference](#).

The organization of the technology file is different from the way the LEF file is organized. The technology file is organized into the following sections:

- **Controls** (`controls`) specify data that can be used throughout the technology file.
- **Layer definitions** (`layerDefinitions`) define the layers that can be used throughout the technology file and in design sessions.
- **Layer rules** (`layerRules`) specify layer attributes, such as layer materials, manufacturing grid resolutions, and current densities.
- **Constraint groups** (`constraintGroups`) specify design constraints.
- **Site definitions** (`siteDefs`) define scalar site definitions and arrays of scalar site definitions.
- **Via definitions** (`viaDefs`) define standard and custom vias.
- **Via specifications** (`viaSpecs`) define arrays of via definitions.

- **Devices** (`devices`) define Cadence-predefined MOS and `ruleContact` devices and multipart path templates.

For more information on the technology file, see the [*Technology File and Display Resource File User Guide*](#).

Version of LEF

The `VERSION` statement indicates the LEF version that was used to create a LEF file. *File – Import – LEF* ignores the `VERSION` statement. *File – Export – LEF* writes the `VERSION` statement as the first line of the LEF file.

```
VERSION number ;
```

Units

The `UNITS` statement block describes the units of measurement used in the LEF file and the precision for each measure in database units. The values show how to interpret the numbers in the LEF file. Following is the `UNITS` block syntax in LEF.

```
[UNITS
    [ TIME NANOSECONDS convertFactor ; ]
    [ CAPACITANCE PICOFARADS convertFactor ; ]
    [ RESISTANCE OHMS convertFactor ; ]
    [ POWER MILLIWATTS convertFactor ; ]
    [ CURRENT MILLIAMPS convertFactor ; ]
    [ VOLTAGE VOLTS convertFactor ; ]
    [ DATABASE MICRONS convertFactor ; ]
    [ FREQUENCY MEGAHERTZ convertFactor ; ]
END UNITS ]
```

The LEF statements in the `UNITS` section of LEF are mapped to the `techParams` subsection of the `controls` section in the technology file.

The syntax of the `techParams` subsection in the technology file is as follows:

```
controls(
    techParams(
        ;(parameter value)
    )
)
```

Design Data Translators Reference

Mapping LEF to the Technology File

Comparison of the UNIT statements in LEF

LEF UNIT Statements	Technology File DFII on OpenAccess
TIME NANOSECONDS <i>convertFactor</i>	LEFDEF_TIME_UNIT <i>value</i>
FREQUENCY MEGAHERTZ <i>convertFactor</i>	LEFDEF_FREQUENCY_UNIT <i>value</i>
RESISTANCE OHMS <i>convertFactor</i>	LEFDEF_RESISTANCE_UNIT <i>value</i>
POWER MILLIWATTS <i>convertFactor</i>	LEFDEF_POWER_UNIT <i>value</i>
VOLTAGE VOLTS <i>convertFactor</i>	LEFDEF_VOLTAGE_UNIT <i>value</i>
CURRENT MILLIAMPS <i>convertFactor</i>	LEFDEF_CURRENT_UNIT <i>value</i>
CAPACITANCE PICOFARADS <i>convertFactor</i>	LEFDEF_CAPACITANCE_UNIT <i>value</i>

Design Data Translators Reference

Mapping LEF to the Technology File

Examples

Example of LEF UNIT Statements	Example of Technology File DFII on OpenAccess
<pre> UNITS TIME NANOSECONDS 100 ; CAPACITANCE PICO FARADS 10 ; RESISTANCE OHMS 10000 ; POWER MILLIWATTS 10000 ; CURRENT MILLIAMPS 10000 ; VOLTAGE VOLTS 1000 ; FREQUENCY MEGAHERTZ 10 ; DATABASE MICRONS 2000 ; END UNITS </pre>	<pre> ;***** ; CONTROLS ;***** controls(techParams(; (parameter value) ; (----- -----) (LEFDEF_BUSBIT_CHARS"<>") (LEFDEF_ANTENNA_INPUT_GATE_AREA1.0) (LEFDEF_USEMINSPACING_OBSn1l) (LEFDEF_OVERLAP_LAYER_NAME"OVERLAP") (LEFDEF_FREQUENCY_UNIT10) (LEFDEF_VOLTAGE_UNIT1000) (LEFDEF_CURRENT_UNIT10000) (LEFDEF_POWER_UNIT10000) (LEFDEF_RESISTANCE_UNIT10000) (LEFDEF_CAPACITANCE_UNIT10) (LEFDEF_TIME_UNIT100) (LEFDEF_DIVIDER_CHARS":")) ;techParams </pre>

Property Definitions

The PROPERTYDEFINITIONS section of the LEF file lists the properties you can provide to libraries, pins, macros, and vias. Although the PROPERTYDEFINITION statements are not stored in OpenAccess, they are used to validate the PROPERTY attributes on objects when the LEF file is being parsed. Every property used in the LEF file must first be declared in this section of the LEF file. If a property that is not declared in the PROPERTYDEFINITIONS section is encountered, an error message is generated and the property is skipped.

Design Data Translators Reference

Mapping LEF to the Technology File

Examples

Example of LEF Property Definition Statements	Example of Technology File DFII on OpenAccess
<pre> PROPERTYDEFINITIONS LIBRARY NAME STRING "Cadence96" ; LIBRARY intNum INTEGER 20 ; LIBRARY realNum REAL 21.22 ; PIN TYPE STRING ; PIN intProp INTEGER ; PIN realProp REAL ; MACRO stringProp STRING ; MACRO integerProp INTEGER ; MACRO WEIGHT REAL RANGE 1.0 100.0 ; VIA stringProperty STRING ; VIA realProperty REAL ; VIA COUNT INTEGER RANGE 1 100 ; LAYER lsp STRING ; LAYER lip INTEGER ; LAYER lrp REAL ; VIARULE vrsp STRING ; VIARULE vrip INTEGER ; VIARULE vrrp REAL ; NONDEFAULTRULE ndrsp STRING ; NONDEFAULTRULE ndrrip INTEGER ; NONDEFAULTRULE ndrrip REAL ; END PROPERTYDEFINITIONS </pre>	<pre> techLayerProperties((PropName Layer1 [Layer2] PropValue) (-----) (lip POLYS 1) (lrp POLYS 2.300000) (lsp POLYS "top")) </pre>

If the property definitions were not present for lip, lrp, and lsp in the LEF file, then these would have been ignored and the generated ASCII Technology File would not have these property names.

Layers

There are four types of layers in a LEF file: routing layer, cut layer, masterslice, and overlap layer. The layer statements in a LEF file define layer names and associate place and route design rules with each routing layer.

The OpenAccess database defines layers in the technology file. For details on defining layers in the technology file see the [*Technology File and Display Resource File User Guide*](#).

The following topics explain the syntax and the statements or constructs in each LEF layer and maps these statements to their technology file equivalents.

Routing Layer

Following is the syntax of the routing layer in LEF 5.6.

```
LAYER layerName
    TYPE ROUTING ;
    DIRECTION {HORIZONTAL | VERTICAL | DIAG45 | DIAG135} ;
    PITCH {distance | xDistance yDistance} ;
    [DIAGPITCH {distance | diag45Distance diag135Distance} ;]
    WIDTH defaultWidth ;
    [OFFSET {distance | xDistance yDistance} ;]
    [DIAGWIDTH diagWidth ;]
    [DIAGSPACING diagSpacing ;]
    [DIAGMINEDGELENGTH diagLength ;]
    [AREA minArea ;]
    [MINSIZE minWidth minLength [minWidth2 minLength2] ... ;]
    [ [SPACING minSpacing
        [ RANGE minWidth maxWidth
            [ USELENGTHTHRESHOLD
                | INFLUENCE value [RANGE stubMinWidth stubMaxWidth]
                | RANGE minWidth maxWidth]
            | LENGTHTHRESHOLD maxLength [RANGE minWidth maxWidth]
        ]
    ] ...
    ] [SPACINGTABLE
        PARALLELRUNLENGTH {length} ...
        {WIDTH width {spacing} ...} ... ;
        [SPACINGTABLE
            INFLUENCE {WIDTH width WITHIN distance SPACING spacing} ... ;
        ] ]
    ]
    [WIREEXTENSION value ; ]
    [MINIMUMCUT numCuts WIDTH width
        [FROMABOVE | FROMBELOW]
        [LENGTH length WITHIN distance] ;] ...
    [MAXWIDTH width ;]
    [MINWIDTH width ;]
    [MINSTEP minStepLength
        [INSIDECORNER | OUTSIDECORNER | STEP]
        [LENGTHSUM maxLength] ;]
    [MINENCLOSEDAREA area [WIDTH width] ;] ...
    [PROTRUSIONWIDTH width1 LENGTH length WIDTH width2 ;]
    [RESISTANCE RPERSQ value ;]
    [CAPACITANCE CPERSQDIST value ;]
    [HEIGHT distance ;]
    [THICKNESS distance ;]
    [SHRINKAGE distance ;]
    [CAPMULTIPLIER value ;]
    [EDGECAPACITANCE value ;]
    [SLOTWIREWIDTH minWidth ;]
    [SLOTWIRELENGTH minLength ;]
```

Design Data Translators Reference

Mapping LEF to the Technology File

```
[SLOTWIDTH minWidth ;]
[SLOTLENGTH minLength ;]
[MAXADJACENTSLOTSPACING spacing ;]
[MAXCOAXIALSLOTSPACING spacing ;]
[MAXEDGESLOTSPACING spacing ;]
[SPLITWIREWIDTh minWidth ;]
[MINIMUMDENSITY minDensity ;]
[MAXIMUMDENSITY maxDensity ;]
[DENSITYCHECKWINDOW windowLength windowWidth ;]
[DENSITYCHECKSTEP stepValue ;]
[FILLACTIVESPACING spacing ;]
[ANTENNAMODEL {OXIDE1 | OXIDE2 | OXIDE3 | OXIDE4} ;] ...
[ANTENNAAREARATIO value ;] ...
[ANTENNADIFFAREARATIO {value | PWL ( ( d1 r1 ) ( d2 r2 ) ... ) } ;] ...
[ANTENNACUMAREARATIO value ;] ...
[ANTENNACUMDIFFAREARATIO {value | PWL ( ( d1 r1 ) ( d2 r2 ) ... ) } ;] ...
[ANTENNAAREAFACtor value [DIFFUSEONLY] ;] ...
[ANTENNASIDEAREARATIO value ;] ...
[ANTENNADIFFSIDEAREARATIO {value | PWL ( ( d1 r1 ) ( d2 r2 ) ... ) } ;] ...
[ANTENNACUMSIDEAREARATIO value ;] ...
[ANTENNACUMDIFFSIDEAREARATIO {value | PWL ( ( d1 r1 ) ( d2 r2 ) ... ) } ;] ...
[ANTENNASIDEAREAFACtor value [DIFFUSEONLY] ;] ...
[PROPERTY propName propVal ;] ...
[ACCURRENTDENSITY {PEAK | AVERAGE | RMS}
  { value
    | FREQUENCY freq_1 freq_2 ... ;
    [WIDTH width_1 width_2 ... ;]
    TABLEENTRIES
      v_freq_1_width_1 v_freq_1_width_2 ...
      v_freq_2_width_1 v_freq_2_width_2 ...
      ...
  } ;]
[DCCURRENTDENSITY AVERAGE
  { value
    | WIDTH width_1 width_2 ... ;
    TABLEENTRIES value_1 value_2 ...
  } ;]
END layerName
```

Following are the technology file equivalents for each line of code in the routing layer in the LEF file.

Design Data Translators Reference

Mapping LEF to the Technology File

The routing **PITCH** of a routing layer in the LEF file is mapped to the `routingGrids` subsection of the `constraintGroups` section in the technology file. The following table shows the syntax and example of the LEF and technology files.

LEF File	Technology File
<code>PITCH <i>distance</i>;</code>	<code>(horizontalPitch <i>layerName Pitch</i>)</code> <code>(verticalPitch <i>layerName Pitch</i>)</code> <code>(leftDiagPitch <i>layerName Pitch</i>)</code> <code>(rightDiagPitch <i>layerName Pitch</i>)</code>
<pre>LAYER metall TYPE ROUTING; PITCH 1.8; DIAGPITCH 1.34 1.38 ; ... END metall</pre>	<pre>constraintGroups (.... routingGrids (.... (horizontalPitch "M2" 1.8) (verticalPitch "M2" 1.8) (leftDiagPitch "M2" 1.34) (rightDiagPitch "M2" 1.38)) )</pre> where M2 is the layer name followed by the pitch values. PITCH in LEF is translated as horizontalPitch and verticalPitch each having equal values in the technology file. Similarly, DIAGPITCH in LEF is mapped to leftDiagPitch and rightDiagPitch.

The routing **OFFSET** for a routing layer in the LEF file is mapped as `routingGrids` in the `constraintGroups` section of the technology file.

LEF	Technology File
<code>OFFSET <i>distance</i>;</code>	<code>(horizontalOffset <i>layerName Offset</i>)</code> <code>(verticalOffset <i>layerName Offset</i>)</code>

Design Data Translators Reference

Mapping LEF to the Technology File

LEF	Technology File
<pre> LAYER RX TYPE ROUTING; OFFSET 0.9; ... END RX </pre>	<pre> constraintGroups (.... routingGrids (.... (horizontalOffset "RX" 0.9) (verticalOffset "RX" 0.9)) ) </pre> where RX is the layer name and 0.9 the <code>maxWidth</code> value.

MAXWIDTH is mapped as *maxWidth* in the `spacings` subsection of the `constraintGroups` section in the technology file.

LEF	Technology File
<pre> MAXWIDTH <i>width</i> </pre>	<pre> (maxWidth <i>layerName width</i>) </pre>
<pre> LAYER RX TYPE ROUTING; MAXWIDTH 10.11111; END RX </pre>	<pre> constraintGroups (.... spacings (.... (maxWidth "RX" 10.111)) ) </pre> where RX is the layer name and 10.111 is the <code>maxWidth</code> value.

The MAXWIDTH rule applies only for routing layers and is a float in units of μm . If MAXWIDTH is specified in any other layer, following warning message is displayed:

```
*WARNING* MAXWIDTH can be defined only in the routing layer: Ignoring the rule for layer layerName.
```

Design Data Translators Reference

Mapping LEF to the Technology File

PROTRUSIONWIDTH in the LEF file is mapped as `protrusionWidth` in the `spacings` subsection of the `constraintGroups` section in the technology file.

LEF	Technology File
PROTRUSIONWIDTH <i>width1</i> LENGTH <i>length</i> WIDTH <i>width2</i> ;	(<code>protrusionWidth</code> <i>layer</i> (<i>length</i> <i>width1</i> <i>width2</i>))
LAYER M1 TYPE ROUTING; PROTRUSIONWIDTH 0.28 LENGTH 0.60 WIDTH 1.20 ; END M1	<code>constraintGroups</code> (.... <code>spacings</code> (.... (<code>protrusionWidth</code> "M1" (0.6 1.2 0.28)))) where M1 is the layer name followed by the length, width, and protrusion width.

In LEF, *width1*, *width2*, and *length* are floats in units of `um`. The PROTRUSIONWIDTH rule applies only to the routing layer. A protrusion width must be greater than or equal to *width1* if the protrusion connects to a wire having width greater than *width2* and length greater than *length*.

MINSTEP is mapped as `minStep` in the `spacings` subsection of the `constraintGroups` section in the technology file. The MINSTEP rule in the LEF file specifies the minimum step size (or shortest edge length). In this rule, *distance* is a float in units of `um` and it specifies the minimum step size. The MINSTEP rule applies only to the routing layers.

LEF	Technology File
MINSTEP <i>minStepLength</i> [OUTSIDECORNER] [LENGTHSUM <i>maxLength</i>] ;	(<code>minStep</code> <i>layerName</i> <i>lengthDefinition</i>)

Design Data Translators Reference

Mapping LEF to the Technology File

LEF	Technology File
<pre> LAYER PC TYPE ROUTING; MINSTEP 0.1 OUTSIDECORNER LENGTHSUM 0.4 ; MINSTEP 0.1 INSIDECORNER LENGTHSUM 0.4 ; MINSTEP 0.1 STEP LENGTHSUM 0.4; MINSTEP 0.2 STEP LENGTHSUM 0.8; END PC </pre>	<pre> constraintGroups(.... spacings(.... (minOutsideCornerEdgeLength "PC"(0.4 0.1)) (minInsideCornerEdgeLength "PC" (0.4 0.1)) (minStepEdgeLength "PC" (0.8 0.2))) ) </pre> where PC is the layer name.

WIDTH in the LEF file is mapped as `minWidth` in the `spacings` section, which is a subsection of the `constraintGroups` section in the technology file.

LEF	Technology File
<pre> WIDTH <i>defWidth</i>; </pre>	<pre> (<i>minWidth</i> <i>layerName</i> <i>value</i>) </pre>
<pre> LAYER RX TYPE ROUTING; WIDTH 1; ... END RX </pre>	<pre> constraintGroups(.... spacings(.... (minWidth "RX" 1)) ) </pre> where RX is the layer name and 1 is the <code>minWidth</code> value.

The AREA rule in the LEF file is mapped as `minArea` in the `spacingRules` subsection of the `physicalRules` section in the technology file.

LEF	Technology File
<pre> AREA <i>minArea</i>; </pre>	<pre> (<i>minArea</i> <i>layerName</i> <i>minAreaValue</i>) </pre>

Design Data Translators Reference

Mapping LEF to the Technology File

LEF	Technology File
<pre> LAYER M1 TYPE ROUTING; AREA 1.0 ; ... END M1 ... </pre>	<pre> constraintGroups (.... spacings (.... (minArea "M1" 1)); spacings ) </pre> where M1 is the layer name followed by the minimum area value.

The **MINWIDTH** statement is mapped as `minWidth` in the `spacings` section, which is a subsection of the `constraintGroups` section in the technology file.

Notice that the precision of the minimum width is curtailed in the technology file as compared to the LEF file.

LEF	Technology File
<pre> MINWIDTH <i>minWidth</i> </pre>	<pre> (<i>minWidth</i> <i>layerName</i> <i>minWidthValue</i>) </pre>
<pre> LAYER RX TYPE ROUTING ; MINWIDTH 0.71111 ; END RX </pre>	<pre> constraintGroups (.... spacings (.... (minWidth "RX" 0.711)); spacings ) </pre>

The **MINIMUMCUT** statement is mapped as `minimumCuts` in the `spacingTables` section, which is a subsection of the `constraintGroups` section in the technology file.

The `minimumCut` rules are stored only on the cut layers in the technology file. Rules for the cut layer are temporarily stored on the current metal layer, since the cut layer would not have been created already. These rules are later moved to the cut layers.

LEF	Technology File
<pre> MINIMUMCUT <i>numCuts</i> WIDTH <i>width</i> </pre>	<pre> (<i>minNumCut</i> <i>width</i> <i>minNumCuts</i>) </pre>

Design Data Translators Reference

Mapping LEF to the Technology File

LEF	Technology File
<pre> LAYER M1 TYPE ROUTING; MINIMUMCUT 4 WIDTH 1.1 ; MINIMUMCUT 6 WIDTH 3.3 ; ... END M1 ... </pre>	<pre> constraintGroups (.... spacings (.... (protrusionWidth "M1" (0.6 1.2 0.28)) (minSpacing "M1" 0.28) (minDiagonalSpacing "M1" 0.45)) ;spacings spacingTables (;(constraint layer1 [layer2] ; ((index1Definitions [index2Defintions]) [defaultValue]) ; (table)) ;(-----) (minNumCut "CA" (("width" nil nil)) (1.1 4 3.3 6))) ;spacingTables where M1 is the layer name. </pre>

MINENCLOSEDAREA in the LEF file maps to the minHoleArea in the spacingTables subsection of the constraintGroups section in the technology file.

In the MINENCLOSEDAREA construct, *area* specifies the minimum area size limit for metal that encloses an empty area. It is a float in units of μm^2 . The *width* statement is optional and is a float in units of μm .

This rule is connected to the slotting process. If the width is specified, then the width applies to the hole-edge starting from the outer-edge of the material surrounding the hole. There can be multiple minHoleArea rules per layer to take into account different width values.

LEF	Technology File
MINENCLOSEDAREA <i>area</i> ;	<i>(minEnclosedArea layerName</i> <i>list(width area))</i>

Design Data Translators Reference

Mapping LEF to the Technology File

LEF	Technology File
<pre> LAYER M1 TYPE ROUTING; MINENCLOSEDAREA 0.5 WIDTH 1.25 ; MINENCLOSEDAREA 0.4 WIDTH 1.15 ; MINENCLOSEDAREA 0.3 WIDTH 1.05 ; ... END M1 ... </pre>	<pre> constraintGroups(.... spacingTables((minHoleArea "RX" (("width" nil nil)) (1.05 0.3 1.15 0.4 1.25 0.5)) );spacingTables </pre>

The SPACINGTABLE rule applies only to the routing layer. In this rule,

PARALLELRUNLENGTH	Specifies that if the maximum width of the wire is greater than <i>width</i> and parallel run length greater than the <i>length</i> , then wire spacing must be greater than or equal to <i>spacing</i> .
INFLUENCE	Specifies that if a wire width is greater than <i>width</i> , and the distance to two other wires is less than <i>distance</i> , then the other wires must be separated by distance greater than or equal to <i>spacing</i> .

where, *length*, *width*, and *spacing* are floats in units of μm .

Design Data Translators Reference

Mapping LEF to the Technology File

1. SPACINGTABLE PARALLELRUNLENGTH maps as minSpacing in the spacingTables subsection of the constraintGroups section in the technology file.

LEF	Technology File
<pre> LAYER M2 TYPE ROUTING; SPACINGTABLE # LEF5.5 PARALLELRUNLENGTH 0.00 0.50 3.00 5.00 WIDTH 0.00 0.15 0.15 0.15 0.15 WIDTH 0.25 0.15 0.20 0.20 0.20 WIDTH 1.50 0.15 0.50 0.50 0.50 WIDTH 3.00 0.15 0.50 1.00 1.00 WIDTH 5.00 0.15 0.50 1.00 2.00 ; </pre>	<pre> constraintGroups (.... spacingTables (.... (minSpacing "M2" (("width" nil nil "length" nil nil)) ((0 0) 0.15 (0 0.5) 0.15 (0 3) 0.15 (0 5) 0.15 (0.2505 0) 0.15 (0.2505 0.5) 0.2 (0.2505 3) 0.2 (0.2505 5) 0.2 (1.5005 0) 0.15 (1.5005 0.5) 0.5 (1.5005 3) 0.5 (1.5005 5) 0.5 (3.0005 0) 0.15 (3.0005 0.5) 0.5 (3.0005 3) 1 (3.0005 5) 1 (5.0005 0) 0.15 (5.0005 0.5) 0.5 (5.0005 3) 1 (5.0005 5) 2))) </pre>

2. SPACINGTABLE PARALLELRUNLENGTH maps to minInfluenceSpacing in the spacingTables subsection of the constraintGroups section in the technology file.

Design Data Translators Reference

Mapping LEF to the Technology File

LEF

Technology File

<pre> LAYER M2 TYPE ROUTING; SPACINGTABLE INFLUENCE WIDTH 1.50 WITHIN 0.50 SPACING 0.55 WIDTH 3.00 WITHIN 1.00 SPACING 1.05 WIDTH 5.00 WITHIN 2.00 SPACING 2.05 ; </pre>	<pre> constraintGroups (.... spacingTables (.... (minInfluenceSpacing "M2" (("width" nil nil "distance" nil nil)) ((1.5 0.5) 0.55 (1.5 1) 0.55 (1.5 2) 0.55 (3 0.5) 0.55 (3 1) 1.05 (3 2) 1.05 (5 0.5) 0.55 (5 1) 1.05 (5 2) 2.05))) ;spacingTables </pre>

The metal slotting and splitting values `SLOTWIREWIDTH`, `SLOTWIRELENGTH`, `SLOTWIDTH`, `SLOTLENGTH`, `MAXADJACENTSLOTSPACING`, `MAXCOAXIALSLOTSPACING`, `MAXEDGESLOTSPACING`, and `SPLITWIREWIDTH` are not stored in the technology file.

The `MINIMUMDENSITY` and `MAXIMUMDENSITY` metal filling values map `minDensity` and `maxDensity` rules in the `spacingTables` subsection of the `constraintGroups` section in the technology file.

LEF

Technology File

<pre> MINIMUMDENSITY minDensity; </pre>	<pre> (minDensity <i>layerName</i> (step <i>minDensity</i>)) (maxDensity <i>layerName</i> (step <i>minDensity</i>)) </pre>
<pre> MAXIMUMDENSITY maxDensity; </pre>	

Design Data Translators Reference

Mapping LEF to the Technology File

LEF	Technology File
<pre> LAYER RX TYPE ROUTING; MINIMUMDENSITY 1.3 ; MAXIMUMDENSITY 4.3 ; ... END RX </pre>	<pre> constraintGroups (.... spacingTables (;(constraint layer1 [layer2] ; ((index1Definitions [index2Definitions]) [defaultValue]) ; (table)) ;(-----) (minDensity "RX" (("step" nil nil)) (0.25 1.3)) (maxDensity "RX" (("step" nil nil)) (0.25 4.3))) ;spacingTables ) </pre> where RX is the layer name

DENSITYCHECKWINDOW is not stored. This value in the technology file is always twice the stepSize.

DENSITYCHECKSTEP is not stored. The value of stepSize is used for setting the minimum and maximum density.

The FILLACTIVESPACING statement maps to the minFillToShapeSpacing statement in the spacings subsection of the constraintGroups section in the technology file.

LEF	Technology File
FILLACTIVESPACING <i>spacing</i>	(minFillToShapeSpacing <i>layerName value</i>)

Design Data Translators Reference

Mapping LEF to the Technology File

LEF	Technology File
<pre> LAYER RX TYPE ROUTING ; FILLACTIVESPACING 0.98 ; END RX </pre>	<pre> constraintGroups(.... spacings((minFillToShapeSpacing "RX" 0.98)) ;spacings ) </pre>

Important

A single SPACING construct is mapped to an `oaIntRule`. If there are SPACING RANGE statements following the generic SPACING statements, the database maps all SPACING constructs to an `oaInt1DTableRule`. If there are SPACING LENGTHTHRESHOLD statements, the database maps all SPACING constructs to an `oaInt2DTableRule`. The `maxWidth` value and optional second RANGE or USELENGTHTHRESHOLD keywords are not stored in OpenAccess.

The routing direction of the routing layers is mapped to the `routingDirections` subsection of the `layerRules` section in the technology file.

LEF	Technology File
<pre> DIRECTION {HORIZONTAL VERTICAL}; </pre>	<pre> layerRules(routingDirections((<i>LayerName PreferredDirection</i>));); </pre>
<pre> LAYER RX TYPE ROUTING; DIRECTION VERTICAL ; </pre>	<pre> layerRules(... routingDirections(;(layer direction) ;(----- -----) (RX "vertical") (PC "horizontal") ...)) ;routingDirections ...);layerRules </pre>

Design Data Translators Reference

Mapping LEF to the Technology File

The electrical rules of the routing layer in the LEF file are mapped to the `techLayerProperties` subsection of the `layerDefinitions` section in the technology file.

The possible electrical rules are `areaCapacitance` (for CAPACITANCE), `edgeCapacitance` (for EDGECAPACITANCE), `sheetResistance` (for RESISTANCE), `height` (for HEIGHT), `thickness` (for THICKNESS), `shrinkage` (for SHRINKAGE), or `capMultiplier` (for CAPMULTIPLIER).

LEF	Technology File
<pre> propertyName propValue </pre>	<pre> layerRules (techLayerProperties ((propName layerName propValue));); </pre>
<pre> LAYER RX TYPE ROUTING; RESISTANCE RPERSQ 0.103; CAPACITANCE CPERSQDIST 0.000156 ; THICKNESS 1 ; SHRINKAGE 0.1 ; CAPMULTIPLIER 1 ; EDGECAPACITANCE 0.00000000000006 ; </pre>	<pre> layerDefinitions (... techLayerProperties (; (PropName Layer1 [Layer2] PropValue) ; (----- -----) (areaCapacitance RX 1.560000e-04) (capMultiplier RX 1) (edgeCapacitance RX 6.000000e-13) (height RX 4.0) (sheetResistance RX 0.103000) (shrinkage RX 0.1) (thickness RX 1.0) ...);techLayerProperties ...);layerRules </pre>

The antenna rules in the routing layers are mapped to the `antennaModels` in the `constraintGroups` section in the technology file.

Transistors with different oxide thickness or doping have different process antenna ratios. Therefore, an optional keyword `ANTENNAMODEL` has been added to the cut and routing layers. `ANTENNAMODEL` has four parameters: `OXIDE1`, `OXIDE2`, `OXIDE3`, or `OXIDE4`. The default parameter for `ANTENNAMODEL` is `OXIDE1`.

`ANTENNAAREAFACITOR` and `ANTENNASIDEAREAFACITOR` are not stored in `OpenAccess`.

Design Data Translators Reference

Mapping LEF to the Technology File

Note: Currently, only OXIDE1 and OXIDE2 are supported. OXIDE3 and OXIDE4 are ignored.

The examples from the LEF file and its corresponding technology file for ANTENAMODELS is as follows:

LEF Example

```
LAYER RX
TYPE ROUTING;
....
ANTENNAMODEL OXIDE1 ;# LEF5.5
ANTENNAAREARATIO 1 ;
ANTENNADIFFAREARATIO PWL ( ( 1 1 ) ( 2 2 ) ) ;
ANTENNACUMAREARATIO 3000 ;
ANTENNACUMDIFFAREARATIO PWL ( ( 1 4 ) ( 2 5 ) ) ;
ANTENNASIDEAREARATIO 7 ;
ANTENNADIFFSIDEAREARATIO PWL ( ( 1 8 ) ( 2 9 ) ) ;
ANTENNACUMSIDEAREARATIO 10 ;
ANTENNACUMDIFFSIDEAREARATIO PWL ( ( 1 11 ) ( 2 12 ) ) ;

ANTENNAMODEL OXIDE2 ;          # LEF5.5
ANTENNAAREARATIO 1 ;
ANTENNADIFFAREARATIO PWL ( ( 1 1 ) ( 2 2 ) ) ;
ANTENNACUMAREARATIO 3000 ;
ANTENNACUMDIFFAREARATIO PWL ( ( 1 4 ) ( 2 5 ) ) ;
ANTENNASIDEAREARATIO 7 ;
ANTENNADIFFSIDEAREARATIO PWL ( ( 1 8 ) ( 2 9 ) ) ;
ANTENNACUMSIDEAREARATIO 10 ;
ANTENNACUMDIFFSIDEAREARATIO PWL ( ( 1 11 ) ( 2 12 ) ) ;

ANTENNAMODEL OXIDE3 ;          # LEF5.5
ANTENNAAREARATIO 1 ;
ANTENNADIFFAREARATIO PWL ( ( 1 1 ) ( 2 2 ) ) ;
ANTENNACUMAREARATIO 3000 ;
ANTENNACUMDIFFAREARATIO PWL ( ( 1 4 ) ( 2 5 ) ) ;
#ANTENNAAREAFACOR 6 DIFFUSEONLY ;
ANTENNASIDEAREARATIO 7 ;
ANTENNADIFFSIDEAREARATIO PWL ( ( 1 8 ) ( 2 9 ) ) ;
ANTENNACUMSIDEAREARATIO 10 ;
ANTENNACUMDIFFSIDEAREARATIO PWL ( ( 1 11 ) ( 2 12 ) ) ;
#ANTENNASIDEAREAFACOR 2 DIFFUSEONLY ;

ANTENNAMODEL OXIDE4 ;          # LEF5.5
ANTENNAAREARATIO 1 ;
ANTENNADIFFAREARATIO PWL ( ( 1 1 ) ( 2 2 ) ) ;
ANTENNACUMAREARATIO 3000 ;
ANTENNACUMDIFFAREARATIO PWL ( ( 1 4 ) ( 2 5 ) ) ;
#ANTENNAAREAFACOR 6 DIFFUSEONLY ;
ANTENNASIDEAREARATIO 7 ;
ANTENNADIFFSIDEAREARATIO PWL ( ( 1 8 ) ( 2 9 ) ) ;
ANTENNACUMSIDEAREARATIO 10 ;
ANTENNACUMDIFFSIDEAREARATIO PWL ( ( 1 11 ) ( 2 12 ) ) ;
#ANTENNASIDEAREAFACOR 2 DIFFUSEONLY ;

...
```

Design Data Translators Reference

Mapping LEF to the Technology File

Technology File Example

```
constraintGroups (
  ....
  antennaModels (
    ; ( model )
    ; ( ----- )
    ( "default"
      antenna (
        ( areaRatio          "RX"          1 )
        ( diffAreaRatio      "RX"
          (
            ( 1 1 )
            ( 2 2 )
          )
        )
      ) ;antenna

      antenna (
        ( areaRatio          "CUT12"        14 )
        ( diffAreaRatio      "CUT12"
          (
            ( 2 15 )
            ( 3 16 )
          )
        )
      ) ;antenna

      antenna (
        ( areaRatio          "PC"           1 )
        ( diffAreaRatio      "PC"           1 )
      ) ;antenna

    ...
  ) ;default

  ( "second"
    antenna (
      ...
      cumulativeMetalAntenna (
        ( areaRatio          3000 )
        ( diffAreaRatio      21 )
      ) ;cumulativeMetalAntenna
      cumulativeViaAntenna (
        ( areaRatio          1700 )
        ( diffAreaRatio      3000 )
      ) ;cumulativeViaAntenna
    ) ;default
  )
  ...
)
```

Design Data Translators Reference

Mapping LEF to the Technology File

The AC and DC current density is mapped to the `currentDensityTables` subsection of the `layerRules` section in the technology file.

LEF	Technology File
<pre> LAYER <i>layerName</i> ... [ACCURRENTDENSITY PEAK { value FREQUENCY <i>freq_1</i> <i>freq_2</i> ... ; [WIDTH <i>width_1</i> <i>width_2</i> ... ;] TABLEENTRIES <i>v_freq_1_width_1</i> <i>v_freq_1_width_2</i> ... <i>v_freq_2_width_1</i> <i>v_freq_2_width_2</i> ;] ... END <i>layerName</i> </pre>	<pre> layerRules(currentDensityTables(;(rule <i>layer1</i> ; ((<i>index1Definitions</i> [<i>index2Definitions</i>]) [<i>defaultValue</i>]) ; (table)) </pre>

where,

Keyword	Values
<i>rule</i>	ACCURRENTDENSITY PEAK, ACCURRENTDENSITY AVERAGE, ACCURRENTDENSITY RMS, and DCCURRENTDENSITY AVERAGE.
<i>layer</i>	Layer name
<i>index1Definition</i>	AC Current Density for Cut and Routing Layers FREQUENCY (<i>f1 f2... fm</i>) nil DC Current Density for Routing Layer FREQUENCY (<i>f1 f2... fm</i>) nil WIDTH (<i>w1 w2... wm</i>) nil DC Current Density for Cut Layer FREQUENCY (<i>f1 f2... fm</i>) nil CUTAREA (<i>a1 a2... am</i>) nil

Design Data Translators Reference

Mapping LEF to the Technology File

Keyword	Values
<i>index2Definition</i> (optional)	AC Current Density for Routing Layer <code>WIDTH (w1 w2... wm) nil</code> AC Current Density for Cut Layer <code>CUTAREA (a1 a2... am) nil</code> DC Current Density <code>nil</code>
<i>defaultValue</i>	Value of the table depends on <i>index1Definition</i> and <i>index2Definition</i> .

Design Data Translators Reference

Mapping LEF to the Technology File

LEF Example

```
LAYER RX
TYPE ROUTING;
....
ACCURRENTDENSITY PEAK
FREQUENCY 1E6 100E6 400E6 ;
WIDTH      0.4 0.8 10.0 50.0 100.0 ;
TABLEENTRIES
#          w1      w2      w3      w4      w5
#f1
2.0E-6 1.9E-6 1.8E-6 1.7E-6 1.5E-6
#f2
1.4E-6 1.3E-6 1.2E-6 1.1E-6 1.0E-6
#f3
0.9E-6 0.8E-6 0.7E-6 0.6E-6 0.4E-6 ;
ACCURRENTDENSITY AVERAGE
FREQUENCY 1E6 100E6 400E6 ;
WIDTH      0.4 0.8 10.0 50.0 100.0 ;
TABLEENTRIES
# 5 widths above but values are for 6 widths
#          w1      w2      w3      w4      w5      w6
#f1
0.6E-6 0.5E-6 0.4E-6 0.3E-6 0.2E-6 0.1E-6
#f2
0.6E-6 0.5E-6 0.4E-6 0.3E-6 0.2E-6 0.1E-6
#f3
0.6E-6 0.5E-6 0.4E-6 0.3E-6 0.2E-6 0.1E-6 ;
ACCURRENTDENSITY RMS
FREQUENCY 100E6 400E6 800E6 ;
WIDTH      0.4 0.8 10.0 50.0 100.0 ;
TABLEENTRIES
#          w1      w2      w3      w4      w5
#f1
2.0E-6 1.9E-6 1.8E-6 1.7E-6 1.5E-6
#f2
1.4E-6 1.3E-6 1.2E-6 1.1E-6 1.0E-6
#f3
0.9E-6 0.8E-6 0.7E-6 0.6E-6 0.4E-6 ;
END RX
```

Design Data Translators Reference

Mapping LEF to the Technology File

Technology File Example

```
layerRules(  
  currentDensityTables(  
    ;( rule          layer1  
    ;  (( index1Definitions[index2Definitions]) [defaultValue] )(table))  
    ;( -----)  
    ( "peakACCurrentDensity""RX"  
      (("frequency" nil nil "width" nil nil))  
      (  
        (1000000.0 0.4)2e-06  
        (1000000.0 0.8)1.9e-06  
        (1000000.0 10.0)1.8e-06  
        (1000000.0 50.0)1.7e-06  
        (1000000.0 100.0)1.5e-06  
        (1e+08 0.4)1.4e-06  
        (1e+08 0.8)1.3e-06  
        (1e+08 10.0)1.2e-06  
        (1e+08 50.0)1.1e-06  
        (1e+08 100.0)1e-06  
        (4e+08 0.4)9e-07  
        (4e+08 0.8)8e-07  
        ...  
      )  
    )  
    ( "avgACCurrentDensity""RX"  
      (("frequency" nil nil "width" nil nil))  
      (  
        (1000000.0 0.4)6e-07  
        (1000000.0 0.8)5e-07  
        (1000000.0 10.0)4e-07  
        (1000000.0 50.0)3e-07  
        (1000000.0 100.0)2e-07  
        (1e+08 0.4)1e-07  
        (1e+08 0.8)6e-07  
        (1e+08 10.0)5e-07  
        (1e+08 50.0)4e-07  
        (1e+08 100.0)3e-07  
        (4e+08 0.4)2e-07  
        (4e+08 0.8)1e-07  
        ...  
      )  
    )  
    ( "rmsACCurrentDensity""RX"  
      (("frequency" nil nil "width" nil nil))  
      (  
        (1e+08 0.4)2e-06  
        (1e+08 0.8)1.9e-06  
        (1e+08 10.0)1.8e-06  
        (1e+08 50.0)1.7e-06  
        (1e+08 100.0)1.5e-06  
        (4e+08 0.4)1.4e-06  
        (4e+08 0.8)1.3e-06  
        (4e+08 10.0)1.2e-06  
        (4e+08 50.0)1.1e-06  
        (4e+08 100.0)1e-06  
        (8e+08 0.4)9e-07  
        ...  
      )  
    )  
  )  
)
```

Cut Layer

Following is the syntax of the cut layer in LEF 5.6.

```
TYPE CUT ;
    [SPACING minSpacing
        [ LAYER 2ndLayerName
            | ADJACENTCUTS {3 | 4} WITHIN distance]
            | CENTERTOCENTER]
    ;] ...
    [PROPERTY propName propName ;] ...
    [ACCURRENTDENSITY {PEAK | AVERAGE | RMS}
        { value
            | FREQUENCY freq_1 freq_2 ... ;
                [CUTAREA cutArea_1 cutArea_2 ... ;]
                TABLEENTRIES
                    v_freq_1_cutArea_1 v_freq_1_cutArea_2 ...
                    v_freq_2_cutArea_1 v_freq_2_cutArea_2 ...
                    ...
                } ;]
    [DCCURRENTDENSITY AVERAGE
        { value
            | CUTAREA cutArea_1 cutArea_2 ... ;
                TABLEENTRIES value_1 value_2 ...
            } ;]
    [ANTENNAMODEL {OXIDE1 | OXIDE2 | OXIDE3 | OXIDE4} ;] ...
    [ANTENNAAREARATIO value ;] ...
    [ANTENNADIFFAREARATIO {value | PWL ( ( d1 r1 ) ( d2 r2 ) ... )} ;] ...
    [ANTENNACUMAREARATIO value ;] ...
    [ANTENNACUMDIFFAREARATIO {value | PWL ( ( d1 r1 ) ( d2 r2 ) ... )} ;] ...
    [ANTENNAAREAFACOR value ;] ...
```

Following are the technology file equivalents for each line of code of the cut layer in the LEF file.

Design Data Translators Reference

Mapping LEF to the Technology File

The SPACING *minSpacing* statement is mapped as *minSpacing* in the *spacings* section, which is a subsection of the *constraintGroups* section in the technology file.

LEF	Technology File
<pre>LAYER CUT01 TYPE CUT; SPACING 0.24 CENTERTOCENTER ; SPACING 0.20 ; SPACING 0.24 CENTERTOCENTER ; SPACING 0.22 ADJACENTCUTS 5 WITHIN 0.25 ; ... END CUT01</pre>	<pre>constraintGroups(.... spacings((minCenterToCenterSpacing "CUT01" 0.24) (minSpacing "CUT01" 0.2) (viaSpacing "CUT01" (3 0.25 0.22)) (minWidth "RX" 0.711) ...);spacings ...)</pre>

A new optional statement with ADJACENTCUTS and WITHIN has been added to the existing SPACING syntax in the cut layer. In the above syntax,

- *minSpacing* and *distance* are floats in units of μm .
- *numCuts* is an integer that is either three or four.

If the ADJACENTCUTS *numCuts* WITHIN construct is specified, *minSpacing* applies only when the number of cuts are three or four specified within the *distance* spacing.

Mapping of other rules is same for cut and routing layers. For details, see [Routing Layer](#).

Design Data Translators Reference

Mapping LEF to the Technology File

Masterslice Layer

Masterslice layers are typically polysilicon layers and are only needed if the cell MACROs have pins on the polysilicon layer. In LEF, the overlap layer is used to store geometries that represent the prboundary. The overlap layer should normally be named OVERLAP.

LEF	Technology File
<pre>LAYER <i>MastersliceLayerName</i> TYPE {MASTERSLICE OVERLAP}; [PROPERTY <i>propName propVal</i> ;] ... END <i>MastersliceLayerName</i> }</pre>	<pre>techLayerProperties(;(<i>PropName Layer1</i>[<i>Layer2</i>] <i>PropValue</i>))</pre>
<pre>LAYER POLYS TYPE MASTERSLICE ; PROPERTY lsp "top" lip 1 lrp 2.3 ; END POLYS</pre>	<pre>techLayerProperties (;(<i>PropName Layer1</i>[<i>Layer2</i>] <i>PropValue</i>) ;(----- -----) (lip POLYS 1) (lrp POLYS 2.300000) (lsp POLYS "top") ;....);techLayerProperties</pre>

Implant Layer

Some new technologies require high-threshold and low-threshold cells to be combined together. This has been made possible by creating the implant layer.

Design Data Translators Reference

Mapping LEF to the Technology File

Width is mapped as `minWidth` in the `spacings` section, which is a subsection of the `constraintGroups` section of the technology file. Similarly, spacing is mapped to the `minSpacing` statement in the `spacings` section.

LEF	Technology File
<pre>LAYER <i>layerName</i> TYPE IMPLANT ; [WIDTH <i>minWidth</i> ;] [SPACING <i>minSpacing</i> [LAYER <i>layerName2</i> ;] ... [PROPERTY <i>propName propVal</i> ;] ... END <i>layerName</i></pre>	<pre>spacings((<i>minWidth layerName</i> <i>minWidthValue</i>)) layerRules (... functions (; (<i>layer</i> <i>function</i> [<i>maskNumber</i>]) ; (----- ----- -----) ... (<i>implant1</i> "nplus" 13) (<i>implant2</i> "nplus" 14)) ;functions ...);layerRules constraintGroups (... spacings ((<i>minWidth</i> "<i>implant1</i>" 0.5) (<i>minSpacing</i> "<i>implant1</i>" 0.5) (<i>minWidth</i> "<i>implant2</i>" 0.5) (<i>minSpacing</i> "<i>implant2</i>" 0.5)) ;spacings ...)</pre>

BEGINEXT Statement

The LEF extension is not stored in OpenAccess. The LEF syntax is:

```
[BEGINEXT "tag"
  extension
ENDEXT]
```


BUSBITCHARS Statement

The OpenAccess tools use *delimiterPair* to identify bus bit characters for name mapping.

BUSBITCHARS is mapped in the technology file in the `techParams` section, which is a subsection of the `controls` section.

LEF	Technology File
BUSBITCHARS " <i>delimiterPair</i> ";	(LEFDEF_BUSBIT_CHARS " <i>value</i> ")
BUSBITCHARS "<>" ;	<pre>controls(techParams(;(parameter value) ;(----- -) (LEFDEF_BUSBIT_CHARS "<>") ...);techParams ...);controls</pre>

CLEARANCEMEASURE Statement

Specifies the clearance distance used for spacing rule checks.

```
CLEARANCEMEASURE {MAXXY | EUCLIDEAN};
```

In the above syntax, `MAXXY` uses the larger X and Y distances for spacing rule checks. Similarly, `EUCLIDEAN` uses Euclidean distance for spacing rule check. The default value is `EUCLIDEAN`.

MACRO Section

The `MACRO` section in the LEF file contains definitions for the macrocell models in the library. Macros are not mapped in the technology file.

DIVIDERCHAR Statement

The LEF divider character is used to set the divider character in the `oaLefNS` namespace.

Design Data Translators Reference

Mapping LEF to the Technology File

DIVIDERCHAR is mapped in the technology file in the `techParams` section, which is a subsection of the `controls` section.

LEF	Technology File
DIVIDERCHARS " <i>character</i> ";	(LEFDEF_DIVIDER_CHARS " <i>value</i> ")
DIVIDERCHAR "<>" ;	<pre>controls(techParams(;(parameter value) ;(----- -----) (LEFDEF_DIVIDER_CHARS ":") ...);techParams ...);controls</pre>

MANUFACTURINGGRID Statement

Manufacturing grid is used to control the distance between the two snap grids.

The snap grid controls pointer movement in the interactive drawing and editing modes. When the snap grid is defined, the pointer snaps to the closest grid point as you move it within the work area.

Distance between the two snap grids must be a multiple of the manufacturing grid. The value specified for the manufacturing grid should be positive. The numeric range is `float`.

MANUFACTURINGGRID is mapped in the technology file in the `controls` section.

LEF	Technology File
[MANUFACTURINGGRID <i>value</i> ;]	<code>mfgGridResolution (<i>grid_value</i>)</code>
MANUFACTURINGGRID 0.0005 ;	<pre>controls(... mfgGridResolution((0.000500)) ;mfgGridResolution);controls</pre>

Design Data Translators Reference

Mapping LEF to the Technology File

Maximum Stacked Via

Some processes have an overall limit on the number of single-cut stacked vias that are allowed on top of each other. The `MAXVIASTACK` keyword has been added to limit the number of stacked vias. Currently, four stacked vias are permitted on top of each other.

The syntax of the `MAXVIASTACK` statement in LEF 5.6 is as follows.

```
[MAXVIASTACK value [ RANGE bottomLayer topLayer];]
```

where,

Syntax	Description
<i>value</i>	An integer that limits the number of single stacked vias that are allowed on top of each other.
<i>bottomLayer</i> <i>topLayer</i>	Routing layer names

`MAXVIASTACK` is mapped in the technology file in the `viaStackingLimits` section, which is a subsection of the `constraintGroups` section.

LEF	Technology File
<code>MAXVIASTACK value [RANGE bottomLayer topLayer];</code>	<code>viaStackingLimits ((value [bottomLayer topLayer]))</code>
<pre># Only one MAXVIASTACK is allowed in a lef file MAXVIASTACK 3 RANGE M1 MT ;</pre>	<pre>constraintGroups(.... viaStackingLimits((3 "M1" "MT")) ;viaStackingLimits ...);constraintGroups</pre>

If the specified layer names are not routing layer names, the `RANGE` keyword is ignored. The `MAXVIASTACK` should follow the layer definitions in the LEF file.

NAMECASESENSITIVE Statement

The LEF NAMECASESENSITIVE statement is ignored since OpenAccess is always case-sensitive. When you write LEF, always include NAMECASESENSITIVE ON. The LEF syntax is as follows:

```
NAMECASESENSITIVE {ON | OFF} ;
```

SITE Section

The SITE section defines the types and properties of placement sites available in the library for. Each SITE section in a LEF file maps to a cell in the OpenAccess library.

SITE statements are mapped in the technology file in the scalarSiteDefs section.

LEF	Technology File
<pre>{ SITE <i>siteName</i> CLASS { PAD CORE } SYMMETRY { X Y R90 }...; SIZE <i>width</i> BY <i>height</i>; END <i>siteName</i> }...</pre>	<pre>siteDefs(; a list of all site definitions. (<i>site1</i> ...) ...)</pre>
<pre>SITE CORE1 CLASS CORE ; SYMMETRY X ; SIZE 67.2 BY 6 ; END CORE1</pre>	<pre>scalarSiteDefs(;(siteDefName type width height symInX symInY symInR90) ;(-----) -----) (CORE1 core 67.2 6.0 t nil nil) (CORE core 3.6 28.8 nil t nil) (MRCORE core 3.6 28.8 nil t nil) (IOWIRED pad 57.6 432.0 nil nil nil) (IOTEST pad 57.6 432.0 nil nil nil) (IODI1 pad 57.6 432.0 nil nil nil) ...) ;scalarSiteDefs</pre>
<pre>SITE CORE CLASS CORE ; SIZE 3.6 BY 28.8 ; SYMMETRY Y ; END CORE</pre>	
<pre>SITE MRCORE CLASS CORE ; SIZE 3.6 BY 28.8 ; SYMMETRY Y ; END MRCORE</pre>	

SPACING Section

The `SPACING` section in the LEF file provides for the same net spacing rules and stacked vias.

LEF	Technology File
<pre>[SPACING [SAMENET <i>layerName</i> <i>layerName minSpace</i> [STACK];] ... END SPACING]</pre>	<pre>spacings((minSameNetSpacing <i>layerName value</i>) stackable <i>layerName</i> t/nil))</pre>
<pre>SPACING SAMENET CUT01 CA 0.1 STACK ; END SPACING</pre>	<pre>constraintGroups (spacings ((minSameNetSpacing "CA" "V1" 1.5) (stackable "CA" "V1" t) (minSameNetSpacing "M1" 0.28) (minSameNetSpacing "V1" "V2" 1.5) (stackable "V1" "V2" t) ...); spacings ...); constraintGroups</pre>

USEMINSPACING Statement

Defines how minimum spacing is calculated for an obstruction (blockage) or pin geometries. It is mapped as the `USEMINSPACINGOBS` or `USEMINSPACINGPIN` property on the database object in the OpenAccess library.

```
USEMINSPACING {OBS | PIN} {ON | OFF};...
```

In the above syntax, `OBS | PIN` specifies whether the minimum spacing is applied to pin geometries or obstructions. The option `ON | OFF` specifies the calculation method for minimum spacing. If you specify `ON`, the spacing is the minimum layer spacing. By default, the value is assumed to be "OFF", in which case the property is not created.

Design Data Translators Reference

Mapping LEF to the Technology File

VIA Statement

VIA is mapped in the technology file in the `customViaDefs` section, which is a subsection of the `viaDef` section.

LEF	Technology File
<pre> VIA <i>viaName</i> [DEFAULT] [TOPOFSTACKONLY] [FOREIGN <i>foreignCellName</i> [pt [<i>orient</i>]]]; [RESISTANCE <i>value</i> ;] {LAYER <i>layerName</i> ; {RECT <i>pt pt</i> ;} ...} ... [PROPERTY <i>propName propVal</i> :] ... END <i>viaName</i> </pre>	<pre> viaDefs(customViaDefs((<i>viaDefName</i> <i>libName cellName viewName layer1</i> <i>layer2 resistancePerCut</i>) </pre>
<pre> VIA RX_PC DEFAULT # VIA RESISTANCE should be discouraged. Will become obsolete. # Instead RESISTANCE in LAYER will be taken. RESISTANCE 2.012 ; LAYER RX ; RECT -0.7 -0.7 0.7 0.7 ; LAYER CUT12 ; RECT -0.25 -0.25 0.25 0.25 ; LAYER PC ; RECT -0.6 -0.6 0.6 0.6 ; PROPERTY stringProperty "DEFAULT" realProperty 32.33 COUNT 34 ; END RX_PC </pre>	<pre> viaDefs(customViaDefs(;(<i>viaDefName libName cellName viewName</i> <i>layer1 layer2 resistancePerCut</i>) ;(----- -----) (RX_PC techLibfinal RX_PC via RX_PC 2.012) (PC_M1 techLibfinal PC_M1 via PC_M1 1.0) (M1_M2 techLibfinal M1_M2 via M1_M2 1.5) (M2_M3 techLibfinal M2_M3 via M2_M3 1.5) (M3_MT techLibfinal M3_MT via M3_MT 1.5) (M2_M3_PWR techLibfinal M2_M3_PWR via M2_M3 1.2) (VIACENTER12 techLibfinal VIACENTER12 via M1_M2 0.48) (nd1VIARX0 techLibfinal nd1VIARX0 via RX_PC 0.2) (nd1VIA01 techLibfinal nd1VIA01 via PC_M1 0.2) (nd1VIA12 techLibfinal nd1VIA12 via M1_M2 0.2) (nd1VIA23 techLibfinal nd1VIA23 via M2_M3 0.0) (nd1VIA34 techLibfinal nd1VIA34 via M3_MT 0.0));customViaDefs) ;viaDefs </pre>
<pre> VIA PC_M1 DEFAULT RESISTANCE 1 ; LAYER PC ; RECT -0.6 -0.6 0.6 0.6 ; LAYER CA ; RECT -0.25 -0.25 0.25 0.25 ; LAYER M1 ; RECT -0.6 -0.6 0.6 0.6 ; END PC_M1 </pre>	<pre>) ;viaDefs </pre>

Design Data Translators Reference

Mapping LEF to the Technology File

VIARULE Statement

The `VIARULE` section in the LEF file specifies vias to be used at the intersection of special wires of the same net. A via rule can refer to vias that are defined elsewhere in the LEF file or generate its own vias.

`VIARULE` is mapped in the technology file in the `viaSpecs` section.

LEF	Technology File
<pre> VIARULE <i>viaRuleName</i> LAYER <i>layerName</i> ; DIRECTION {HORIZONTAL VERTICAL} ; [WIDTH <i>minWidth</i> TO <i>maxWidth</i> ;] LAYER <i>layerName</i> ; DIRECTION {HORIZONTAL VERTICAL} ; [WIDTH <i>minWidth</i> TO <i>maxWidth</i> ;] {VIA <i>viaName</i> ;} ... [PROPERTY <i>propName</i> <i>propVal</i> ;] ... END <i>viaRuleName</i> </pre>	<pre> viaSpecs((<i>layer1 layer2 (viaDefName ...)</i> [(<i>layer1MinWidth layer1MaxWidth</i> <i>layer2MinWidth layer2MaxWidth</i> (<i>viaDefName ...</i>))])) </pre>
<pre> VIARULE VIALIST12 LAYER M1 ; DIRECTION VERTICAL ; WIDTH 5.0 TO 10.0 ; LAYER M2 ; DIRECTION HORIZONTAL ; WIDTH 5.5 TO 10.5 ; VIA VIACENTER12 ; PROPERTY vrsp "new" vrip 1 vrrp 4.5 ; END VIALIST12 </pre>	<pre> viaSpecs((M1 M2 nil ((1.0 5.0 1.5 5.5 ("VIAGEN12")) (1.0 5.0 5.5 10.5005 ("VIAGEN12")) (1.0 5.0 10.5005 15.5005 ("VIAGEN12")) (5.0 10.0005 1.5 5.5 ("VIAGEN12")) (5.0 10.0005 5.5 10.5005 ("VIACENTER12" "VIAGEN12")) (5.0 10.0005 10.5005 15.5005 ("VIAGEN12")) (10.0005 15.0005 1.5 5.5 ("VIAGEN12")) (10.0005 15.0005 5.5 10.5005 ("VIAGEN12")) (10.0005 15.0005 10.5005 15.5005 ("VIAGEN12")))) (M2 M3 nil ((0.2 100.0005 0.2 100.0005 ("VIAGEN23"))))) ;viaSpecs </pre>

VIARULE GENERATE

In LEF, a via rule using generated vias has the following syntax:

The LEF VIARULE GENERATE objects are stored in OpenAccess by using an `oaViaSpec`. This is similar to the normal LEF VIARULE, however, the VIARULE GENERATE uses a single `oaStdViaDef` object (stored in `oaViaDefArray(0)`) instead of the `oaCustomViaDef` objects.

The generating vias section is mapped to the `standardViaDefs` subsection `viaDefs` section of the technology file.

LEF	Technology File
<pre> VIARULE <i>viaRuleName</i> GENERATE LAYER <i>routingLayerName</i> ; { DIRECTION {HORIZONTAL VERTICAL} ; [OVERHANG <i>overhang</i> ;] [METALOVERHANG <i>metalOverhang</i> ;] ENCLOSURE <i>overhang1</i> <i>overhang2</i> ;} [WIDTH <i>minWidth</i> TO <i>maxWidth</i> ;] LAYER <i>routingLayerName</i> ; { DIRECTION {HORIZONTAL VERTICAL} ; [OVERHANG <i>overhang</i> ;] [METALOVERHANG <i>metalOverhang</i> ;] ENCLOSURE <i>overhang1</i> <i>overhang2</i> ;} [WIDTH <i>minWidth</i> TO <i>maxWidth</i> ;] LAYER <i>cutLayerName</i> ; RECT <i>pt pt</i> ; SPACING <i>xSpacing</i> BY <i>ySpacing</i> ; [RESISTANCE <i>resistancePerCut</i> ;] </pre>	<pre> standardViaDefs((<i>viaDefName</i> <i>layer1 layer2</i> (<i>cutLayer cutWidth cutHeight</i> [<i>resistancePerCut</i>]) (<i>cutRowscutCol(cutSpace)</i>) (<i>layer1Enc</i>) (<i>layer2Enc</i>) (<i>layer1Offset</i>) (<i>layer2Offset</i>) (<i>origOffset</i>) [<i>implant1</i> (<i>implant1Enc</i>) [<i>implant2</i> (<i>implant2Enc</i>)]])) </pre>

Design Data Translators Reference

Mapping LEF to the Technology File

LEF

Technology File

```
# LEF 5.6 adds DEFAULT to VIA GENERATE viaDefs(
VIARULE VIAGEN12 GENERATE
  LAYER M1 ;
    ENCLOSURE 0.03 0.01 ;
      #2 sides need >= 0.03, 2 other
      sides >= 0.01
    WIDTH 1.0 TO 15.0 ;
  LAYER M2 ;
    ENCLOSURE 0.05 0.01 ;
      #2 sides need >= 0.05, 2 other
      sides >= 0.01
    WIDTH 1.5 TO 15.5 ;
  LAYER V1 ;
    RECT -0.1 -0.1 0.1 0.1 ;
      #cut is .20 by .20
    SPACING 0.40 BY 0.40 ;
      #center-to-center spacing
    RESISTANCE 20 ; #ohms per cut
END VIAGEN12

VIARULE VIAGEN23 GENERATE
  LAYER M2 ;
    ENCLOSURE 0.10 0.01 ; # LEF5.5
    WIDTH 0.2 TO 100.0 ;
  LAYER M3 ;
    ENCLOSURE 0.10 0.01 ; # LEF5.5
    WIDTH 0.2 TO 100.0 ;
  LAYER V2 ;
    RECT -0.8 -0.8 0.8 0.8 ;
    SPACING 5.6 BY 6.0 ;
    RESISTANCE 0.2 ; # Timing
END VIAGEN23
```

Design Data Translators Reference

Mapping LEF to the Technology File

List of XStream Messages

The following table provides a list of error, warning, and information messages generated by the XStream translator. To view the table legend, see [Legend](#).

Message ID	Severity	Message
XSTRM-1	Warning	Value of pin attribute number is out of range 1-127. Pins will not be preserved.
XSTRM-2	Error	Library name '%s' is invalid in UNIX namespace.
XSTRM-3	Error	Tech library name '%s' is invalid in UNIX namespace.
XSTRM-4	Error	Failed to locate run directory '%s'. Check that the directory exists and that you have permissions to write in this directory.
XSTRM-5	UN-USED	Run directory '%s' does not exist or is not a directory.
XSTRM-6	Error	Failed to open the template file '%s'. Check that the template file exists and is a regular UNIX file.
XSTRM-7	Error	Failed to open cell name map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-8	Error	Failed to open text font map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-9	Error	Failed to open property map file '%s'. Check that the file exists in the run directory and that you have permission to read it.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-10	Error	Failed to open layer map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-11	Error	Failed to open refLibList file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-12	UN-USED	Failed to open pinTextMap file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-13	Error	Failed to open input Stream file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-14	Error	Failed to open ASCII technology file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-15	Error	Failed to create Stream file '%s' in the run directory. Check that you have permission to create this file.
XSTRM-16	Error	Failed to create summary file '%s' in the run directory. Check that you have permission to create this file.
XSTRM-17	Error	Failed to create log file '%s' in the run directory. Check that you have permission to create this file.
XSTRM-18	Warning	Performing automatic layer mapping because either the layer mapping information is missing or the '-layerMap' option is using the XST_AUTO_LM value. However, failed to create the automatic layer map file '%s' in the run directory. Ensure that you have the write permission to create this file.
XSTRM-19	Warning	Failed to create the automatic cell name map file '%s' in the run directory. Ensure that you have the write permission to create this file.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-20	Warning	Output Stream file '%s' already exists. It will be overwritten.
XSTRM-21	Error	Failed to open the library '%s'. Check that library is present in the path defined in the 'cds.lib' file.
XSTRM-22	Error	Failed to open the technology file '%s' from library '%s'. Check that the library exists in the location defined in the 'cds.lib' file and that you have read permission to the technology file.
XSTRM-23	UN-USED	Purpose '%s' is not defined in the technology file. All the blockages in the design will be considered on purpose 'boundary'.
XSTRM-24	UN-USED	Purpose '%s' is not defined in the technology file. All the blockages in the design will be considered on purpose 'drawing'.
XSTRM-25	Warning	The layer map file '%s' contains an invalid record at line number '%d' and will be ignored. This is because the format of the mapping entry in the layer map file '%s' is invalid. The valid mapping entry format is: <layerName purposeName strmLayer strmType [Material [MaskNumber]] [Qualifier] [PhotomaskColor ColorState]>, where 'PhotomaskColor' and 'ColorState' are valid only when the '-enableColoring' option is set and the permissible range for strmLayer and strmType is 0 - %d. Now, update the mapping entry and then translate again.
XSTRM-26	Warning	The font map file '%s' contains an invalid record at line number '%d' and will be ignored. This is because the format of the mapping entry in the font map file '%s' is invalid. Update the mapping entry format to: <fontNumber fontName> and try again.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-27	Warning	The property map file '%s' contains an invalid record at line number '%d' and will be ignored. This is because the format of the mapping entry in the property map file '%s' is invalid. Update the mapping entry format to: <strmAttribute objectType propName> and try again.
XSTRM-28	Warning	The cell map file '%s' contains an invalid record at line number '%d' and will be ignored. This is because the format of a mapping entry in the cell map file '%s' is invalid. Update the mapping entry format to: <libName cellName viewName structName> and try again.
XSTRM-29	Warning	The refLib file contains invalid record at line number %d. This record in the refLib file will be ignored.
XSTRM-30	UN-USED	
XSTRM-31	Warning	Layer map file '%s' contains a duplicate entry for Stream layer-datatype '%d:%d' at line '%d'. Therefore, this entry will be ignored during translation.
XSTRM-32	Warning	The layer map file '%s' contains unknown layer '%s' at line number '%d' and will be ignored. This is because the layer name specified in the layer map file is not defined in the technology file. Ensure that the layer name specified in the layer map file is both valid and defined in the technology file.
XSTRM-33	Warning	The layer map file '%s' contains unknown purpose '%s' at line number '%d' and will be ignored. This is because the purpose name specified in the layer map file is not defined in the technology file. Ensure that the purpose name specified in the layer map file is both valid and defined in the technology file.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-34	Warning	The objects in the layer-purpose pair '%u:%u' are ignored. This is because either the layer-purpose pair used in the design is not specified in the layer map file or the layer/purpose is not defined in technology file. Ensure that the layer-purpose pair used in the design is specified in the layer map file and the layer/purpose is defined in technology file.
XSTRM-35	Warning	The objects in the layer-purpose pair '%s:%s' are ignored. This is because the layer-purpose pair used in the design is not specified in the layer map file. Specify the layer-purpose pair in the layer map file and try again.
XSTRM-36	Warning	The objects in the stream layer-datatype '%d:%d' are ignored. This is because the stream layer-datatype pair is either not specified in the layer map file or the entry is invalid and was therefore ignored. Specify the stream layer-datatype pair in the layer map file or correct the wrong entry and try again.
XSTRM-37	UN-USED	Layer-datatype range is exhausted. All the shapes on layer-purpose pair '%d:%d' will be ignored.
XSTRM-38	Warning	The library %s in the refLibList file %s could not be found. This is because the library '%s' specified in refLibList file does not exist. Ensure that the library '%s' is defined in the library definition file and also exists in the location specified in the library definition file.
XSTRM-39	Warning	The font map file contains invalid font name '%s' at line number %d. The valid font names are: EuroStyle, Gothic, Math, Roman, Script, Stick, Fixed, Swedish, and MilSpec. Specify a valid font name and try again.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-40	Warning	The font map file contains invalid Stream font number '%d' at line number %d. The valid font numbers are: 0, 1, 2, and 3. Specify the valid font numbers and try again.
XSTRM-41	Warning	The property map file '%s' contains invalid record at line number '%d'. The valid object types are: Dot, Pin, Line, Path, Polygon, Rect, Text, ArrayInst, ScalarInst, Arc, Donut, Ellipse, PathSeg, Route, Marker, Row, AreaBoundary, PRBoundary, SnapBoundary, ClusterBoundary, AreaBlockage, AreaHalo, LayerBlockage, and LayerHalo. Specify the valid object types and try again.
XSTRM-42	Warning	The property map file '%s' contains an invalid format at line number '%d' and will be ignored. The valid mapping entry format is [<-fp/-fv> SP_CH_OPTSEP <-s 'property separator'>]. If '-fp' option is specified during StreamOut, properties are written as <name:value> pairs; and during GDSII/ StreamIn, properties are assumed to be present in the stream file as <name:value> pairs. If '-fv' option is specified during StreamOut, only values of properties are written in the property map file. You can also specify the property separator as [-s 'property separator'] without using '-fp' or '-fv'.
XSTRM-43	Warning	Stream Attribute %d specified in the property map file conflicts with the value of '-pinAttNum' option. This entry will be ignored.
XSTRM-44	UN-USED	Pin text map file contains an unknown layer '%s' at line '%d'. This entry in the pin text map file will be ignored.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-45	Warning	Mapping for layer '%d' will not be written in %s file because layer '%d', which is used in the input design is not defined in the technology file. Ensure that the layer exists in the technology file or use the correct technology file.
XSTRM-46	Warning	Mapping for purpose '%d' will not be written in %s file because purpose '%d', which is used in the input design is not defined in the technology file. Ensure that the purpose exists in the technology file or use the correct technology file.
XSTRM-47	Warning	Library '%s' is used in the design but does not exist. It will not be listed in the summary file.
XSTRM-48	Error	Failed to flatten the Pcell instance '%s' in the cellview '%s/%s/%s'. XStream Out flattens Pcell instances when the Flatten Pcells option is used. Ensure that the Pcell instance is correct by opening the cellview '%s/%s/%s' and flattening the instance in Virtuoso.
XSTRM-49	Error	The pin information could not be translated. This is because the value 'text' is specified for the Convert Pin To option and Stream Out is unable to get the ID associated with the pin connected with the figure at origin (%d, %d). Rectify the input database to prevent data corruption and translate again.
XSTRM-50	Error	The pin information could not be translated. This is because the Pin Attribute Num option is specified during Stream Out to translate the pin information, and Stream Out was unable to get the ID associated with the pin connected with the figure at origin (%d, %d). Rectify the input database to prevent data corruption and translate again.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-51	Error	The pin information could not be translated. This is because the value 'text' is specified for the Convert Pin To option and Stream Out was unable to get the pin name associated with the pin connected with the figure at origin (%d, %d). Rectify the input database to prevent data corruption and translate again.
XSTRM-52	Error	The pin information could not be translated. This is because the Pin Attribute Num option is specified during Stream Out to translate the pin information, and Stream Out was unable to get the pin name associated with the pin connected with the figure at origin (%d, %d). Rectify the input database to prevent data corruption and translate again.
XSTRM-53	Error	The pin information could not be translated. This is because the Pin Attribute Num option is specified during XStream Out to translate the pin information, and XStream Out was unable to get the term name associated with the pin. Rectify the input database to prevent data corruption and translate again.
XSTRM-54	UN-USED	Symbolic pins will not be translated as text because pinTextMap file is not provided.
XSTRM-55	Warning	Vector instance '%s' encountered. It will be translated as SREF in the Stream file.
XSTRM-56	Warning	Library '%s' specified in cell map file is not present in refLibList, its mapping will be replaced with the target library.
XSTRM-57	Warning	Failed to create the new layer '%s'. All the shapes on this layer will be dropped.
XSTRM-58	Warning	Failed to create the new layer-purpose pair '%s:%s'. It will be skipped and all the shapes on this layer-purpose pair will be dropped.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-59	Warning	Failed to create the new purpose '%s'. All the shapes on this purpose will be dropped.
XSTRM-60	Info	Creating a new purpose %s in the technology file.
XSTRM-61	Info	Creating a new layer %s in the technology file.
XSTRM-62	Info	Creating a new layer-purpose pair '%s:%s' in the technology file.
XSTRM-63	Warning	Failed to set the 'visibility attribute' of a newly created layer-purpose pair '%s:%s'.
XSTRM-64	Warning	Failed to set the 'layer selectable' property of a newly created layer-purpose pair '%s:%s'.
XSTRM-65	Warning	Failed to set the 'priority' of a newly created layer-purpose pair '%s:%s'.
XSTRM-66	Warning	Failed to update the 'packet name' of a newly created layer-purpose pair '%s:%s'.
XSTRM-67	Warning	Failed to change the validity of a newly created layer-purpose pair '%s:%s'.
XSTRM-68	Warning	Ignoring 'Load ASCII Tech File' option because target library %s is already attached' to the technology library %s.
XSTRM-69	Info	All the libraries specified in 'cds.lib' will be considered as reference libraries. This is because '-refLibList' option is specified as \XST_CDS_LIB\".
XSTRM-70	Info	The 'Reference Library list' and the 'Keep Stream Cells' options have been used together. Any cell defined in the Stream file will not be searched in reference libraries.
XSTRM-71	Warning	The '-keepStreamCells' option has been overridden for the cells in the target library because %s.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-72	Warning	The 'Load ASCII tech file' and 'Attach Technology Library' options have been used together. The 'Load ASCII tech file' option will be ignored.
XSTRM-73	Warning	The library name column in the cell name map file will be ignored because reference library list is not specified.
XSTRM-74	Error	Translation cannot be continued because layer mapping cannot be done. This is because the layer map is not specified by the '-layerMap' option and the technology library '%s' also does not contain '%s.layermap'. Additionally, automatic layer mapping cannot be done because the technology file is opened in read-only mode due to the library '%s' being attached to it. Use a layer map file to continue.
XSTRM-75	Warning	Target library '%s' is attached to the technology library '%s'. Therefore, the technology file is opened in read-only mode. All the undefined layer-purpose pairs will be dropped.
XSTRM-76	Error	Failed to open technology file '%s:%s' in append mode. Therefore, automatic layer mapping cannot be done. Use the layer map file to continue.
XSTRM-77	UN-USED	Failed to open the technology file '%s:%s' in append mode. All the undefined layer-purpose pairs will be dropped.
XSTRM-78	Error	Failed to load ASCII technology file %s, specified in 'Load ASCII Tech File' option.
XSTRM-79	Error	Failed to attach technology file from the library %s because a technology file already exists in the target library.
XSTRM-80	Error	The technology library '%s' specified in the 'Attach Technology Library' option does not exist.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-81	Error	The technology library '%s' attached to the target library does not exist.
XSTRM-82	UN-USED	Failed to create the library %s.
XSTRM-83	Error	Technology file '%s' cannot be opened from the library '%s'. Ensure that the library '%s' exists and have appropriate permissions.
XSTRM-84	Warning	The 'Skip Undefined LPPs' option will be ignored because a new library is being created.
XSTRM-85	Warning	Case conversion resulted in mapping of STRUCTURE %s to cell name %s. During XStream In, translation continues by overwriting cellview data. However, to avoid the collision do not specify the case conversion option.
XSTRM-86	Warning	Memory prediction is disabled because system parameters could not be found.
XSTRM-87	Warning	Process may require more virtual memory than 32-bit system can handle. You may need to run 64-bit version of the product. Available Memory:%llu MB\tPredicted Memory:%llu MB.
XSTRM-88	Warning	Process might require more virtual memory than available on the system. Available Memory:%llu MB\tPredicted Memory:%llu MB.
XSTRM-89	Error	Failed to flatten via of viaDef '%s' at (%d, %d) in the cellview '%s/%s/%s'. XStream Out flattens a via in the specified cellview when the Flatten Vias option is used. Ensure that the via in the specified cellview is correct by opening the cellview '%s/%s/%s' and flattening the via in Virtuoso.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-90	Warning	Process may require more virtual memory than available on the system for a process. You can increase this limit by using the 'limit' command. Available Memory:%llu MB\tPredicted Memory:%llu MB.
XSTRM-91	Info	Process may require virtual memory near to what it is available on the system for a process. You can increase this limit by using the 'limit' command. Available Memory:%llu MB\tPredicted Memory:%llu MB.
XSTRM-92	UN-USED	Process may require more data segment than available on the system for a process. You can increase this limit by using the 'limit' command. Available Memory:%llu\tPredicted Memory :%llu.
XSTRM-93	UN-USED	Process may require data segment near to available on the system for a process. You can increase this limit by using the 'limit' command. Available Memory:%llu\tPredicted Memory:%llu.
XSTRM-94	UN-USED	The cellView %s/%s is not found in the mapped library %s. This cell will not be translated, and all the instances of this cell will be unbound.
XSTRM-95	UN-USED	Creating an unbound instance of the cellview '%s/%s/%s' as the master was not found in the Stream file.
XSTRM-96	UN-USED	Creating an unbound instance because the master cellview '%s/%s' was not found in the Stream file or the specified reference library '%s'.
XSTRM-97	UN-USED	Creating an unbound instance of the cellview '%s/%s' as the master cannot be found in the mapped library '%s'.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-98	UN-USED	Creating an unbound instance with library as target library %s because the master cellview '%s/%s' was not found in any reference library.
XSTRM-99	UN-USED	Creating an unbound instance with library as target library %s, because the master cellview '%s/%s' was not found in the Stream file or in any of the reference libraries.
XSTRM-100	Error	Instance Name %s has illegal characters. This instance is dropped.
XSTRM-101	Warning	An AREF record in the Stream file contains the illegal angle '%f', this will not be translated. Angles supported by Stream In are 0, 90, 180, 270.
XSTRM-102	Error	The '%s' option specified is invalid. View the available options using the \-help\" command and then specify the valid option."
XSTRM-103	Error	The '%s' option specified in the template file is invalid%s. View the available options using the \-help\" command and then specify the valid option."
XSTRM-104	Warning	The Stream attributes associated with some objects in the Stream file exists multiple times or have multiple properties separated by a property separator. Only the first property value is preserved and the others are ignored.
XSTRM-105	Info	Property map file contains duplicate entry for Stream attribute number '%d', object type '%s' at line number '%d'. This entry will be ignored.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-106	Warning	The undefined layer-purpose pair are skipped because either the Skip Undefined LPPs option is specified or XStream In into an existing library is performed. To avoid skipping of undefined LPPs, ensure that the Skip Undefined LPPs option is not specified or XStream In should be performed into a new library.
XSTRM-107	Warning	Existing cells in the target library will be overwritten if the cell names in the Stream file and the target library are the same. To prevent this, %s.
XSTRM-108	Error	Layer map file is not being processed because the technology file is not available.
XSTRM-109	UN-USED	Pin text map file is not being processed because the technology file is not available.
XSTRM-110	UN-USED	Layer %s not found in the technology file.
XSTRM-111	UN-USED	Purpose %s not found in the technology file.
XSTRM-112	Error	Failed to open the user SKILL file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-113	Warning	Unable to load the SKILL file %s because of the undefined SKILL function and syntactical error. Specify the SKILL function with valid syntax and translate again.
XSTRM-114	Warning	Error in the definition of the SKILL function '%s'. The SKILL function maps the stream %s to %s. Specify the correct definition of the SKILL function and restart the translation.
XSTRM-115	Warning	Error in the definition of 'textFontMap'. Invalid font name for the Stream font number %d
XSTRM-116	Error	Illegal font '%s' mapped to the Stream font number %d. This mapping will be ignored.
XSTRM-117	Warning	Error in the definition of 'piCellNameMap'. Cell name '%s' is mapped to NULL

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-118	Warning	Error in the definition of 'piLayerMap'. No layer string returned for layer '%u' and datatype '%u'.
XSTRM-119	Warning	Error in the definition of 'piLayerMap'. No purpose string returned for layer '%u' and datatype '%u'.
XSTRM-120	Error	Translation failed because the file size of GDSII/Stream file '%s' is zero. Ensure that the correct GDSII file is specified.
XSTRM-121	UN-USED	Could not process the library '%s'.
XSTRM-122	UN-USED	
XSTRM-123	UN-USED	
XSTRM-124	UN-USED	
XSTRM-125	UN-USED	
XSTRM-126	UN-USED	
XSTRM-127	UN-USED	
XSTRM-128	UN-USED	
XSTRM-129	UN-USED	
XSTRM-130	Warning	Error occurred in the user SKILL function 'textFontMap' while mapping '%s'.
XSTRM-131	Warning	Font name '%s' mapped to invalid Stream font number '%d' by the user SKILL function 'textFontMap'. This mapping will be ignored.
XSTRM-132	Warning	Error occurred in the user SKILL function 'poCellNameMap'.
XSTRM-133	Warning	Error occurred in the user SKILL function 'poLayerMap'.
XSTRM-134	Warning	Error in the definition of 'poLayerMap'. Invalid layer number returned for layer '%s'.
XSTRM-135	Warning	Error in the definition of 'poLayerMap'. Invalid layer number returned for purpose '%s'.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-136	Warning	Error occurred in the user SKILL function 'piTextMap' while mapping '%s'. This text will not be translated.
XSTRM-137	Warning	Error occurred in the user SKILL function 'poTextMap' while mapping '%s'. This text will not be translated.
XSTRM-138	UN-USED	
XSTRM-139	UN-USED	
XSTRM-140	Warning	The object map file '%s' contains an invalid record at line number '%d' and will be ignored. This is because the format of the mapping entry in the object map file '%s' is invalid. The valid mapping entry format is: <objectType objectSubType [oaLayer] strmLayer strmType [PhotomaskColor]>, where 'PhotomaskColor' is valid only when the '-enableColoring' option is set. Now, update the mapping entry and then translate again.
XSTRM-141	Warning	The object map file '%s' contains an invalid object type '%s' at line number '%d' and will be ignored. This is because the format of the mapping entry in the object map file '%s' is invalid. Update the mapping entry format to: <objectType objectSubType strmLayer strmType> and try again.
XSTRM-142	Warning	The object map file '%s' contains an invalid sub object '%s' at line number '%d' and will be ignored. This is because the format of the mapping entry in the object map file is invalid. Update the mapping entry format to: <objectType objectSubType strmLayer strmType> and try again.
XSTRM-143	Warning	Object map file %s contains multiple entries for objectType:sub-objectType '%s:%s' at line number '%d'. Only the first record will be used.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-144	Warning	Stream attribute number '%d' specified in property mapping file at line '%d' is not within range, '0-125'. This entry will be ignored.
XSTRM-145	Warning	The structure name '%s' has been changed to '%s' by the translator. This could be because multiple cells with the same cell name exist in different libraries, the same cell name has been mapped to multiple cell names, or the cell name length is greater than the '%d' that was specified using the '-respectGDSIINameLimit' or '-gdsCellNameLength' option.
XSTRM-146	Error	Cell map file contains a duplicate stream structure name '%s' at line number '%d'. This record in the cell map file will be ignored.
XSTRM-147	Warning	The property string associated with the object type '%s' exceeds the maximum permissible length of '%d' characters. Therefore, the property string will be truncated.
XSTRM-148	Error	Failed to open the via map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-149	Error	Failed to create the via map file '%s' in the run directory. Check that you have permission to create this file.
XSTRM-150	Warning	The via map file, '%s' already exists. It will be overwritten.
XSTRM-151	Warning	Failed to find via definition '%s' in the tech file. This entry in the viaMap file will be ignored.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-152	Warning	A polygon at (%d, %d) will be dropped because OpenAccess does not support a polygon which contains less than three coordinates. The number of distinct points required to create a polygon should be three. Rectify the source of the GDSII file and translate again.
XSTRM-153	Error	Failed to open object map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-154	Warning	The object map file does not contain any mapping for the OpenAccess object '%s' and the sub-object '%s'. Therefore, these objects in the design will not be translated. To enable the translation of these objects in the Stream file, provide a valid mapping in the object map file.
XSTRM-155	Warning	Ignoring the Stream layer-datatype pair '%d:%d' at row '%d' in the object map file '%s' because it is also defined in the layer map file. Additionally, shapes in GDS on '%d:%d' with properties indicating they were translated from objects in the OpenAccess database will be translated back to objects. The remaining shapes will be translated to shapes.
XSTRM-156	Warning	The entry at line number '%d' is duplicate in the '%s' layerMap file. Therefore, this entry will be ignored during translation.
XSTRM-157	Warning	The '%s' SKILL procedure has generated a structure name '%s' that has been changed to '%s' by the translator. This could be because multiple cells with the same cell name exist in different libraries, the same cell name has been mapped to multiple cell names, or the cell name length is greater than the '%d' that was specified using the '-respectGDSIINameLimit' or '-gdsCellNameLength' option.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-158	Error	Error in the definition of the SKILL procedure '%s'. The SKILL procedure maps the Pcell submaster name to a NULL string. Correct the definition of the SKILL procedure for this error and restart the translation.
XSTRM-159	Warning	Error in the definition of the SKILL procedure '%s'. Either the evaluation of %s has failed or it has returned an invalid SKILL list. Correct the SKILL procedure and restart the translation
XSTRM-160	Warning	The design has an area halo blockage with owner instance. Property map file is needed to preserve instance name as a property on the instance. Specify property map file at '-propMap' option.
XSTRM-161	Error	Property map file is needed to preserve area halo blockages. Property map file is not specified. Specify Property map file at '-propMap' option.
XSTRM-162	Info	You have not used the '-viaMap' option. If the OpenAccess design has native oaVia instances, use the '-viaMap' option for preserving oaVia instances in a XStream Out - XStream In round trip. Using the '-viaMap' option improves performance and VM usage of applications using the Streamed-In design.
XSTRM-163	Info	You have not used the '-viaMap' option. If a viaMap file, corresponding technology file or technology library exist, then use the '-viaMap' option for creating oaVia instances in the Streamed-In library. Using the '-viaMap' option improves performance and VM usage of applications using the Streamed-In design.
XSTRM-164	Warning	%s. The %s will be dropped.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-165	Error	Multiple stream structures are getting mapped to the same cell '%s' and view '%s' in the target library '%s'. Ensure that you have specified all required reference libraries and updated the cell name map file to avoid many-to-one mapping. Also ensure that input stream file is valid.
XSTRM-166	Error	Translation Failed - %s\nlf problem persists, contact the Cadence Customer Support.
XSTRM-167	Warning	Failed to open the sub-master. Internal automatic name mapping will be done for the sub-master %s as the SKILL function %s cannot be called for this sub-master.\nlf problem persists, contact the Cadence Customer Support.
XSTRM-168	Warning	The structure, %s, in the stream file is getting mapped to a cellview ('%s:%s') that already exists in the target library '%s'. Ignoring this cellview.
XSTRM-169	Warning	The structure, %s, in the stream file is getting mapped to a cellview ('%s:%s') that already exists in the target library '%s'. Overwriting this cellview in the target library.
XSTRM-170	Warning	The '-noObjectProp' option is used without the '-objectMap' option therefore it will be ignored. To use the '-noObjectProp' option you must specify an object map file using the '-objectMap' option.
XSTRM-171	Info	The '-noObjectProp' and the '-objectMap' options have been used together. Therefore, no property data is written to the stream file at property attribute 126 on the stream Boundary record.
XSTRM-172	Warning	Font map file %s exists in the technology library but is not readable. No font mapping will be done.
XSTRM-173	UN-USED	

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-174	Warning	The target library is already attached to technology library '%s'. Therefore, '-techRefs' option will be ignored
XSTRM-175	Warning	'%s' and '%s' are two mutually exclusive options that have been used together. The '%s' option will override the '%s' option.
XSTRM-176	UN-USED	Translation Failed.\nIf problem persists, contact the Cadence Customer Support.
XSTRM-177	Warning	Invalid stream version '%d' specified. Valid values are 3,5 and 6. The translator will use version '%d' as default.
XSTRM-178	Warning	Stream version '%d' and '%s' option have been used together. BOX record in stream file is supported only for stream version 5 and onwards. The translator will ignore '%s' option. If you want to translate rectangles in the design as BOX record in stream file, specify version as '%d' or do not specify '-strmVersion' option.
XSTRM-179	Error	XStream In has detected a conflict in the technology database graph. This could be because the technology library of the target library has conflict in its technology database graph or the specified reference technology libraries have a conflict with the technology library of the target library. Resolve the conflict manually before running the XStream In application.
XSTRM-180	Info	You have not used the '-objectMap' option. The design has instance(s) of at least one of following OpenAccess objects: oaBlockage, oaBoundary, oaRow, and oaMarker. Any information about these objects will not be translated into the generated stream file. Use the '-objectMap' option to translate/preserve these objects using XStream.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-181	Warning	The '%s' label map file contains an unknown layer '%s' at line '%d'. XStream Out will ignore this record in the label map file and continue to run. To fix this error, define the '%s' layer name in the technology file and restart XStream Out.
XSTRM-182	Warning	The '%s' label map file contains an unknown purpose '%s' at line '%d'. XStream Out will ignore this record in the label map file and continue to run. To fix this error, define the '%s' purpose name in the technology file and restart XStream Out.
XSTRM-183	Warning	The '%s' label map file contains an invalid record at line '%d'. XStream Out will ignore this record in the label map file and continue to run.
XSTRM-184	Warning	The '%s' label map file contains a duplicate record for the '%s:%s' OpenAccess layer-purpose pair at line '%d'. XStream Out will ignore this record in the label map file and continue to run.
XSTRM-185	Warning	The value '%s' is specified for the 'Convert Pin To' option simultaneously with the '%s' option. XStream Out will ignore the '%s' option and continue to run.
XSTRM-186	Warning	The value 'text' or 'geometryAndtext' is specified for the 'Convert Pin To' option simultaneously with the 'Label Map' option. XStream Out will map data according to the label map file specified in the 'Label Map' option.
XSTRM-187	UN-USED	
XSTRM-188	Warning	OpenAccess does not support coincident or collinear paths. A path with either coincident or collinear points is encountered at (%d, %d). The coincident or collinear points will be eliminated and an equivalent path will be created during translation.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-189	Warning	A path with an invalid begin or end extension value is encountered at (%d, %d). This path will be ignored during translation.
XSTRM-190	Warning	OpenAccess does not support coincident or collinear polygons. A polygon with either coincident or collinear points is encountered at (%d, %d). The coincident or collinear points will be eliminated and an equivalent polygon will be created during translation.
XSTRM-191	Warning	Found a polygon with self-intersecting points at the origin (%d, %d). This polygon will be translated without any changes. Correct the source of translation if the self-intersecting polygon is not required.
XSTRM-192	Warning	Found a path with self-intersecting points at the origin (%.4f, %.4f). This path will be translated without any changes. Correct the source of translation if the self-intersecting path is not required.
XSTRM-193	Warning	An open polygon is encountered at (%d, %d). This polygon will be ignored during translation.
XSTRM-194	Warning	OpenAccess does not support paths that have less than two coordinates. A path starting at (%d, %d) has less than two coordinates and therefore will be ignored during translation.
XSTRM-195	UN-USED	Value '%d' specified for option '%s' is %s than the valid range that is %s. %s value for this option has been set.
XSTRM-196	Warning	The design has %s with octagonal extensions in the cellview '%s/%s/%s'. These paths will be written as PATHTYPE1 (circular) in the Stream/GDSII file. You can retain the geometry of these %s by using the Convert Path to Polygon option.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-197	Warning	The Stream/GDSII files has some PATH records with PATHTYPE 1 (circular). Virtuoso workbench will view extensions of these paths as octagonal. There could be potential problems and you should correct these at the source of the Stream/GDSII file.
XSTRM-198	Warning	The Stream layer-datatype '%d:%d' pair at row number '%d' in the object map file '%s' is already mapped. This entry in object map file will be ignored.
XSTRM-199	Warning	The Stream layer-datatype '%d:%d' is mapped to sub-object '%s' of layerBlockage which is not supported in DFII.
XSTRM-200	Warning	XStream In is running in automatic layer mapping mode and hence '-skipUndefinedLPP' option will be ignored. Specify layer map file name for XStream In to respect '-skipUndefinedLPP' option.
XSTRM-201	Warning	A path with self-intersecting points is encountered at (%d, %d). This path will be translated as is. Correct the source of the translation if the self-intersecting path is not required.
XSTRM-202	Warning	The '-topCell' option is used without any view name. During XStream Out, the translator will translate all the views of maskLayout type for each of the specified topCell(s). Specify the view name that needs to be translated.
XSTRM-203	Warning	The '-view' name is not specified. During XStream Out, the translator will translate all the views of maskLayout type of the specified library. Specify the view name that needs to be translated.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-204	Warning	The following issue occurred while processing cellview '%s': %s The corresponding structure definition will not be added to the GDSII/stream file, and therefore, importing the GDSII file may result in unbound instances. %s
XSTRM-205	Info	The Stream layer-datatype pair %d:%d is mapped to layer-purpose pair %s:%s, which does not exist in the technology library. All shapes on XStream layer-datatype pair %d:%d will be skipped. To avoid skipping shapes on the undefined LPPs, do not specify the '-skipUndefinedLPP' option.
XSTRM-206	Warning	%s The path would be written in Stream file as is. To resolve this, fix the reported path in the input design and run the translator again.
XSTRM-207	Warning	%s The path would be created with the dimensions in Stream file. To resolve this, fix the source from which the Stream file is generated and then run the translator again with the new stream file.
XSTRM-208	Error	License initialization failed due to some internal error. Try again and if problem persists, contact Cadence Customer Support.
XSTRM-209	Error	The required license for XStream translator could not be found. Either the license server is down or the license is not available. Ensure that the valid license is available.
XSTRM-210	Error	Failed to initialize %s subsystem. For more information, contact the Cadence customer support.
XSTRM-211	Error	The argument with option '%s' could not be found. This is because the value for the argument is not specified. Specify the valid value and then try again.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-212	Error	The value for the scaleTextHeight argument is invalid. This is because the value of the argument is less than zero. Specify the positive integer value and then try again.
XSTRM-213	Error	The value for argument '%s' is invalid. This is because the argument '%s' contains a value other than upper, lower, or preserve. Specify the value for the argument '%s' as upper, lower, or preserve and then try again.
XSTRM-214	Error	The value for '-strmTextNS' is invalid. Valid values are cdba, verilog, or vhdl. Specify the correct value and translate again.
XSTRM-215	Error	The option in the command line is missing. This is because either both '-strmFile' and '-library' options are not specified or the '-cellListFile' option is not specified. Specify '-strmFile' and '-library' options or the '-cellListFile' option and then translate again.
XSTRM-216	Error	
XSTRM-217	Info	Reading the %s file, %s
XSTRM-218	Error	The parsing of the template file, '%s', cannot be completed. This is because the undefined UNIX environment variable, '%s', is used in the value at line number '%d'. Ensure that the UNIX environment variable is defined and then try again.
XSTRM-219	UN-USED	The value for the maxCellsInTargetLib argument is invalid. This is because the value of the argument is less than zero. Specify the positive integer value of the argument and try again.
XSTRM-220	Info	Translating STRUCTURE %s as cellView %s/%s/%s.
XSTRM-221	Info	Cell contains a zero width path. The zero width path will not be translated.
XSTRM-222	Info	%s

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-223	Info	%ld. Translating %s %s/%s/%s as STRUCTURE %s.
XSTRM-224	Warning	Unable to find library %s
XSTRM-225	UN-USED	Error in reading template file %s
XSTRM-226	Warning	File name for %s is not specified
XSTRM-227	Warning	Error in opening file %s
XSTRM-228	Error	Failed to open object map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-229	Error	Failed to create viaDef in the technology library as the technology library is attached to the target library. ViaDefs are being created because '-detectVias' option is specified. Detach the technology library from the target library and use '-techRefs' option to create incremental technology or don't specify the '-detectVias' option.
XSTRM-230	Error	Failed to create viaDef in the technology library as it is read-only. ViaDefs are being created because '-detectVias' option is specified. Change the permissions of the technology library or don't specify the '-detectVias' option.
XSTRM-231	Error	Pcell evaluation failed for '%s' because the Pcell SKILL code contains either a syntax error or an unsupported XStream function. Check the standard output or the Virtuoso log file for more information. Cadence recommends correcting the Pcell SKILL code to resolve the issue. However, to ignore these errors and continue the translation, you may use the '-ignorePcellEvalFail' option.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-232	Warning	Pcell evaluation failed for '%s' because the Pcell SKILL code contains either a syntax error or an unsupported XStream function. Check the standard output or the Virtuoso log file for more information. The translation will continue because the '-ignorePcellEvalFail' option is enabled, but the output should be checked to ensure it meets your requirements.
XSTRM-233	Error	The creation of a run directory '%s' has failed. Ensure that you have appropriate permissions to create the directory.
XSTRM-234	Info	Translation completed. '%d' error(s) and '%d' warning(s) found.
XSTRM-235	Warning	A LayerBlockage will be dropped because the layer used for the layerBlockage is not defined in the technology library. Ensure that the correct technology library is used and then translate again.
XSTRM-236	Info	A path with a rounded extension, an acute angle or a length segment less than half width is encountered at (%d, %d). This path will be written as a boundary record in the GDS file.
XSTRM-237	Warning	The layer-purpose pair '%u:%u' does only exist with \floating keyword\". The \"floating\" layer will also be used to map \"non floating\" shapes."
XSTRM-238	Warning	The layer-purpose pair '%s:%s' does only exist with \floating keyword\". The \"floating\" layer will also be used to map \"non floating\" shapes."
XSTRM-239	Warning	The value '%d' specified for the '-viaCutArefThreshold' option is less than 15. The recommended value of the option is greater than or equal to 15. Therefore, the value of the option will be defaulted to 15.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-240	Warning	Property %s will not be put on merged PATH in the GDS as another pathSeg has already has this property. You have specified '-mergePathSegToPath' option and translating properties.
XSTRM-241	Error	Unable to create the ASCII technology file '%s' in the run directory. Ensure that you have the write permission to create this file.
XSTRM-242	Error	Unable to create the LayerMap file '%s' in the run directory. Ensure that you have the write permission to create this file.
XSTRM-243	UN-USED	Unable to open the Cadence default technology library \"%s\". Ensure that the library has been installed properly."
XSTRM-244	Warning	The default mapping, without any optional fields, for layer-purpose pair '%u:%u' is not defined in the layer map file. All the objects on this layer-purpose pair that map to the default mapping will be ignored.
XSTRM-245	Warning	The default mapping, without any optional fields, for layer-purpose pair '%s:%s' is not defined in the layer map file. All the objects on this layer-purpose pair that map to the default mapping will be ignored.
XSTRM-246	Warning	Ignoring the record at line number '%d' in the layer map file '%s'. This is because the qualifier \"cutsize\" specified for the layer '%s' is not valid for the layer's material type '%s'. Valid material types for the qualifier are cut
XSTRM-247	Warning	Layer map file '%s' contains an invalid record for OpenAccess layer-purpose pair '%s:%s' and Stream layer-datatype pair '%d:%d' at line number '%d' and will be ignored. This is because the \"floating\" qualifier is specified for the %s purpose whose type is neither fill nor oaFillOPC."

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-248	Error	Failed to open label map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-249	Error	The cellListFile does not contain any valid records to process. Ensure that the cellListFile contains valid records in the following format: <LibraryName> <CellName> <ViewName> <GDSFileName>
XSTRM-250	Error	Failed to create output directory '%s'. Ensure that you have appropriate permissions to create this directory, or provide a different output directory.
XSTRM-251	Error	Failed to open cell list file '%s'. Check that the file exists in the run directory and you have appropriate permissions to read it.
XSTRM-252	Warning	The cellListFile contains invalid record at line number '%d'. This record in cellListFile will be ignored. Ensure that each line in cellListFile contains all the four entries in the following format: <LibraryName> <CellName> <ViewName> <GDSFileName>
XSTRM-253	Warning	Multiple records with different library names exist in the cellListFile and will be ignored. Presently, only one library name is supported in the cellListFile.
XSTRM-254	Warning	The '-view' option in the command line is missing and '*' is specified in the view entry of the cellListFile. The entry will be ignored because the value of the '-view' option that needs to replace '*' is missing. Specify the '-view' option or view name in cellListFile.
XSTRM-255	Warning	The cellListFile contains duplicate entries for cellView '%s/%s/%s' and will be ignored. Ensure that only one record for each cellView exists in the cellListFile.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-256	Warning	The cellView '%s/%s/%s' specified in the cellListFile does not exist on-disk, and entry will be ignored. Ensure that the cellView exists and the path of the library '%s' is specified in library definition file.
XSTRM-257	Warning	Library name specified in the cellListFile is different than specified with '-library' option and will be ignored. Presently, only one library name is supported in the cellListFile.
XSTRM-258	Warning	The '-library' option in the command line is missing and '*' is specified in the lib entry of the cellListFile. This entry will be ignored because the value of the '-library' option that needs to replace '*' is missing. Specify the '-library' option or lib name in cellListFile.
XSTRM-259	Warning	The property map file '%s' contains invalid record at line number '%d'. The valid object types are: Dot, Line, Path, Polygon, Rect, Text, ArrayInst, ScalarInst, Arc, Donut, Ellipse, PathSeg, Route, Marker, Row, AreaBoundary, PRBoundary, SnapBoundary, ClusterBoundary, AreaBlockage, AreaHalo, LayerBlockage, and LayerHalo. Specify the valid object types and try again.
XSTRM-260	Error	The option in the command line is missing. This is because the '-strmFile' option is not specified and either of the '-strmTechGen' or '-library' option is not specified. Specify the '-strmFile' option and either of the '-strmTechGen' or '-library' option and then translate again.
XSTRM-261	Warning	The object map file does not contain any mapping for the '%s' LayerBlockage on layer '%s'. Therefore, this LayerBlockage in the design will not be translated. To enable the translation of this LayerBlockage in the Stream file, provide a valid mapping in the object map file.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-262	Error	Failed to translate cellView, '%s/%s/%s'. This is because the cellView is opened by another process in the edit mode. Close the cellView and then translate again.
XSTRM-263	Error	Failed to translate cellView, '%s/%s/%s'. This is because the '%s/%s/%s/layout.oa' file does not have appropriate permissions. Set appropriate permissions on the file and then translate again.
XSTRM-264	Error	Failed to translate cellView, '%s/%s/%s' - '%s'.\nIf problem persists, contact the Cadence Customer Support.
XSTRM-265	Info	The Property map file '%s' contains multiple property names for Stream attribute number '%d', object type '%s'. This may cause problem while XStream In.
XSTRM-266	Warning	The property map file '%s' contains a record with more than one property name at line number '%d'. Only the first property name will be considered in XStream In and others will be ignored.
XSTRM-267	Warning	%s Ensure that the referenced cell exists and library is defined in the cds.lib file.
XSTRM-268	UN-USED	
XSTRM-269	Warning	Layer map file '%s' contains value, which is not within the range '0 - %d' for Stream layer at line number '%d'. However, this entry will not be ignored and all objects will be translated.
XSTRM-270	Warning	Layer map file '%s' contains value, which is not within the range '0 - %d' for Stream datatype at line number '%d'. However, this entry will not be ignored and all objects will be translated.
XSTRM-271	UN-USED	Value '%d' specified for option '%s' is %s than the valid range that is %s. %s value for this option has been set.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-272	UN-USED	Ignoring the numThreads option because the stream file specified is a compressed stream file, and compressed file is not supported with multiple threads.
XSTRM-273	Info	Translation failed. '%d' error(s) and '%d' warning(s) found.
XSTRM-274	Info	To improve performance of flattening Pcell instances the preserving Pcell pins is not done during XStream Out. This is because preserving Pcell pins is not required as the value 'geometry' or 'ignore' is specified for the Convert Pin To option.
XSTRM-275	Info	Number of vertexes in the '%s' with starting point (%d, %d) on layer '%u' is %d.%s This number exceeds the maximum limit, %d, defined by '%s'. The '%s' will be chopped into smaller %ss during XStream Out translation.
XSTRM-276	Error	Failed to chop '%s' with starting point (%d, %d) on layer '%u'. Delete or chop this '%s' and continue translation.
XSTRM-277	Error	Failed to open excludeMapToVia file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XSTRM-278	Warning	The '%s' excludeMapToVia file contains an invalid record at line '%d'. XStream In will ignore this record in the excludeMapToVia file and continue to run.
XSTRM-279	Error	Translation failed due to an internal error. This problem occurred while creating FIFO. Translate again and if problem persists, contact the Cadence Customer Support.
XSTRM-280	Info	The output file will be a compressed file because the file name has extension .gz
XSTRM-281	Info	The input file '%s' will be read as a compressed file because the file name has extension .gz

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-282	Info	The '-dbuPerUU' option will be ignored because the value (%d) does not match the 'dbuPerUU' value (%d) in technology database '%s'. Using the 'dbuPerUU' value from the technology database.
XSTRM-283	Warning	Data precision might be lost because the value (%d) set for the '-dbuPerUU' option does not match the 'dbuPerUU' value (%d) in technology database '%s'.
XSTRM-284	Warning	DbuPerUU (%d) specified in the technology database or using '-dbuPerUU' is not divisible by the value (%d) set in the GDSII file. This can cause loss in data precision during translation.
XSTRM-285	Warning	Data precision is lost while scaling a point. This is because of difference in the original point (%d, %d) and the scaled point (%d,%d).
XSTRM-286	UN-USED	The '-numThreads' option will be ignored. This is because the value of the argument is less than 1. The value of the argument should be greater than or equal to 1.
XSTRM-287	Info	The STRUCTURE '%s' in the input GDSII/Stream file will not be translated. This is because either the '-refLibList' option is used and the cellview is available in the reference library '%s' or the STRUCTURE is not mapped to the target library using the cell map file/piCellMap SKILL function. Therefore, all the instances of the cellview will be referenced from '%s/%s/%s'.
XSTRM-288	Info	Performing automatic layer mapping because either the layer mapping information is not provided or the XST_AUTO_LM value is used with the '-layerMap' option. Additionally, the '%s' file has been generated and the mapping information has been added to it.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-289	Info	Performing automatic layer mapping because either the layer mapping information is not provided or the XST_AUTO_LM value is used with the '-layerMap' option.
XSTRM-290	UN-USED	%s
XSTRM-291	Info	Output GDSII/Log file '%s' already exists. Therefore, XStream Out will create a backup file with the name, '%s.bak' as the '-backupGdsLogFiles' option is specified.
XSTRM-292	Warning	Both output GDSII/Log file '%s' and it's backup file, '%s.bak' already exist. Therefore, XStream Out will remove the existing '%s.bak' and create a new backup file, '%s.bak' as the '-backupGdsLogFiles' option is specified.
XSTRM-293	Error	Translation stopped. Both output GDSII/Log file '%s' and it's backup file, '%s.bak' already exist. Therefore, XStream Out will try to remove the existing '%s.bak' and create a new backup file, '%s.bak' as the '-backupGdsLogFiles' option is specified. However, the '%s.bak' file does not have the write permission. Ensure that the '%s.bak' file have the appropriate permissions and then try again.
XSTRM-294	Warning	Both '-append' and '-noOverwriteCell' options have been specified. The '-noOverwriteCell' option will be ignored.
XSTRM-295	Warning	Data precision is lost while scaling an integer value. This is because of the difference in the original value (%d) and the scaled value (%d).
XSTRM-296	Warning	Data precision is lost while scaling an integer value. This is because of the difference in the original value (%u) and the scaled value (%u).

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-297	Warning	The pin information of the GDS shape is ignored. This is because the GDS shape is created as an OpenAccess object, and a pin cannot be attached to an OpenAccess object.
XSTRM-298	UN-USED	The Stream layer-datatype '%d:%d' pair at row number '%d' in the object map file '%s' is also defined in the layer map file. Remove this duplicate entry either from the layer map file or object map file to avoid the XStream Out-XStream In round-trip issue.
XSTRM-299	Error	The specified library '%s' is not a valid primary virtual memory library created by XStream In. Use the 'xstInGetVMLibs' SKILL function to get a list of valid primary virtual memory libraries created by XStream In.
XSTRM-300	Error	The specified destination path '%s' does not exist. Ensure that the destination path exists.
XSTRM-301	Error	The directory '%s' required to save the virtual memory library already exists at the specified destination path '%s'. Specify a destination path that does not have a directory with the virtual memory library name.
XSTRM-302	Error	The technology library '%s' attached to the library '%s' does not exist. Ensure that the library '%s' is attached to a valid technology library.
XSTRM-303	Error	Failed to open the '%s' file. Check that the file exists and that you have the permission to modify it.
XSTRM-304	Warning	Disk space is low (< %dMB of free space) and may fill up during translation. To avoid any data loss, free up some disk space.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-305	Warning	Writing PATH record with the coincident points or less than two points at (%d %d) because the oaDesign contains a line with such points.
XSTRM-306	UN-USED	
XSTRM-307	Warning	The value for \text\" or \"geometryAndtext\" is specified for the Convert Pin To field simultaneously with the \"Label Map File\" found in the technology library. XStream Out will map data according to the \"Label Map File\" picked from the technology library."
XSTRM-308	Warning	The '%s' option will be ignored in command line. This option is only available in GUI. Use the \-help\" command to view the options available in command line."
XSTRM-309	Warning	The '%s' option specified in the template file will be ignored in command line. This option is only available in GUI. Use the \-help\" command to view the options available in command line."
XSTRM-310	Warning	The '-disableLocking' option is now obsolete. XStream In, by default, translates the library in non-locking mode. Use the '-enableLocking' option to translate the library in locking mode.
XSTRM-311	Warning	The '-disableLocking' option specified in the template file is now obsolete. XStream In, by default, translates the library in non-locking mode. Use the '-enableLocking' option to translate the library in locking mode.
XSTRM-312	Info	The '-enableLocking' option is selected so the output cellviews in the Cadence database will be separately locked. The translation may take more time as locking is enabled.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-313	Warning	The '-disableLocking' option specified in the xstInSetField/xstInGetField SKILL function or streamInKeys SKILL list is now obsolete. XStream In, by default, translates the library in non-locking mode. Use the '-enableLocking' option to translate the library in locking mode.
XSTRM-314	Warning	The value '%s' specified for the '%s' option is invalid. This is because the value of the argument is either negative, zero, or any other invalid value. Specify a valid non-zero positive float value for the '%s' option and translate again.
XSTRM-315	Warning	The cellview '%s/%s/%s' has a label with multiple lines. This label will be translated as is. If a multiple lines label is not required, correct the label in the source file.
XSTRM-316	Warning	The STRUCTURE '%s' has a TEXT record with multiple lines. This TEXT record will be translated as is. If a multiple lines TEXT record is not required, correct the TEXT record in the source file.
XSTRM-317	Info	The '%s/%s/%s' has a pathSeg with diagonal orientation, or end style as custom or chamfer at (%d, %d). This pathSeg will be translated as a boundary record in the GDS file during XStream Out.
XSTRM-318	Info	The Property map file '%s' contains duplicate entries for property name '%s', object type '%s' at line number '%d'. Since the same entry already exists for Stream attribute number '%d', this entry will be ignored.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-319	Warning	The %s map file '%s' contains color aware mapping(s) and will not be visible in GUI. This is because the 'Enable Coloring' option is not selected. Select the 'Enable Coloring' option to display these entries in the table else these entries will be ignored during translation.
XSTRM-320	Warning	The %s map file '%s' contains a color aware entry at line number '%d' and this entry will be ignored. To consider this entry specify the '-enableColoring' option to do color aware mapping.
XSTRM-321	Error	Translation cannot be continued as the '-enableColoring' option is used, but the layer map file is not found. This is because either the -layerMap option is not specified or technology does not have the layer map file. Use a valid layer map file and then translate again.
XSTRM-322	Error	Translation cannot be continued as the '-enableColoring' option is used, but technology is not found. Provide a technology either by using the '-attachTechFileOfLib' option, or ensure that library either has valid technology or is attached to the valid technology in case the target library already exist and then translate again.
XSTRM-323	Warning	The layer map file '%s' contains inconsistent color aware entry at line number '%d' and this entry will be ignored. This is because layer '%s' does not support either coloring or Photomask color '%s'.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-324	Warning	Colored shapes are found on layer-purpose pair '%u:%u' with photomask color '%s' and color state '%s', and a non-color-aware mapping is done for these shapes. This is because, color mapping is not specified for photomask color '%s', color state '%s', and layer-purpose pair '%u:%u'.
XSTRM-325	Warning	Colored shapes are found on layer-purpose pair '%s:%s' with photomask color '%s' and color state '%s', and a non-color-aware mapping is done for these shapes. This is because, color mapping is not specified for photomask color '%s', color state '%s', and layer-purpose pair '%s:%s'.
XSTRM-326	Warning	Connected pathSegs on colorable layers will not be merged and each pathSeg will be written as separate PATH record in GDS. This is because both '-enableColoring' and '-mergePathSegsToPath' options have been specified. Therefore, the '-mergePathSegsToPath' option will not be considered for the pathSegs on colorable layers.
XSTRM-327	Warning	Both '-enableColoring' and '-viaCutArefThreshold' options have been specified. The '-viaCutArefThreshold' option is not supported with the '-enableColoring' option. Therefore, the '-viaCutArefThreshold' option will be ignored.
XSTRM-328	Info	The '-enableColoring' and '-topCell' options are specified but '-arrayInstToScalar' option is not specified. Additionally, the layer map file has unlocked mapping entries. The GDS file will be generated with the effective color for the unlocked color shapes. However, array instances might be written as multiple SREF records in the GDS file because the effective color requires color variant generation.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-329	Warning	The GDS file will be generated without the effective color for the unlocked color shapes. This is because the '-topCell' option is not specified, although the '-enableColoring' option is specified and the layer map file has unlocked mapping entries. The effective color requires color variant generation that depends on the hierarchy context. To generate the GDS file with the effective color, define the hierarchy context using the '-topCell' option.
XSTRM-330	Info	The layer map file '%s' contains a duplicate entry for OpenAccess layer-purpose pair '%s:%s' and Stream layer-datatype pair '%d:%d' at line number '%d'.
XSTRM-331	Info	The '-enableColoring' and '-topCell' options have been specified, but the layer map file does not have any unlocked mapping entry. Therefore, all the unlocked color shapes in the OpenAccess database will be translated according to noncolor-aware mapping in the GDS file.
XSTRM-332	Warning	Encountered a cellview '%s/%s/%s' that has a view type different from maskLayout. It will not be saved on the disk.
XSTRM-333	Warning	The '-enableColoring' option will be ignored during %s. This is because there is no color entry in the layer map and the object map file.
XSTRM-334	Error	Translation has been aborted because the '-compress' option has not been specified with the '-compressLevel' option. When the '-compressLevel' option is used, the '-compress' option must be specified.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-335	Error	Translation has been aborted because invalid value, '%s' has been specified for the '-compressLevel' option. Valid values for the '-compressLevel' option are between 1 and 9.
XSTRM-336	Warning	The '%s' option will be ignored during XStream %s because the '%s' option has not been specified.
XSTRM-337	UN-USED	
XSTRM-338	Error	The translation terminated because the cellview '%s/%s/%s' contains colored data that is incompatible with this version of the translator. Use an advanced node version of the translator (for example ICADV12.1) to process this data.
XSTRM-339	Error	The translation terminated because the GDSII/Stream file '%s' contains colored data that is incompatible with this version of the translator. Use an advanced node version of the translator (for example ICADV12.1) to process this data.
XSTRM-340	Error	Failed to attach technology '%s' to the library '%s' because read permission is not set for the database of at least one of the libraries or technologies defined in cds.lib. Check that you have appropriate permissions and try again.
XSTRM-341	Error	Translation failed while processing '%s' because of the following reason: %s %s
XSTRM-342	Warning	The input Stream/GDSII file is empty; therefore, XStream In has no structures to translate.
XSTRM-343	Warning	The input Stream/GDSII file is empty and has no structures that can be translated. The preexisting library '%s\' will remain as it is.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-344	UN-USED	The colors set for the locked color shapes in the master design are ignored and the colors set for the locked hierarchical color shapes in the top cell are used instead.
XSTRM-345	UN-USED	The design contains unlocked color shapes and also has an overlay cell. The unlocked color shapes have been mapped based on the unlocked color mappings defined in the layer map file.
XSTRM-346	Warning	The '-%s' option is no longer supported. Use the '-writeMode' option with value '%s' instead.
XSTRM-347	Warning	The '%s' option specified in the template file is no longer supported. Use the 'writeMode' option with value '%s' instead.
XSTRM-348	Warning	The '%s' option specified in the 'xstInSetField' or 'xstInGetField' SKILL functions or the streamInKeys SKILL list is no longer supported. Use the 'writeMode' option with value '%s' instead.
XSTRM-349	Error	An invalid value '%s' was specified for the '-writeMode' option. Set '-writeMode' to either 'noOverwrite', 'overwrite', 'append', or 'rename' and run the command again.
XSTRM-350	Warning	The '-overwriteRootCells' option has been ignored because it is valid only when '-writeMode' is set to 'rename'.
XSTRM-351	UN-USED	Value '%d' specified for option '%s' is %s than the valid range that is %s. %s value for this option has been set.
XSTRM-352	Warning	The '-exportHierarchicalColorLocks' option is now obsolete. XStream Out, by default, exports the hierarchical color locks.
XSTRM-353	Warning	The 'exportHierarchicalColorLocks' option specified in the template file is now obsolete. XStream Out, by default, exports the hierarchical color locks.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-354	Warning	The technology library '%s' contains colored data that is incompatible with this version of the translator and should be used ideally with an advanced node version of the translator (for example ICADV12.1). Make sure that you are using the correct version.
XSTRM-355	Warning	The 'translateUnmappedLPP' option has been ignored during XStream %s because it does not impact automatic layer mapping. Use this option with the 'layerMap' file or the layer mapping SKILL function '%s'.
XSTRM-356	Info	Applied automatic layer mapping to the layer-purpose pair '%u:%u'.
XSTRM-357	Info	Applied automatic layer mapping to the layer-purpose pair '%s:%s'.
XSTRM-358	Warning	The new purpose '%s' will be created in memory only and will not be saved to the technology because the technology has been opened in read-only mode.
XSTRM-359	Warning	The new layer '%s' will be created in memory only and will not be saved to the technology because the technology has been opened in read-only mode.
XSTRM-360	Warning	The new layer-purpose pair '%s:%s' will be created in memory only and will not be saved to the technology because the technology has been opened in read-only mode.
XSTRM-361	Info	Ignoring the '-translateUnmappedLPP' option because the '-skipUndefinedLPP' option has been defined.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-362	Error	Translation cannot be continued. This is because the '-translateUnmappedLPP' option is used, which requires creation of layers, purposes, and LPPs, and technology library '%s' is opened in read-only mode due to the library '%s' being attached to it. To continue, either do not specify the '-translateUnmappedLPP' option or specify a target library that contains 'tech.db' or ITDB.
XSTRM-363	Warning	The technology library has been modified during XStream In translation. Cannot save the changes in the technology file because it is open in read-only mode.
XSTRM-364	Warning	Ignoring the layer-datatype pair '%s:%s' in the layer map file '%s' at line number '%d' because the technology library is read-only and the layer-datatype pair is not defined in the technology file.
XSTRM-365	Warning	The layer-datatype pair '%s:%s' in 'piLayerMap' will be ignored. This is because technology is read-only and the layer-datatype pair is not defined in the technology file. This layer-datatype pair is now mapped automatically for XStream In and XStream Out.
XSTRM-366	Warning	The '-strmTechGen' option requires only a GDSII file as its input. Therefore, ignoring all options other than '-strmTechGen' and '-strmFile.'
XSTRM-367	Warning	Ignoring the '-strmTechGen' option because the '-library' option has been defined.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-368	Warning	The '-enableIncrementalImport' option is ignored during XStream In because the library specified is not the same as the primary library used for previous incremental import project. To continue in the same incremental import project, specify the '%s' library as the primary library. You can also start a new incremental project with a different primary library by running the translation in a new directory or by deleting the '.cadence/geoTrans' directory present in the current directory.
XSTRM-369	Warning	The instance '%s' instantiated in '%s/%s/%s' with origin at (%d, %d) is unplaced but will be written to Stream/GDSII file '%s'. If you do not want to export unplaced instances, use the option '-noOutputUnplacedInst'.
XSTRM-370	UN-USED	
XSTRM-371	Error	The translation failed because automatic layer mapping cannot be done. This is because none of the stream %s numbers from the valid range ('0-65535') is available. Define the %s you want to map in the layerMap file and start the translation again.
XSTRM-372	Info	XStream Out is propagating the locks because it is in the 'lock same color group' mode.
XSTRM-373	Info	XStream Out will not be in the 'lock same color group' mode. This is because %s.
XSTRM-374	Error	The design '%s' contains array instance '%s' of design '%s' with %d columns and %d rows. This does not comply with GDS format. Split this array instance such that number of columns and rows of the resultant array instances does not exceed 65536, and run translation again.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-375	Warning	The design '%s' contains array instance '%s' of design '%s' with %d columns and %d rows. This is incompatible with some GDS readers, which only allow a maximum of 32768 AREF columns and rows. To avoid issues during translation, split this array instance such that the number of columns and rows of the resultant instances does not exceed 32768.
XSTRM-375	Warning	%s
XSTRM-376	Info	The '-translateUnmappedLPP' option overrides any entry with 'ignoreLPP' qualifier specified in layermap file '%s'.
XSTRM-377	Error	The translation failed because the option '-guiHistory' has been specified but no successful translation was done from the XStream %s GUI. Remove the '-guiHistory' option and rerun the translation.
XSTRM-378	UN-USED	
XSTRM-380	UN-USED	
XSTRM-381	Info	The '-enableColoring' and '-mapAllColorToLocked' options have been enabled. Unlocked shapes will be mapped according to the locked color mappings. Unlocked colored mappings will be ignored in the layer map file.
XSTRM-382	UN-USED	
		Translation has been stopped because the layer map file '%s' contains insufficient number of locked color-aware mappings for layer '%s'. Provide locked color-aware mappings for all the 'color masks' or 'photo masks' for this layer and translate again.
XSTRM-383	Warning	Found a rectangle with zero-area at the origin (%.4f, %.4f). This rectangle will be translated without any changes. Correct the source of translation if the zero-area rectangle is not required.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-384	Info	The number of cells in library \"%s\" exceeds the limit of %d. A new library \"%s\" will be created."
XSTRM-385	Warning	The '-numThreads' option is no longer supported in XStream In. The number of cores in the CPU will be detected automatically and an appropriate number of threads will be used for translation.
XSTRM-386	Warning	The 'numThreads' option specified in the template file is no longer supported in XStream In. The number of cores in the CPU will be detected automatically and an appropriate number of threads will be used for translation.
XSTRM-387	Warning	The 'numThreads' option specified in the xstInSetField or xstInGetField SKILL function or the streamInKeys SKILL list is no longer supported in XStream In. The number of cores in the CPU will be detected automatically and an appropriate number of threads will be used for translation.
XSTRM-388	Warning	The '-detectVias' option is no longer supported in XStream In. The vias will be detected automatically. Use the '-noDetectVias' option if via detection is not required.
XSTRM-389	Warning	The '-detectVias' option specified in the template file is no longer supported in XStream In. The vias will be detected automatically. Use the '-noDetectVias' option if via detection is not required.
XSTRM-390	Warning	The '-detectVias' option specified in the xstInSetField or xstInGetField SKILL function or the streamInKeys SKILL list is no longer supported in XStream In. The vias will be detected automatically. Use the '-noDetectVias' option if via detection is not required.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-391	Warning	Running the replay function in support mode. This replay function was created using an older release, such as IC6.1.6 or ICADV12.1 and is now obsolete. Regenerate the replay function using the redesigned XStream GUI.
XSTRM-392	Warning	Detected old replay functions that are not supported in the current release. The XStream GUI has been redesigned and any settings through replay functions from an older release, such as IC6.1.6 or ICADV12.1 will be ignored. Regenerate the replay function using the redesigned XStream GUI.
XSTRM-393	Warning	Ignoring color aware entries in the via map file '%s'. To consider these entries during translation, specify the '-enableColoring' option.
XSTRM-394	Warning	Ignoring color aware entries in the via map file '%s'. To consider these entries for via preservation with coloring information, use an advanced node version of the translator (for example ICADV12.2), and specify the '-enableColoring' option.
XSTRM-395	Warning	Ignoring via submaster record '%s' in the via map file '%s' because it contains an invalid color attribute parameter.
XSTRM-396	Warning	Ignoring rectangles with inverted bounding box because translating them may result in an invalid GDS file.
XSTRM-397	Warning	Ignoring option '%s' %s because it is incompatible with this version of the translator. Use this option with an advanced node version of the translator (for example ICADVM20.1).
XSTRM-398	UN-USED	Value '%d' specified for option '%s' is %s than the valid range that is %s. %s value for this option has been set.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-399	Warning	The STRUCT '%s' generated for '%s/%s/%s' in GDS is empty because either the design is empty or no shape or object was translated.
XSTRM-400	Warning	Both option '-flattenVias' and CDS environment variable 'xstIgnoreCoveredViaShapes' have been specified. The 'xstIgnoreCoveredViaShapes' variable is not supported with the '-flattenVias' option. Therefore, the 'xstIgnoreCoveredViaShapes' variable will be ignored.
XSTRM-401	Error	Translation failed because bounding box coordinates of the array instance %s in the cellview '%s/%s/%s' exceed the limits allowed in the GDSII format (-2147483647, 2147483648). Use a smaller DBU value or use the OASIS format to export the design.
XSTRM-402	Info	Ignoring %s file '%s' in the technology library because it is not readable.
XSTRM-403	Warning	The '-flattenViaShapesAsViaPurp' option is no longer supported. Use '-outputViaShapesToViaPurp' with '-flattenVias' to flatten shapes of vias to via purpose.
XSTRM-404	Warning	The 'flattenViaShapesAsViaPurp' option specified in template file is no longer supported. Use 'outputViaShapesToViaPurp' with 'flattenVias' to flatten shapes of vias to via purpose.
XSTRM-405	Warning	The 'flattenViaShapesAsViaPurp' option specified in the xstOutSetField or xstOutGetField SKILL function or the streamOutKeys SKILL list is no longer supported. Use 'outputViaShapesToViaPurp' with 'flattenVias' to flatten shapes of vias to via purpose.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-406	Warning	Both locked and unlocked shapes for '%s:%s:%s'%s are mapped to same Stream layer-datatype '%d:%d' in layer map file '%s' at line number '%d'.
XSTRM-407	Warning	The top cell '%s' contains a modified Concurrent Layout cellview. Changes in the top and hierarchical Concurrent Layout design partition views are not automatically exported from virtual memory to GDS. To export these changes, right-click in the Concurrent Layout assistant and choose 'XStream Out from VM for DRC'.
XSTRM-408	Warning	Option '%s' specified in the template file is invalid in XStream %s GUI. Run \"%s -help\" at command prompt to view list of available options and then specify the valid option in the template file."
XSTRM-409	Warning	Both '-noDetectVias' and '%s' options have been specified. The '%s' option works when via detection is enabled but the '-noDetectVias' option disables via detection. Therefore, the '%s' option will be ignored.
XSTRM-410	Warning	Layer map file '%s' contains a duplicate entry for Stream layer-datatype '%d:%d' at line '%d' which maps the shapes to '%s:%s:%s:%s' but is already mapped to '%s:%s:%s:%s'. This entry will be ignored during translation.
XSTRM-411	Error	Failed to open the flatten Pcell exclusion file '%s'. Ensure that the file exists in the run directory and that you have permission to read it.
XSTRM-412	Info	Cannot flatten instances of Pcell master '%s/%s/%s' in cellView '%s/%s/%s' because the Pcell master is specified in the flatten Pcell exclusion file '%s'.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-413	Error	Translation failed because the specified technology is non-IC type. This is not supported by XStream Out.
XSTRM-414	Error	An invalid value '%s' was specified for the '-curvedPathEndCapStyle' option. Set '-curvedPathEndCapStyle' to 'round', 'square', or 'flush' and run the command again.
XSTRM-415	Error	An invalid value '%s' was specified for the '-curvedShapeEdgePlacement' option. Set '-curvedShapeEdgePlacement' to either 'inside' or 'outside' and run the command again.
XSTRM-416	Warning	Value '%d' specified for the '-curvedShapeNumSides' option is not a multiple of 4, which may cause discretization to generate asymmetric geometry. For symmetric geometries, use a value that is a multiple of 4.
XSTRM-417	Warning	Value '%d' specified for the '-curvedShapeNumSides' option is higher than 512. This may require more time and memory during discretization and might result in a larger GDS file. Use a number larger than 512 for this option only if it is really needed.
XSTRM-418	Info	DbuPerUU (%d) specified in the technology database or using '-dbuPerUU' does not match with the value (%d) set in the GDSII file.
XSTRM-419	Warning	Exported a shape with starting point (%d, %d) and '5' vertices. The shape will be represented using '6' points in the GDS. Although 'maxVertices' is set to '%d', the shape cannot be chopped further because it has any-angle edges. Chop the shape manually and translate again.
XSTRM-420	UN-USED	Translation Failed - %s Enable the '-detectOverflow' option and try again.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-421	Warning	Ignoring option '-pinLabelPurp' because purpose '%s' specified for this option is not found in the technology library. Ensure that the specified purpose exists in the technology library and translate again.
XSTRM-422	Warning	Cannot find layer mapping for the layer-purpose pair '%s:%s' in the layer map file. Exported pin labels will have same layer-purpose pair as the pin shape. To avoid this issue, ensure that the layer-purpose pair exists in the layer map file corresponding to the purpose 'pinLabelPurp'.
XSTRM-423	Warning	Ignoring the Stream layer-datatype pair '%d:%d' in the layer map file because it is also defined in the object map file '%s' at row '%d' and a user-specified setting gives the object map file higher precedence over the layer map file.
XSTRM-424	Error	Translation Failed. %s Ensure that the specified cellview exists on disk and that you have permission to open it and then rerun the translation.
XSTRM-425	Warning	%s The corresponding structure definition will not be added to the GDSII file. Importing the GDSII file may result in unbound instances. Ensure that the master cellview exists and the specified library is defined in the library definition file.
XSTRM-426	Info	%ld. Translating cut shape of standard via '%s' as STRUCTURE %s.
XSTRM-427	Info	Based on current hardware configuration, running XStream Out translation in multi-threaded mode with %d threads.
XSTRM-428	Error	%s failed due to unknown reason. Run the translator again. If the problem persists, contact Cadence Customer Support for assistance.

Design Data Translators Reference

List of XStream Messages

Message ID	Severity	Message
XSTRM-429	Warning	Cannot translate because an existing non-VM library has been specified. Specify either a non-existing library or a pre-existing VM library, created using XStream In VM mode and translate again.
XSTRM-430	Info	%s

Legend

Format	Data Type
%d	Signifies that this value will be replaced by an integer value.
%f	Signifies that this value will be replaced by a floating-point number.
%s	Signifies that this value will be replaced by a character string.
%u	Signifies that this value will be replaced by an unsigned integer.

List of XOasis Messages

The following table provides a list of error, warning, and information messages generated by the XOasis translator. To view the table legend, see [Legend](#).

Message ID	Severity	Message
XOASIS-1	UN-USED	
XOASIS-2	Error	Library name '%s' is invalid in UNIX NameSpace.
XOASIS-3	Error	Tech library name '%s' is invalid in UNIX NameSpace.
XOASIS-4	Error	Failed to locate run directory '%s'. Check that the directory exists and that you have permissions to write in this directory.
XOASIS-5	UN-USED	Run directory '%s' does not exist or is not a directory.
XOASIS-6	Error	Failed to open the template file '%s'. Check that the template file exists and is a regular UNIX file.
XOASIS-7	Error	Failed to open cell name map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-8	UN-USED	Failed to open text font map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-9	Error	Failed to open property map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-10	Error	Failed to open layer map file '%s'. Check that the file exists in the run directory and that you have permission to read it.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-11	Error	Failed to open refLibList file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-12	UN-USED	Failed to open pinTextMap file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-13	Error	Failed to open input Oasis file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-14	Error	Failed to open ASCII techfile '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-15	Error	Failed to create Oasis file '%s' in the run directory. Check that you have permission to create this file.
XOASIS-16	Error	Failed to create summary file '%s' in the run directory. Check that you have permission to create this file.
XOASIS-17	Error	Failed to create log file '%s' in the run directory. Check that you have permission to create this file.
XOASIS-18	Warning	Performing automatic layer mapping because either the layer mapping information is missing or the -layerMap option is using the XOASIS_AUTO_LM value. However, failed to create the automatic layer map file '%s' in the run directory. Ensure that you have the write permission to create this file.
XOASIS-19	UN-USED	Failed to create the automatic cell name map file '%s' in the run directory.
XOASIS-20	Warning	Output Oasis file '%s' already exists. It will be overwritten.
XOASIS-21	Error	Failed to open the library '%s'. Check that library is present in the path defined in the 'cds.lib' file.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-22	Error	Failed to open the technology file '%s' from library '%s'. Check that the library exists in the location defined in the 'cds.lib' file and that you have read permission to the technology file.
XOASIS-23	UN-USED	Purpose '%s' is not defined in the technology file. All the blockages in the design will be considered on purpose 'boundary'.
XOASIS-24	UN-USED	Purpose '%s' is not defined in techfile. All the blockages in the design will be considered on purpose 'drawing'.
XOASIS-25	Warning	The layer map file '%s' contains an invalid record at line number '%d' and will be ignored. This is because the format of the mapping entry in the layer map file is invalid. The valid mapping entry format is: <layerName purposeName oasisLayer oasisType [Material [MaskNumber]] [Qualifier] [PhotomaskColor ColorState]>, where 'PhotomaskColor' and 'ColorState' are valid only when the '-enableColoring' option is set and the permissible range for oasisLayer and oasisType is 0 - %u. Now, update the mapping entry and then translate again.
XOASIS-26	UN-USED	The font map file '%s' contains an invalid record at line number '%d' and will be ignored. This is because the format of the mapping entry in the font map file '%s' is invalid. Update the mapping entry format to: <fontNumber fontName> and try again.
XOASIS-27	Warning	The property map file '%s' contains an invalid entry at line number '%d' and will be ignored. This is because the format of the mapping entry in the property map file '%s' is invalid. Update the mapping entry format to: <[property ID] objectType propName> and then translate again.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-28	Warning	The cell map file '%s' contains an invalid record at line number '%d' and will be ignored. This is because the format of a mapping entry in the cell map file '%s' is invalid. Update the mapping entry format to: <libName cellName viewName structName> and try again.
XOASIS-29	Warning	refLib file contains invalid record at line number '%d'. This record in the refLib file will be ignored.
XOASIS-30	UN-USED	
XOASIS-31	Warning	Layer map file '%s' contains a duplicate entry for Oasis layer:datatype '%d:%d' at line '%d'. Therefore, this entry will be ignored during translation.
XOASIS-32	Warning	The layer map file '%s' contains unknown layer '%s' at line number '%d' and will be ignored. This is because the layer name specified in the layer map file is not defined in the technology file. Ensure that the layer name specified in the layer map file is both valid and defined in the technology file.
XOASIS-33	Warning	The layer map file '%s' contains unknown purpose '%s' at line number '%d' and will be ignored. This is because the purpose name specified in the layer map file is not defined in the technology file. Ensure that the purpose name specified in the layer map file is both valid and defined in the technology file.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-34	Warning	The objects in the layer-purpose pair '%u:%u' are ignored. This is because either the layer-purpose pair used in the design is not specified in the layer map file or the layer/purpose is not defined in technology file. Ensure that the layer-purpose pair used in the design is specified in the layer map file and the layer/purpose is defined in technology file.
XOASIS-35	Warning	The objects in the layer-purpose pair '%s:%s' are ignored. This is because the layer-purpose pair used in the design is not specified in the layer map file. Specify the layer-purpose pair in the layer map file and try again.
XOASIS-36	Warning	The objects in the oasis layer-datatype '%d:%d' are ignored. This is because the oasis layer-datatype pair is either not specified in the layer map file or the entry is invalid and was therefore ignored. Specify the oasis layer-datatype pair in the layer map file or correct the wrong entry and try again.
XOASIS-37	UN-USED	Layer-Datatype range is exhausted. All the shapes on layer-purpose '%d:%d' will be ignored.
XOASIS-38	Warning	The library %s in the refLibList file %s could not be found. This is because the library '%s' specified in refLibList file does not exist. Ensure that the library '%s' is defined in the library definition file and also exists in the location specified in the library definition file.
XOASIS-39	UN-USED	The font map file contains invalid font name '%s' at line number '%d'. The valid font names are: EuroStyle, Gothic, Math, Roman, Script, Stick, Fixed, Swedish, and MilSpec. Specify a valid font name and try again.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-40	UN-USED	The font map file contains invalid Oasis font number '%d' at line number '%d'. The valid font numbers are: 0, 1, 2, and 3. Specify the valid font numbers and try again.
XOASIS-41	Warning	The property map file '%s' contains invalid record at line number '%d'. The valid object types are: Dot, Pin, Line, Path, Polygon, Rect, Text, ArrayInst, ScalarInst, Arc, Donut, Ellipse, PathSeg, Route, Marker, Row, AreaBoundary, PRBoundary, SnapBoundary, ClusterBoundary, AreaBlockage, AreaHalo, LayerBlockage, and LayerHalo. Specify the valid object types and try again.
XOASIS-42	UN-USED	
XOASIS-43	UN-USED	
XOASIS-44	UN-USED	pintext map file contains an unknown layer '%s' at line '%d'. This entry in the pin text map file will be ignored.
XOASIS-45	Warning	Mapping for layer '%d' will not be written in xOasisOut_layerMap.txt file because layer '%d', which is used in the input design is not defined in the technology file. Ensure that the layer exists in the technology file or use the correct technology file.
XOASIS-46	Warning	Mapping for purpose '%d' will not be written in xOasisOut_layerMap.txt file because purpose '%d', which is used in the input design is not defined in the technology file. Ensure that the purpose exists in the technology file or use the correct technology file.
XOASIS-47	Warning	Library '%s' is used in the design but does not exist. It will not be listed in the summary file.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-48	Error	Failed to flatten the pcell instance '%s' in the cellview '%s/%s/%s'. OasisOut flattens pcell instances when the Flatten Pcells option is used. Ensure that the pcell instance is correct by opening the cellview '%s/%s/%s' and flattening the instance in Virtuoso.
XOASIS-49	UN-USED	The pin information could not be translated. This is because the value 'text' is specified for the Convert Pin To option and Oasis Out is unable to get the ID associated with the pin. Rectify the input database to prevent data corruption and translate again.
XOASIS-50	UN-USED	The pin information could not be translated. This is because the Pin Attribute Num option is specified during Oasis Out to translate the pin information, and Oasis Out was unable to get the ID associated with the pin. Rectify the input database to prevent data corruption and translate again.
XOASIS-51	UN-USED	The pin information could not be translated. This is because the value 'text' is specified for the Convert Pin To option and Oasis Out was unable to get the pin name associated with the pin. Rectify the input database to prevent data corruption and translate again.
XOASIS-52	UN-USED	The pin information could not be translated. This is because the Pin Attribute Num option is specified during Oasis Out to translate the pin information, and Oasis Out was unable to get the pin name associated with the pin. Rectify the input database to prevent data corruption and translate again.
XOASIS-53	UN-USED	The pin information could not be translated. This is because the Pin Attribute Num option is specified during Oasis Out to translate the pin information, and Oasis Out was unable to get the term name associated with the pin. Rectify the input database to prevent data corruption and translate again.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-54	UN-USED	Symbolic pins will not be translated as text because pinTextMap file is not provided.
XOASIS-55	UN-USED	Vector instance '%s' encountered. It will be translated as SREF in the Oasis file.
XOASIS-56	Warning	Library '%s' specified in cell map file is not present in refLibList, its mapping will be replaced with the target library.
XOASIS-57	Warning	Failed to create the new layer '%s'. All the shapes on this layer will be dropped.
XOASIS-58	Warning	Failed to create the new layer purpose pair '%s:%s'. It will be skipped and all the shapes on this LPP will be dropped.
XOASIS-59	Warning	Failed to create the new purpose '%s'. All the shapes on this purpose will be dropped.
XOASIS-60	Info	Creating a new purpose %s in the technology file.
XOASIS-61	Info	Creating a new layer %s in the technology file.
XOASIS-62	Info	Creating a new layer:purpose pair %s:%s in the technology file.
XOASIS-63	Warning	Failed to set the 'visibility attribute' of a newly created layer-purpose pair '%s:%s'.
XOASIS-64	Warning	Failed to set the 'layer selectable' property of a newly created layer-purpose pair '%s:%s'.
XOASIS-65	Warning	Failed to set the 'priority' of a newly created layer-purpose pair '%s:%s'.
XOASIS-66	Warning	Failed to update the 'packet name' of a newly created layer-purpose pair '%s:%s'.
XOASIS-67	Warning	Failed to change the validity of a newly created layer-purpose pair '%s:%s'.
XOASIS-68	Warning	Ignoring 'Load ASCII Techfile' option because target library %s is already attached' to the technology library %s.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-69	Info	All the libraries specified in 'cds.lib' will be considered as reference libraries. This is because refLibList options is specified as \XOASIS_CDS_LIB\" or the refLibList file contains \"XOASIS_CDS_LIB\"."
XOASIS-70	Info	The 'Reference Library list' and the 'Keep Oasis Cells' options have been used together. Any cell defined in the Oasis file will not be searched in reference libraries.
XOASIS-71	Warning	The '-keepOasisCells' option has been overridden for the cells in the target library because '-writeMode' is set to 'noOverwrite'
XOASIS-72	Warning	The 'Load ASCII tech file' and 'Attach Technology Library' options have been used together. The 'Load ASCII tech file' option will be ignored.
XOASIS-73	Warning	The library name column in the cell name map file will be ignored because reference library list is not specified.
XOASIS-74	Error	Translation cannot be continued because layer mapping cannot be done. This is because the layer map is not specified by the '-layerMap' option and the technology library '%s' also does not contain '%s.layermap'. Additionally, automatic layer mapping cannot be done because the technology file is opened in read-only mode due to the library '%s' being attached to it. Use a layer map file to continue.
XOASIS-75	Warning	Target library '%s' is attached to the technology library '%s'. Therefore, the technology file is opened in read-only mode. All the undefined layer-purpose pairs will be dropped.
XOASIS-76	Error	Failed to open technology file '%s:%s' in the append mode. Therefore, automatic layer mapping cannot be done. Use the layer map file to continue.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-77	UN-USED	Failed to open the technology file '%s:%s' in the append mode. All the undefined layer-purpose pairs will be dropped.
XOASIS-78	Error	Failed to load ASCII technology file %s, specified in 'Load ASCII techfile' option.
XOASIS-79	Error	Failed to attach technology file from the library %s because a technology file already exists in the target library.
XOASIS-80	Error	The technology library '%s' specified in the 'Attach Technology Library' option does not exist.
XOASIS-81	Error	The technology library '%s' attached to the target library does not exist.
XOASIS-82	UN-USED	Failed to create the library %s.
XOASIS-83	Error	Technology file '%s' cannot be opened from the library '%s'. Ensure that the library '%s' exists and have appropriate permissions.
XOASIS-84	Warning	The 'Skip Undefined LPPs' option will be ignored because a new library is being created.
XOASIS-85	Warning	Case conversion resulted in mapping of STRUCTURE %s to cell name %s. During OasisIn, translation will continue by overwriting cellview data. However, to avoid the collision do not specify the case conversion option.
XOASIS-86	UN-USED	Memory prediction is disabled because system parameters could not be found.
XOASIS-87	UN-USED	Process may require more virtual memory than 32-bit system can handle. You may need to run 64-bit version of the product. Available Memory:%llu MB Predicted Memory:%llu MB.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-88	UN-USED	Process might require more virtual memory than available on the system.\n Available Memory:%llu MB\tPredicted Memory:%llu MB.
XOASIS-89	Error	Failed to flatten via of viaDef '%s' at (%f, %f) in the cellview '%s/%s/%s'. OasisOut flattens a via in the specified cellview when the Flatten Vias option is used. Ensure that the via in the specified cellview is correct by opening the cellview '%s/%s/%s' and flattening the via in Virtuoso.
XOASIS-90	UN-USED	Process may require more virtual memory than available on the system for a process. You can increase this limit by using the 'limit' command.\nAvailable Memory:%llu MB\tPredicted Memory:%llu MB.
XOASIS-91	UN-USED	Process may require virtual memory near to what it is available on the system for a process. You can increase this limit by using the 'limit' command.\nAvailable Memory:%llu MB\tPredicted Memory:%llu MB.
XOASIS-92	UN-USED	Process may require more data segment than available on the system for a process. You can increase this limit by using the 'limit' command.\nAvailable Memory:%llu\tPredicted Memory :%llu.
XOASIS-93	UN-USED	Process may require data segment near to available on the system for a process. You can increase this limit by using the 'limit' command.\nAvailable Memory:%llu\tPredicted Memory:%llu.
XOASIS-94	UN-USED	The cellView %s/%s is not found in the mapped library %s. This cell will not be translated, and all the instances of this cell will be unbound.
XOASIS-95	UN-USED	Creating an unbound instance of the cellview '%s/%s/%s' as the master was not found in the Oasis file.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-96	UN-USED	Creating an unbound instance because the master cellview '%s/%s' was not found in the Oasis file or the specified reference library '%s'.
XOASIS-97	UN-USED	Creating an unbound instance of the cellview '%s/%s' as the master cannot be found in the mapped library '%s'.
XOASIS-98	UN-USED	Creating an unbound instance with library as target library %s because the master cellview '%s/%s' was not found in any reference library.
XOASIS-99	UN-USED	Creating an unbound instance with library as target library %s, because the master cellview '%s/%s' was not found in the Oasis file or in any of the reference libraries.
XOASIS-100	UN-USED	Instance Name %s has illegal characters. This instance is dropped.
XOASIS-101	Warning	An AREF record in the Oasis file contains the illegal angle '%f', this will not be translated. Angles supported by Oasis In are 0, 90, 180, 270.
XOASIS-102	Warning	The '%s' option specified is invalid. View the available options using the \-help\" command and then specify the valid option."
XOASIS-103	Warning	The '%s' option specified in the template file is invalid%s. View the available options using the \-help\" command and then specify the valid option."
XOASIS-104	UN-USED	The oasis attributes associated with some objects in the Oasis file exists multiple times or have multiple properties separated by a property separator. Only the first property value is preserved and the others are ignored.
XOASIS-105	UN-USED	

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-106	Warning	The undefined layer-purpose pair (LPP) are skipped because either the Skip Undefined LPPs option is specified or Oasis In into an existing library is performed. To avoid skipping of undefined LPPs, ensure that the Skip Undefined LPPs option is not specified or Oasis In should be performed into a new library.
XOASIS-107	Warning	Existing cells in the target library may be overwritten if the cell name in the Oasis file is same as the cell name in the target library. Specify the Do Not Overwrite Existing Cells option and try again.
XOASIS-108	Error	Layer map file is not being processed because the technology file is not available.
XOASIS-109	UN-USED	Pin text map file is not being processed because the technology file is not available.
XOASIS-110	UN-USED	Layer %s not found in the technology file.
XOASIS-111	UN-USED	Purpose %s not found in the technology file.
XOASIS-112	Error	Failed to open the user SKILL file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-113	Warning	Unable to load the SKILL file %s because of the undefined SKILL function and syntactical error. Specify the SKILL function with valid syntax and translate again.
XOASIS-114	Warning	Error in the definition of the SKILL function '%s'. The SKILL function maps the oasis %s to %s. Specify the correct definition of the SKILL function and restart the translation.
XOASIS-115	UN-USED	Error in the definition of 'textFontMap()'. Invalid font name for the Oasis font id '%d'
XOASIS-116	Error	Illegal font '%s' mapped to the Oasis font id '%d'. This mapping will be ignored.
XOASIS-117	Warning	Error in the definition of 'piCellNameMap()'. Cell name '%s' is mapped to NULL

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-118	Warning	Error in the definition of 'piLayerMap()'. No layer string returned for layer '%u' and datatype '%u'.
XOASIS-119	Warning	Error in the definition of 'piLayerMap()'. No purpose string returned for layer '%u' and datatype '%u'.
XOASIS-120	Error	Translation failed because the file size of Oasis file '%s' is zero. Ensure that the correct Oasis file is specified.
XOASIS-121	UN-USED	Could not process the library '%s'.
XOASIS-122	UN-USED	
XOASIS-123	UN-USED	
XOASIS-124	UN-USED	
XOASIS-125	UN-USED	
XOASIS-126	UN-USED	
XOASIS-127	UN-USED	
XOASIS-128	UN-USED	
XOASIS-129	UN-USED	
XOASIS-130	UN-USED	Error occurred in the user SKILL function 'textFontMap' while mapping '%s'.
XOASIS-131	UN-USED	Font name '%s' mapped to invalid Oasis font number '%d' by the user SKILL function 'textFontMap'. This mapping will be ignored.
XOASIS-132	Warning	Error occurred in the user SKILL function 'poCellNameMap'.
XOASIS-133	Warning	Error occurred in the user SKILL function 'poLayerNameMap'.
XOASIS-134	Warning	Error in the definition of 'poLayerMap()'. Invalid layer number returned for layer '%s'.
XOASIS-135	Warning	Error in the definition of 'poLayerMap()'. Invalid purpose number returned for purpose '%s'.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-136	Warning	Error occurred in the user SKILL function 'piTextMap' while mapping '%s'. This text will not be translated.
XOASIS-137	Warning	Error occurred in the user SKILL function 'poTextMap' while mapping '%s'. This text will not be translated.
XOASIS-138	Info	The 'enableLocking' option is selected so the output cellViews in the Cadence database will be separately locked. The translation may take more time as locking is enabled.
XOASIS-139	UN-USED	
XOASIS-140	UN-USED	The object map file '%s' contains an invalid record at line number '%d' and will be ignored. This is because the format of the mapping entry in the object map file is invalid. The valid mapping entry format is: <objectType objectSubType [oaLayer] oasisLayer oasisType [PhotomaskColor]>, where 'PhotomaskColor' is valid only when the '-enableColoring' option is set. Now, update the mapping entry and then translate again.
XOASIS-141	UN-USED	The object map file '%s' contains an invalid object type '%s' at line number '%d' and will be ignored. This is because the format of the mapping entry in the object map file is invalid. Update the mapping entry format to: <objectType objectSubType oasisLayer oasisType> and try again.
XOASIS-142	UN-USED	The object map file '%s' contains an invalid sub object '%s' at line number '%d' and will be ignored. This is because the format of the mapping entry in the object map file is invalid. Update the mapping entry format to: <objectType objectSubType oasisLayer oasisType> and try again.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-143	UN-USED	Object map file '%s' contains multiple entries for objectType:sub-objectType '%s:%s' at line number '%d'. Only the first record will be used.
XOASIS-144	Warning	Property ID '%d' specified in property mapping file at line '%d' is not within range, '0-125'. This entry will be ignored.
XOASIS-145	UN-USED	The structure name '%s' has been changed to '%s' by the translator. This is because of either the multiple cells with the same cell name are present in different libraries, a cell name mapping is done to the same cell names, or cell names greater than 32 characters are used when the Respect GDSII Name Limit option is selected.
XOASIS-146	Warning	Cell map file contains a duplicate oasis structure name '%s' at line number '%d'. This record in the cell map file will be ignored.
XOASIS-147	UN-USED	The property string associated with the object type '%s' exceeds the maximum permissible length of '%d' characters. Therefore, the property string will be truncated.
XOASIS-148	Error	Failed to open the via map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-149	Error	Failed to create the via map file '%s' in the run directory. Check that you have permission to create this file.
XOASIS-150	Warning	The via map file, '%s' already exists. It will be overwritten.
XOASIS-151	Warning	Failed to find via definition '%s' in the tech file. This entry in the viaMap file will be ignored.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-152	UN-USED	A polygon at (%d, %d) will be dropped because OpenAccess does not support a polygon which contains less than three coordinates. The number of distinct points required to create a polygon should be three. Rectify the source of the Oasis file and translate again.
XOASIS-153	Error	Failed to open object map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-154	UN-USED	The object map file does not contain any mapping for the OpenAccess object '%s' and the sub-object '%s'. Therefore, these objects in the design will not be translated. To enable the translation of these objects in the Oasis file, provide a valid mapping in the object map file. For details on the format of object map file, refer to the \Design Data Translator's Reference\" guide for XOasis."
XOASIS-155	UN-USED	Ignoring the OASIS layer:datatype pair '%d:%d' at row '%d' in the object map file '%s' because it is also defined in the layer map file. Additionally, shapes in OASIS on '%d:%d' with properties indicating they were translated from objects in the OpenAccess database will be translated back to objects. The remaining shapes will be translated to shapes.
XOASIS-156	Warning	The entry at line number '%d' is duplicate in the '%s' layerMap file. Therefore, this entry will be ignored during translation.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-157	UN-USED	'%s' SKILL procedure has generated a structure name '%s' that has been changed to '%s' by the translator. This is because of either the multiple cells with the same cell name are present in different libraries, a cell name mapping is done to the same cell names, or cell names greater than 32 characters are used when the Respect GDSII Name Limit option is selected.
XOASIS-158	Error	Error in the definition of the SKILL procedure '%s'. The SKILL procedure maps the PCell subMaster name to a NULL string. Correct the definition of the SKILL procedure for this error and restart the translation.
XOASIS-159	Warning	Error in the definition of the SKILL procedure '%s'. Either the evaluation of %s has failed or it has returned an invalid SKILL list. Correct the SKILL procedure and restart the translation
XOASIS-160	UN-USED	The design has an areaHalo blockage with owner instance. Property map file is needed to preserve instance name as a property on the instance. Specify property map file at -propMap option.
XOASIS-161	UN-USED	Property map file is needed to preserve areaHalo blockages. Property map file is not specified. Specify Property map file at -propMap option.
XOASIS-162	UN-USED	When an OpenAccess design has oaVia instances, Cadence recommends that you use the '-viaMap' option to preserve the oaVia instances during the XOasis Out - XOasis In round trip. The 'viaMap' option improves the performance by effectively using virtual memory during translation.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-163	Info	You have not used the viaMap option. If a viaMap file, corresponding techfile or technology library exist, then use the -viaMap option for creating oaVia instances in the imported library. Using the -viaMap option improves performance and VM usage of applications using the imported design. For details on the viaMap option, refer to the \Design Data Translator's Reference\" guide for XOasis."
XOASIS-164	UN-USED	%s. The %s will be dropped.
XOASIS-165	UN-USED	Multiple oasis structures are getting mapped to the same cell '%s' and view '%s' in the target library '%s'. Ensure that you have specified all required reference libraries and updated the cell name map file to avoid many-to-one mapping. Also ensure that input oasis file is valid.
XOASIS-166	Error	Translation Failed - %s\nlf problem persists, contact Cadence Customer Support.
XOASIS-167	Warning	Failed to open the sub-master. Internal automatic name mapping will be done for the sub-master %s as the SKILL function %s cannot be called for this sub-master.\nlf problem persists, contact Cadence Customer Support.
XOASIS-168	UN-USED	The structure, %s, in the oasis file is getting mapped to a cellview ('%s:%s') that already exists in the target library '%s'. Ignoring this cellview.
XOASIS-169	UN-USED	The structure, %s, in the oasis file is getting mapped to a cellview ('%s:%s') that already exists in the target library '%s'. Overwriting this cellview in the target library.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-170	Warning	The 'noObjectProp' option is used without the 'objectMap' option therefore it will be ignored. To use the noObjectProp option you must specify an object map file using the objectMap option.
XOASIS-171	Info	The 'noObjectProp' and the 'objectMap' options have been used together. Therefore, no property data is written to the oasis file on the oasis Boundary record.
XOASIS-172	UN-USED	Font map file %s exists in the technology library but is not readable. No font mapping will be done.
XOASIS-173	UN-USED	
XOASIS-174	Warning	The target library is already attached to technology library '%s'. Therefore, 'techRefs' option will be ignored
XOASIS-175	Warning	Mutually exclusive options '%s' and '%s' have been used together. Option '%s' will override option '%s' and option '%s' will be ignored.
XOASIS-176	Error	Translation Failed.\nlf problem persists, contact Cadence Customer Support.
XOASIS-177	UN-USED	Invalid oasis version '%d' specified. Valid values are 3,5 and 6. The translator will use version '%d' as default.
XOASIS-178	UN-USED	Oasis version '%d' and '%s' option have been used together. BOX record in oasis file is supported only for oasis version 5 and onwards. The translator will ignore '%s' option. If you want to translate rectangles in the design as BOX record in oasis file, specify version as '%d' or do not specify 'oasisVersion' option.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-179	Error	OasisIn has detected a conflict in the technology database graph. This could be because the technology library of the target library has conflict in its technology database graph or the specified reference technology libraries have a conflict with the technology library of the target library. Resolve the conflict manually before running the OasisIn application.
XOASIS-180	UN-USED	You have not used the objectMap option. The design has instance(s) of at least one of following OpenAccess objects: oaBlockage, oaBoundary, oaRow, and oaMarker. Any information about these objects will not be translated into the generated oasis file. Use the objectMap option to translate/preserve these objects using XOasis. For details on the objectMap option, refer to the \Design Data Translator's Reference\" guide for XOasis."
XOASIS-181	Warning	The '%s' label map file contains an unknown layer '%s' at line '%d'. Oasis Out will ignore this record in the label map file and continue to run. To fix this error, define the '%s' layer name in the technology file and restart Oasis Out.
XOASIS-182	Warning	The '%s' label map file contains an unknown purpose '%s' at line '%d'. Oasis Out will ignore this record in the label map file and continue to run. To fix this error, define the '%s' purpose name in the technology file and restart Oasis Out.
XOASIS-183	Warning	The '%s' label map file contains an invalid record at line '%d'. Oasis Out will ignore this record in the label map file and continue to run.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-184	Warning	The '%s' label map file contains a duplicate record for the '%s:%s' OA layer;purpose at line '%d'. Oasis Out will ignore this record in the label map file and continue to run.
XOASIS-185	Warning	The value 'ignore' is specified for the 'Convert Pin To' option simultaneously with the 'Label Map' option. Oasis Out will ignore the 'Label Map' option and continue to run.
XOASIS-186	Warning	XOasis Out has mapped the data based on the 'Label Map' file. This is because the value 'text' or 'geometryAndText' has been specified using 'convertPin' or 'convertPcellPin' along with a 'Label Map' file that has been specified using 'labelMap'.
XOASIS-187	UN-USED	
XOASIS-188	UN-USED	OpenAccess does not support coincident or collinear paths. A path with either coincident or collinear points is encountered at (%d, %d). The coincident or collinear points will be eliminated and an equivalent path will be created during translation.
XOASIS-189	UN-USED	A path with an invalid begin or end extension value is encountered at (%d, %d). This path will be ignored during translation.
XOASIS-190	UN-USED	OpenAccess does not support coincident or collinear polygons. A polygon with either coincident or collinear points is encountered at (%d, %d). The coincident or collinear points will be eliminated and an equivalent polygon will be created during translation.
XOASIS-191	UN-USED	A polygon with self-intersecting points is encountered at (%d, %d). This polygon will be translated as is. Correct the source of the translation if the self-intersecting polygon is not required.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-192	UN-USED	A path with self-intersecting points is encountered at (%f, %f). This path will be translated as is. Correct the source of the translation if the self-intersecting path is not required.
XOASIS-193	UN-USED	An open polygon is encountered at (%d, %d). This polygon will be ignored during translation.
XOASIS-194	UN-USED	OpenAccess does not support paths that have less than two coordinates. A path starting at (%d, %d) has less than two coordinates and therefore will be ignored during translation.
XOASIS-195	UN-USED	Value '%d' specified for option '%s' is %s than the valid range that is %s. %s value for this option has been set.
XOASIS-196	UN-USED	The design has %s with octagonal extensions in the cellview '%s/%s/%s'. These paths will be written as PATHTYPE1 (circular) in the Oasis file. You can retain the geometry of these %s by using the Convert Path to Polygon option.
XOASIS-197	UN-USED	The Oasis files has some PATH records with PATHTYPE 1 (circular). Virtuoso workbench will view extensions of these paths as octagonal. There could be potential problems and you should correct these at the source of the Oasis file.
XOASIS-198	UN-USED	The oasis layer-datatype '%d:%d' pair at row number '%d' in the object map file '%s' is already mapped. This entry in object map file will be ignored.
XOASIS-199	UN-USED	The oasis layer:datatype '%d:%d' is mapped to sub-object '%s' of layerBlockage which is not supported in DFII.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-200	Warning	OasisIn is running in automatic layer mapping mode and hence 'skipUndefinedLPP' option will be ignored. Specify layer map file name for OasisIn to respect 'skipUndefinedLPP' option.
XOASIS-201	UN-USED	A path with self-intersecting points is encountered at (%d, %d). This path will be translated as is. Correct the source of the translation if the self-intersecting path is not required.
XOASIS-202	Warning	The 'topCell' option is used without any view name. During OasisOut, the translator will translate all the views of maskLayout type for each of the specified topCell(s). Specify the view name that needs to be translated.
XOASIS-203	Warning	The 'view' name is not specified. During OasisOut, the translator will translate all the views of maskLayout type of the specified library. Specify the view name that needs to be translated.
XOASIS-204	UN-USED	%s. The corresponding structure definition of this master cell view will be missed in the oasis file. Importing this Oasis file may result in unbound instances. Ensure that the master cell view exists and the Lib is defined in library definition file.
XOASIS-205	Info	The Oasis layer-datatype pair %d:%d is mapped to layer-purpose %s:%s, which does not exist in the technology library. All shapes on Oasis layer-datatype pair %d:%d will be skipped. To avoid skipping shapes on the undefined LPPs, do not specify the '-skipUndefinedLPP' option.
XOASIS-206	UN-USED	%s The path would be written in Oasis file as is. To resolve this, fix the reported path in the input design and run the translator again.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-207	Warning	%s The path would be created with the dimensions in Oasis file. To resolve this, fix the source from which the Oasis file is generated and then run the translator again with the new Oasis File.
XOASIS-208	Error	License initialization failed due to some internal error. Try again and if problem persists, contact Cadence Customer Support.
XOASIS-209	Error	The required license for XOasis translator could not be found. Either the license server is down or the license is not available. Ensure that the valid license is available.
XOASIS-210	Error	Failed to initialize %s subsystem. For more information, contact Cadence customer support.
XOASIS-211	Error	The argument with option '%s' could not be found. This is because the value for the argument is not specified. Specify the valid value and then try again.
XOASIS-212	Error	The value for the textHeight argument is invalid. This is because the value of the argument is less than zero. Specify the positive integer value and then try again.
XOASIS-213	Error	The value for argument '%s' is invalid. This is because the argument '%s' contains a value other than upper, lower, or preserve. Specify the value for the argument '%s' as upper, lower, or preserve and then try again.
XOASIS-214	Error	The value for the argument '%s' is invalid. Specify the value as cdba, lef, def, native, spf, spf, spice, verilog, or vhdl and then try again.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-215	Error	The option in the command line is missing. This is because either both -oasisFile and -library options are not specified or the -cellListFile option is not specified. Specify -oasisFile and -library options or the -cellListFile option and then translate again.
XOASIS-216	Error	
XOASIS-217	Info	Reading the %s file, %s
XOASIS-218	Error	The parsing of the template file, '%s' cannot be completed. This is because the undefined UNIX environment variable, '%s' is used in the value at line number '%d'. Ensure that the UNIX environment variable is defined and then try again.
XOASIS-219	UN-USED	The value for the maxCellsInTargetLib argument is invalid. This is because the value of the argument is less than zero. Specify the positive integer value of the argument and try again.
XOASIS-220	Info	Translating STRUCTURE %s as cellview %s/%s/%s.
XOASIS-221	UN-USED	Cell contains a zero width path. The zero width path will not be translated.
XOASIS-222	Info	%s
XOASIS-223	UN-USED	%d. Translating cellView %s/%s/%s as STRUCTURE %s.
XOASIS-224	Warning	Unable to find library %s
XOASIS-225	UN-USED	Error in reading template file %s
XOASIS-226	Warning	File name for %s is not specified
XOASIS-227	Warning	Error in opening file %s
XOASIS-228	Error	Failed to open object map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-229	UN-USED	

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-230	UN-USED	
XOASIS-231	Error	Pcell evaluation failed for '%s' because the Pcell SKILL code contains either a syntax error or an unsupported XOasis function. Check the standard output or the Virtuoso log file for more information. Cadence recommends correcting the Pcell SKILL code to resolve the issue. However, to ignore these errors and continue the translation, you may use the 'ignorePcellEvalFail' option.
XOASIS-232	Warning	Pcell evaluation failed for '%s' because the Pcell SKILL code contains either a syntax error or an unsupported XOasis function. Check the standard output or the Virtuoso log file for more information. The translation will continue because the 'ignorePcellEvalFail' option is enabled, but the output should be checked to ensure it meets your requirements.
XOASIS-233	Error	The creation of a run directory '%s' has failed. Ensure that you have appropriate permissions to create the directory.
XOASIS-234	Info	Translation completed. '%d' error(s) and '%d' warning(s) found.
XOASIS-235	UN-USED	A LayerBlockage will be dropped because the layer used for the layerBlockage is not defined in the technology library. Ensure that the correct technology library is used and then translate again.
XOASIS-236	UN-USED	A path with a rounded extension, an acute angle or a length segment less than half width is encountered at (%d, %d). This path will be written as a boundary record in the Oasis file.
XOASIS-237	UN-USED	The Layer-purpose pair '%u:%u' does only exist with \floating keyword\. The \"floating\" layer will also be used to map \"non floating\" shapes."

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-238	UN-USED	The Layer-purpose pair '%s:%s' does only exist with \floating keyword\. The \"floating\" layer will also be used to map \"non floating\" shapes."
XOASIS-239	UN-USED	The value '%d' specified for the viaCutArefThreshold option is less than 15. The recommended value of the option is greater than or equal to 15. Therefore, the value of the option will be defaulted to 15.
XOASIS-240	UN-USED	
XOASIS-241	UN-USED	Unable to create the ASCII techfile '%s' in the run directory. Ensure that you have the write permission to create this file.
XOASIS-242	UN-USED	Unable to create the LayerMap file '%s' in the run directory. Ensure that you have the write permission to create this file.
XOASIS-243	UN-USED	Unable to open the Cadence default technology library \"%s\". Ensure that the library has been installed properly."
XOASIS-244	UN-USED	The default mapping, without the cutSize qualifier, for Layer-purpose pair '%u:%u' is not defined in the layer map file. All the objects which map to default mapping on this LPP will be ignored.
XOASIS-245	UN-USED	The default mapping, without the cutSize qualifier, for Layer-purpose pair '%s:%s' is not defined in the layer map file. All the objects which map to default mapping on this LPP will be ignored.
XOASIS-246	Warning	Ignoring the record at line number '%d' in the layer map file '%s'. This is because the qualifier \cutsize\" specified for the layer '%s' is not valid for the layer's material type '%s'. Valid material types for the qualifier are cut

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-247	Warning	Layer map file '%s' contains an invalid record for OA layer:purpose '%s:%s' and Oasis layer:datatype '%d:%d' at line number '%d' and will be ignored. This is because the \floating\" qualifier is specified for the %s purpose whose type is neither fill nor oaFillOPC."
XOASIS-248	Error	Failed to open label map file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-249	Error	The cellListFile does not contain any valid records to process. Ensure that the cellListFile contains valid records in the following format: <LibraryName> <CellName> <ViewName> <OasisFileName>
XOASIS-250	Error	Failed to create output directory '%s'. Ensure that appropriate permissions are set to create this directory, or provide a different output directory.
XOASIS-251	Error	Failed to open cellList file '%s'. Check that the file exists in the run directory and you have appropriate permissions to read it.
XOASIS-252	Warning	The cellListFile contains invalid record at line number '%d'. This record in cellListFile will be ignored. Ensure that each line in cellListFile contains all the four entries in the following format: <LibraryName> <CellName> <ViewName> <OasisFileName>
XOASIS-253	Warning	Multiple records with different library names exist in the cellListFile and will be ignored. Presently, only one library name is supported in the cellListFile.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-254	Warning	The -view option in the command line is missing and '*' is specified in the view entry of the cellListFile. The entry will be ignored because the value of the -view option that needs to replace '*' is missing. Specify the -view option or view name in cellListFile.
XOASIS-255	Warning	The cellListFile contains duplicate entries for cellView '%s/%s/%s' and will be ignored. Ensure that only one record for each cellView exists in the cellListFile.
XOASIS-256	Warning	The cellView '%s/%s/%s' specified in the cellListFile does not exist on-disk, and entry will be ignored. Ensure that the cellView exists and the path of the library '%s' is specified in library definition file.
XOASIS-257	Warning	Library name specified in the cellListFile is different than specified with -library option and will be ignored. Presently, only one library name is supported in the cellListFile.
XOASIS-258	Warning	The -library option in the command line is missing and '*' is specified in the lib entry of the cellListFile. This entry will be ignored because the value of the -library option that needs to replace '*' is missing. Specify the -library option or lib name in cellListFile.
XOASIS-259	Warning	The property map file '%s' contains invalid record at line number '%d'. The valid object types are: Dot, Line, Path, Polygon, Rect, Text, ArrayInst, ScalarInst, Arc, Donut, Ellipse, PathSeg, Route, Marker, Row, AreaBoundary, PRBoundary, SnapBoundary, ClusterBoundary, AreaBlockage, AreaHalo, LayerBlockage, and LayerHalo. Specify the valid object types and try again.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-260	Error	The option in the command line is missing. This is because the -oasisFile option is not specified and either of the -oasisTechGen or -library option is not specified. Specify the -oasisFile option and either of the -oasisTechGen or -library option and then translate again.
XOASIS-261	UN-USED	The object map file does not contain any mapping for the '%s' LayerBlockage on layer '%s'. Therefore, this LayerBlockage in the design will not be translated. To enable the translation of this LayerBlockage in the Oasis file, provide a valid mapping in the object map file. For details on the format of object map file, refer to the \Design Data Translator's Reference\" guide for XOasis."
XOASIS-262	Error	Failed to translate cellView, '%s/%s/%s'. This is because the cellView is opened by another process in the edit mode. Close the cellView and then translate again.
XOASIS-263	UN-USED	
XOASIS-264	UN-USED	Failed to translate cellView, '%s/%s/%s' - '%s'.\nIf problem persists, contact Cadence Customer Support.
XOASIS-265	UN-USED	
XOASIS-266	UN-USED	The property map file '%s' contains a record with more than one property name at line number '%d'. Only the first property name will be considered in XOasis In and others will be ignored.
XOASIS-267	Warning	%s Ensure that the referenced cell exists and library is defined in the cds.lib file.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-268	UN-USED	%s The required standard via definition does not exist in the technology database, which is used by the input design library. This is because either the required via definition is removed from the technology database after the design is created or the design library is attached to a different technology library. Ensure that the correct technology database is used by the input design library and then translate again.
XOASIS-269	Error	Failed to open excludeMapToVia file '%s'. Check that the file exists in the run directory and that you have permission to read it.
XOASIS-270	Warning	The excludeMapToVia file '%s' contains an invalid record at line '%d'. This record will be ignored during translation.
XOASIS-271	UN-USED	Value '%d' specified for option '%s' is %s than the valid range that is %s. %s value for this option has been set.
XOASIS-272	UN-USED	
XOASIS-273	Warning	The 'hierDepth' option is used without the 'topCell' option therefore it will be ignored. To use the hierDepth option you must specify topCell option.
XOASIS-274	Error	The value for argument '-convertDot' is invalid. This is because the argument '-convertDot' contains a value other than circle, polygon, or ignore. Specify the value for the argument '-convertDot' as circle, polygon, or ignore and then try again.
XOASIS-275	Error	The value for '%s' is invalid. Valid values are geometry, text, geometryAndText or ignore.
XOASIS-276	Error	The font name '%s' specified is an invalid font name. The valid font names are: euroStyle, gothic, math, roman, script, stick, fixed, swedish, or milSpec. Specify a valid font name and try again.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-277	Error	The orientation name '%s' specified is an invalid orientation name. The valid orientation names are: R0, R90, R180, R270, MY, MYR90, MX, or MXR90. Specify a valid orientation name and try again.
XOASIS-278	UN-USED	
XOASIS-279	UN-USED	
XOASIS-280	UN-USED	
XOASIS-281	UN-USED	
XOASIS-282	Info	Translation Failed. '%d' error(s) and '%d' warning(s) found.
XOASIS-283	Error	Translation Failed due to unknown error at line number - %s\nIf problem persists, contact Cadence Customer Support.
XOASIS-284	UN-USED	XOASIS In and XOASIS Out translators are only available to beta customers. For more information, contact the Cadence Sales Representative
XOASIS-285	Info	Performing automatic layer mapping because either the layer mapping information is not provided or the XOASIS_AUTO_LM value is used with the - layerMap option. Additionally, the '%s' file has been generated and the mapping information has been added to it.
XOASIS-286	Info	Performing automatic layer mapping because either the layer mapping information is not provided or the XOASIS_AUTO_LM value is used with the - layerMap option.
XOASIS-287	Info	Output OASIS/Log file '%s' already exists. Therefore, OasisOut will create a backup file with the name, '%s.bak' as the - backupOasisLogFiles option is specified.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-288	Warning	Both output OASIS/log file '%s' and its backup file '%s.bak' already exist. Therefore, OasisOut will remove the existing '%s.bak' and create a new backup file '%s.bak' because the '-backupOasisLogFiles' option is specified.
XOASIS-289	Error	Translation stopped. Both output OASIS/log file '%s' and its backup file '%s.bak' already exist. Therefore, OasisOut will try to remove the existing '%s.bak' and create a new backup file '%s.bak' because the '-backupOasisLogFiles' option is specified. However, the '%s.bak' file does not have the write permission. Ensure that the '%s.bak' file have the appropriate permissions and then try again.
XOASIS-290	Warning	The '%s' option will be ignored in command line. This option is only available in GUI. Use the \-help\" command to view the options available in command line."
XOASIS-291	Warning	The '%s' option specified in the template file will be ignored in command line. This option is only available in GUI. Use the \-help\" command to view the options available in command line."
XOASIS-292	UN-USED	
XOASIS-293	Info	The 'numThreads' option is used. Messages in the logFile may not be in deterministic order.
XOASIS-294	UN-USED	
XOASIS-295	Info	Value '%d' specified for option '%s' is %s than the valid range that is %s. %s value for this option has been set.
		The property map file '%s' contains duplicate entries for property name '%s', object type '%s' at line number '%d'. This entry in the property map file will be ignored.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-296	Warning	The %s map file '%s' contains color aware mapping(s) and will not be visible in GUI. This is because the 'Enable Coloring' option is not selected. Select the 'Enable Coloring' option to display these entries in the table else these entries will be ignored during translation.
XOASIS-297	Warning	The %s map file '%s' contains a color aware entry at line number '%d' and this entry will be ignored. To consider this entry specify the '-enableColoring' option to do color aware mapping.
XOASIS-298	Error	Translation cannot be continued as the '-enableColoring' option is used, but the layer map file is not found. This is because either the -layerMap option is not specified or technology does not have the layer map file. Use a valid layer map file and then translate again.
XOASIS-299	Error	Translation cannot be continued as the '-enableColoring' option is used, but technology is not found. Provide a technology either by using the '-attachTechFileOfLib' option, or ensure that library either has valid technology or is attached to the valid technology in case the target library already exist and then translate again.
XOASIS-300	Warning	The layer map file '%s' contains inconsistent color aware entry at line number '%d' and this entry will be ignored. This is because layer '%s' does not support either coloring or Photomask color '%s'.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-301	Warning	Connected pathSegs on colorable layers will not be merged and each pathSeg will be written as separate PATH record in GDS. This is because both '-enableColoring' and '-mergePathSegsToPath' options have been specified. Therefore, the '-mergePathSegsToPath' option will not be considered for the pathSegs on colorable layers.
XOASIS-302	UN-USED	Array instances will be written as multiple SREF records in OASIS file. This is because both '-enableColoring' and '-topCell' options have been specified and the layer map file has unlocked mapping entries. Therefore, different color variant might need to be generated for each instance of an array instance.
XOASIS-303	Warning	The OASIS file will be generated without the effective color for the unlocked color shapes. This is because the '-topCell' option is not specified, although the '-enableColoring' option is specified and the layer map file has unlocked mapping entries. The effective color requires color variant generation that depends on the hierarchy context. To generate the OASIS file with the effective color, define the hierarchy context using the '-topCell' option.
XOASIS-304	Error	Value '%d' specified for option '%s' is %s than the valid range that is %s. %s value for this option has been set.
XOASIS-305	Error	Translation has been aborted because the -compress option has not been specified with the -compressLevel option. When the -compressLevel option is used, the -compress option must be specified.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-306	Warning	The '-enableColoring' option will be ignored during %s. This is because there is no color entry in the layer map and the object map file.
XOASIS-307	Info	To improve performance of flattening Pcell instances the preserving Pcell pins is not done during XOasis Out. This is because preserving Pcell pins is not required as the value 'geometry' or 'ignore' is specified for the Convert Pin To option.
XOASIS-308	Warning	The '-doNotPreservePcellPins' option will be ignored during xOasis Out because the '-flattenPcells' option has not been specified.
XOASIS-309	UN-USED	
XOASIS-310	Error	Failed to attach technology '%s' to the library '%s' because read permission is not set for the database of at least one of the libraries or technologies defined in cds.lib. Check that you have appropriate permissions and try again.
XOASIS-311	Warning	The input OASIS file is empty; therefore, XOasis In has no cells to translate.
XOASIS-312	Warning	The input OASIS file is empty and has no cells that can be translated. The preexisting library '%s' will remain as it is.
XOASIS-313	Warning	Data precision might be lost because the value (%d) set for the 'dbuPerUU' option does not match the 'dbuPerUU' value (%d) in technology database '%s'.
XOASIS-314	Warning	DbuPerUU (%d) specified in the technology database or using '-dbuPerUU' is not divisible by the value (%d) set in the OASIS file. This can cause loss in data precision during translation.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-315	Info	The 'dbuPerUU' option will be ignored because the value (%d) does not match the 'dbuPerUU' value (%d) in technology database '%s'. Using the 'dbuPerUU' value from the technology database.
XOASIS-316	Warning	The technology library '%s' contains colored data that is incompatible with this version of the translator and should be used ideally with an advanced node version of the translator (for example ICADV12.1). Make sure that you are using the correct version.
XOASIS-317	Info	The '-enableColoring' and '-topCell' options are specified but '-arrayInstToScalar' option is not specified. Additionally, the layer map file has unlocked mapping entries. The OASIS file will be generated with the effective color for the unlocked color shapes. However, array instances might be written as multiple SREF records in the OASIS file because the effective color requires color variant generation.
XOASIS-318	Info	The '-enableColoring' and '-topCell' options have been specified, but the layer map file does not have any unlocked mapping entry. Therefore, all the unlocked color shapes in the OA database will be translated according to noncolor-aware mapping in the OASIS file.
XOASIS-319	Warning	The '%s' option will be ignored during XOasis %s because the '%s' option has not been specified.
XOASIS-320	Warning	The technology library has been modified during XOasis In translation. Cannot save the changes in the technology file because it is open in read-only mode.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-321	Warning	XOasis Out has mapped the data based on the 'Label Map' file. This is because the value 'text' or 'geometryAndText' has been specified using 'convertPin' or 'convertPcellPin' along with a 'Label Map' file that is located in the technology library.
XOASIS-322	Info	xOasis Out is propagating the locks because it is in the 'lock same color group' mode.
XOASIS-323	Info	xOasis Out will not be in the 'lock same color group' mode. This is because %s.
XOASIS-324	Error	The translation failed because the option '-guiHistory' has been specified but no successful translation was done from the XOasis %s GUI. Remove the '-guiHistory' option and rerun the translation.
XOASIS-325	Warning	Ignoring the layer-datatype pair '%s:%s' in the layer map file '%s' at line number '%d' because the technology library is read-only and the layer-datatype pair is not defined in the technology file.
XOASIS-326	Warning	Ignoring the layer-datatype pair '%s:%s' in 'piLayerMap' because the technology library is read-only and the layer-datatype pair is not defined in it. XOasis In will automatically map this layer-datatype pair.
XOASIS-327	Error	Cannot translate because the '-translateUnmappedLPP' option has been specified but the technology library '%s' has target library '%s' attached to it and is therefore open in read-only mode. Due to this, layers, purposes, and LPPs cannot be created with '-translateUnmappedLPP'. Either do not specify the '-translateUnmappedLPP' option or specify a target library that contains a 'tech.db' or incremental technology file.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-328	Warning	Ignoring the '-translateUnmappedLPP' option during XOasis %s because it does not impact automatic layer mapping. Use this option with the 'layerMap' file or the layer mapping SKILL function '%s'.
XOASIS-329	Info	Ignoring the '-translateUnmappedLPP' option during XOasis In because the '-skipUndefinedLPP' option has been defined.
XOASIS-330	Info	Applied automatic layer mapping to the layer-purpose pair '%u:%u'.
XOASIS-331	Info	Applied automatic layer mapping to the layer-purpose pair '%s:%s'.
XOASIS-332	Warning	Ignoring option '%s' %sbecause it is incompatible with this version of the translator. Use this option with an advanced node version of the translator (for example ICADV12.3).
XOASIS-333	Info	The STRUCTURE '%s' in the input OASIS file will not be translated. This is because the refLibList option is used and the cellview is available in the reference library '%s'. Therefore, all the instances of the cellview will be referenced from '%s/%s/%s'.
XOASIS-334	Warning	Ignoring color aware entries in the via map file '%s'. To consider these entries during translation, specify the '-enableColoring' option.
XOASIS-335	Warning	Ignoring color aware entries in the via map file '%s'. To consider these entries for via preservation with coloring information, use an advanced node version of the translator (for example ICADV12.3), and specify the '-enableColoring' option.
XOASIS-336	Warning	Ignoring via subMaster record '%s' in the via map file '%s' because %s

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-337	UN-USED	Value '%d' specified for option '%s' is %s than the valid range that is %s. %s value for this option has been set.
XOASIS-338	Error	The value '%s' specified for the scale option is invalid. This is because the value of the argument is either negative or zero. Specify the appropriate positive value for the scale option and rerun the translation.
XOASIS-339	Warning	The STRUCT '%s' generated for '%s/%s/%s' in OASIS is empty because either the design is empty or no shape or object was translated.
XOASIS-340	Warning	Running the replay function in support mode. This replay function was created using an older release, such as IC6.1.7 or ICADV12.3 and is now obsolete. Regenerate the replay function using the redesigned XOasis GUI.
XOASIS-341	Warning	The '-detectVias' option is no longer supported in XOasis In. The vias will be detected automatically. Use the '-noDetectVias' option if via detection is not required.
XOASIS-342	Warning	The '-detectVias' option specified in the template file is no longer supported in XOasis In. The vias will be detected automatically. Use the '-noDetectVias' option if via detection is not required.
XOASIS-343	Warning	The '-detectVias' option specified in the xoasInSetField or xoasInGetField SKILL function or the oasisInKeys SKILL list is no longer supported in XOasis In. The vias will be detected automatically. Use the '-noDetectVias' option if via detection is not required.
XOASIS-344	Warning	The '-%s' option is no longer supported. Use the '-writeMode' option with value '%s' instead.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-345	Warning	The '%s' option specified in the template file is no longer supported. Use the '-writeMode' option with value '%s' instead.
XOASIS-346	Warning	The '%s' option specified in the 'xoasInSetField' or 'xoasInGetField' SKILL functions or the oasisInKeys SKILL list is no longer supported. Use the '-writeMode' option with value '%s' instead.
XOASIS-347	Error	An invalid value '%s' was specified for the '-writeMode' option. Set '-writeMode' to either 'noOverwrite', 'overwrite', 'append', or 'rename' and run the command again.
XOASIS-348	Warning	The '-overwriteRootCells' option has been ignored because it is valid only when '-writeMode' is set to 'rename'.
XOASIS-349	Warning	The '-enableIncrementalImport' option is ignored during XOasis In because the library specified is not the same as the primary library used for previous incremental import project. To continue in the same incremental import project, specify the '%s' library as the primary library. You can also start a new incremental project with a different primary library by running the translation in a new directory or by deleting the '.cadence/geoTrans' directory present in the current directory.
XOASIS-350	Info	The number of cells in library '%s' exceeds the limit of %d. A new library '%s' has been created.
XOASIS-351	Error	Failed to open the '%s' file. Check that the file exists and that you have the permission to modify it.
XOASIS-352	Warning	The 'numThreads' option is no longer supported in XOasis In. The number of cores in the CPU will be detected automatically and an appropriate number of threads will be used for translation.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-353	Warning	The 'numThreads' option specified in the template file is no longer supported in XOasis In. The number of cores in the CPU will be detected automatically and an appropriate number of threads will be used for translation.
XOASIS-354	Warning	The 'numThreads' option specified in the xoasInSetField or xoasInGetField SKILL function or the oasisInKeys SKILL list is no longer supported in XOasis In. The number of cores in the CPU will be detected automatically and an appropriate number of threads will be used for translation.
XOASIS-355	Warning	The '-scaleTextHeight' option is no longer supported in XOasis In. Use the '-textHeight' option instead.
XOASIS-356	Warning	The '-scaleTextHeight' option specified in the template file is no longer supported in XOasis In. Use the '-textHeight' option instead.
XOASIS-357	Warning	The '-scaleTextHeight' option specified in the xoasInSetField or xoasInGetField SKILL function or the oasisInKeys SKILL list is no longer supported in XOasis In. Use the '-textHeight' option instead.
XOASIS-358	Warning	Translation failed while processing '%s' because of the following reason: %s Ensure that the specified cellview exists on disk and that you have permission to open it and then rerun the translation.
XOASIS-359	Info	Ignoring %s file '%s' in the technology library because it is not readable.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-360	Warning	Ignoring the qualifier 'OASISLayerName' in the layer map file '%s' at line number '%d' because the layer-datatype pair '%d:%d' is already mapped to a different OASISLayerName. However, the layer-mapping record is not ignored, and shapes on the layer-purpose pair '%s:%s' will be mapped to the layer-datatype pair '%d:%d'.
XOASIS-361	Warning	Option '%s' specified in the template file is invalid in XOasis %s GUI. Run \"%s -help\" at command prompt to view list of available options and then specify the valid option in the template file."
XOASIS-362	Warning	Both '-noDetectVias' and '%s' options have been specified. The '%s' option works when via detection is enabled but the '-noDetectVias' option disables via detection. Therefore, the '%s' option will be ignored.
XOASIS-363	Warning	Disk space is low (< %dMB of free space) and may fill up during translation. To avoid any data loss, free up some disk space.
XOASIS-364	Warning	The topcell '%s' contains a modified Concurrent Layout cellview. Changes in the top and hierarchical Concurrent Layout design partition views are not automatically exported from virtual memory to OASIS. To export these changes, right-click in the Concurrent Layout assistant and choose 'XOasis Out from VM for DRC'.
XOASIS-365	Warning	Ignoring the qualifier 'OASISLayerName' in the object map file '%s' at line number '%d' because the layer-datatype pair '%d:%d' is already mapped to a different OASISLayerName%s. However, the object-mapping record is not ignored, and the objectType:sub-objectType '%s:%s' will be mapped to the layer-datatype pair '%d:%d'.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-366	Info	The '-translateUnmappedLPP' option overrides any entry with 'ignoreLPP' qualifier specified in layermap file '%s'.
XOASIS-367	Warning	Layer map file '%s' contains a duplicate entry for Oasis layer:datatype '%d:%d' at line '%d' which maps the shapes to '%s:%s:%s:%s' but is already mapped to '%s:%s:%s:%s'. This entry will be ignored during translation.
XOASIS-368	Error	Failed to open the flatten Pcell exclusion file '%s'. Ensure that the file exists in the run directory and that you have permission to read it.
XOASIS-369	Info	Cannot flatten instances of Pcell master '%s/%s/%s' in cellView '%s/%s/%s' because the Pcell master is specified in the flatten Pcell exclusion file '%s'.
XOASIS-370	Info	There are multiple property names with property ID '%d' and object type '%s' in the property map file '%s'. This may cause issues during XOasis In translation.
XOASIS-371	Info	Values of property name '%s' and object type '%s' at line number '%d' in the property map file '%s' are same as the values for the property ID '%d'. Entry at line '%d' will be ignored during translation.
XOASIS-372	Info	The values of property ID '%d' and object type '%s' at line number '%d' in the property map file '%s' are same as the values for the property name '%s'. Entry at line '%d' will be ignored during translation.
XOASIS-373	Warning	The property map file '%s' contains an entry with more than one property name at line number '%d'. Only the first property name will be considered during XOasis In translation. All other entries will be ignored.

Design Data Translators Reference

List of XOasis Messages

Message ID	Severity	Message
XOASIS-374	Warning	The '-textHeight' option has been specified. This option will be ignored while translating labels with text attribute property.
XOASIS-375	Warning	The property map file '%s' contains an invalid format at line number '%d' and will be ignored. The valid mapping entry format is [<-fp/-fv> SP_CH_OPTSEP <-s 'property separator'>].
XOASIS-376	Info	DbuPerUU (%d) specified in the technology database or using '-dbuPerUU' does not match with the value (%d) set in the OASIS file.
XOASIS-377	UN-USED	Translation Failed - %s Enable the 'detectOverflow' option and try again.
XOASIS-378	Warning	Cannot translate because an existing non-VM library has been specified. Specify either a non-existing library or a pre-existing VM library, created using XOasis In VM mode and translate again.
XOASIS-379	Info	%s

Legend

Format	Data Type
%d	Signifies that this value will be replaced by an integer value.
%f	Signifies that this value will be replaced by a floating-point number.
%s	Signifies that this value will be replaced by a character string.
%u	Signifies that this value will be replaced by an unsigned integer.

Design Data Translators Environment Variables

This appendix covers the design data translators-specific environment variable names, descriptions, types, and values.

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

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

.cdsenv File

Add environment variables to the `.cdsenv` file when the settings are applied while launching Layout XL.

Use the following syntax:

```
xstream environmentVariableName dataType value
```

For example:

```
xstream XST_AUTO_LOAD_TEMPLATE_NAME 'string /home/user1/strmIn.tl
```

.cdsinit File

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

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

```
envSetVal("xstream" "XST_AUTO_LOAD_TEMPLATE_NAME" 'string "/home/user1/  
strmIn.tl")
```



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

CIW

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

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

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

Displaying the Current Value of an Environment Variable

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

```
envGetVal("xstream" "environmentVariableName")
```

For example:

```
("xstream" "XST_AUTO_LOAD_TEMPLATE_NAME")
```

Setting Shell Environment Variables

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

```
setenv environmentVariableName
```

For example:

```
setenv XST_AUTO_LOAD_TEMPLATE_NAME ""
```


List of Design Data Translators Environment Variables

This section covers the following design data environment variables:

XStream SKILL Environment Variables

- XST_AUTO_LOAD_TEMPLATE_NAME
- XST_IN
- XST_OUT
- xstAlwaysLookupMapFilesInTech
- xstEnableGUIHistory
- xstHideMapAllColorToLocked
- xstIgnoreCoveredViaShapes
- xstInOverwriteLibPopup
- xstOutAutoPopulateIfEmptyLCV
- xstShowCompletionMsgBox
- strmFilenameSuffix

XStream Fast Export SKILL Environment Variables

- convertPin
- layerMap
- logFile
- objectMap
- summaryFile
- enableColoring
- strmFilenamePrefix
- strmFilenameSuffix

Design Data Translators Reference

Design Data Translators Environment Variables

XStream Shell Environment Variables

- XST_ALL_VIEWS
- XST_AUTO_LOAD_TEMPLATE_NAME
- XST_IN
- XST_IN_VM_NT
- XST_OUT
- XST_OUT_CC_CHOP

XOasis SKILL Environment Variables

- xoasEnableGUIHistory
- XOASIS_AUTO_LOAD_TEMPLATE_NAME
- XOASIS_IN
- XOASIS_OUT
- xoasAlwaysLookupMapFilesInTech
- xoasOutAutoPopulateIfEmptyLCV
- xoasShowCompletionMsgBox
- oasisFileExtension

XOasis Fast Export SKILL Environment Variables

- enableColoring
- oasisFilenamePrefix
- oasisFilenameSuffix

XOasis Shell Environment Variables

- XOASIS_ALL_VIEWS
- XOASIS_AUTO_LOAD_TEMPLATE_NAME
- XOASIS_IN
- XOASIS_IN_VM_NT

Design Data Translators Reference

Design Data Translators Environment Variables

■ XOASIS_OUT

XST_AUTO_LOAD_TEMPLATE_NAME

```
xstream XST_AUTO_LOAD_TEMPLATE_NAME string "template_file_name"
```

Description

Loads the specified template file.

The default is " ", which loads the default template file.

Setting the Default Information in XStream

You can set the default values of XStream In and XOasis Out options by using the XST_IN and XST_OUT environment variables along with XST_AUTO_LOAD_TEMPLATE_NAME. The following table summarizes the various possibilities and the behavior.

Template File Specified in the XST_IN/XST_OUT Environment Variables	Default values specified in the XST_AUTO_LOAD_TEMPLATE_NAME Environment Variable	DF II	Command Line
Yes	default		In the first invocation, loads the template file pointed by the XST_IN/XST_OUT environment variable. In subsequent invocations, loads the values from its previous session.
No	default		In the first invocation, loads the GUI with empty fields because no template file is specified in the XST_IN/XST_OUT environment variable. In subsequent invocations, it loads the values from its previous session.
Yes	" "		Always loads the template file pointed by the <u>XST_IN</u> / <u>XST_OUT</u> environment variables.

Design Data Translators Reference

Design Data Translators Environment Variables

Template File Specified in the XST_IN/XST_OUT Environment Variables	Default values specified in the XST_AUTO_LOAD_TEMPLATE_NAME Environment Variable	DF II	Command Line
No	""		Always loads the GUI with empty fields because no template file is specified in the XST_IN/XST_OUT environment variable and it does not retain the values of the previous session.

GUI Equivalent

None

Examples

```
envGetVal("xstream" "XST_AUTO_LOAD_TEMPLATE_NAME")  
envSetVal("xstream" "XST_AUTO_LOAD_TEMPLATE_NAME" 'string "")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

XST_IN

```
xstream XST_IN string "template_file_path"
```

Description

Specifies the path for the template file to be used during XStream In translation.

The default is "".

You can set the default values of the XStream In option by using the `XST_IN` environment variable. This variable contains the path to the XStream In template file. The template file consists of the values for different XStream In options. Therefore, each time you run XStream In, the values of these options are taken from the template files and the path of the template file is taken from the `XST_IN` environment variables.

You can use the `XST_IN` environment variables while running XStream from GUI. For example, set `XST_IN` and invoke XStream In by using the following command:

```
setenv XST_IN /home/user1/templateIn.txt
```

XStream In is invoked in the interactive mode and the values of XStream In options are taken from the `templateIn.txt` template file specified in the `XST_IN` environment variable.

You can set `XST_IN` in your `.cdsenv` file by using the following command:

```
xstream XST_IN string "/home/user1/strmIn.tl"
```

For more information, about setting default values in XStream In using environment variables, see [Setting the Default Information in XStream](#).

GUI Equivalent

None

Examples

```
envGetVal("xstream" "XST_IN")  
envSetVal("xstream" "XST_IN" 'string "/home/user1/templateIn.txt")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

XST_OUT

```
xstream XST_OUT string "template_file_path"
```

Description

Specifies the path for the template file to be used during XOasis Out translation. The environment variable also helps in retaining the last successful translation option settings for next Virtuoso session.

The default is "".

You can set the default values of the XOasis Out option by using the XST_OUT environment variable. This variable contains the path to the XOasis Out template file. The template file consists of the values for different XOasis Out options. Therefore, each time you run XOasis Out, the values of these options are taken from the template files and the path of the template file is taken from the XST_OUT environment variables.

You can use the XST_OUT environment variables while running XStream from GUI. For example, set XST_OUT and invoke XOasis Out by using the following command:

```
setenv XST_OUT /home/user1/templateOut.txt
```

XOasis Out is invoked in the interactive mode and the values of XOasis Out options are taken from the `templateOut.txt` template file specified in the XST_OUT environment variable.

You can set XST_OUT in your `.cdsenv` file by using the following command:

```
xstream XST_OUT string "/home/user1/strmOut.tl"
```

For more information, about setting default values in XOasis Out using environment variables, see [Setting the Default Information in XStream](#).

GUI Equivalent

None

Examples

```
envGetVal("xstream" "XST_OUT")  
envSetVal("xstream" "XST_OUT" 'string "/home/user1/strmIn.tl")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

xstAlwaysLookupMapFilesInTech

```
xstream xstAlwaysLookupMapFilesInTech boolean { t | nil }
```

Description

Specifies whether the XStream automatically opens the technology library including ITDB hierarchy to search for map files in the technology libraries. If the technology library is already open, then setting this variable has no effect.

Fields for map files, if empty, are always automatically populated when technology library is already open in the current Virtuoso session. However, if the technology library is not open in the current Virtuoso session then XStream, by default, will open the technology library including the ITDB hierarchy to search for the map files ('<techlibname>.layermap' for layer map file, '<techlibname>.objectmap' for object map file, and '<techlibname>.labelmap' for label map file). Opening the technology library can take some time. Default behavior of opening the technology library including ITDB can be disabled by setting the value of this environment variable to `nil`.

Note: This setting only prevents auto population of layer map files in GUI. Translation will still traverse through all the technology libraries to search for layer map files.

The default is `t`.

GUI Equivalent

None

Examples

```
envGetVal("xstream" "xstAlwaysLookupMapFilesInTech")  
envSetVal("xstream" "xstAlwaysLookupMapFilesInTech" 'boolean t)  
envSetVal("xstream" "xstAlwaysLookupMapFilesInTech" 'boolean nil)
```

Related Topics

XStream In Main form – [Layer Map](#)

XStream In Main form – [Object Map](#)

XStream Out Main form – [Layer Map](#)

XStream Out Main form – [Object Map](#)

Design Data Translators Reference

Design Data Translators Environment Variables

XStream Out More Options form – Label Case Sensitivity

xstEnableGUIHistory

```
xstream xstEnableGUIHistory boolean { t | nil }
```

Description

Saves the GUI history that lets you use the last preserved GUI history at command line. This environment variable also helps in retaining the option settings used during the last successful translation in the next Virtuoso session.

The default is `nil`, which means that the GUI history is not saved.

GUI Equivalent

None

Examples

```
("xstream" "xstEnableGUIHistory")  
envSetVal("xstream" "xstEnableGUIHistory" 'boolean t)  
envSetVal("xstream" "xstEnableGUIHistory" 'boolean nil)
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Preserving GUI History in XStream](#)

xstHideMapAllColorToLocked

```
xstream xstHideMapAllColorToLocked boolean { t | nil }
```

Description

Hides the *Map All Colors to Locked* option.

The default is `t`, which means that the *Map All Colors to Locked* option is hidden.

GUI Equivalent

None

Examples

```
envGetVal("xstream" "xstHideMapAllColorToLocked")  
envSetVal("xstream" "xstHideMapAllColorToLocked" 'boolean t)  
envSetVal("xstream" "xstHideMapAllColorToLocked" 'boolean nil)
```

Related Topics

[List of Design Data Translators Environment Variables](#)

xstIgnoreCoveredViaShapes

```
xstream xstIgnoreCoveredViaShapes boolean { t | nil }
```

Description

Specifies whether standard via shapes covered by another shape on the same LPP should be exported during XOasis Out translation. The default is `nil`, which means that all standard via shapes are exported during XOasis Out translation.

When set to `t`, XOasis Out does not export standard via shapes if there is another shape on the same LPP inside the same standard via master that covers the first shape completely. This may happen in local-interconnect vias (with cut layer equal to either layer1 or layer2).

Note: This option does not work with the `-flattenVias` option.

GUI Equivalent

None

Examples

```
envGetVal("xstream" "xstIgnoreCoveredViaShapes")
envSetVal("xstream" "xstIgnoreCoveredViaShapes" 'boolean t)
envSetVal("xstream" "xstIgnoreCoveredViaShapes" 'boolean nil)
```

Related Topics

[List of Design Data Translators Environment Variables](#)

xstInOverwriteLibPopup

```
xstream xstInOverwriteLibPopup boolean { t | nil }
```

Description

Enables a warning message that is displayed during the XStream In translation to an existing library to warn you that cells in the target library will be overwritten or appended to in this process.

This message is displayed when Write Mode is set to *overwrite* or *append*.

The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("xstream" "xstInOverwriteLibPopup")  
envSetVal("xstream" "xstInOverwriteLibPopup" 'boolean t)  
envSetVal("xstream" "xstInOverwriteLibPopup" 'boolean nil)
```

Related Topics

[List of Design Data Translators Environment Variables](#)

xstOutAutoPopulateIfEmptyLCV

```
xstream xstOutAutoPopulateIfEmptyLCV boolean { t | nil }
```

Description

Specifies whether the *Stream File*, *Library*, *TopCell(s)*, and *View(s)* fields on the XOasis Out main form should be filled automatically only when they are empty. If these fields already have some information and if the environment variable is set to `t`, then this information is not replaced even if a cellview is open.

The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("xstream" "xstOutAutoPopulateIfEmptyLCV")  
envSetVal("xstream" "xstOutAutoPopulateIfEmptyLCV" 'boolean t)  
envSetVal("xstream" "xstOutAutoPopulateIfEmptyLCV" 'boolean nil)
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[XStream Out Form](#)

xstShowCompletionMsgBox

```
xstream xstShowCompletionMsgBox boolean { t | nil }
```

Description

Specifies whether the *Stream in translation complete* message box should display the number of errors and warnings after the translation is complete.

The default is `t`.

GUI Equivalent

GUI Equivalent

Command	<i>Virtuoso XOasis Out – More Options – General –</i>
Field	<u>Show Completion Message Box</u>

Examples

```
envGetVal("xstream" "xstShowCompletionMsgBox")
envSetVal("xstream" "xstShowCompletionMsgBox" 'boolean t)
envSetVal("xstream" "xstShowCompletionMsgBox" 'boolean nil)
```

Related Topics

List of Design Data Translators Environment Variables

strmFileExtension

```
xstream strmFileExtension string "Extension_Added_to_GDS_Files"
```

Description

Specifies the file extension to be added to the output top cell file name when the XStream Out translation is done using the *Layout XL — File — Export Stream from VM option*.

The default is `gds`.



This environment variable replaces `xstOutExt`. Support for the `xstOutExt` environment variable will be removed in a future release. Instead use `strmFileExtension`.

GUI Equivalent

None

Examples

```
envGetVal("xstream.fastExport" "strmFileExtension")  
envSetVal("xstream.fastExport" "strmFileExtension" 'string "sf")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Stream Out](#)

Design Data Translators Reference

Design Data Translators Environment Variables

convertPin

```
xstream.fastExport convertPin string { "geometry" | "text" | "geometryAndText" |  
    "ignore" }
```

Description

Specifies whether pins should be translated as geometries or text when Stream files are exported using the *Layout XL – File – Export Stream from VM* command.

The default is `geometry`.

GUI Equivalent

None

Examples

```
envGetVal("xstream.fastExport" "convertPin")  
envSetVal("xstream.fastExport" "convertPin" 'string "text")  
envSetVal("xstream.fastExport" "convertPin" 'string "geometry")  
envSetVal("xstream.fastExport" "convertPin" 'string "geometryAndText")  
envSetVal("xstream.fastExport" "convertPin" 'string "ignore")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Stream Out](#)

[Convert Pin to](#)

Design Data Translators Reference

Design Data Translators Environment Variables

layerMap

```
xstream.fastExport layerMap string { "name_of_layer_map_file" }
```

Description

Specifies the name of the layer map file to be used when Stream files are exported using the *Layout XL – File – Export Stream from VM* command.

The default is " ".

GUI Equivalent

None

Examples

```
envGetVal("xstream.fastExport" "layerMap")  
envSetVal("xstream.fastExport" "layerMap" 'string "strmLayerMap")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Stream Out](#)

[Layer Map File](#)

logFile

```
xstream.fastExport logFile string { name_of_log_file }
```

Description

Specifies the name of the log file to be used when Stream files are exported using the *Layout XL – File – Export Stream from VM* command.

The default is " ".

GUI Equivalent

None

Examples

```
envGetVal("xstream.fastExport" "logFile")  
envSetVal("xstream.fastExport" "logFile" 'string "strmLogFile")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Stream Out](#)

maxVertices

```
xstream.fastExport maxVertices int integer
```

Description

Specifies the maximum number of coordinates for polygons and paths in Stream. *Export – Stream* divides polygons and paths in DFII that have vertices more than or equal to `maxVertices` into smaller objects.

Polygons and paths are stored differently in Design Framework II and Stream formats. The DFII database does not store both the start and end points of a polygon because they are the same point. The Stream format requires that both points be stored individually. The maximum limit imposed by the GDSII format is 200 and that by the Stream format is 8191. Therefore, the Stream/GDSII reader/writer of each EDA vendor is responsible for handling restrictions imposed by their databases with this option.

The largest DFII polygon or path *Export – Stream* can process a maximum of 8191 vertices, which translates into 8191 Stream vertices.

Valid Values: 5 to 8191

The default is 200.

Command-line option: `maxVertices`

GUI Equivalent

None

Examples

```
envGetVal("xstream.fastExport" "maxVertices")  
envSetVal("xstream.fastExport" "maxVertices" 'int 500)
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Max Vertices](#)

[Fast Stream Out](#)

Design Data Translators Reference

Design Data Translators Environment Variables

objectMap

```
xstream.fastExport objectMap string { "name_of_object_map_file" }
```

Description

Specifies the name of the object map file to be used when Stream files are exported using the *Layout XL – File – Export Stream from VM* command.

The default is " ".

GUI Equivalent

None

Examples

```
envGetVal("xstream.fastExport" "objectMap")  
envSetVal("xstream.fastExport" "objectMap" 'string "strmObjMapFile")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Stream Out](#)

[Object Map File](#)

Design Data Translators Reference

Design Data Translators Environment Variables

summaryFile

```
xstream.fastExport summaryFile string { "name_of_summary_file" }
```

Description

Specifies the name of the summary map file to be used when Stream files are exported using the *Layout XL – File – Export Stream from VM* command.

The default is " ".

GUI Equivalent

None

Examples

```
envGetVal("xstream.fastExport" "summaryFile")  
envSetVal("xstream.fastExport" "summaryFile" 'string "strmObjMapFile")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Stream Out](#)

[Sample Summary File](#)

enableColoring

```
xstream.fastExport enableColoring boolean { t | nil }
```

Description

Specifies the value for the `enableColoring` option when Stream files are exported from *Layout XL – File – Export Stream from VM*.

The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("xstream.fastExport" "enableColoring")  
envSetVal("xstream.fastExport" "enableColoring" 'boolean t)  
envSetVal("xstream.fastExport" "enableColoring" 'boolean nil)
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Stream Out](#)

strmFilenameSuffix

```
xstream.fastExport strmFilenameSuffix string  
    "Suffix_Added_to_Exported_VM_Files"
```

Description

Specifies the suffix to be added to the names of the Stream files exported using the *Layout XL — File — Export Stream from VM* command.

The default is " ".



This environment variable replaces the `xstream` environment variable `xstExportFromVmFilenameSuffix`. Support for the `xstExportFromVmFilenameSuffix` environment variable will be removed in a future release. Instead use the `xstream.fastExport` environment variable `strmFilenameSuffix`.

GUI Equivalent

None

Examples

```
envGetVal("xstream.fastExport" "strmFilenameSuffix")  
envSetVal("xstream.fastExport" "strmFilenameSuffix" 'string "_VM")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Stream Out](#)

strmFilenamePrefix

```
xstream.fastExport strmFilenamePrefix string  
    "Prefix_Added_to_Exported_VM_Files"
```

Description

Specifies the prefix to be added to the names of the Stream files exported using the *Layout XL – File – Export Stream from VM* command.

The default is " ".



This environment variable replaces the `xstream` environment variable `xstExportFromVmFilenamePrefix`. Support for the `xstExportFromVmFilenamePrefix` environment variable will be removed in a future release. Instead use the `xstream.fastExport` environment variable `strmFilenamePrefix`.

GUI Equivalent

None

Examples

```
envGetVal("xstream.fastExport" "strmFilenamePrefix")  
envSetVal("xstream.fastExport" "strmFilenamePrefix" 'string "VM_")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Stream Out](#)

XStream Shell Environment Variables

XST_ALL_VIEWS

Description

Shows all the views currently listed for the selected cell in the library, cell, and view browser.

Example

```
setenv XST_ALL_VIEWS ""
```

XST_AUTO_LOAD_TEMPLATE_NAME

See [XST_AUTO_LOAD_TEMPLATE_NAME](#).

Example

```
setenv XST_AUTO_LOAD_TEMPLATE_NAME ""
```

XST_IN

See [XST_IN](#).

Example

```
setenv XST_IN /home/user1/strmIn.tl
```

XST_IN_VM_NT

Description

Specifies the number of threads to be used in Virtual Memory mode.

Example

For single thread use:

```
setenv XST_IN_VM_NT 1
```

XST_OUT

See XST_OUT.

Example

```
setenv XST_OUT /home/user1/strmIn.tl
```

XST_OUT_CC_CHOP

Description

Enables the alternate any-angle fracturing algorithm for any-angle non-CurvyCore polygons. This algorithm, fractures shapes by creating new edges between existing vertices and breaking the shape along the new edge.

By default, the classical fracturing algorithm is used for any-angle non-CurvyCore polygons. In the classic fracturing algorithm, a shape is chopped in half along its x or y axis until all the chopped shapes have fewer than the maximum number of vertices.

The maximum number of vertices is 200 by default and can be specified by user using the `-maxVertices` option.

The XStream Out translator uses:

- The classic fracturing algorithm:
 - ❑ For polygons with 45 or 90 degree angle edge.
 - ❑ For all any-angle non-CurvyCore polygons.
- The alternate any-angle fracturing algorithm:
 - ❑ For any-angle CurvyCore polygons. This is enabled by default.
 - ❑ For all any-angle non-CurvyCore polygons when the SHELL environment variable `XST_OUT_CC_CHOP` is set.

No special Photonics license is needed to set this variable.

Example

```
setenv XST_OUT_CC_CHOP
```

Related Topics

[ccGetFromFig](#)

[Geometry: Limit Options \(Max Vertices\)](#)

xoasEnableGUIHistory

```
xoasis xoasEnableGUIHistory boolean { t | nil }
```

Description

Saves the GUI history that lets you use the last preserved GUI history at command line. This environment variable also helps in retaining the option settings used during the last successful translation in the next Virtuoso session.

The default is `nil`, which means that the GUI history is not saved.

GUI Equivalent

None

Examples

```
envGetVal("xoasis" "xoasEnableGUIHistory")
envSetVal("xoasis" "xoasEnableGUIHistory" 'boolean t)
envSetVal("xoasis" "xoasEnableGUIHistory" 'boolean nil)
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[xoasEnableGUIHistory](#)

[Preserving GUI History in XOasis](#)

XOASIS_AUTO_LOAD_TEMPLATE_NAME

```
xoasis XOASIS_AUTO_LOAD_TEMPLATE_NAME string "template_file_name"
```

Description

Loads the specified template file.

The default is " " that loads the default template file.

Setting Default Values in XOasis

You can set the default values of XOasis In and XOasis Out options by using the XOASIS_IN and XOASIS_OUT environment variables along with XOASIS_AUTO_LOAD_TEMPLATE_NAME. The following table summarizes the various possibilities and the behavior.

Template File Specified in the XOASIS_IN/XOASIS_OUT Environment Variables	Default values specified in the XOASIS_AUTO_LOAD_TEMPLATE_NAME Environment Variable	DF II	Command Line
Yes	default		In the first invocation, loads the template file pointed by the XOASIS_IN/XOASIS_OUT environment variable. In subsequent invocations, loads the values from its previous session.
No	default		In the first invocation, loads the GUI with empty fields because no template file is specified in the XOASIS_IN/XOASIS_OUT environment variable. In subsequent invocations, it loads the values from its previous session.

Design Data Translators Reference

Design Data Translators Environment Variables

Template File Specified in the XOASIS_IN/ XOASIS_OUT Environment Variables	Default values specified in the XOASIS_AUTO_LOAD_TEMPLATE_NAME Environment Variable	DF II	Command Line
Yes	" "		Loads the template file pointed by the XOASIS_IN/ XOASIS_OUT environment variables.
No	" "		Loads the GUI with empty fields because no template file is specified in the XOASIS_IN/ XOASIS_OUT environment variable and it does not retain the values of the previous session.

GUI Equivalent

None

Examples

```
envGetVal("xoasis" "XOASIS_AUTO_LOAD_TEMPLATE_NAME")
envSetVal("xoasis" "XOASIS_AUTO_LOAD_TEMPLATE_NAME" 'string "/home/user1/
oasIn.tl")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

XOASIS_IN

```
xoasis XOASIS_IN string "template_file_path"
```

Description

Specifies the path for the template file to be used during XOasis In translation.

The default is "".

The `XOASIS_IN` environment variable sets the default values of the corresponding options as well as the template files. A template file consists of the values of the XOasis In option. Therefore, each time you run XOasis In, the values for the options are taken from the template files and the path of the template file is taken from the `XOASIS_IN` environment variables.

You can use the `XOASIS_IN` environment variable while running XOasis from GUI. For example, set `XOASIS_IN` and invoke XOasis In by using the following commands:

```
setenv XOASIS_IN /home/user1/templateIn.txt
```

XOasis In is invoked in interactive mode and the values of XOasis In options are taken from the `templateIn.txt` template file specified in the `XOASIS_IN` environment variable.

You can set `XOASIS_IN` in your `.cdsenv` file by using the following command:

```
xoasis XOASIS_IN string "/home/user1/xoasisIn.tl"
```

By default, XOasis retains the values of various XOasis In options from its previous session. Let us consider a scenario where the GUI is invoked. If you modify the value in the `General` tab of the XOasis In Options form to 6 and click the `OK` button, in the next session value, 6 will be displayed. However, in the session after that, if you modify the value to 3 and click the `Cancel` button, then the `General` tab of the XOasis In Options form will retain the previous value, 6.

If you run XOasis in the GUI mode, then the GUI form is populated with the values in the template file specified using these variables at startup. You can override these values by specifying different values explicitly.

Additionally, in case of XOasis in the GUI mode, CDS environment variables take precedence over the UNIX environment variables.

For more information, see [Setting Default Values in XOasis](#).

GUI Equivalent

None

Design Data Translators Reference

Design Data Translators Environment Variables

Examples

```
envGetVal("xoasis" "XOASIS_IN")  
envSetVal("xoasis" "XOASIS_IN" 'string "/home/user1/templateIn.txt")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

YOASIS_OUT

```
yoasis YOASIS_OUT string "template_file_path"
```

Description

Specifies the path for the template file to be used during YOasis Out translation.

The default is "".

The YOASIS_OUT environment variable sets the default values of the corresponding options as well as the template files. A template file consists of the values of the YOasis Out option. Therefore, each time you run YOasis Out, the values for the options are taken from the template files and the path of the template file is taken from the YOASIS_OUT environment variables.

You can use the YOASIS_OUT environment variable while running YOasis from GUI. For example, set YOASIS_OUT and invoke YOasis Out by using the following commands:

```
setenv YOASIS_OUT /home/user1/templateOut.txt
```

YOasis Out is invoked in interactive mode and the values of YOasis Out options are taken from the `templateOut.txt` template file specified in the YOASIS_OUT environment variable.

You can set YOASIS_OUT in your `.cdsenv` file by using the following command:

```
yoasis YOASIS_OUT string "/home/user1/yoasisOut.tl"
```

By default, YOasis retains the values of various YOasis Out options from its previous session. Let us consider a scenario where the GUI is invoked. If you modify the value in the **General** tab of the YOasis Out Options form to 6 and click the **OK** button, in the next session value, 6 will be displayed. However, in the session after that, if you modify the value to 3 and click the **Cancel** button, then the **General** tab of the YOasis Out Options form will retain the previous value, 6.

If you run YOasis in the GUI mode, then the GUI form is populated with the values in the template file specified using these variables at startup. You can override these values by specifying different values explicitly.

Additionally, in case of YOasis in the GUI mode, CDS environment variables take precedence over the UNIX environment variables.

For more information, see [Setting Default Values in YOasis](#).

Design Data Translators Reference

Design Data Translators Environment Variables

GUI Equivalent

None

Examples

```
envGetVal("xoasis" "XOASIS_OUT")  
envSetVal("xoasis" "XOASIS_OUT" 'string "/home/user1/strmIn.tl")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

xoasAlwaysLookupMapFilesInTech

```
xoasis xoasAlwaysLookupMapFilesInTech boolean { t | nil }
```

Description

Specifies whether the XOasis automatically opens the technology library including ITDB hierarchy to search for map files in the technology libraries. If the technology library is already open, setting this variable has no effect.

Fields for map files, if empty, are always automatically populated when technology library is already open in the current Virtuoso session. However, if the technology library is not open in the current Virtuoso session then XOasis, by default, will open the technology library including the ITDB hierarchy to search for the map files ('<techlibname>.layermap' for layer map file, '<techlibname>.objectmap' for object map file, and '<techlibname>.labelmap' for label map file). Opening the technology library can take some time. Default behavior of opening the technology library including ITDB can be disabled by setting the value of this environment variable to `nil`.

Note: This setting only prevents auto population of layer map files in GUI. Translation will still traverse through all the technology libraries to search for layer map files.

The default is `t`.

GUI Equivalent

None

Examples

```
envGetVal("xoasis" "xoasAlwaysLookupMapFilesInTech")
envSetVal("xoasis" "xoasAlwaysLookupMapFilesInTech" 'boolean t)
envSetVal("xoasis" "xoasAlwaysLookupMapFilesInTech" 'boolean nil)
```

Related Topics

XOasis In Main form – [Layer Map](#)

XOasis In Main form – [Object Map](#)

XOasis Out Main form – [Layer Map](#)

XOasis Out Main form – [Object Map](#)

xoasOutAutoPopulateIfEmptyLCV

```
xoasis xoasOutAutoPopulateIfEmptyLCV boolean { t | nil }
```

Description

Specifies whether the *OASIS File*, *Library*, *TopCell(s)*, and *View(s)* fields on the XOasis Out main form should be filled automatically only when they are empty. If these fields already have some information and if the environment variable is set to `t`, this information is not replaced even if a cellview is open.

The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("xoasis" "xoasOutAutoPopulateIfEmptyLCV")  
envSetVal("xoasis" "xoasOutAutoPopulateIfEmptyLCV" 'boolean t)  
envSetVal("xoasis" "xoasOutAutoPopulateIfEmptyLCV" 'boolean nil)
```

Related Topics

[List of Design Data Translators Environment Variables](#)

xoasInOverwriteLibPopup

```
xoasis xoasInOverwriteLibPopup boolean { t | nil }
```

Description

Enables a warning message that is displayed during the XOasis In translation to an existing library to warn you that cells in the target library will be overwritten or appended to in this process.

This message is displayed when Write Mode is set to *overwrite* or *append*.

The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("xoasis" "xoasInOverwriteLibPopup")  
envSetVal("xoasis" "xoasInOverwriteLibPopup" 'boolean t)  
envSetVal("xoasis" "xoasInOverwriteLibPopup" 'boolean nil)
```

Related Topics

[List of Design Data Translators Environment Variables](#)

xoasShowCompletionMsgBox

```
xoasis xoasShowCompletionMsgBox boolean { t | nil }
```

Description

Specifies whether the *Oasis in translation complete* message box should display the number of errors and warnings after the translation is complete.

The default is `t`.

GUI Equivalent

GUI Equivalent

Command	<i>Virtuoso XOasis Out</i>
Field	<u>Show Completion Message Box</u>

Examples

```
envGetVal("xoasis" "xoasShowCompletionMsgBox")
envSetVal("xoasis" "xoasShowCompletionMsgBox" 'boolean t)
envSetVal("xoasis" "xoasShowCompletionMsgBox" 'boolean nil)
```

Related Topics

List of Design Data Translators Environment Variables

oasisFileExtension

```
xoasis oasisFileExtension string "Extension_Added_to_OASIS_Files"
```

Description

Specifies the file extension to be added to the output top cell file name when the XOasis Out translation is done using the *Layout XL — File — Export Oasis from VM option*.

The default is `oasis`.



This environment variable replaces `xoasOutExt`. Support for the `xoasOutExt` environment variable will be removed in a future release. Instead use `oasisFileExtension`.

GUI Equivalent

None

Examples

```
envGetVal("xoasis.fastExport" "oasisFileExtension")  
envSetVal("xoasis.fastExport" "oasisFileExtension" 'string "of")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Oasis Out](#)

enableColoring

```
xoasis.fastExport enableColoring boolean { t | nil }
```

Description

Specifies the value for the `enableColoring` option when OASIS files are exported from *Layout XL – File – Export Oasis from VM*.

The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("xoasis.fastExport" "enableColoring")  
envSetVal("xoasis.fastExport" "enableColoring" 'boolean t)  
envSetVal("xoasis.fastExport" "enableColoring" 'boolean nil)
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Oasis Out](#)

oasisFilenamePrefix

```
xoasis.fastExport oasisFilenamePrefix string  
    "Prefix_Added_to_Exported_VM_Files"
```

Description

Specifies the prefix to be added to the names of OASIS files exported from *Layout XL — File — Export Oasis from VM*.

The default is " ".

GUI Equivalent

None

Examples

```
envGetVal("xoasis.fastExport" "oasisFilenamePrefix")  
envSetVal("xoasis.fastExport" "oasisFilenamePrefix" 'string "VM_")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Oasis Out](#)

oasisFilenameSuffix

```
xoasis.fastExport oasisFilenameSuffix string  
    "Suffix_Added_to_Exported_VM_Files"
```

Description

Specifies the suffix to be added to the names of the OASIS files exported from *Layout XL — File — Export Oasis from VM*.

The default is " ".

GUI Equivalent

None

Examples

```
envGetVal("xoasis.fastExport" "oasisFilenameSuffix")  
envSetVal("xoasis.fastExport" "oasisFilenameSuffix" 'string "_VM")
```

Related Topics

[List of Design Data Translators Environment Variables](#)

[Fast Oasis Out](#)

XOasis Shell Environment Variables

XOASIS_ALL_VIEWS

Description

Shows all views currently listed in a cellview of the selected cell in the library, cell, and view browser.

Example

```
setenv XOASIS_ALL_VIEWS ""
```

XOASIS_AUTO_LOAD_TEMPLATE_NAME

See XST AUTO LOAD TEMPLATE NAME.

Example

```
setenv XOASIS_AUTO_LOAD_TEMPLATE_NAME ""
```

XOASIS_IN

See XOASIS_IN.

Example

```
setenv XOASIS_IN /home/user1/oasIn.tl
```

XOASIS_IN_VM_NT

Description

Specifies the number of threads to be used in Virtual Memory mode.

Example

For single thread use:

```
setenv XOASIS_IN_VM_NT
```

XOASIS_OUT

See [XOASIS_OUT](#).

Example

```
setenv XOASIS_OUT /home/user1/oasmIn.tl
```