

SIEMENS EDA

Calibre® Layout Comparison and Translation Guide

DBdiff, FastXOR, fdi2gds,
fdi2oasis, and fdiBA

Software Version 2021.2
Document Revision 14

Unpublished work. © 2021 Siemens

This material contains trade secrets or otherwise confidential information owned by Siemens Industry Software, Inc., its subsidiaries or its affiliates (collectively, "Siemens"), or its licensors. Access to and use of this information is strictly limited as set forth in Customer's applicable agreement with Siemens. This material may not be copied, distributed, or otherwise disclosed outside of Customer's facilities without the express written permission of Siemens, and may not be used in any way not expressly authorized by Siemens.

This document is for information and instruction purposes. Siemens reserves the right to make changes in specifications and other information contained in this publication without prior notice, and the reader should, in all cases, consult Siemens to determine whether any changes have been made. Siemens disclaims all warranties with respect to this document including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement of intellectual property.

The terms and conditions governing the sale and licensing of Siemens products are set forth in written agreements between Siemens and its customers. Siemens' **End User License Agreement** may be viewed at: www.plm.automation.siemens.com/global/en/legal/online-terms/index.html.

No representation or other affirmation of fact contained in this publication shall be deemed to be a warranty or give rise to any liability of Siemens whatsoever.

TRADEMARKS: The trademarks, logos, and service marks ("Marks") used herein are the property of Siemens or other parties. No one is permitted to use these Marks without the prior written consent of Siemens or the owner of the Marks, as applicable. The use herein of third party Marks is not an attempt to indicate Siemens as a source of a product, but is intended to indicate a product from, or associated with, a particular third party. A list of Siemens' trademarks may be viewed at: www.plm.automation.siemens.com/global/en/legal/trademarks.html. The registered trademark Linux[®] is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a world-wide basis.

Support Center: support.sw.siemens.com

Send Feedback on Documentation: support.sw.siemens.com/doc_feedback_form

Revision History

Revision	Changes	Status/ Date
14	Modifications to improve the readability and comprehension of the content. Approved by Lucille Woo. All technical enhancements, changes, and fixes listed in the <i>Calibre Release Notes</i> for this products are reflected in this document. Approved by Michael Buehler.	Released April 2021
13	Modifications to improve the readability and comprehension of the content. Approved by Lucille Woo. All technical enhancements, changes, and fixes listed in the <i>Calibre Release Notes</i> for this products are reflected in this document. Approved by Michael Buehler.	Released January 2021
12	Modifications to improve the readability and comprehension of the content. Approved by Lucille Woo. All technical enhancements, changes, and fixes listed in the <i>Calibre Release Notes</i> for this products are reflected in this document. Approved by Michael Buehler.	Released October 2020
11	Modifications to improve the readability and comprehension of the content. Approved by Lucille Woo. All technical enhancements, changes, and fixes listed in the <i>Calibre Release Notes</i> for this products are reflected in this document. Approved by Michael Buehler.	Released July 2020

Author: In-house procedures and working practices require multiple authors for documents. All associated authors for each topic within this document are tracked within the Siemens EDA documentation source. For specific topic authors, contact the Siemens Digital Industries Software documentation department.

Revision History: Released documents maintain a revision history of up to four revisions. For earlier revision history, refer to earlier releases of documentation which are available on <https://support.sw.siemens.com/>.

Table of Contents

Revision History

Chapter 1

Overview	13
Database Comparison	13
Database Translation	15
Database Backannotation	16
Syntax Conventions	17

Chapter 2

Standard Dual Database XOR Comparison	19
Using Calibre nmDRC-H to Perform LVL	20
Layer Bump	21
Special Semantics for Hierarchical Applications	23

Chapter 3

FastXOR Database Comparison	25
Getting Started with FastXOR	26
Functional Overview	26
Differences Between FastXOR and Standard Dual Database Processing	27
Generating the LVL Comparison Rule File	28
Using FastXOR to Perform LVL	29
Comparing FastXOR and Calibre nmDRC-H Run Results	31
Improving FastXOR Performance	33
FastXOR Command Line Syntax	34
XOR Rules Generated by DBdiff	34
FastXOR Layer Mapping and Comparison	37
Customized Layer Mapping	39
Layer Mapping Based Upon Originating Design	39
Layer Mapping with Include and Exclude Areas	40
Excluded Cells in FastXOR	41
NOT Operations in FastXOR	41
FastXOR Run Transcript Features	42
DBdiff Command Execution in FastXOR	43
LVL Run Configuration	46
Restrictions	47
Troubleshooting FastXOR Runs	50
Problem: Some Items in Third-Party Databases Are Not Compared	51
Problem: Corresponding Layers Not Being Compared	52
Problem: How Not to Compare All Layers in Both Designs	52
Problem: Results from FastXOR Not Matching Usual XOR Run	53
Problem: DBdiff Reports Designs as Different But FastXOR Finds No Differences	54

Problem: Layout Input Exceptions Not Matching Usual XOR Run	58
Problem: Differences in Handling of Non-Orthogonal Shapes	58
Problem: Verification of OpenAccess Database is Failing	59
Problem: Verification of LEF/DEF Databases is Failing	59
FastXOR Environment Variables	60
Chapter 4	
DBdiff Database Comparison	63
Getting Started with DBdiff.	64
Input and Output	64
Using DBdiff to Perform LVL	65
Comparing Text with DBdiff	67
DBdiff Usage Examples	68
Layout Comparison Command Line Syntax	79
Write XOR Rules Command Line Syntax	98
Summary of Differences in the ASCII Report File	105
OASIS Format Restrictions	107
LEF/DEF Design Considerations	107
OpenAccess Layer Mapping	109
Chapter 5	
Database Translation Utilities	111
Third-Party Database Support	111
License Requirements	114
FDI Environment Variables	114
GDS to OASIS Translation	119
Getting Started with fdi2gds and fdi2oasis	121
Create and Use a LEF/DEF Mapping File	121
Translating LEF/DEF to GDS or OASIS	122
Translating a LEF Library Using Directories and Outputting to Standard Out	123
Translating LEF/DEF Properties	124
Translating OpenAccess to OASIS or GDS	125
Summary Tables	126
Command Line Reference	130
LEF/DEF FDI Usage	130
OpenAccess FDI Usage	131
Complete FDI Command Line Syntax	133
FDI File Reference	155
LEF/DEF Mapping	156
LEF/DEF Map File Format	157
LEF/DEF Text Mapping	163
Rows and Sites	164
Example: LEF/DEF Map File	165
OA Mapping	167
Place and Route Layer Mapping	167
OpenAccess Layer Mapping File	169
Layer Map File for fdi2gds and fdi2oasis	172
Object Map File	176

Table of Contents

Cell Mapping File	179
Layer Names File	181
Mapping Information File	182
Net Properties File	184
Nets File	185
Property Map File	187
Input Exception File	188
Template File	190
 Chapter 6	
Via and Fill Backannotation	191
Directly Backannotating to DEF Using SVRF	191
Getting Started with fdiBA	195
Backannotating Fill and OPC Fill to GDS or OASIS	195
Backannotating Line End Extensions and Vias to DEF	197
fdiBA Examples	197
fdiBA Command Line Syntax	199
Mapping Files for Backannotation	204
Layer Map File for fdiBA	205
Layer Map File for DFM Databases in fdiBA	209
 Appendix A	
Calibre Utility Messages	213
FDI Messages	213
DBdiff Messages	237
FastXOR Messages	256
 Index	
 Third-Party Information	


List of Figures

Figure 1-1. fdi2gds and fdi2oasis Flow	16
Figure 3-1. Standard XOR	27
Figure 3-2. FastXOR	27
Figure 3-3. FastXOR Does Not Report Duplicate Shapes On The Same Layer	54
Figure 3-4. FastXOR Does Not Report Enclosed Shapes On The Same Layer	55
Figure 3-5. FastXOR Does Not Report Hierarchy Of Same-Layer Shapes	56
Figure 4-1. Hierarchy Changes	75
Figure 4-2. Comparison with -compareallplacedcells	84
Figure 4-3. Layermap Format for GDS and OASIS	89
Figure 4-4. Layermap Format for Place and Route (OpenAccess)	90
Figure 4-5. Comparing Unmerged Shapes with DBdiff	95
Figure 4-6. -comparemergeddiffshape Behavior	96
Figure 5-1. CUTSIZE:y:x Dimensions	174
Figure 6-1. Workflow for Direct Backannotation to DEF	192
Figure 6-2. Backannotating Calibre DFM Enhancements to P&R with fdiBA	195
Figure 6-3. CUTSIZE:y:x Dimensions for -layermap in fdiBA	207

List of Tables

Table 1-1. Syntax Conventions	17
Table 3-1. Supported Layer Operations and Specification Statements	48
Table 3-2. FastXOR Environment Variables	60
Table 4-1. DBdiff Usage Examples	69
Table 5-1. Mentor Graphics Variables	113
Table 5-2. Cadence Variables	113
Table 5-3. FDI Environment Variables	115
Table 5-4. OA Paths to Polygons	147
Table 5-5. Extracted Net Types	150
Table 5-6. FDI File Summary	155
Table 5-7. Objects and Subtypes	158
Table 5-8. Default Object Mapping	160
Table 5-9. Default Text Mapping	163
Table 5-10. OpenAccess Default Layer Mapping	167
Table 5-11. LEF/DEF and OA object_type and subtype Values	176
Table A-1. FDI Error Messages	214
Table A-2. fdi2gds and fdi2oasis Warning Messages	215
Table A-3. fdi2gds and fdi2oasis Map File Messages	229
Table A-4. fdiBA Warning Messages	231
Table A-5. Calibre View Messages	232
Table A-6. Numbered DBdiff Warning Messages	237
Table A-7. Input GDS File Errors for DBdiff	241
Table A-8. Input OASIS File Errors for DBdiff	241
Table A-9. Input LEF/DEF File Errors for DBdiff	245
Table A-10. DBdiff Comparison Errors	245
Table A-11. DBdiff Invocation Errors	246
Table A-12. Miscellaneous Errors for DBdiff	252
Table A-13. DBdiff Warnings	252
Table A-14. FastXOR Error Messages	256
Table A-15. FastXOR Warning Messages	258

Layout comparison tools include FastXOR, DBdiff, and standard XOR with Calibre nmDRC. The fdi2gds and fdi2oasis tools translate LEF/DEF and OpenAccess (OA) databases to GDS or OASIS^{®1}. The fdiBA utility backannotates fill and via enhancements to a LEF/DEF or OA database.

<p>Try It!</p> 	<p>Calibre Layout Comparison Tutorial and Example Kit (eKit)</p> <p>The eKit demonstrates basic procedures for performing layout versus layout using standard XOR, FastXOR, and DBdiff, and includes all files and data needed to run the tool.</p> <p>Go to this page on Support Center to download the eKit (Documentation tab, Document Types=Getting Started Guide). The link goes to the latest release.</p>
---	--

Database Comparison 13

Database Translation..... 15

Database Backannotation 16

Syntax Conventions 17

Database Comparison

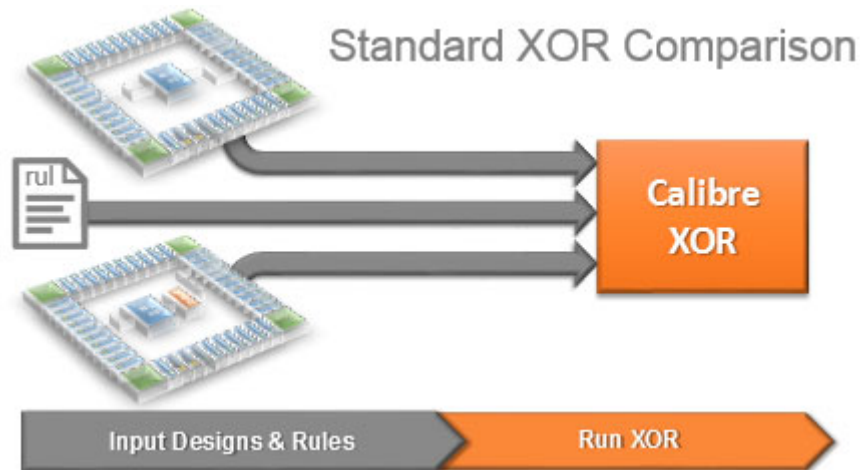
Layout Versus Layout (LVL) comparison comprises three fundamental flows: FastXOR, the standard dual-database flow, and DBdiff.

The three database comparison flows are described as follows:

- **Standard XOR** — Calibre dual-database flow where two databases are compared with no pre-processing. This flow uses Calibre nmDRC to perform the SVRF XOR operation

1. OASIS[®] is a registered Trademark of Thomas Grebinski and licensed for use to SEMI, San Jose, California.

on layers in your database. The entire layouts are involved in the XOR operations. An XOR rules file is required. This is sometimes referred to as traditional XOR.

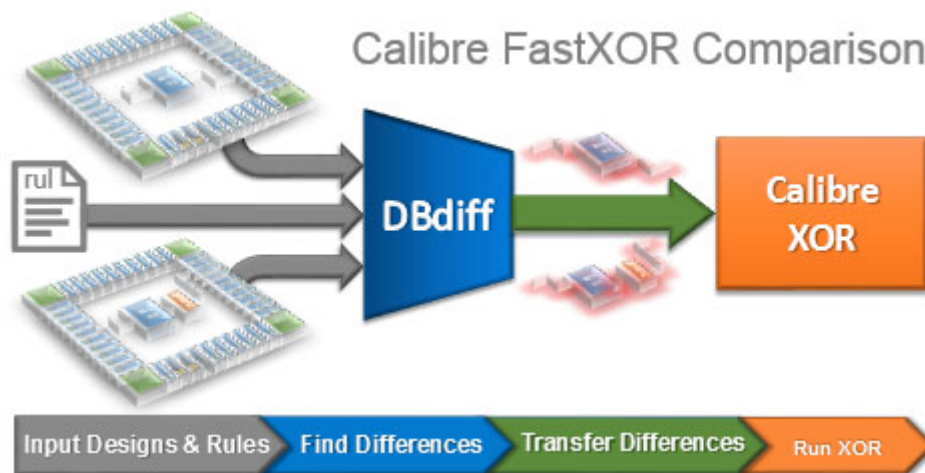


Tip

i See [“Differences Between FastXOR and Standard Dual Database Processing”](#) on page 27 for more details.

- **FastXOR** — FastXOR uses pre-processing by DBdiff to analyze the layout databases to determine the objects of the databases that are different. Then, Calibre nmDRC-H is used to perform an XOR on just the objects and layers that have been determined to be different by DBdiff.

The run times of FastXOR and Calibre nmDRC-H usually differ. In most cases, FastXOR is faster than Calibre nmDRC-H for the same runtime environment. However, Calibre nmDRC-H can be faster in designs where the runtime is short, or where the hierarchies are dissimilar. FastXOR is optimized for designs using similar hierarchies.



- **DBdiff** — The DBdiff application is a command-line driven Calibre tool that compares all hierarchy, shapes, properties, and text between two databases. This comparison is not

meant to be a replacement for the standard XOR operation, but instead provides designers a way to find the differences between two databases quickly. Trade-offs in accuracy and performance are discussed in “[DBdiff Database Comparison](#)” on page 63.

Invocation

Utility	Command-line	GUI
Standard XOR	<code>calibre -drc xor_rules_file</code>	Calibre DESIGNrev > Utilities > Layout DIFF Calibre Interactive nmDRC
FastXOR	<code>calibre -fx -drc xor_rules_file</code>	Calibre Interactive nmDRC > Inputs > FastXOR
DBdiff	<code>dbdiff ...</code>	N/A

Related Topics

[Standard Dual Database XOR Comparison](#)

[DBdiff Database Comparison](#)

[FastXOR Database Comparison](#)

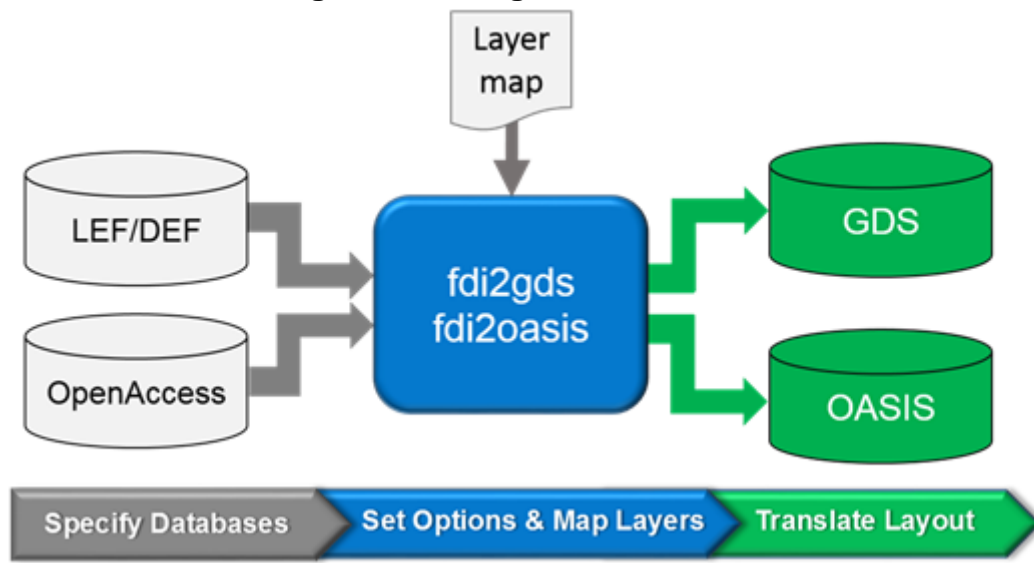
Database Translation

Calibre has two command line utilities that translate third-party databases to either GDSII or OASIS format. The utilities are called `fdi2gds` and `fdi2oasis`. The acronym FDI stands for Foreign Database Interface.

The Calibre FDI utilities are located in the `$CALIBRE_HOME/bin` directory.

The FDI utilities are also called by other Calibre processes if you specify third-party databases in your SVRF rule file. The utilities are also used to import databases in Calibre DESIGNrev.

Figure 1-1. fdi2gds and fdi2oasis Flow



Related Topics

[Database Translation Utilities](#)

Database Backannotation

Calibre includes direct DEF output capabilities in addition to a stand-alone utility to backannotate Calibre enhancements, vias, and fill to third-party databases.

Direct Backannotation

The Calibre DEF backannotation flow enables you to generate geometries (such as redundant vias, line extensions, and fill) for a LEF/DEF layout in Calibre and then export the new geometries to a full or incremental DEF file. This flow is controlled by the [DFM RDB DEF](#) statement.

Using this integrated flow, you can read in a LEF/DEF design using the direct read functionality in the Layout Path statement, generate enhancement shapes with DFM or other statements, and then export the enhancement shapes with the original DEF design, all within a single SVRF rule file.

See “[Directly Backannotating to DEF Using SVRF](#)” on page 191 for details.

fdiBA Utility

The fdiBA utility is located in the `$CALIBRE_HOME/bin` directory and can perform backannotation to OpenAccess and LEF/DEF systems. See “[Getting Started with fdiBA](#)” on page 195 for details.

Related Topics

[Via and Fill Backannotation](#)

Syntax Conventions

The command descriptions use font properties and several metacharacters to document the command syntax.

Table 1-1. Syntax Conventions

Convention	Description
Bold	Bold fonts indicate a required item.
<i>Italic</i>	Italic fonts indicate a user-supplied argument.
Monospace	Monospace fonts indicate a shell command, line of code, or URL. A bold monospace font identifies text you enter.
<u>Underline</u>	Underlining indicates either the default argument or the default value of an argument.
UPPercase	For certain case-insensitive commands, uppercase indicates the minimum keyword characters. In most cases, you may omit the lowercase letters and abbreviate the keyword.
[]	Brackets enclose optional arguments. Do not include the brackets when entering the command unless they are quoted.
{ }	Braces enclose arguments to show grouping. Do not include the braces when entering the command unless they are quoted.
‘ ’	Quotes enclose metacharacters that are to be entered literally. Do not include single quotes when entering braces or brackets in a command.
or	Vertical bars indicate a choice between items. Do not include the bars when entering the command.
...	Three dots (an ellipsis) follows an argument or group of arguments that may appear more than once. Do not include the ellipsis when entering the command.
Example: DEVICE { <i>element_name</i> [‘(‘ <i>model_name</i> ‘)’] } <i>device_layer</i> { <i>pin_layer</i> [‘(‘ <i>pin_name</i> ‘)’] ... } [‘<’ <i>auxiliary_layer</i> ‘>’ ...] [‘(‘ <i>swap_list</i> ‘)’ ...] [<u>BY NET</u> BY SHAPE]	

Related Topics

[Third-Party Database Support](#)


[License Requirements](#)

Chapter 2

Standard Dual Database XOR Comparison

A major application of the Calibre dual-database capability is layout-versus-layout (LVL), or dual-database comparison.

Note

 Dual database capability is not the same as using multiple filenames in the [Layout Path](#) or [Layout Path2](#) specification statements. Calibre applications treat these multiple files as a single input layout database (for example, there is only one top-level cell). In a dual database application, there are two distinct layout hierarchies.

In dual database applications, the following specification statements read in the first database:

- [Layout System](#)
- [Layout Path](#)
- [Layout Primary](#)

Similarly, the following statements read in the second database:

- [Layout System2](#)
- [Layout Path2](#)
- [Layout Primary2](#)

The [Layout Bump2](#) specification statement is used in dual-database applications to manage layers having identical numbers in both layouts. This statement is discussed in detail in the next section.

Only the GDSII, OASIS, LEF/DEF, and OpenAccess formats are supported. You can compare databases of differing formats if you choose. In that case, an Calibre nmDRC-H license is required.

If you are comparing an OpenAccess database to a database of a different format, or another OpenAccess database that uses a different layer scheme, you need a layer mapping file. This is discussed under “[Treatment of Third-Party Layout Databases](#)” in the *SVRF Manual*. See the -layerMap option under the discussion of the MGC_CALIBRE_DB_READ_OPTIONS variable. The layer mapping file may also be specified by setting the MGC_CALIBRE_LAYER_MAP environment variable. See “[FDI Environment Variables](#)” on page 114 for details.

You can specify Layout Path and Layout Path2 any number of times. When you specify multiple layout paths using either of these statements, this results in a concatenation of layouts for the statement in which multiple layouts are specified. Two distinct design hierarchies are maintained: one for Layout Path designs and the other for Layout Path2 designs.

All other layout database specification statements such as [Layer](#), [Layer Map](#), [Exclude Cell](#), [Layout Rename Cell](#), [Layout Text](#), and so forth apply equally to both databases in a dual-database application.

Note



DFM property comparison is unsupported in dual-database comparison mode.

Other methods of performing layout comparison are discussed under “[FastXOR Database Comparison](#)” on page 25 and “[DBdiff Database Comparison](#)” on page 63.

Using Calibre nmDRC-H to Perform LVL	20
Layer Bump	21
Special Semantics for Hierarchical Applications	23

Using Calibre nmDRC-H to Perform LVL

Use Calibre nmDRC-H for performing LVL comparison.

This type of run can be compared to a FastXOR run to see which gets better performance. Calibre nmDRC-H often has better performance than FastXOR when design hierarchies are substantially different.

See “[Standard Dual Database XOR Comparison](#)” on page 19.

Prerequisites

- An XOR comparison rule file. See “[Generating the LVL Comparison Rule File](#).” You can use an XOR rule file not generated with DBdiff; however, if you plan to compare the normal XOR run results with the FastXOR results, then you should use a rule file generated by DBdiff that was also used for the FastXOR run.
- The rule file should handle layer mapping correctly so the appropriate layers are compared.
- Two complete layout databases to be compared.
- If an OpenAccess database is being used as input, see “[Third-Party Database Support](#)” on page 111 for further instructions.
- Sufficient Calibre nmDRC/Calibre nmDRC-H license pairs for the number of CPUs in your run.

Procedure

1. If you are planning to compare the results of this run to a FastXOR run, you should use different working directories for the two runs. This prevents overwriting of files.
2. Execute the following commands:

```
calibre -drc -hier -turbo rules.xor >& xor.log &  
tail -f xor.log
```

3. Check the [DRC Summary Report](#) file for warnings and results statistics. This line in the report tells you the total number of hierarchical results and an estimate of the flat result count.

```
TOTAL DRC Results Generated: 58698 (58700)
```

Results

If you used DBdiff to generate your rule file, the run produces the following by default:

- DRC Summary Report file: *rule_file.summary*
- ASCII [DRC Results Database](#): *rule_file.asc*
- Geometric output database: *rule_file.gds*
- Run transcript

The names depend upon what is specified in the rule file.

You can open your reference layout design in your layout editor and the ASCII DRC Results Database file in Calibre RVE to check any XOR differences.

Layer Bump

For a dual-database model to have meaningful applications, primitive objects (shapes and text) must be distinguishable between the two databases. The problem that arises is when identical layer numbers are used in both databases.

Calibre nmDRC addresses this by incrementing the primitive layer number (before applying any [Layer Map](#) specification statements) of all objects in the second database by a constant value. You specify this value in the [Layout Bump2](#) specification statement. Each object in the second database of a dual-database application behaves exactly as if its primitive layer number increases by the specified layer bump value.

Layer and Layer Map specification statements apply to each of the two layout databases equally in a dual-database model. Thus, you must be careful when constructing original layer definitions in a rule file and any layer mappings in a dual database application.

In the following example, assume you want to compare GDSII databases A and B, each contains layer 2 (diff) and layer 45 (metal1). You do not want diff from both databases to be

read in on layer 2 or you cannot compare the diff layers from each database. Similarly, you do not want metall from both databases read in on layer 45.

To handle the assignment of distinct layer numbers for the second database, the Layout Bump2 statement is used:

Example 2-1. Dual-database (LVL) Comparison Rule File

```
LAYOUT SYSTEM GDSII
LAYOUT PATH a.gds
LAYOUT PRIMARY ATOP

LAYOUT SYSTEM2 GDSII
LAYOUT PATH2 b.gds
LAYOUT PRIMARY2 BTOP      // Could be ATOP also.

LAYOUT BUMP2 100          // increment the layer numbers for the
                          // second database by 100

LAYOUT USE DATABASE PRECISION YES

LAYER diff 2              // first database gets these layer numbers
LAYER metall 45

LAYER diff_2 102          // second database gets these layer numbers
LAYER metall_2 145

A { diff XOR ( SIZE diff_2 BY 0.01 ) }
B { metall NOT metall_2 }
C { metall_2 NOT metall }
```

The choice of 100 as a Layout Bump2 value is arbitrary. The value needs to be larger than 45, which is the highest referenced simple layer number from the first database. (A simple layer number refers to the primitive layer number of an object after applying any Layer Map specification statements.)

If you choose a layer bump value of 20, then an object on layer 25 in the second database would become layer 45, and would appear on the original metall layer. This is because [Layer](#) specification statements apply equally to each of the two databases.

For dual database applications, Calibre nmDRC ignores all objects in the first database whose primitive layer number is greater than or equal to the value of Layout Bump2. In the previous example, objects on layer 102 in the first database, if this layer exists, are placed on the original layer diff_2.

Construction of original layer definitions with Layer Map specification statements in a dual-database application is more complicated. You need to recall the exact definition of how a layer map works and choose the layer bump value that is greater than the highest simple (not primitive) layer number from the first database. The layer bump value also applies to text objects, including those defined in [Layout Text](#) specification statements.

Special Semantics for Hierarchical Applications

For flat dual-database applications, the layer bump value from the Layout Bump2 specification statement controls all semantics for combining the two input layout databases. The internal combination process is more complicated in hierarchical applications because they preserve the input hierarchy instead of flattening it.

Calibre nmDRC-H does this as follows:

1. Rename all cells in database one internally (such as A -> A\$1\$).
2. Rename all cells in database two internally (such as A -> A\$2\$), as in Step 1.
3. Create a new top-level cell and instantiate the top-level cells T1\$1 and T2\$2\$ from databases one and two with identity transforms. The name of the new top-level cell is that of the top-level cell from database one. (This means that text objects are no longer at the primary level. Adjustment of the Text Depth statement may be necessary for certain flows.)
4. Attempt merging at critical points in the construction of the hierarchical database. Create a new cell A, if it has not been done already, such that whenever two placements A\$1\$ and A\$2\$ are in an exact overlap situation (equal extents, transforms, and array characteristics), all objects from A\$1\$ and A\$2\$ are placed into cell A, and replace the two placements of A\$1\$ and A\$2\$ with a placement of cell A that has the same transform and array characteristics.
5. Retain internal names of cell templates that were not merged in all placements and keep them as part of the hierarchy. If the top-level cell from database one was merged, then the merged cell retains an internal name to avoid naming conflicts with the new top-level cell.

Processes that require cell names, such as Inside Cell operations and hcell detection, always use the original cell name (never the internally-generated name). Also, by using the Layout Rename Cell specification statement, you can force the merging of differently-named, or in some cases, similarly-named cells from the two databases.

If the hierarchies of the two databases are not drastically different, the automatic hierarchical database construction processes of dense overlap removal and hierarchical injection make them as similar as possible. This prevents an adverse impact on performance. If the hierarchies are drastically different, you should consider running the application flat.

Chapter 3

FastXOR Database Comparison

Calibre FastXOR uses the DBdiff utility to decrease Layout Versus Layout (LVL) XOR comparison run times. FastXOR often runs five times faster than a normal dual-database XOR run.


You can also start a FastXOR run from Calibre Interactive for DRC, as explained in the section [“Performing a FastXOR Layout Comparison Run in Calibre Interactive”](#) in the *Calibre Interactive User’s Manual*.

Getting Started with FastXOR	26
FastXOR Command Line Syntax	34
XOR Rules Generated by DBdiff	34
FastXOR Layer Mapping and Comparison	37
Excluded Cells in FastXOR	41
NOT Operations in FastXOR	41
FastXOR Run Transcript Features	42
DBdiff Command Execution in FastXOR	43
LVL Run Configuration	46
Restrictions	47
Troubleshooting FastXOR Runs	50
FastXOR Environment Variables	60

Getting Started with FastXOR

You can run FastXOR from the command-line or from the Calibre Interactive nmDRC GUI.

See “[FastXOR Layout Comparison in Calibre Interactive](#)” in the *Calibre Interactive User’s Manual* for details on FastXOR in Calibre Interactive.

Try It! 	Calibre Layout Comparison Tutorial and Example Kit (eKit) The eKit demonstrates basic procedures for performing layout versus layout using standard XOR, FastXOR, and DBdiff, and includes all files and data needed to run the tool. Go to this page on Support Center to download the eKit (Documentation tab, Document Types=Getting Started Guide). The link goes to the latest release.
---	---

Functional Overview	26
Differences Between FastXOR and Standard Dual Database Processing	27
Generating the LVL Comparison Rule File	28
Using FastXOR to Perform LVL	29
Comparing FastXOR and Calibre nmDRC-H Run Results	31
Improving FastXOR Performance.....	33

Functional Overview

FastXOR performs layout comparison as a two stage process. The first stage uses the DBdiff utility to perform cell-by-cell object comparison in order to exclude any unchanged data from further processing. DBdiff then submits data that has been changed to the second stage. In the second stage, Calibre nmDRC-H performs an XOR comparison of the layers that have differences.

FastXOR achieves optimum performance when the two input design hierarchies are similar and polygon differences do not create significant processing overhead.

The DBdiff utility can also be used to generate a set of XOR rules based on the two input designs. The XOR rule file can be used for FastXOR. See “[Generating the LVL Comparison Rule File](#)” on page 28.

“[DBdiff Database Comparison](#)” on page 63 includes complete details on the DBdiff utility.

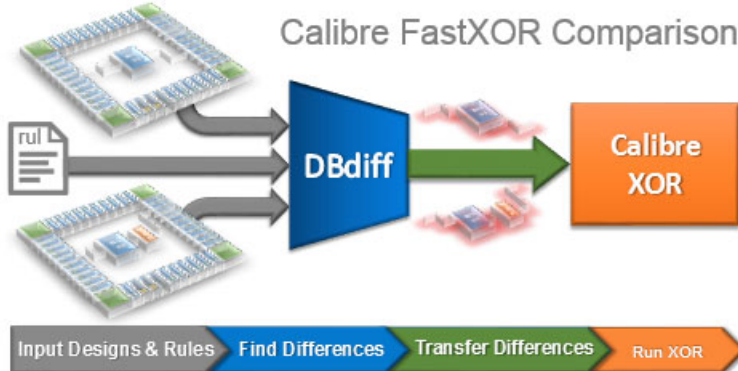
Differences Between FastXOR and Standard Dual Database Processing

There are several functional differences between FastXOR, DBdiff, and standard XOR layout comparison.

Figure 3-1. Standard XOR



Figure 3-2. FastXOR



The following are the differences you will see when comparing FastXOR processing versus the standard Calibre nmDRC XOR processing of two databases:

- FastXOR (`calibre -fx`) reduces hardware requirements. A FastXOR job running on four CPUs typically runs in less time than a traditional DRC job on 16 CPUs. You may see similar performance gains.
- In a FastXOR run, the layouts are automatically pre-processed by the DBdiff utility to determine the layers and regions of differences between the layouts. In a traditional DRC run using XOR operations, no such pre-processing occurs.
- In a FastXOR run, if DBdiff establishes that a cell is the same in both input designs, then some layers may not be presented to the DRC executive process. Any operations in the rule file that might depend upon such layers will not see those layers because the layers are not being submitted for XOR comparison. This can lead to unexpected results for

layer operations that expect non-empty input from layers that have been filtered out by DBdiff.

- The Calibre transcript statistics of the number of shapes read during the layout input step during a calibre -fx run will be significantly different when compared to the data present in the original layout files. This is because of the internal DBdiff processing that filters out areas of the designs that are not different.
- The number of results generated in calibre -fx mode may not match the number of results generated in the normal -drc mode, but a comparison of the overall XOR results of the two methods will not show geometric differences. This happens because the data and hierarchy seen in calibre -fx mode is different from the data and hierarchy seen in the normal -drc mode. See “[Comparing FastXOR and Calibre nmDRC-H Run Results](#)” on page 31.
- A comparison of differences between output of calibre -fx and a normal -drc run may not be the same when there are non-orthogonal shapes in the designs and there are differences between the designs in the non-orthogonal shapes. There may be 1 dbu differences in where edges are snapped between FastXOR and normal -drc runs. This can be ignored.

Generating the LVL Comparison Rule File

Produce an LVL rule file using DBdiff. The rule file employs XOR rule checks to find differences between two designs. The layers used in the layouts are assumed to correspond by number.

Prerequisites

- A Calibre nmDRC/Calibre nmDRC-H license pair.
- Two complete GDS, OASIS, or OpenAccess layout databases to be compared. If an OpenAccess database is being used as input, see “[Database Translation Utilities](#)” on page 111 for further instructions.

Procedure

1. Run the following command:

```
$MGC_HOME/bin/dbdiff -system GDS \  
-design layout_mod.gds TOPCELL \  
-refdesign layout.gds TOPCELL \  
-write_xor_rules rules_diff.xor \  
-turbo
```

- The -system argument specifies the layout format. GDS, LEF/DEF, OA, or OASIS can be specified.
- The -design argument specifies one design file and its top-level cell name.
- The -refdesign argument specifies a reference design file and its top-level cell name.

- The `-write_xor_rules` argument specifies the output rule file name.
 - The `-system` and `-refsystem` arguments may differ.
2. Verify that the run completes without any errors.

Results

The run produces a rule file that can be used for LVL comparison. The rule file contains an **XOR** rule for each corresponding layer in the two specified layouts. This is the type of rule file you should use when using the calibre `-fx` option.

Note



It is not recommended to modify the rule file.

During DBdiff file rule generation, layers are assumed to correspond by number between the two designs. For example, if metal1 is on layer 7, datatype 1 in one layout, it is assumed the same is true in the other layout.

Read through the generated rule file to familiarize yourself with its contents. For each compared layer, the tool generates statements similar to the following (without the comments):

```
LAYER MAP 1 DATATYPE 0 65535 // first layer in -design database
LAYER MAP 12 DATATYPE 0 65534 // matching layer in -refdesign database
LAYER L1_D0 65535 // internally-assigned layer names
LAYER B_L1_D0 65534
DRC CHECK MAP L1_D0 GDSII 1 0 "rules.xor.gds" // GDS results output
DRC CHECK MAP L1_D0 ASCII // ASCII results output
L1_D0 { @ compare layer 1 datatype 0 // XOR rule check
      XOR L1_D0 B_L1_D0
}
```

If you want to apply user-specified layer names in the output rule checks, use the `-layermap filename` option. If you want to control the specification statements that are written to the output rules, use the `-include_rules filename` option.

For more details, see “[FastXOR Command Line Syntax](#)” on page 34.

Using FastXOR to Perform LVL

Use FastXOR to compare two layouts.

FastXOR is best suited for designs with similar hierarchies, for example comparing the layouts of two design iterations. For designs that include significant differences in hierarchy or geometry, it is recommended to first set the DBdiff maximum difference threshold using the `-maxdiff` option to auto, or to use standard Calibre XOR. Setting the maximum difference threshold switches optimized performance by causing FastXOR to switch to standard XOR when a certain number of differences is detected.

For instructions on using Calibre Interactive to run FastXOR, see the section “[Performing a FastXOR Layout Comparison Run in Calibre Interactive](#)” in the *Calibre Interactive User’s Manual*.

Prerequisites

- An XOR comparison rule file that has only supported rule file statements. The rule file should handle layer mapping correctly so the appropriate layers are compared. See “[Generating the LVL Comparison Rule File](#).”
- Two complete layout databases to be compared.
- If an OpenAccess database is being used as input, see “[Third-Party Database Support](#)” on page 111 for further instructions.
- The rule file may not specify “*” as the [Layout Primary](#) or [Layout Primary2](#) names.
- The [Layout Path](#) and [Layout Path2](#) may only specify a single file each.
- If your rule file was not generated using the dbdiff -write_xor_rules option, ensure your rule file has this statement in it:

```
LAYOUT INPUT EXCEPTION SEVERITY DATATYPE_MAP_TARGET 2
```

- Sufficient Calibre nmDRC/Calibre nmDRC-H license pairs for the number of CPUs in the run.

Procedure

1. Execute the following commands:

```
% calibre -drc -hier -turbo -fx rules.xor |& tee fx.log
```

The first command calls DBdiff internally as part of the run. There may be nothing written to the log file for significant periods of time. Run the top command and look for dbdiff.mod or dbdiff_64.mod to see that the process is running.

You will see differences between this log file and one produced by a run that does not use the -fx option. For example, notice these lines:

```
-----  
-----      EXECUTING FAST HIERARCHICAL COMPARE WITH DBDIFF      -----  
-----  
--- DBDIFF: READING DESIGNS...  
--- DBDIFF: READING DESIGNS COMPLETED. CPU TIME = 232 REAL TIME = 64  
--- DBDIFF: AUTOMATCHING...  
--- DBDIFF: AUTOMATCHING COMPLETED. CPU TIME = 32 REAL TIME = 34  
--- DBDIFF: COMPARING DESIGNS...  
--- DBDIFF: COMPARING DESIGNS COMPLETED. CPU TIME = 63 REAL TIME =  
64  
--- DBDIFF: HIERARCHICAL COMPARE COMPLETE. TOTAL CPU TIME = 339 REAL  
TIME = 173  
--- DBDIFF: DESIGNS ARE DIFFERENT
```

2. Check the [DRC Summary Report](#) file for warnings and results statistics. This line in the report tells you the total number of hierarchical results and an estimate of the flat result count, just like the typical DRC run.

```
--- TOTAL RESULTS GENERATED = 58634 (58704)
```

Results

The run produces the following:

- DRC Summary Report file: *rule_file.summary*
- ASCII [DRC Results Database](#): *rule_file.asc*
- Geometric output database: *rule_file.gds*
- Run transcript

File names depend on what is specified in the rule file.

Notice the following at the end of the transcript:

```
--- DBDIFF: CPU TIME = 476 REAL TIME = 310 MEMORY USED = 4276  
--- FASTXOR: CPU TIME = 479 REAL TIME = 320 MEMORY USED = 4293
```

The first line shows statistics just for the DBdiff portion of the run. The second line shows statistics for the complete FastXOR run. The MEMORY USED in the second line is the sum of memory used by DBdiff and the Calibre DRC executive process until the Calibre database construction module.

You can open your reference layout design in your layout editor and the ASCII DRC Results Database file in Calibre RVE to check any XOR differences. You can open the geometric output database in your layout editor for viewing.

See “[Restrictions](#)” on page 47 and “[Troubleshooting FastXOR Runs](#)” on page 50 for information about constraints on FastXOR runs and debugging information.

Comparing FastXOR and Calibre nmDRC-H Run Results

Compare FastXOR (`calibre -fx`) and traditional Calibre nmDRC-H LVL runs on the same two layouts when using similar runtime environments for the two runs.

FastXOR is often appreciably faster than standard Calibre nmDRC-H when the run times are greater than a half hour and the hierarchies of the two designs are similar. The number of results the two methods find can differ, but in the flat view, the results are typically identical. See “[Differences Between FastXOR and Standard Dual Database Processing](#)” on page 27.

Prerequisites

- Results from a FastXOR run. See “[Using FastXOR to Perform LVL.](#)”

- Results from a Calibre nmDRC-H run that are in a different directory than the FastXOR run. See “[Using Calibre nmDRC-H to Perform LVL.](#)”
- Sufficient Calibre nmDRC/Calibre nmDRC-H license pairs for the number of CPUs in your run.

Procedure

1. Open the run transcript from your normal Calibre nmDRC-H run and search for these lines near the end of the file:

```
--- TOTAL RESULTS GENERATED = 58698 (58700)
...
--- TOTAL CPU TIME = 16809 REAL TIME = 3370
```

2. Open the run transcript from your FastXOR run and look for lines like these at the end of the file:

```
--- DBDIFF: CPU TIME = 476 REAL TIME = 310 MEMORY USED = 4276
--- FASTXOR: CPU TIME = 479 REAL TIME = 320 MEMORY USED = 4293
```

The first line shows statistics just for the DBdiff portion of the run. The second line shows statistics for the complete FastXOR run.

3. In the run transcript from your FastXOR run, search for the TOTAL RESULTS GENERATED line.

Note that the hierarchical result counts between the runs can differ. In the following steps, we will see that these results represent the same thing.

4. If you used the comparison rule file generated by DBdiff for both of your LVL runs, then make the following edits to your rule file:

```
//LAYOUT PATH "<original_layout_path>"
//LAYOUT PATH2 "<original_layout_path>"
LAYOUT PATH "<dbdiff_run_directory>/<rule_file>.gds"
LAYOUT PATH2 "<drc-h_run_directory>/<rule_file>.gds"
```

When DBdiff is used to generate the comparison rules, the generated file specifies geometric output of the DRC results in addition to ASCII results. By default, that geometric results database uses the output rule file name as the portion of the file name before the filename extension. Comparing those two output databases will verify whether or not the results of the FastXOR and normal Calibre nmDRC runs are identical.

5. Run DRC as follows:

```
% calibre -drc -hier -turbo rules.xor |& tee verify.log
```

6. Verify that the results are empty by checking the tail of the run transcript:

```
--- TOTAL RESULTS GENERATED = 0 (0)
```


Results

If you used DBdiff to generate your rule file, the run produces the following:

- DRC Summary Report file: *rule_file.summary*
- ASCII [DRC Results Database](#): *rule_file.asc*
- Geometric output database: *rule_file.gds*
- Run transcript

The names depend upon what is specified in the rule file.

If you do not want the overhead of generating a geometric results database, comment out or remove the DRC Check Map statements in the rule file and save the file to a new name. Use that file for your DRC validation run.

See [“Restrictions”](#) on page 47 and [“Troubleshooting FastXOR Runs”](#) on page 50 for information about constraints on FastXOR runs and debugging information.

Improving FastXOR Performance

In some cases, the default FastXOR configuration may not be optimal for your environment or layout databases. There are several options you can define that can have a significant impact on performance.

A key to achieving the best possible Calibre performance is to ensure that Calibre constructs an optimal hierarchical database (HDB) from your layout. The Layout Base Layer command is extremely important for HDB construction. Base layers are considered to be device creation layers, such as “diffusion”, “contact” (not vias) and “polysilicon” layers. These layers are likely to occur in lower levels of hierarchy and not at the top level. The “well”, “substrate”, “metal”, and “via” layers are *not* typically good candidates for base layers.

You can define base layers when writing XOR rules from DBdiff or by inserting the Layout Base Layer SVRF command in your Calibre XOR rules file.

For this procedure, assume that your design contains the following base layers:

Layer Name	Layer Number	Layer Datatype Number
poly	1	0
diff	2	0
contact	3	0

Procedure

1. Do either of the following:


- Generate an XOR rules file using DBdiff with the `-base_layers` option:

```
dbdiff -system GDS -design lay2.gds top -refdesign lay1.gds top \
-write_xor_rules xor.rul diff -base_layers 1.0 2.0 3.0
```

- Enter the following statement into your existing SVRF XOR rules file:

```
LAYOUT BASE LAYER {base_layer ...}
```

Note

 When using Layout Base Layer, it is important to exclude interconnect layers, marker layers, artificial cell boundaries (for example, the prBoundary layer), or any layers you are not sure about.

2. The XOR rules file automatically generated by DBdiff contains the following statements:

```
//=====
//BASE LAYERS STATEMENT
//=====
LAYOUT BASE LAYER L1_D0 B_L1_D0 L2_D0 B_L2_D0 L3_D0 B_L3_D0
```

3. Run FastXOR.

FastXOR Command Line Syntax

FastXOR is triggered by the calibre `-fx` command-line option.

The following command is used to invoke FastXOR:

```
calibre -drc -hier -fx [<drc_options>] rule_file
```

Any of the Calibre nmDRC command line options may be specified. The `-drc` option is required. Calibre nmDRC and Calibre nmDRC-H licenses are checked out in conformance to the type of run that the command line specifies.

The procedure for running the tool is covered under “[Using FastXOR to Perform LVL](#)” on page 29.

XOR Rules Generated by DBdiff

You can generate rules for LVL comparison by using the DBdiff `-write_xor_rules` option. The rules generated by this command are recommended for all LVL XOR runs.

A typical rule file generation command is the following:

```
dbdiff -system GDS -design TOP.gds TOP -refdesign TOP1.gds TOP \  
-write_xor_rules rules.xor -comparetext
```

The following is an excerpt of the rules.xor file for a typical run:

```
LAYOUT SYSTEM GDS  
LAYOUT SYSTEM2 GDS  
LAYOUT PRIMARY TOP  
LAYOUT PRIMARY2 TOP  
//=====  
LAYOUT PATH "TOP.gds"  
LAYOUT PATH2 "TOP1.gds"  
LAYOUT ERROR ON INPUT NO  
DRC KEEP EMPTY NO  
//=====  
DRC RESULTS DATABASE "rules.xor.asc" ASCII  
DRC SUMMARY REPORT "rules.xor.summary" HIER  
DRC CELL NAME YES CELL SPACE XFORM  
DRC MAXIMUM RESULTS ALL  
  
LAYOUT BUMP2 11  
LAYOUT USE DATABASE PRECISION YES  
LAYOUT PROCESS BOX RECORD YES  
LAYOUT INPUT EXCEPTION SEVERITY DATATYPE_MAP_TARGET 2  
//=====  
// XOR RULES  
//=====  
LAYER MAP 1 DATATYPE 0 65535  
LAYER MAP 12 DATATYPE 0 65534  
LAYER L1_D0 65535  
LAYER B_L1_D0 65534  
DRC CHECK MAP L1_D0 GDSII 1 0 "rules.xor.gds"  
DRC CHECK MAP L1_D0 ASCII  
L1_D0 { @ compare layer 1 datatype 0  
        XOR L1_D0 B_L1_D0  
}  
...
```

See “[Layout Comparison Command Line Syntax](#)” on page 79 for the complete DBdiff command line.

Control of Specification Statements in Generated XOR Rules

You can control the statements that are added to the FastXOR rules file by applying different options in DBdiff.

By default, the DBdiff -write_xor_rules option generates the following statements in the output rule file:

```
LAYOUT ERROR ON INPUT NO
DRC KEEP EMPTY NO
DRC CELL NAME YES CELL SPACE XFORM
DRC MAXIMUM RESULTS ALL
LAYOUT USE DATABASE PRECISION YES
LAYOUT PROCESS BOX RECORD YES
LAYOUT INPUT EXCEPTION SEVERITY DATATYPE_MAP_TARGET 2
```

In some cases, it may be desirable to change these statements for the output rule file. The -include_rules *filename* option enables you to modify the SVRF statements that you want to include in the generated rule file.

When you specify -include_rules, DBdiff generates only the following SVRF statements:

```
DRC CELL NAME YES CELL SPACE XFORM
DRC MAXIMUM RESULTS ALL
LAYOUT PROCESS BOX RECORD YES
LAYOUT INPUT EXCEPTION SEVERITY DATATYPE_MAP_TARGET 2
```

Additionally, the specification statements in the -include_rules rule file are imported directly into the rule file generated by DBdiff. There is no sanity check performed by DBdiff for the imported statements. Hence, you need to ensure that the imported statements are valid and do not conflict with the other SVRF statements generated by DBdiff.

FastXOR Layer Mapping and Comparison

The FastXOR rule file that is written using the DBdiff -write_xor_rules option maps layers for comparison based upon corresponding numbers. Layers that have the same numbers in both databases, respectively, are compared in the rule file that gets written.

If you do not use a rule file described in the preceding paragraph, you may have [Layer Map](#) statements in your rule file. FastXOR handles the majority of layer mapping and comparison scenarios in the way you would expect.

For the following discussion, we define the terms *simple layer*, *layer group*, and *member layer*.

- A *simple layer* is a single drawn layer in a layout database. It may or may not have a datatype.
- A *layer group* is formed when multiple simple layers are mapped to a single layer. Here, layer 100 is a layer group formed from layers 1 and 2:

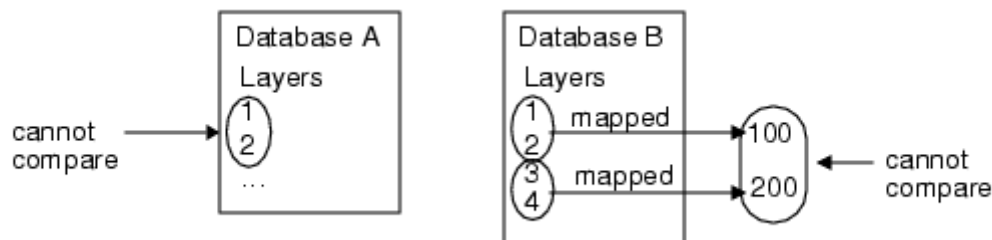
```
LAYER MAP > 0 < 3 DATATYPE 0 100
```

- A *member layer* is a simple layer that is a member of a layer group.

From the preceding Layer Map statement, Calibre layer 100 is a layer group composed of member layers 1.0 and 2.0.

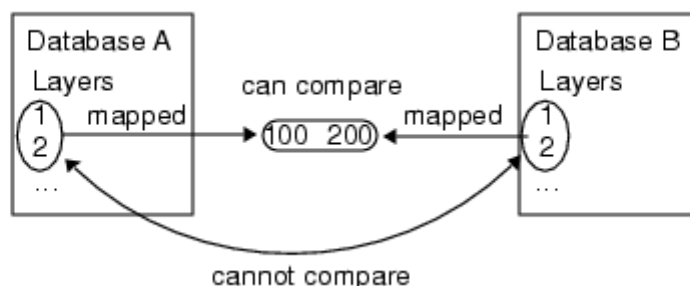
The following are unsupported layer comparison configurations:

- **Comparison of two layers or layer groups from the same database** — Suppose that layers 1 and 2 are both from the same layout database. These layers cannot be compared to each other. This applies to both simple layers and layer groups.

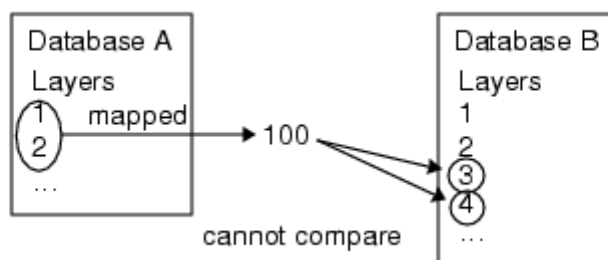


- **Comparison of member layers from different layer groups** — Suppose that layers 1 and 2 are mapped to layer 100 in database A. Suppose that layers 1 and 2 are mapped to

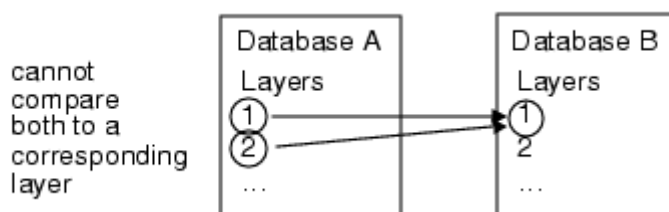
layer 200 in database B. You may compare layer 100 to layer 200 (corresponding layer groups). However, the member layers of layer groups 100 and 200 cannot be compared.



- **Comparison of group** — Suppose layers 1 and 2 are mapped to layer 100 in database A. Layer group 100 cannot be compared to multiple distinct layers in database B.



- **Comparison of group** — Suppose you have layers 1 and 2 in database A. These layers cannot both be compared to layer 1 or to layer 2 in database B.



The layer mapping behavior described previously can be disabled in a calibre -fx run by setting this environment variable:

CALIBRE_FX_DISABLE OPS=1

This causes only matching layers to be compared.

Customized Layer Mapping **39**

Customized Layer Mapping

The DBdiff -layermap file is used for customized layer mapping for comparing layers on differing layer numbers, datatypes, and so forth. The -layermap option can be specified when writing the rules with the dbdiff -write_xor_rules option, or it can be specified with the CALIBRE_FX_DB_DIFF_OPTIONS environment variable. In the latter case, it applies during the FastXOR run and the rules are expected to handle the layers after mapping.

See also “[Place and Route Layer Mapping](#)” on page 167.

The layer mapping file can contain information in two forms, depending on usage. In GDS or OASIS format, columns refer to layer number constraint, datatype number constraint, and layer name, respectively. Layer number and datatype constraints use the customary Calibre forms. For example, this file could be used for mapping GDS and OASIS layers:

#Layer Number	Datatype	Layer Name
>2<=5	>0<4	MET1
>5<7	< 5	MET2
8	0	MET3
1	0	MY_LAYER

Using the dbdiff -layermap option during rule generation with the preceding entries causes this to be written in the rule file output for MY_LAYER:

```
LAYER MY_LAYER 65535
LAYER B_MY_LAYER 65534
DRC CHECK MAP MY_LAYER GDSII 1 0 "rules.xor.gds"
DRC CHECK MAP MY_LAYER ASCII
MY_LAYER { @ compare layer MY_LAYER
            XOR MY_LAYER B_MY_LAYER
        }
```

Note the presence of MY_LAYER instead of the system-generated layer and check names.

FastXOR Environment Variables

Layer Mapping Based Upon Originating Design	39
Layer Mapping with Include and Exclude Areas	40

Layer Mapping Based Upon Originating Design

You can use an optional column in a layer mapping file that specifies a CURrent or REference keyword. Using these keywords creates a correspondence between disparate layers in two databases. These two keywords can be abbreviated with their first three letters.

CURrent refers to a current layer to compare to a reference layer. CURrent is defined either by the DBdiff **-design** parameter or the **-lef** and **-def** parameters. REference specifies the original layer for comparison with the current layer. REference is defined by either **-redesign** or **-reflef** and **-refdef**.

For example, if the layer mapping file contains this:

```
1  >=0 met1 CUR
11 >=0 met1 REF
2  >=0 met2 CUR
12 >=0 met2 REF
```

using the dbdiff -layermap option during rule generation with the preceding entries causes this to be written in the rule file output:

```
//assume LAYOUT BUMP2 103

LAYER MAP 1 DATATYPE >=0 65535
LAYER MAP 114 DATATYPE >=0 65534
LAYER met1 65535
LAYER B_met1 65534
DRC CHECK MAP met1 GDSII 1 0 "rules.gds"
met1 { @ compare layer met1
      XOR met1 B_met1

LAYER MAP 2 DATATYPE >=0 65533
LAYER MAP 115 DATATYPE >=0 65532
LAYER met2 65533
LAYER B_met2 65532
DRC CHECK MAP met2 GDSII 2 0 "rules.gds"
met2 { @ compare layer met2
      XOR met2 B_met2
}
```

Layer Mapping with Include and Exclude Areas

You can use an optional column in the layer mapping file that specifies a WINDEL_LAYER or WINDOW_LAYER keyword. The keywords provide a way to exclude or include areas covered by specific layers in a design.

Note



Using the WINDEL_LAYER or WINDOW_LAYER keywords negates the need for [Layout Windel Layer](#) or [Layout Window Layer](#) rule file statements.

The CUR and REF keywords may be used along with the WINDEL_LAYER or WINDOW_LAYER keywords to control which layers and from which design to act as include or exclude regions. For example:

```
11 0 MET2 REF WINDEL_LAYER
99 0 MARKER CUR WINDOW_LAYER
```

This causes MET2 in the reference design to be treated as a Layout Windel Layer (exclude layer). The MARKER layer in the current design is treated as a Layout Window Layer (include layer).

Optionally, an integer layer number can be specified as a destination layer for the included or excluded layers. For example:

```
11 0 MET2 REF WINDEL_LAYER 1001
```

Here, MET2 is handled as a Layout Windel Layer as before, but it is mapped to layer 1001. If a number is not provided, a layer number is generated by the tool automatically.

Excluded Cells in FastXOR

FastXOR rule files may have the Exclude Cell specification statement in them. When Exclude Cell statements are present in the rule file, the same semantics for a typical Calibre nmDRC run apply to the DBdiff portion of a FastXOR run. Such statements can appear in the FastXOR rule file when the DBdiff -include_rules option is used.

When cells are excluded, all the data in the cells is filtered by DBdiff. The excluded cell names do not appear in the DRC Summary Report file.

If an excluded cell cannot be located, then warnings like this are issued:

```
WARNING: EXCLUDE CELL name <name> is not located within input layout  
database <file>.
```

The severity of this condition is controlled by the [Layout Input Exception Severity](#) EXCLUDE_CELL_NAME exception.

The [Layout Primary](#) cell may not be specified as an Exclude Cell parameter.

NOT Operations in FastXOR

Boolean NOT operations are supported in FastXOR with certain restrictions.

Note



NOT operations are supported only if these conditions are met:

- A NOT operation may only appear inside of a rule check. It may not be used for layer derivation.
 - The layer arguments to the NOT operation must appear in an XOR rule check.
 - The input layers to the operation are original.
-

For example, this is permitted:

```
A_B {XOR A B}  
A_NOT_B {A NOT B}  
B_NOT_A {B NOT A}
```

But these are not permitted:

```
A_NOT_C {A NOT C} // not allowed. C is not paired with A in an XOR rule.  
D = A NOT B      // layer derivations with NOT are disallowed.
```

You can generate an XOR rule file with NOT operations in it as shown previously by using the `dbdiff -write_xor_rules -enhanced` options. When using these options, the generated rules contain rule checks like these:

```
L1_D0 { @ compare layer 1 datatype 0  
      XOR L1_D0 B_L1_D0  
}  
L1_D0_NEW_SHAPES { @ New shapes in layout path on layer 1 datatype 0  
      L1_D0 NOT B_L1_D0  
}  
L1_D0_MISSING_SHAPES { @ missing shapes in layout path on layer 1 datatype  
0  
      B_L1_D0 NOT L1_D0  
}
```

FastXOR Run Transcript Features

FastXOR outputs run-specific entries to the transcript that are useful for debugging.

When you run `calibre -fx`, the following entries appear in the run transcript that are specific to FastXOR.

```
-----  
-----          EXECUTING FAST HIERARCHICAL COMPARE WITH DBDIFF          -----  
-----  
--- DBDIFF: READING DESIGNS...  
--- DBDIFF: READING DESIGNS COMPLETED. CPU TIME = 232 REAL TIME = 64  
--- DBDIFF: AUTOMATCHING...  
--- DBDIFF: AUTOMATCHING COMPLETED. CPU TIME = 32 REAL TIME = 34  
--- DBDIFF: COMPARING DESIGNS...  
--- DBDIFF: COMPARING DESIGNS COMPLETED. CPU TIME = 63 REAL TIME = 64  
--- DBDIFF: HIERARCHICAL COMPARE COMPLETE. TOTAL CPU TIME = 339 REAL TIME  
= 173  
--- DBDIFF: DESIGNS ARE DIFFERENT
```

This section shows the DBdiff processing steps and runtime statistics.

At the end of the transcript, you will see this:

```
--- DBDIFF: CPU TIME = 476 REAL TIME = 310 MEMORY USED = 4276  
--- FASTXOR: CPU TIME = 479 REAL TIME = 320 MEMORY USED = 4293
```

This shows the DBdiff statistics for that portion of the run, followed by the statistics for the complete `calibre -fx` run. The FASTXOR: MEMORY USED is the sum of memory used by DBdiff plus the Calibre DRC executive until the Calibre database construction module.

DBdiff Command Execution in FastXOR

During a FastXOR run, DBdiff is automatically called as an initial processing step to analyze the layouts for differences. It is unnecessary to call DBdiff explicitly.

Setting DBdiff Options

You can specify DBdiff options using different methods. The tool reads the options in the following order:

- **Template file in the Calibre installation directory** — Internally, DBdiff uses a template file that contains default options. The template file is stored in `<calibre_install_dir>/shared/pkgs/fdi/templates/fastxor.template`. This template file is used by default in all runs.

You can copy and modify the template file in a custom location and specify it with the `-template` option using the SVRF statement or environment variable. If you specify a custom template file, the options from the custom file are used and the default template settings are ignored.

- **SVRF statement in rule file** — Set options using the [SVRF DBdiff Options](#) statement in your rule file. If the statement is present and the environment variable is not set, the tool *only* uses the options defined in the statement and the default template file is ignored.

You can only specify the SVRF statement once in your rule file; additional SVRF DBdiff Options statements are ignored without indication.

- **Environment variable** — Additional command options can be specified using the `CALIBRE_FX_DB_DIFF_OPTIONS` environment variable. The environment variable options are added to the default template and the SVRF DBdiff Options statement (if the statement does not include the `-template` option). The environment variable options override both the default template file and SVRF statement if the options are the same.

See “[Layout Comparison Command Line Syntax](#)” on page 79 for the complete DBdiff command line. Executing `dbdiff -h` as a shell command shows a list of options.

Examples

The following examples cause DBdiff to rotate the design that is compared to the reference design by 90 degrees, counter-clockwise. There are additional options for translating (offset) or reflecting (mirroring) the compared design.

```
SVRF DBDIFF OPTIONS "-transform R90"
```

```
setenv CALIBRE_FX_DB_DIFF_OPTIONS "-transform R90"
```

If there is at least one unique layer (deleted or added) between compared designs, consider doing one of the following to improve performance:

- Include the following statement in your SVRF rule file:

```
SVRF DBDIFF OPTIONS "-passthroughlayer"
```

- Set the following environment variable:

```
setenv CALIBRE_FX_PASS_THROUGH_LAYER_ENABLE 1
```

When enabled, FastXOR passes unique layers directly to the traditional DRC XOR engine, bypassing unnecessary dbdiff operations.

Management of Large Numbers of Differences

In cases where the number of differences between two designs is large, it is more efficient to enable a maximum difference threshold to switch from FastXOR mode to conventional XOR mode automatically.

Note



The -maxdiff option is provided for rare instances when unexpectedly large difference counts exist between the input databases. The time spent before a FastXOR run is halted by the threshold plus the total runtime for restarting the conventional XOR job must be less than the time it would have taken for FastXOR to complete for the option to be advantageous.

To automatically stop a FastXOR run and restart it as a standard XOR run when a certain number of total differences is exceeded, set the CALIBRE_FX_DB_DIFF_OPTIONS environment variable to include the -maxdiff argument set as follows:

```
setenv CALIBRE_FX_DB_DIFF_OPTIONS "-maxdiff 5.0e+06"
```

In this example, the FastXOR run halts at five million violations and *restarts* the run as a conventional XOR comparison. The numeric argument must evaluate to a non-negative integer and may be written in scientific notation, as shown.

You can also specify the “auto” keyword instead of specifying a number of differences. This instructs the tool to use a built-in value for the number of differences. For example:

```
setenv CALIBRE_FX_DB_DIFF_OPTIONS "-maxdiff auto"
```

The auto keyword enables an internal algorithm that switches from FastXOR to XOR when high difference counts are detected.

You can enable this new feature using any of the following methods:

- Set the MGC_DBDIFF_MAXDIFF_AUTO environment variable to any non-null value before running dbdiff -write_xor_rules to create XOR rules. DBdiff includes the -maxdiff auto option in the generated rule file.
- Include the [SVRF DBdiff Options](#) statement in your XOR rule file as follows:

```
SVRF DBDIFF OPTIONS "-maxdiff auto"
```

- Set the CALIBRE_FX_DB_DIFF_OPTIONS environment variable with the new option as follows:

```
setenv CALIBRE_FX_DB_DIFF_OPTIONS "-maxdiff auto"
```

The total difference count is the sum of object (shape) differences taken per-cell instance over all cell instances in the compared designs, as follows:

- If a cell instance appears in one design but not the other, all objects in the corresponding master cell contribute to the total difference count.
- If a cell instance appears in both designs and its master cell differs between the designs, then the objects that differ between the two versions of the master cell contribute to the total difference count.

If the -maxdiff threshold is reached at any point, the run shifts to conventional XOR mode.

The following sections provide a detailed explanation of how DBdiff counts differences when you apply the -maxdiff option.

Missing and New Shapes

Assume your current layout consists only of a square polygon and your reference layout contains a triangular polygon. If you run layout comparison on these two layouts, DBdiff reports a total difference count of two as follows:

- 1 MISSING SHAPE
- 1 NEW SHAPE

In other words, DBdiff could not find the triangular polygon that is in the reference design and additionally found a new square polygon that was not in the reference design.

Instance Differences

If a cell is modified between the current and reference designs, then the total difference count depends on the number of placed instances of that cell as follows:

$$\text{diff_count} = \text{num_cell_differences} \times \text{num_cell_instances}$$

FastXOR Log File

The generated FastXOR log file includes additional information when you set the -maxdiff option.

- Summary of the flat object count of both designs and the applied -maxdiff value (if you specified the auto keyword, then the log includes the automatically calculated value here). For example:

```
--- DBDIFF: FLAT OBJECTS COUNT = 30925013533 MAXDIFF THRESHOLD =  
137724865
```

- Current progress regarding the found differences between the designs. The tool writes the percentage of found differences versus the -maxdiff threshold and displays the value at intervals of 10%. For example:

```
--- DBDIFF: FOUND 110180046 OBJECT DIFFERENCES (80%)
```

- If the -maxdiff threshold is not exceeded, the log includes the number of found differences. For example:

```
--- DBDIFF: MAXDIFF DETECTED DIFFERENCES COUNT = 148492029
```

LVL Run Configuration

You should consider running FastXOR, traditional XOR, and doing a comparison of the two runs in three separate directories.

Here is a sample script for doing this:

Example 3-1. LVL Run Script

```
#!/bin/sh

# remove any files from a previous run
rm -rf *XOR xor.rules

# Generate the XOR rule file using dbdiff
# Messages to STDOUT
dbdiff -system GDS -design top_ic_2.gds top \
-refdesign top_ic.gds top -write_xor_rules xor.rules -turbo

mkdir FASTXOR NORMALXOR VALIDATION_FASTXOR

cd FASTXOR
ln -s ../xor.rules
ln -s ../top_ic_2.gds
ln -s ../top_ic.gds
# Run FASTXOR
calibre -drc -hier -turbo -hyper -fx xor.rules > FASTXOR.log

cd ../NORMALXOR
ln -s ../xor.rules
ln -s ../top_ic_2.gds
ln -s ../top_ic.gds
# Run regular XOR
calibre -drc -hier -turbo -hyper xor.rules > NORMALXOR.log

# Run validation of FASTXOR and normal XOR results
# dbdiff messages to STDOUT
cd ../VALIDATION_FASTXOR
dbdiff -system GDS -design ../NORMALXOR/xor.rules.gds \
-refdesign ../FASTXOR/xor.rules.gds -write_xor_rules validation.rules \
-turbo

calibre -drc -hier -turbo -hier -hyper validation.rules \
> VALIDATION.log

#Look for number of results of validation run
echo "`grep 'TOTAL RESULTS GENERATED' ../VALIDATION.log`"
```

Restrictions

You must understand the limitations of FastXOR before you perform database comparison.

The following restrictions are in effect for using FastXOR (calibre -fx).

- FastXOR can only be run using Calibre nmDRC-H.
- The only rule file layer operations that are executed during a FastXOR run are single-layer **OR** (for layer merging), **XOR** (for comparison), **NOT** (for comparison) and the Drawn family of layer operations.

- The supported layout formats are GDSII, OASIS, LEF/DEF, and OpenAccess.
- [Layout Path](#) and [Layout Path2](#) can only specify a single file each.
- [Layout Primary](#) and [Layout Primary2](#) may not specify “*”.
- The calibre -fx rule file must have [Layout Input Exception Severity](#) DATATYPE_MAP_TARGET 2 set.
- TVF rule file format is not supported.

Most SVRF layer operations and specification statements are *not permitted* in a rule file used for FastXOR and cause errors if used. The following table shows the elements that are allowed.

Table 3-1. Supported Layer Operations and Specification Statements

Drawn family (e.g. Drawn Acute, Drawn Offgrid) ¹	DRC Summary Report	Layout Error On Input	Layout Windel Layer
DRC Cell Name	DRC Tolerance Factor	Layout Input Exception RDB ²	Layout Window
DRC Check Map	DRC Unselect Check	Layout Input Exception Severity ²	Layout Window Cell
DRC Check Text	DRC Unselect Check By Layer	Layout Magnify	Layout Window Clip
DRC Exclude False Notch	Exclude family (such as Exclude Acute, Exclude Offgrid) ¹	Layout Path	Layout Window Layer
DRC Keep Empty	Flag family (such as Flag Acute, Flag Offgrid) ¹	Layout Path2	OR ³
DRC Magnify Results	Flatten	Layout Precision	Precision
DRC Maximum Cell Name Length	Group	Layout Primary	Resolution
DRC Maximum Results	Include	Layout Primary2	SVRF Error
DRC Maximum Vertex	Layer	Layout System	SVRF Message
DRC Results Database	Layer Directory	Layout System2	SVRF Version
DRC Results Database Libname	Layer Map	Layout Top Layer	Title

Table 3-1. Supported Layer Operations and Specification Statements (cont.)

DRC Results Database Precision	Layout Base Cell	Layout Use Database Precision	Variable
DRC Select Check	Layout Base Layer	Layout Windel	XOR
DRC Select Check By Layer	Layout Bump2	Layout Windel Cell	

1. Results may be different from a dual-database comparison run because FastXOR only compares layers and sections of the layouts that differ.
2. Only the exceptions in the databases after DBdiff pre-processing are reported.
3. Only single-layer OR operations are supported.

Troubleshooting FastXOR Runs

This section discusses some of the problems you may encounter when doing LVL comparisons using FastXOR.

The following are some general guidelines for debugging XOR runs and improving performance.

- When the normal XOR run is under a half hour, you may not see dramatic improvements in runtime when using FastXOR. However, there have been cases where dramatic improvements are seen in runtime for runs that take less than a half hour.
- If there are many differences between the layout versions on many layers, then the performance advantage of FastXOR over traditional XOR diminishes greatly. This can occur when large cell instances (such as a cell that contains the majority of the functional blocks of the chip) are moved even slightly. In such cases, it is better to use the large cell that was moved as the primary cell for both designs and run FastXOR using that configuration. See “[Management of Large Numbers of Differences](#)” on page 44 for details when this approach will not work.
- Deletion of metal layers that occur in nearly all of the cells will cause a runtime issue. This has a negative effect on DBdiff performance and FastXOR run times will increase accordingly. For layers that occur in a few cells, this is usually not an issue.
- Reticle-like designs where the top cell contains arrays of full chip cells can be slow. Run these designs with the primary cell being the chip cell, not the top cell.
- Setting the CALIBRE_FX_VERBOSE variable to a non-null value is useful for tracking what occurs during the DBdiff portion of the run. This writes a dbdiff.summary file for the DBdiff portion of the calibre -fx run. Be sure to check for error or warning messages and understand what they mean. This variable also causes the run transcript to show a section like this:

```
--- CONTENTS OF DBDIFF INCLUDE_LAYER FILE ...
1 0
10 0
11 0
12 0
```

which shows the layers and datatypes that were used for the run.

- Setting the CALIBRE_FX_TRANSCRIPT_ENABLE variable to a non-null value generates a DBdiff transcript in addition to the FastXOR transcript. When the variable is set, DBdiff writes a transcript of just the DBdiff execution to the file dbdiff<\$pid>.transcript in the working directory. The pid is the process identifier of the dbdiff job.
- Text is only compared in FastXOR if the XOR rules file was generated with the -comparetext DBdiff invocation option.

The following are common problems to debug:

Problem: Some Items in Third-Party Databases Are Not Compared	51
Problem: Corresponding Layers Not Being Compared.....	52
Problem: How Not to Compare All Layers in Both Designs	52
Problem: Results from FastXOR Not Matching Usual XOR Run	53
Problem: DBdiff Reports Designs as Different But FastXOR Finds No Differences ..	54
Problem: Layout Input Exceptions Not Matching Usual XOR Run	58
Problem: Differences in Handling of Non-Orthogonal Shapes	58
Problem: Verification of OpenAccess Database is Failing.....	59
Problem: Verification of LEF/DEF Databases is Failing.....	59

Problem: Some Items in Third-Party Databases Are Not Compared

If you are comparing third-party databases, such as LEF/DEF and OpenAccess you may encounter situations where all items are not compared.

Symptoms

Two layouts are reported as the same after a comparison run, but there are differences between the designs.

For example, one LEF/DEF design may include fill objects, but the other design does not, yet FastXOR and DBdiff report the designs as the same

Causes

DBdiff and FastXOR convert third-party databases to an internal layout format (OASIS) in order to perform comparison. All items in the internal databases are compared.

The problem described in this topic is caused because the FDI translators do not convert all objects in the third-party database *by default*. For example, routing layers and cell instances are converted by default, but fill objects are not.

Solution

In order to avoid this problem, you must map all desired layers and objects using the DBdiff command line options.

Provide an FDI layer mapping file and set the appropriate environment variables as indicated under “[Customized Layer Mapping](#)” on page 39.

Problem: Corresponding Layers Not Being Compared

Troubleshoot layers that are not compared.

Symptoms

Layers in the two designs that seem to correspond in the rule file are not being compared.

Causes

This mainly occurs because the DBdiff phase only scans the layers present in the database and does not read anything in the rule file to establish the context of that layer in the rules. This is typically a rule file configuration problem.

Solution

Use an XOR rule file generated by DBdiff. See “[Generating the LVL Comparison Rule File](#)” on page 28.

Related Topics

[Troubleshooting FastXOR Runs](#)

Problem: How Not to Compare All Layers in Both Designs

Troubleshoot undesired layers that are compared.

Symptoms

The XOR rule file generated using the DBdiff -write_xor_rules options includes all layers in both designs in the comparison rules. How can this be changed to exclude certain layers or include certain layers from the design?

Causes

By default, the -write_xor_rules option takes into account all the layers in both designs.

Solution

If you want to exclude or include a subset of layers in the designs, you can use the DBdiff -exclude_layer or -include_layer options. See “[Layout Comparison Command Line Syntax](#)” on page 79. A file specified using the -exclude_layer or -include_layer options has lines this form:

```
<layer_constraint> <datatype_constraint>
```

which specify the layers and datatypes to exclude or include for the run. The constraint parameters use the same syntax as the [Layer Map](#) specification statement in the rule file. Here is an example of a line that specifies all layers less than or equal to 10 and the datatype of 0:

```
<=10 ==0
```

The `-include_layer` option may not be specified with `-exclude_layer`, or `-layermap` either in this case.

Related Topics

[Troubleshooting FastXOR Runs](#)

Problem: Results from FastXOR Not Matching Usual XOR Run

Troubleshoot mismatches between FastXOR and standard XOR.

Symptoms

The numbers of results in the FastXOR run do not match the numbers of results from a traditional DRC XOR run. This is evident from the tail of the run transcript or the DRC Summary Report.

Causes

FastXOR processes the layout differently from Calibre nmDRC alone. This is expected behavior.

Solution

Export a geometric results database from FastXOR and one from the standard XOR run. Then perform a validation XOR on those two databases. See “[Comparing FastXOR and Calibre nmDRC-H Run Results](#)” on page 31. It is best to use an XOR rule file generated by DBdiff for both runs.

Concentrate on just the layers that have differences. Use [DRC Select Check By Layer](#) in a validation XOR rule file to run just the rule checks that involve layers with differences.

If there are minor differences in the validation XOR run, highlight the polygons in the original designs and visually correlate shapes involved in differences.

Note



If you are running without `-hyper` and want the same hierarchy as a `-hyper` run, set the `CALIBRE_FLATTEN_LIKE_HYPER` environment variable to YES.

Related Topics

[Troubleshooting FastXOR Runs](#)

Problem: DBdiff Reports Designs as Different But FastXOR Finds No Differences

Troubleshoot differences between DBdiff and FastXOR runs.

Symptoms

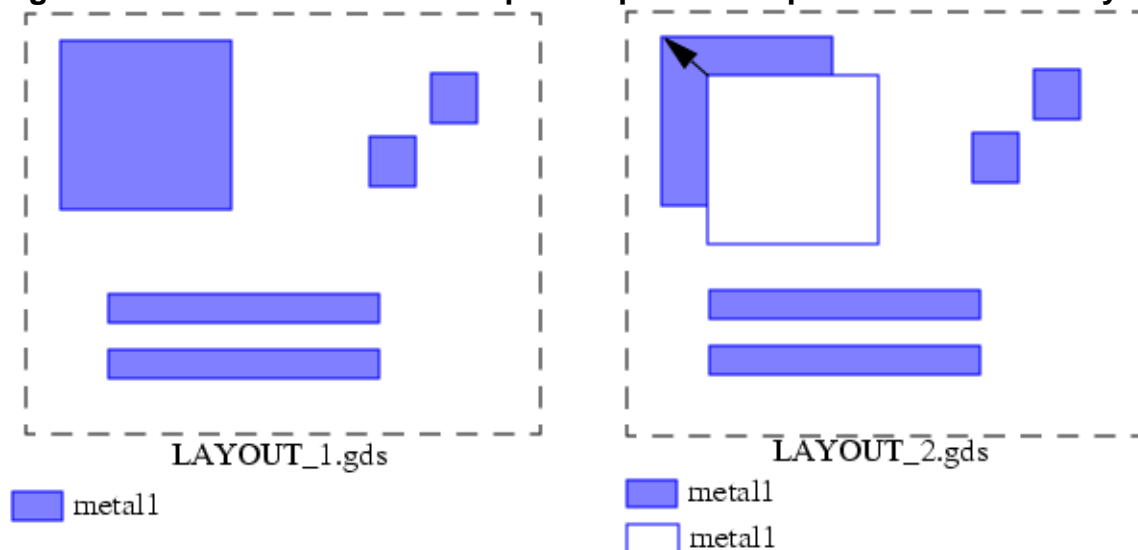
When you perform an LVL comparison, the DBdiff application reports that the two designs are different. When you run FastXOR on the same two designs, no results are produced, indicating that the designs are the same.

Causes

This can be caused by a number of factors. In these cases, the differences reported by DBdiff may not appear in FastXOR results because they do not affect the mask-level data. The most common causes are listed as follows:

- **Duplicate shapes** — Your layout contains exact duplicate shapes on the same layers, at the same coordinates. For example, two metal1 polygons occupy the same space in the second layout, but there is only one metal1 polygon of equal dimensions in the first layout. The following figure demonstrates this scenario:

Figure 3-3. FastXOR Does Not Report Duplicate Shapes On The Same Layer



In *LAYOUT_1.gds*, there is a single metal1 polygon in the upper-left corner of the design. *LAYOUT_2.gds* contains the same metal1 polygon at the same coordinates, however there is also a second metal1 polygon of the same dimensions placed at the same coordinates. The new metal1 polygon is completely superimposed over the

Problem: DBdiff Reports Designs as Different But FastXOR Finds No Differences

existing metall polygon. In this case, the DBdiff tool finds a new polygon and reports a difference in the report file as follows:

```
----- Difference Table -----
| Design | Cell Name | Difference | Type          | Bounding Box          |
...| Master Name| Layer | DataType|
| Current | TOPCELL  | New Object | Rectangle    | -30090 -170 -19860 10000 |
...| 0          | 0          |

***** Summary Report*****

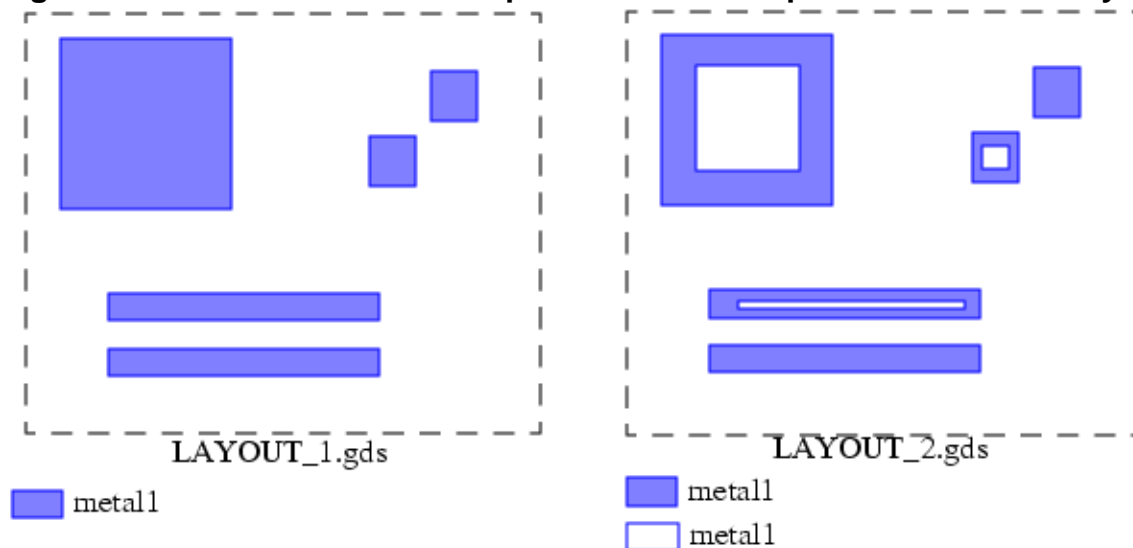
*****Shape difference summary begin*****
Layer  Datatype  NewObjs  MissingObjs  TotalDiffObjs  DesignObjs
RefdesignObjs  TotalObjs  Change (%)
0         0         1         0             1             6             5
11        9.09

*****Shape difference summary end*****
```

For the same layout comparison, FastXOR reports no differences (TOTAL RESULTS GENERATED = 0 (0)) because the superimposed metall polygon is merged in the mask layer.

- **Enclosed shapes** — A shape on one layer is entirely enclosed by a shape on the same layer. The following figure demonstrates this scenario:

Figure 3-4. FastXOR Does Not Report Enclosed Shapes On The Same Layer



In this example, three metal1 polygons are added to *LAYOUT_2.gds*. DBdiff reports that there are three new polygons in the design, as shown in the following transcript output:

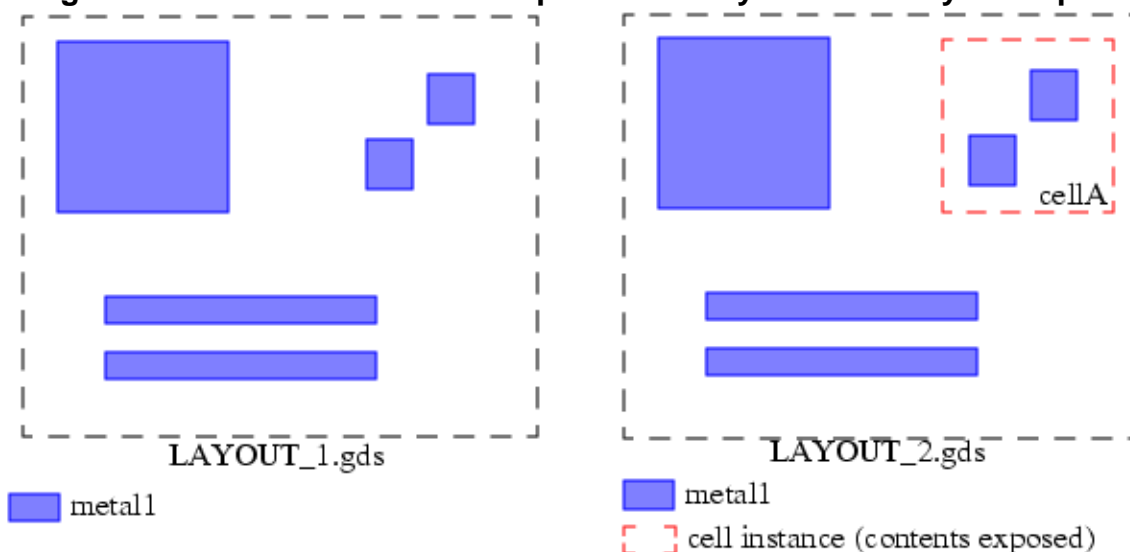
```
----- Difference Table -----
| Design | Cell Name | Difference | Type | Bounding Box |
...|Master Name| Layer | DataType |
| Current | TOPCELL | New Object | Rectangle | -27760 2580 -21790 7425 |
| | 0 | 0 |
| Current | TOPCELL | New Object | Rectangle | -15815 -9845 6940 -8555 |
| | 0 | 0 |
| Current | TOPCELL | New Object | Rectangle | 1130 1290 3710 3710 |
| | 0 | 0 |
***** Summary Report*****
*****Shape difference summary begin*****
Layer Datatype NewObjs MissingObjs TotalDiffObjs DesignObjs
RefdesignObjs TotalObjs Change (%)
0 0 3 0 3 8 5
13 23.08

*****Shape difference summary end*****
```

For the same comparison, FastXOR reports no differences (TOTAL RESULTS GENERATED = 0 (0)) because the new metal1 polygons are completely contained within the existing metal1 shapes; there are no differences in the merged mask-level data for metal1.

- **Different hierarchy** — Shapes are in the same location, but at different levels of hierarchy in the design. The following figure demonstrates this scenario:

Figure 3-5. FastXOR Does Not Report Hierarchy Of Same-Layer Shapes



Problem: DBdiff Reports Designs as Different But FastXOR Finds No Differences

In this example, the two metal1 polygons in the upper-right corner of the design are in the same location in *LAYOUT_1.gds* and *LAYOUT_2.gds*, but they are contained within *cellA* in *LAYOUT_2.gds*. DBdiff reports that a new instance appears in the design (*cellA*) and that two polygons are missing, as shown in the following report:

```
----- Difference Table -----
| Design | Cell Name | Difference | Type | Bounding Box |
...|Master Name | Layer | DataType |
| Current | TOPCELL | New Object | Instance | 0 0 10925 10845 |
| cellA | | | | |
| Current | TOPCELL | Missing Object | Rectangle | 0 0 5000 5000 |
| | 0 | 0 | | |
| Current | TOPCELL | Missing Object | Rectangle | 5925 5845 10925 10845 |
| | 0 | 0 | | |

***** Summary Report*****

*****Shape difference summary begin*****

Layer Datatype NewObjs MissingObjs TotalDiffObjs DesignObjs
RefdesignObjs TotalObjs Change (%)

0 0 0 2 2 5 5
10 20.00

*****Shape difference summary end*****

*****Instance difference summary begin*****

CellName RefCellName NewInsts MissingInsts TotalDiffInsts DesignInsts
RefdesignInsts TotalInsts Change(%)

TOPCELL TOPCELL 1 0 1 1 0
1 100.00

*****Instance difference summary end*****
```

For the same comparison, FastXOR reports no differences (TOTAL RESULTS GENERATED = 0 (0)). Although the two polygons are referenced at a different level of hierarchy, they are still present in the metal1 mask, so FastXOR ignores the hierarchy differences.

Solution

If your design has any of the above features (duplicate shapes, enclosed shapes, or different hierarchy), and you are interested in viewing these differences, use DBdiff instead of FastXOR.

Related Topics

[Troubleshooting FastXOR Runs](#)

Problem: Layout Input Exceptions Not Matching Usual XOR Run

Troubleshoot differences in input exception handling between DBdiff and Calibre nmDRC.

Symptoms

DBdiff appears to handle layout exceptions differently from Calibre nmDRC.

Causes

Calibre batch tools have a rich set of layout input exception handling options through the [Layout Input Exception Severity](#) specification statement. These controls are not available in DBdiff; hence, DBdiff can handle certain exceptions differently from the Calibre batch tools.

Solution

The best approach may be to get the layouts to be free from any input exceptions when using Calibre nmDRC.

Related Topics

[Troubleshooting FastXOR Runs](#)

Problem: Differences in Handling of Non-Orthogonal Shapes

Troubleshoot differences in incorrect comparison results due to non-orthogonal shapes.

Symptoms

Run results differ by 1 dbu for skew edges.

Causes

DBdiff handles skew geometry slightly differently from Calibre nmDRC when it comes to placement of skew edges. DBdiff can place an edge offset by 1 dbu from a usual DRC run; hence, DBdiff could view certain skew edges as matching during comparison where Calibre nmDRC may not.

Solution

These 1 dbu differences must be understood and accepted. In general, it is best to use the normal XOR run results for tape out verification.

Related Topics

[Troubleshooting FastXOR Runs](#)

Problem: Verification of OpenAccess Database is Failing

Troubleshoot problems in OA database comparisons.

Symptoms

At least one input database is OpenAccess and there are mismatches between the layers.

Causes

By default, if an FDI layer map file is not provided (see “[Layer Map File for fdi2gds and fdi2oasis](#)” on page 172), then the Calibre database constructor reads all OpenAccess datatypes and purpose names together for a given layer. For example, the database constructor reads layers 1.0, 1.1, 1.2 as layer 1.0. However, DBdiff (which is used in FastXOR) gives consideration to datatypes and purpose names, so it reads the layers as 1.0, 1.1, and 1.2 respectively. If comparing OpenAccess format to either GDS or OASIS format, the layer numbers are likely different. If comparing OpenAccess format to OpenAccess format and the layer schemes differ, they may not be compared correctly.

Solution

Provide an FDI layer mapping file and set the appropriate environment variables as indicated under “[Customized Layer Mapping](#)” on page 39.

Related Topics

[Troubleshooting FastXOR Runs](#)

Problem: Verification of LEF/DEF Databases is Failing

Troubleshoot problems in LEF/DEF database comparisons.

Symptoms

Two LEF/DEF databases are being compared but there are mismatches between the layers.

Causes

There are two different LEF files being compared, and the layers are not in the same order in each file. By default, the tool will match the layers in the order they appear in the LEF.

Solution

Provide an FDI layer mapping file and set the appropriate environment variables as indicated under “[Customized Layer Mapping](#)” on page 39.

Related Topics

[Troubleshooting FastXOR Runs](#)

FastXOR Environment Variables

FastXOR uses several environment variables to control the tool flow.

See also “[FDI Environment Variables](#)” on page 114.

Table 3-2. FastXOR Environment Variables



Variable and settings	Description
CALIBRE_FX_TRANSCRIPT_ENABLE 1	Writes the DBdiff transcript to a <i>dbdiff.pid.transcript</i> file.
CALIBRE_FX_DB_DIFF_OPTIONS “ <i>dbdiff_options</i> ”	DBdiff command line options can be specified using this variable. The specified options are used during FastXOR runs. The entire list of options are printed for this command: <code>dbdiff -h</code>  Note: You can also set DBdiff options using the SVRF DBdiff Options statement in the rule file. The options in the environment variable take precedence.
CALIBRE_FX_VERBOSE 1	Triggers verbose reporting in the run transcript and writes a <i>dbdiff.summary</i> report file. This is useful in debugging.
CALIBRE_FX_DISABLE_OPS 1	Causes only matching layers (by database number) to be compared rather than resolving layer maps.
CALIBRE_FXOR_GDS_MODE 1	Specifies that FastXOR runs in legacy mode.
CALIBRE_FX_PASS_THROUGH_LAYER_ENABLE 1	Improves performance if one or more layers between the compared designs has been added or removed before a FastXOR comparison. Setting this environment variable instructs FastXOR to pass unique layers directly to the standard dual-database comparison engine, bypassing unnecessary dbdiff operations.  Note: You can also specify this behavior by setting the <code>-passthroughlayer</code> DBdiff option in the SVRF DBdiff Options statement.

Table 3-2. FastXOR Environment Variables (cont.)

Variable and settings	Description
DBDIFF_ENABLE_EXTRA_PATHMASTERS	Set to a non-null value to increase the default memory for path processing in DBdiff by 100 times the default. This is useful only if your layout has large numbers of different paths and you are experiencing issues.
MGC_CALIBRE_CURRENT_DESIGN_LAYERMAP_FILE <i>pathname</i>	Specifies a DBdiff layer mapping file to be applied for the current design during the FastXOR run. This environment variable overrides CALIBRE_FX_DB_DIFF_OPTIONS if both are specified.
MGC_CALIBRE_REFERENCE_DESIGN_LAYERMAP_FILE <i>pathname</i>	Specifies a DBdiff layer mapping file to be applied for the reference design during the FastXOR run. This environment variable overrides CALIBRE_FX_DB_DIFF_OPTIONS if both are specified.
MGC_CALIBRE_LAYERMAP_FILE <i>pathname</i>	Specifies a DBdiff layer mapping file to be applied during the FastXOR run. This environment variable overrides CALIBRE_FX_DB_DIFF_OPTIONS if both are specified.
MGC_TMPDIR " <i>pathname</i> "	Controls where a temporary file is written for layer processing. By default, the location is <i>\$MGC_HOME/tmp</i> . If MGC_TMPDIR is set, then the specified pathname is used. If neither MGC_HOME nor MGC_TMPDIR is set, then the current directory is used.

Chapter 4

DBdiff Database Comparison

DBdiff enables you to compare two design databases and isolate differences between each one. The DBdiff application is a command-line-driven Calibre tool that compares all hierarchy, shapes, properties, and text between two databases. This comparison is not meant to be a replacement for the standard XOR operation, but instead provides designers a way to find the differences between two databases quickly.

Note



See “[Troubleshooting FastXOR Runs](#)” on page 50 if you encounter problems with DBdiff. Since FastXOR uses the DBdiff application, the debugging steps are largely the same.

DBdiff has the following additional applications:


- **Layout versus layout rule file creation** — The `-write_xor_rules` command line option creates a SVRF rule file that can be used for layout versus layout (LVL) comparison. See [Write XOR Rules Command Line Syntax](#).
- **Layout comparison for FastXOR** — FastXOR (`calibre -fx`) uses the DBdiff application to identify the areas that differ between the two input designs and only runs XOR on the regions that differ. See “[Getting Started with FastXOR](#)” on page 26.

Licensing for DBdiff and FastXOR is discussed in the *[Calibre Administrator's Guide](#)*.

Getting Started with DBdiff	64
Layout Comparison Command Line Syntax	79
Write XOR Rules Command Line Syntax	98
Summary of Differences in the ASCII Report File	105
OASIS Format Restrictions	107
LEF/DEF Design Considerations	107
OpenAccess Layer Mapping	109

Getting Started with DBdiff

DBdiff provides a summary of differences between two layout databases. Differences are written to a report file and an optional Calibre results database. The application can also generate XOR rules for use with standard XOR or FastXOR.

<p>Try It!</p> 	<p>Calibre Layout Comparison Tutorial and Example Kit (eKit)</p> <p>The eKit demonstrates basic procedures for performing layout versus layout using standard XOR, FastXOR, and DBdiff, and includes all files and data needed to run the tool.</p> <p>Go to this page on Support Center to download the eKit (Documentation tab, Document Types=Getting Started Guide). The link goes to the latest release.</p>
---	--

This section contains the following topics:

Input and Output	64
Using DBdiff to Perform LVL	65
Comparing Text with DBdiff	67
DBdiff Usage Examples	68

Input and Output

DBdiff accepts GDS and OASIS databases in addition to the supported third-party layout databases.

For information on supported third-party layout databases, view “[Third-Party Database Support](#)” on page 111.

DBdiff text input files that contain blank lines or lines that begin with “#” are ignored. In-line comments are also supported using the “#” character.

The following output options are available:

- An ASCII report describing the changes found.
- A Calibre RVE results database (RDB) describing the changes found.

DBdiff supports 64-bit address space.

Related Topics

[Layout Comparison Command Line Syntax](#)

[Write XOR Rules Command Line Syntax](#)

Using DBdiff to Perform LVL

Use DBdiff to perform fast Layout versus Layout comparisons. DBdiff performs a fast cell-by-cell comparison of shapes, instances, properties (if enabled), and text between two layouts.

The performance advantage of DBdiff over traditional LVL XOR depends on the number of differences between the layouts. DBdiff can be very fast, but it can also consume significantly more memory if the layouts contain large differences. For layouts with minor differences, you can expect DBdiff to finish significantly faster than a traditional XOR.

However, DBdiff must only be used with these considerations in mind:

- The two designs must contain a small number of differences, otherwise the memory overhead is too great. DBdiff should not be used to determine detailed differences between two layouts.
- DBdiff does not indicate the location of a noted differences, it only indicates that the polygons differ between layouts (no mask-level information is provided). Therefore, the results are more difficult to interpret than with FastXOR.
- DBdiff runs in batch mode only.

See [“Differences Between FastXOR and Standard Dual Database Processing”](#) on page 27.

Prerequisites

- Two complete layout databases to be compared. The input layouts should use 32-bit coordinate space.
- If an OpenAccess database is being used as input, see [“Third-Party Database Support”](#) on page 111.
- Sufficient Calibre nmDRC/Calibre nmDRC-H license pairs for the number of CPUs in the run.

Procedure

1. Enter the following command to invoke DBdiff and compare *layout.gds* and *layout_modified.gds*:

```
dbdiff -system GDS -design layout_modified.gds TOPCELL \  
-refdesign layout.gds TOPCELL -report dbdiff.rpt -turbo 4 \  
-comparetext -rdb dbdiff.rdb -sortlayer
```

The full list of command options can be found under the section [“Layout Comparison Command Line Syntax”](#) on page 79.

During the run, the transcript is updated with information about the design and it indicates whether the two designs differ, as shown here:

```
COMPARING DESIGNS

COMPARED CELL NewCell with NewCell(REF DESIGN)
CELLS COMPLETE = 1 OF 2  ELAPSED TIME = 1
COMPARED CELL TOPCELL with TOPCELL(REF DESIGN)
CELLS COMPLETE = 2 OF 2  ELAPSED TIME = 1

Cells in comparison are DIFFERENT
```

2. When the run completes, open the ASCII report file (*dbdiff.report*) in a text editor and view the differences. The report indicates differences between your two layouts in a manner similar to this example:

----- Difference Table -----

Design	Cell Name	Difference	Type	Bounding Box	Layer	Data Type
Current	TOPCELL	New Object	Rectangle	800 1100 900 1700	1	0
Current	TOPCELL	Missing Object	Rectangle	0 0 10 10	1	0

3. Open your layout design (*layout_modified.gds*) in a supported layout viewer.
4. Start Calibre RVE, using **Verification > Start RVE** from Calibre DESIGNrev, or using **Calibre > Start RVE** from other supported layout viewers.
5. In the Calibre RVE dialog box, select **File > Open Database**. Specify the *dbdiff.rdb* file that you generated in Step 1 in the database field and select a database type of DRC/ERC.

The layout differences are shown as checks in Calibre RVE. See “[Using Calibre RVE for DRC](#)” in the *Calibre RVE User’s Manual*.

6. View the differences by highlighting the DBdiff results in Calibre RVE.

Results

DBdiff was used to compare two layouts. The following files were generated by the DBdiff application:

- *dbdiff.report*
- *dbdiff.rdb*

You used the transcript and the DBdiff *dbdiff.report* report file to see whether the two layouts contained differences. You further used Calibre RVE with the generated *dbdiff.rdb* results database to view results in the modified layout.

This procedure can be used to quickly determine if two layouts differ. If DBdiff reports no differences, you have saved the computational expense of performing a full XOR comparison. If DBdiff reports significant differences, or if the DBdiff run consumes large amounts of memory, a traditional XOR run is highly recommended instead of DBdiff.

Comparing Text with DBdiff

DBdiff generates a summary of differences in a text report and a Calibre RVE results database when you apply the `-report` and `-rdb` options, respectively.

Differences generated by DBdiff are always classified as “New” or “Missing” with respect to the *current* database. For example, if you delete a text object from the current design and then run a comparison with the modified current design and the reference design, DBdiff reports a “Missing” text object. Similarly, if you add a text object to the current design and run a comparison to the reference design, DBdiff reports a “New” text object.

DBdiff text comparison considers the text location, layer, and string for all cells in the design. Changes to the text location, layer, and string are reported as “New” or “Missing” text objects, as described in the previous example. Changing a text location in the current design results in a “New” text object at the new location and a “Missing” text object at the original location. Likewise, the layer and string properties result in a “New” and “Missing” text object if they are changed in the current design.

Note



If you use FastXOR to perform layout comparison with text, you must generate a FastXOR rules file with DBdiff and set the `-comparetext` option using *one* of the following methods:

- Include the following statement in your rule file:

```
SVRF DBDIFF OPTIONS "-comparetext -rdb rdb_file -sortlayer"
```

- Set the following environment variable:

```
setenv CALIBRE_FX_DB_DIFF_OPTIONS "-comparetext -rdb rdb_file -sortlayer"
```

Procedure

1. Invoke DBdiff with the `-comparetext` option:

```
dbdiff -system GDS -turbo
      -refdesign ref.gds TOP \
      -design current.gds TOP -automatch -multimatch \
      -comparetext -include_layer include_text_layers.txt \
      -report compare_text.report -rdb compare_text.rdb \
      -sortlayer >& compare_text.log
```

2. Open the *compare_text.report* file in a text editor.

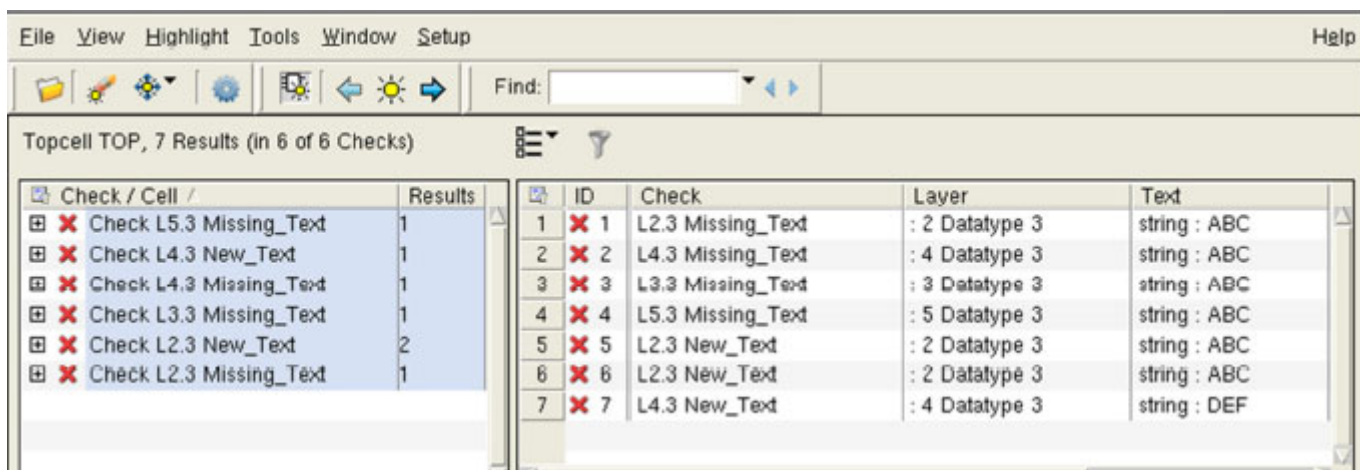
The text report includes a list of differences and runtime information. View the “Text difference summary” section of the report. This includes the number and type of text differences. The following is an excerpt:

```
*****Text difference summary begin*****
Layer      Datatype  NewTexts    MissingTexts  TotalDiffTexts  DesignTexts
2          3         2           1              3                2
4          3         1           1              2                1
3          3         0           1              1                0
5          3         0           1              1                0
*****Text difference summary end*****
```

3. Open your current design and reference design in Calibre DESIGNrev.
4. Open the results database in Calibre RVE:

```
calibre -rve compare_text.rdb
```

The Calibre RVE results database enables you to highlight reported differences in Calibre DESIGNrev. The database results are 2-point polygons that are 1 dbu × 1 dbu in size. The lower-left corner of these polygons is the original location of the text.



5. Highlight each result and refer to the two layouts to understand the text differences between the designs.

DBdiff Usage Examples

The DBdiff invocation syntax depends on the type of library and the information that you want to report.

The following examples demonstrate the syntax for different DBdiff operations.

Table 4-1. DBdiff Usage Examples

Example	Command Line
Compare Two Homogeneous Databases with DBdiff	<code>dbdiff -system GDS -refsystem GDS -design DIGBLK.gds -refdesign DIGBLK2.gds</code>
Compare Two Heterogeneous Databases with DBdiff	<code>dbdiff -system GDS -refsystem OASIS -design DIGBLK.gds -refdesign DIGBLK2.oas</code>
Compare LEF designs with DBdiff	<code>dbdiff -system LEFDEF -lef tech.lef NEW/DAC.lef -reflef tech.lef OLD/DAC.lef -report dbdiff_lef.report -rdb DAC_lef_comparison.rdb</code>
Generate an ASCII Comparison Report with DBdiff	<code>dbdiff -system GDS -design DIGBLK.gds TOPCELL -refdesign DIGBLK2.gds TOPCELL -report digblk.report</code>
Generate a Calibre RVE Results Database (RDB) File with DBdiff	<code>dbdiff -system GDS -design DIGBLK_v2.gds DIGBLK -refdesign DIGBLK_v1.gds DIGBLK -rdb digblk.rdb</code>
Use -automatch with pcells in DBdiff	<code>dbdiff -system GDS -design DIGBLK_v2.gds DIGBLK -refdesign DIGBLK_v1.gds -automatch automatch_file.txt -multimatch</code>
Handling Hierarchy Changes in DBdiff	<code>dbdiff -system GDS -design ip.gds TOP -refdesign chip.gds CHIP -automatch -multimatch -nocompare -report ip_check.report</code>
Use -automatch to identify Third-Party IP in a Database with DBdiff	<code>dbdiff -system GDS -design ip.gds TOP -refdesign chip.gds CHIP -automatch -multimatch -nocompare -report ipcheck.report</code>
Errors when using -write_xor_rules with CURrent and REFeRence in DBdiff	N/A
DBdiff -enhanced option output	N/A
Write XOR Usage Example	<code>dbdiff ... -write_xor_rules</code>

Compare Two Homogeneous Databases with DBdiff

The following example compares two databases together using the DBdiff application:

```
dbdiff -system GDS -design DIGBLK.gds TOPCELL -refdesign DIGBLK2.gds
TOPCELL
```

The DBdiff output shows the two designs are different.

```
// Calibre dbdiff v2013.3_8.0008 Wed Jul 24 13:24:23 PDT 2013
//
// Copyright Mentor Graphics Corporation 2008-2013
// All Rights Reserved.
// THIS WORK CONTAINS TRADE SECRET AND PROPRIETARY INFORMATION
// WHICH IS THE PROPERTY OF MENTOR GRAPHICS CORPORATION
// OR ITS LICENSORS AND IS SUBJECT TO LICENSE TERMS.
//
// Mentor Graphics software executing under x86-64 Linux
// 64 bit virtual addressing enabled

READING DESIGN DIGBLK.gds ELAPSED TIME = 0
READING DESIGN DIGBLK2.gds ELAPSED TIME = 0
COMPARING DESIGNS

COMPARED CELL NewCell with NewCell(REF DESIGN)
CELLS COMPLETE = 1 OF 2 ELAPSED TIME = 1
COMPARED CELL TOPCELL with TOPCELL(REF DESIGN)
CELLS COMPLETE = 2 OF 2 ELAPSED TIME = 1

Cells in comparison are DIFFERENT

Data Read Time    :: 0 Seconds
Compare Time      :: 1 Seconds

***** Summary Report *****

Total Elapsed Time    :: 1 Seconds
Peak Virtual Memory   :: 33.69 MB
```

Compare Two Heterogeneous Databases with DBdiff

The following example compares two different types of databases together using the DBdiff application:

```
dbdiff -system GDS -refsystem OASIS -design DIGBLK.gds -refdesign
DIGBLK2.oas
```

The DBdiff output shows the two heterogeneous designs are different.

```
// Calibre dbdiff v2013.3_8.0008 Wed Jul 24 13:24:23 PDT 2013
//
// Copyright Mentor Graphics Corporation 2008-2013
// All Rights Reserved.
// THIS WORK CONTAINS TRADE SECRET AND PROPRIETARY INFORMATION
// WHICH IS THE PROPERTY OF MENTOR GRAPHICS CORPORATION
// OR ITS LICENSORS AND IS SUBJECT TO LICENSE TERMS.
//
// Mentor Graphics software executing under x86-64 Linux
//
// Command: $MGC_HOME/bin/dbdiff -system GDS -design DIGBLK_6x6.gds
// -refsystem OASIS -refdesign DIGBLK_6x6_rev2.oas

READING DESIGN DIGBLK_6x6.gds ELAPSED TIME = 0
READING DESIGN DIGBLK_6x6_rev2.oas ELAPSED TIME = 3

COMPARING DESIGNS

TOP CELL TOPCELL AND TOPCELL (REF DESIGN) ARE DIFFERENT
MISSING TOP CELL IN CURRENT DESIGN TOP_WB27
MISSING TOP CELL IN CURRENT DESIGN TOP_WB41
MISSING TOP CELL IN CURRENT DESIGN TOP_WB49

DESIGNS IN COMPARISON ARE DIFFERENT

Data Read Time   :: 4 Seconds
Compare Time     :: 2 Seconds

***** Summary Report *****

Total Elapsed Time      :: 6 Seconds
Total Elapsed CPU Time  :: 6 Seconds
Peak Virtual Memory     :: 225.46 MB
```

Compare LEF designs with DBdiff

The following example demonstrates how to compare two LEF designs using DBdiff:

```
dbdiff -system LEFDEF \
-lef tech.lef NEW/DAC.lef \
-reflef tech.lef OLD/DAC.lef \
-report dbdiff_lef.report -rdb DAC_lef_comparison.rdb
```

Note



To achieve the best results, it is recommended to first translate third-party databases to GDS or OASIS before running a comparison.

In this example, new metal shapes are placed on the M5 layer of the new DAC design. The generated report describes the differences in detail:

```
// dbdiff v2014.2_x

Current   Lib Name : tech.lef NEW/DAC.lef
Reference Lib Name : tech.lef OLD/DAC.lef
MGC_HOME   :    /<calibre_install>/
HostName   :    hostname
Time       :    Thu Apr 17 13:01:31 2014

TOP CELL DETECTED IN CURRENT DESIGN : dac6_op2
TOP CELL DETECTED IN REFERENCE DESIGN : dac6_op2

...

In below table * ,if any ,denotes the difference in the instance due to
Master Cell change
----- Difference Table -----
| Design | Cell Name | Difference | Type       | Bounding Box |...
| Current| dac6_op2 | New Object | Rectangle | 71255 76140 161950 79130
| Current| dac6_op2 | New Object | Rectangle | 71255 4665 72755 10640

***** Summary Report*****

*****Shape difference summary begin*****
Layer Datatype NewObjs MissingObjs TotalDiffObjs DesignObjs
RefdesignObjs TotalObjs Change (%)
10 0 2 0 2 2 0
2 100.00
*****Shape difference summary end*****

*****Different cells summary begin*****

CellName ReferenceCellName

dac6_op2 dac6_op2
*****Different cells summary end*****

Command Line Executed : dbdiff -system LEFDEF -lef tech.lef NEW/DAC.lef
-reflef tech.lef OLD/DAC.lef -report dbdiff_lef.report
-rdb DAC_lef_comparison.rdb

Total Elapsed Time :: 1 Seconds
Total Elapsed CPU Time :: 0 Seconds
Peak Virtual Memory :: 61.05 MB
```

Generate an ASCII Comparison Report with DBdiff

The following example generates an ASCII report highlighting the differences between the two layout databases:

```
dbdiff -system GDS -design DIGBLK.gds TOPCELL -refdesign DIGBLK2.gds
TOPCELL -report digblk.report
```


In this example, a new polygon on layer 1 was added to DIGBLK, and a polygon on layer 1 was removed.

```
// dbdiff v2013.3_8.008
```

```
Current   Lib Name :   DIGBLK.gds
Current   Top Cell :   TOPCELL
Reference Lib Name :   DIGBLK2.gds
Reference Top Cell :   TOPCELL
MGC_HOME           :   /mynet/my_dir/Mgc_home
HostName           :   my_host
Time               :   Wed Jul 31 13:52:09 2013
```

In table * ,if any ,denotes the difference in the instance due to Master Cell change

```
----- Difference Table -----
| Design | Cell Name | Difference | Type | Bounding Box
| Layer | DataType |           |      | |
|---|---|---|---|---|
| Current | TOPCELL | New Object | Rectangle | 800 1100 900 1700 |
1 | 0 | | | |
| Current | TOPCELL | Missing Object | Rectangle | 0 0 10 10 |
1 | 0 | | | |
```

```
***** Summary Report*****
```

```
Command Line Executed : dbdiff -system GDS -design DIGBLK.gds TOPCELL
-refdesign DIGBLK2.gds TOPCELL -report /mynet/dir/calibre/digblk.report
```

```
Total Elapsed Time      : 3 Seconds
Total Elapsed CPU Time   : 0 Seconds
Peak Virtual memory: 33.69 MB
```

Generate a Calibre RVE Results Database (RDB) File with DBdiff

The following example shows how to use the DBdiff application to generate a RDB output file digblk.rdb.

```
dbdiff -system GDS -design DIGBLK_v2.gds DIGBLK -refdesign DIGBLK_v1.gds
DIGBLK -rdb digblk.rdb
```

To view the layout differences in Calibre RVE, follow these steps:

1. Open the design being evaluated in a supported layout viewer.
2. Start Calibre RVE, using **Verification > Start RVE** from DESIGNrev, or using **Calibre > Start RVE** from other supported layout viewers.
3. In the Calibre RVE dialog box, select **File > Open Database**. Enter the name of the RDB in the database field and select a database type of DRC/ERC. The layout differences are shown as checks in Calibre RVE. See [“Using Calibre RVE for DRC”](#) in the *Calibre RVE User’s Manual*.

Use -automatch with pcells in DBdiff

If your design contains pcells, the names generated in the GDS file when these pcells are streamed out are cellname\$\$<number>. If you generate two versions of the same design, the number at the end of the pcell name can be different. By default, DBdiff interprets these pcells to be different even though they are geometrically identical.

Follow these steps to use the -automatch option to enable DBdiff to interpret these pcells as being the same:

1. Create the automatch driver file. Create the file automatch_file.txt, and enter the following line:

```
*$$* *$$*
```

This string instructs DBdiff to apply automatching between any cell in the reference design that contains \$\$ and any cell in the current design that contains \$\$.

2. If the cell names have changed between design versions, you may want to add an additional second line to the file as shown:

```
*$$* *$$*
```

```
* *
```

This instructs automatch to first try to automatch any cells that have \$\$ and then to try to match all other cells afterwards.

3. Run DBdiff:

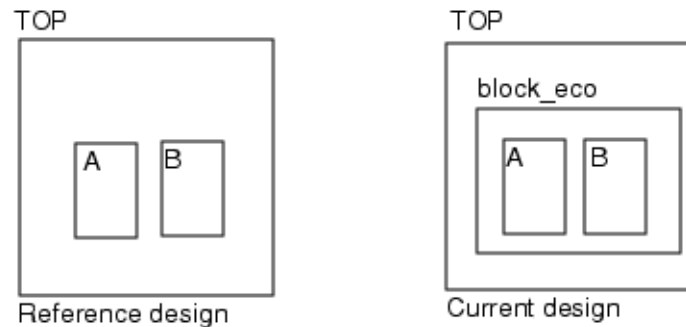
```
dbdiff -system GDS -design DIGBLK_v2.gds DIGBLK -refdesign  
DIGBLK_v1.gds -automatch automatch_file.txt -multimatch
```

Handling Hierarchy Changes in DBdiff

A change in the design hierarchy can result in many differences being reported by DBdiff. Flattening the hierarchy may give a better match between the two designs. You can use the -flattencell option to provide DBdiff with a more equal hierarchy for comparing the two designs.

The following figure illustrates a case when you might want to use the `-flattencell` option:

Figure 4-1. Hierarchy Changes



In [Figure 4-1](#) the cell `block_eco` has been added to the hierarchy in the current design. Without the `-flattencell` option, this design change causes DBdiff to report the entire region of `block_eco` as a difference.

To instruct DBdiff to flatten the cell `block_eco`, add this command line option:

```
-flattencell flatten_file
```

where *flatten_file* is a text file with this line in it:

```
block_eco current
```

Use `-automatch` to identify Third-Party IP in a Database with DBdiff

Follow this process to find a known golden cell or IP cell named `my_ip` in your current design. The current design is named `chip.gds` with a top-level cell name of `CHIP`.

1. Create a new layout database named `ip.gds` and instantiate the cell `my_ip` in the top level.
2. Run DBdiff with the `-automatch` and `-nocompare` options:

```
dbdiff -system GDS -design ip.gds TOP \
-refdesign chip.gds CHIP -automatch -multimatch -nocompare \
-report ip_check.report
```

In the transcript, you should see the automatching information about matching cells:

```
...
AUTOMATCHED CELLS my_ip TO cell_1
AUTOMATCHED CELLS my_ip TO cell_2
GEOMETRICALLY EQUIVALENT CELL SETS IN CURRENT DESIGN
SET 1 (
cell_1
cell_2
)
...
```

Or, you can find the same information in the report file:

```
...
Automatching Report Start:
----- AutoMatched Cell Table -----
Current design cellname | Reference design cellname
my_ip | cell_1
my_ip | cell_2
GEOMETRICAL EQUIVALENT CELLS IN CURRENT DESIGN
SET 1 (
cell_1
cell_2
) Automatching report End ...
```

This indicates that DBdiff found two geometric matches (cell_1 and cell_2) for my_ip in chip.gds.

You can use the `-include_layer` option to do the same comparison but focus on specific layers such as poly or diffusion. The cell matching process takes place only on the layers specified in the `-include_layer` list.

Errors when using `-write_xor_rules` with CURrent and REFerence in DBdiff

If the layer map file contains this:

```
1 0 A CUR
2 0 A CUR
3 0 A CUR
6 0 A REF
5 2 C REF
10 2 met1
```

Then an error is generated for layer number 10. If one of the lines uses the CUR or REF keyword, then all of them must.

If the final line of the layer map file is removed, then the generated rule file would be this:

```
LAYER MAP 1 DATATYPE 0 65535
LAYER MAP 2 DATATYPE 0 65534
LAYER MAP 3 DATATYPE 0 65533
LAYER MAP 106 DATATYPE 0 65531
LAYER A 65535 65534 65533
LAYER B_A 65531
DRC CHECK MAP A GDSII 1 0 "rules.gds"
DRC CHECK MAP A ASCII
A { @compare layer A
    XOR A B_A
}
```

A warning is issued for layer number 5. Layer C is specified as a REF layer, but not a CUR layer.

DBdiff -enhanced option output

Here is an example output when the -enhanced option is used. Notice the inclusion of the NOT operations.

```
LAYER MAP 1 DATATYPE 0 65535
LAYER MAP 16 DATATYPE 0 65534
LAYER L1_D0 65535
LAYER B_L1_D0 65534
DRC CHECK MAP L1_D0 GDSII 1 0 "rules.gds"
DRC CHECK MAP L1_D0 ASCII
L1_D0 { @ compare layer 1 datatype 0
      XOR L1_D0 B_L1_D0
}
//=====
DRC CHECK MAP L1_D0_NEW_SHAPES GDSII 65535 0 "rules.gds"
DRC CHECK MAP L1_D0_NEW_SHAPES ASCII
L1_D0_NEW_SHAPES { @ New shapes in layout path on layer 1 datatype 0
L1_D0 NOT B_L1_D0
}
//=====
DRC CHECK MAP L1_D0_MISSING_SHAPES GDSII 65534 0 "rules.gds"
DRC CHECK MAP L1_D0_MISSING_SHAPES ASCII
L1_D0_MISSING_SHAPES { @ missing shapes in layout path on layer 1 datatype
0
B_L1_D0 NOT L1_D0
}
//=====
```

Write XOR Usage Example

The rule file created with the -write_xor_rules argument includes the following SVRF specification statements:

```
LAYOUT ERROR ON INPUT NO
DRC KEEP EMPTY NO
DRC CELL NAME YES CELL SPACE XFORM
DRC MAXIMUM RESULTS ALL
LAYOUT USE DATABASE PRECISION YES
LAYOUT PROCESS BOX RECORD YES
LAYOUT INPUT EXCEPTION SEVERITY DATATYPE_MAP_TARGET 2
```

If the -include_rules option is used, then these statements are written to the output rule file:

```
DRC CELL NAME YES CELL SPACE XFORM
DRC MAXIMUM RESULTS ALL
LAYOUT PROCESS BOX RECORD YES
LAYOUT INPUT EXCEPTION SEVERITY DATATYPE_MAP_TARGET 2
```

in addition to the statements specified in the -include_rules file.

For additional information on the generated rule file, see the section [“Generating the LVL Comparison Rule File”](#) on page 28.

Related Topics

[DBdiff Messages](#)

[Write XOR Rules Command Line Syntax](#)

[Input and Output](#)

Layout Comparison Command Line Syntax

The DBdiff application compares databases of different types. This section covers the syntax and arguments for DBdiff.

Usage

```
dbdiff -system { GDS | OASIS | LEFDEF | { OA [version] } }  
  { { -design { library | filename } [cell] [view] } |
```

{ **-lef** *filename* [*filename ...*] [-def *filename*] [*cell*] }

{ { **-refdesign** { *library* | *filename* } [*cell*] [*view*] } |

{ **-reflef** *filename* [*filename ...*] [-refdef *filename*] [*cell*] }

[**-refsystem** { **GDS** | **OASIS** | **LEFDEF** | **OA** }]
[**-automatch** [*filename*] [-multimatch] [-nocompare]]
[**-cellmap** *filename*]
[**-checkcell** *filename*]
[**-compareallplacedcells**]
[**-comparearefsassrefs**]
[**-comparemergeddiffshape**]
[**-compareproperties** [*property1 property2 ...*]]
[**-compareshapesaspolygons**]
[**-comparetext**]
[**-comparezerowidthpaths**]
[**-comparezerowidthpolygons**]
[**-exclude_instance**]
[**-exclude_cell** *filename*]
[**-exclude_layer** *filename*]
[**-flattencell** *filename*]
[**-h**]
[**-hierarchyonly**]
[**-ignoreemptycells**]
[**-ignoreboxrecords**]
[**-include_layer** *filename*]
[**-layermap** *layermap_file*]
[**-max_cell_diff_count** *N*]
[**-maxdiff** { *N* | auto }]
[**-nowait**]
[**-optimizerepetitions**]
[**-passthroughlayer**]
[**-rdb** *filename* [**FC** | **FP** | **FT**] [-sortlayer]]
[**-removeduplicateinsts**]
[**-report** *filename*]
[**-template** *filename*]
[**-transform** { [auto_offset | *x_offset y_offset*] [**R0** | **R90** | **R180** | **R270**] [**MX** | **MY**]}]
[**-turbo** *N* [-turbo_all]]

[\[-version\]](#)
[\[-wait *N*\]](#)

Arguments

- **-system** { **GDS** | **OASIS** | **LEFDEF** | { **OA** [*version*] } }

A required parameter set that specifies the input database framework. The options correspond to the database formats of the same name or acronym. Here are some special notes.

OASIS — See the section “[OASIS Format Restrictions](#)” on page 107 for a list of constructs that are not supported.

LEFDEF — LEF/DEF design. When this option is used, the **-lef** and optional **-def** keywords must be used instead of **-design**.

OA [*version*] — The default OpenAccess version is 22.43. The following OA versions are supported:

- 22.43
- 22.50
- 22.60

Note



You can also specify the OpenAccess version by setting the `MGC_FDI_OA_VERSION` environment variable to one of the supported OA version numbers. The command line version takes precedence over the environment variable value.

DBdiff ignores zero-length paths and zero-area polygons in GDS and OASIS format input designs unless the [-comparezerowidthpaths](#) or [-comparezerowidthpolygons](#) are applied. DBdiff prints a warning when such structures are found.

- **-design** { *filename* | *library* } [*cell*] [*view*]

Parameter set that specifies a current design to compare to a reference design. This parameter set is not used if the design is LEF/DEF.

The arguments are defined as follows:

filename* | *library — Specifies the current design as follows, depending on your layout system:

GDS or OASIS — The GDS or OASIS filename.

OpenAccess — The library name.

cell — The primary cell of the layout.

If the layout system is GDS or OASIS, the *cell* parameter is optional. When no cell is specified, DBdiff attempts to locate all top-level cells present in the design and compares those cells. If no top cells are present, DBdiff compares all cells in the libraries.

view — An optional parameter used for OpenAccess databases. If *view* is not specified, “layout” is the default.

- **-lef *filename*** [*filename ...*] [-def *filename*] [*cell*]

Parameter set that specifies a LEF-only or LEF/DEF design to compare to a reference design.

The arguments are defined as follows:

-lef *filename* [*filename ...*] — Specifies the LEF file(s). The files may be flat files or directories. The first *filename* must either be a technology LEF file or a directory that contains it. If it is a directory, the technology file must be lexicographically first in the directory. The LEF files in a directory are expected to have the extension .lef. If more than one file or directory is specified, then they are read in the order they appear on the command line. Flat files take precedence over files of the same name that appear in a directory.

-def *filename* — Optional DEF file. If this is not specified, comparison is performed on LEF files only.

cell — Optional parameter used with **-def** that specifies the primary cell name.

Note



LEF-only systems cannot be compared to a -reflef that includes a DEF file.

- **-refdesign {*filename* | *library*}** [*cell*] [*view*]

Parameter set that specifies the original (reference) design for comparison with the current design. This parameter set is not used if the design is LEF/DEF.

The arguments are defined as follows:

filename* | *library — Specifies the current design as follows, depending on your layout system:

GDS or OASIS — The GDS or OASIS filename.

OpenAccess — The library name.

cell — The primary cell of the layout.

If the layout system is GDS or OASIS, the *cell* parameter is optional. When no cell is specified, DBdiff attempts to locate all top-level cells present in the design and compares those cells. If no top cells are present, DBdiff compares all cells in the libraries.

view — An optional parameter used for OpenAccess databases. If *view* is not specified, “layout” is the default.

- **-reflef *filename*** [*filename ...*] [-refdef *filename*] [*cell*]

Parameter set that specifies a LEF-only or LEF/DEF reference design to compare to the **-lef** design.

The arguments are defined as follows:

-reflef *filename* [*filename* ...] — Specifies the LEF file(s). The semantics are identical to the **-lef** parameter set. The **-reflef** parameter set may be omitted if both **-system** and **-refsystem** are **LEFDEF**.

-refdef *filename* — Optional DEF file.

cell — Optional parameter used with **-def** that specifies the primary cell name.

- **-refsystem** {GDS | OASIS | LEFDEF | OA}

An optional parameter set that specifies the format of the reference design. This parameter set must be specified when the two input designs use different formats. The descriptions for the allowed formats are the same as for **-system**. If LEFDEF is specified, then **-reflef** and optionally **-refdef** must be used instead of **-refdesign**.

- **-automatch** [*filename*] [-multimatch] [-nocompare]

An optional parameter set that instructs DBdiff to search for exact geometric matches between current design cells and previous design cells based upon the patterns in the optional *filename*.

If *filename* is not specified, then the automatch process attempts to match all cells present in the top-level cell of the current design. This option is useful when some cells of the current design are renamed and you have no information about the name mapping.

By default, the automatch process stops after the first matching cell is found. This behavior is changed with the **-multimatch** option.

filename — An optional parameter that specifies the path to an ASCII driver file. The driver file contains custom cell naming patterns used during the automatch process.

The file requires two entries per line, is case-sensitive, and allows the use of the wildcard character (*), which matches zero or more characters. For example:

```
cut* via*
genvia* via*
```

These lines instruct DBdiff to try to match all cells in the current design starting with “cut” or “genvia” to cells in the reference design starting with “via”.

-multimatch — An optional argument that enables multiple matching for a cell. If specified, cells in the current design are matched to all equivalent cells in the **-refdesign** file. Without this option, the automatch process stops at the first matched cell. If **-system LEFDEF** is specified, then this option is enabled automatically.

The **-multimatch** option also outputs the geometrically equivalent sets that are found in each of the designs.

-nocompare — An optional argument that causes processing to stop after the automatch (with optional multimatch) process is complete. This option is useful if you are only interested in identifying cells that match. You cannot specify **-rdb** with **-nocompare**. See [“Use -automatch to identify Third-Party IP in a Database with DBdiff”](#) on page 75.

If the `-cellmap` option is also specified with the `-automatch` option, priority is given to the entries in the `-cellmap` file. In this case, automatching is not applied to cell names that appear in the cell mapping file.

If the `-comparetext` option is specified with the `-automatch` option, DBdiff considers text matching during the automatch operation.

When used with the `-report` option, the automatched cells are reported as shown in this excerpt:

```
----- AutoMatched Cell Table -----
Current design cellname | Reference design cellname
ML_M1_C$$41077804 | ML_M1_C$$36753452
ML_M1_C$$41080876 | ML_M1_C$$36754476
```

When the options `-multimatch` and `-report` are both specified, the geometrically equivalent sets are reported as shown in this excerpt:

```
GEOMETRICAL EQUIVALENT CELLS IN REFERENCE DESIGN

SET 1 (
M1_M0_C$$33505324
M1_M0_C$$42710060
)

GEOMETRICAL EQUIVALENT CELLS IN CURRENT DESIGN

SET 1 (
ML_M1_C$$33499180
ML_M1_C$$36752428
)

SET 2 (
ML_M1_C$$36753452
ML_M1_C$$36754476
ML_M1_C$$36756524
)
```

Also see [Use -automatch with pcells in DBdiff](#).

- `-cellmap filename`

An optional parameter set that specifies the path of a cell mapping file. This option allows differently-named cells to be compared. This file contains the new name (current) and old name (reference) for the cells that are renamed in the current design. When `-cellmap` is not specified, cells with different names are not compared, and instances of those cells are treated as though they are unique.

The new name and old name are separated by whitespace, for example:

```
...
Newnor2x1 nor2x1
Newxnor2x1 xnor2x1
via via_eco
and and_eco
```

- `-checkcell filename`

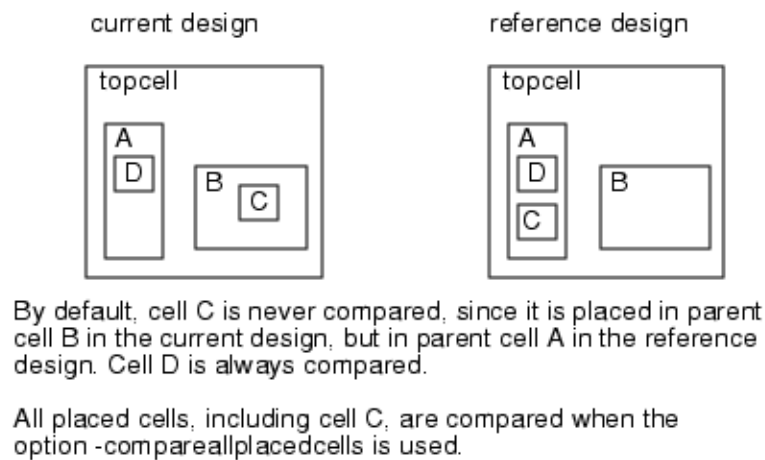
An optional parameter set that specifies the path of a cell checking file. This file contains the list of cells to check for differences. When this option is used, DBdiff checks only the specified cells, which speeds up the comparison process. The cell names appear in a single column. This option is mutually exclusive with the `-exclude_cell`, `-automatch`, and `-multimatch` options.

- `-compareallplacedcells`

An optional argument that instructs DBdiff to compare all cells that are referenced in the primary cells in both designs.

By default, if a child cell exists in both designs but is placed in different parent cells, the child cell is reported as a missing or new instance in the parent cell, but the child cell is not compared. This is shown in [Figure 4-2](#).

Figure 4-2. Comparison with `-compareallplacedcells`



- `-comparearefsassrefs`

An optional argument that instructs DBdiff to break AREFs into individual SREFs and then compares the resulting instances. For example, a 2×2 AREF is broken into four SREFs and then compared. This is useful when AREF or SREF conversion has occurred.

For `-system OASIS`, the option `-comparearefsassrefs` is always enabled.

- `-comparemergeddiffshape`

An optional argument that causes overlapping shapes to be merged and compared. This is useful when a shape in one design consists of multiple polygons in the other design. See [“Compare Differences as Merged Shapes”](#) on page 95 for complete details.

- `-compareproperties`

An optional argument that specifies to compare property records in the databases.

When -compareproperties is specified, the following appears in the DBdiff report (formatted for page width):

```
----- Property Difference Table -----
| PROPERTY_OWNER | Design | Cell Name | Type | Bounding Box | Master Name | Layer | DataType |
... | PROPERTY_DIFFERENCE | Type | Property Name | Property Value | [Property Value Reference] |
```

Here is an example of the report output:

```
|PROPERTY_OWNER|dffrx1|Property|Polygon|0 6580 18400 9350| |8|0|
|PROPERTY_DIFFERENCE| |New|"MyProperty1"|"MyVal11"|
|PROPERTY_DIFFERENCE| |Missing|"MyProperty2"|"MyVal21"|
|PROPERTY_DIFFERENCE| |Difference|"MyProperty3"|"MyVal31"|"MyVal32"|
```

A “New” property is one that is in the current database. A “Missing” property is one that is in the reference database but not the current database. A “Difference” occurs when the property is in both databases but the values differ.

The following are examples of property differences as a result of this comparison option:

- An object in layout1 has a property name, but the matched object in layout2 does not.
- An object in layout1 has a property name “abc” but the matched object in layout2 has a property name “def”.
- An object in layout1 has a property name “abc” with value “1”, but the matched object in layout2 has a property name “abc” with a value of “0”.

Here is an example of the property difference summary table:

```
*****Shape property difference summary begin*****

Layer      Datatype  NewProps      MissingProps  ChangedProps  Total
8           0         1             1             1             3
12          0         0             1             1             2
13          0         1             1             1             3

*****Shape property difference summary end*****

*****Instance property difference summary begin*****

CellName  RefCellName  NewProps      MissingProps  ChangedProps  Total
aoi33x1   aoi33x1      2             1             3             6

*****Instance property difference summary end*****
```

Property difference reporting in the RDB file is similar to the reporting for other objects. The differences are shown as property differences. Here is an RDB example:

```
Property_Difference
1 1 0
p 1 4
CN MyCell c 1 0 0 1 0 0
New Property Name and Value: name1, value1
Missing Property Name and Value: name2, value2
Changed Property Name and Value: name3, value3, valu4
-1153 4474
-1152 4473
-1143 4482
-1144 4483
```

- **-compareshapaspolygons**

An optional argument that specifies DBdiff interprets rectangles, paths, and polygons as equivalent, and polygon comparison is performed. When **-compareshapaspolygons** is specified, DBdiff reports the rectangles and paths objects as their original object types and not as polygons in the report output.

- **-comparetext**

An optional argument that enables text comparison. DBdiff performs a comparison of the text string, location, and the layer on which the text occurs. See [“Comparing Text with DBdiff”](#) on page 67 for an example.

Differences from the comparison operation are output to the files specified by the **-report** and **-rdb** arguments. Polygons of 1×1 dbu are created in the RDB file to show locations where text was modified. The text string is also generated in the RDB file. This text is viewable in the Calibre RVE property window. For example, the following information is viewable in the Calibre RVE property window when a check result is selected.

```
1)Check missing_text, Cell TOPCELL:2-Vertex Polygon
String PWR
Status Missing
Layer 3
Datatype 0
328.500 307.700
328.501 307.701

1)Check new_text, Cell TOPCELL:2-Vertex Polygon
String GND
Status New
Layer 3
Datatype 0
328.500 307.700
328.501 307.701
```

If the **-report** option is specified, a new column is added in the difference table to display the text string value and the text location is printed in the bounding box column.

When the `-sortlayer` option is specified along with the `-rdb` option for `-comparetext`, text differences are generated on a sorted layer basis in order to view the generated differences in Calibre RVE by layer.

- `-comparezerowidthpaths`

An optional argument that enables zero-width path comparison. Paths with zero widths are ignored by default.

- `-comparezerowidthpolygons`

An optional argument that enables zero-width polygon comparison. Polygons with zero widths are ignored by default.

- `-exclude_cell filename`

An optional parameter set that specifies only those cells that are not listed in the *filename* are compared. The filename contains the cell names in a single column, such as in this example:

```
inv_x1
inv_x2
inv_A
```

- `-exclude_instance`

An optional argument that instructs DBdiff to exclude instance differences from the report. Only shapes are compared.

- `-exclude_layer filename`

An optional parameter set that specifies only those layers that are not listed in the *filename* are compared. The *filename* contains lines of this format:

```
layer_constraint datatype_constraint
```

The *layer_constraint* and *datatype_constraint* use the same syntax as the constraints in the [Layer Map](#) SVRF specification statement, but whitespace in a constraint string may not be used.

The *filename* is a path to a file that contains a list of layers and datatypes. Layers that meet the constraints are excluded from the run. If no datatype is given, a datatype of 0 is assumed.

Here is an example of an exclude file:

```
2 0
3 0
4 1
5 >2<5
```

DBdiff compares shapes on all layers of the design except for layer 2 datatype 0, layer 3 datatype 0, and layer 4 datatype 1. The final line with constraint notation indicates layer 5 datatypes 3 and 4 are also excluded.

The `-exclude_layer` argument may not be specified with `-include_layer`.

- `-flattencell filename`

An optional parameter set that specifies the path of the file containing a list of cells that are flattened. The cells are specified as:

cellname [*design*]

cellname — The name of the cell. The wildcard character (*) may be used to match any number of characters.

design — Optionally specifies the design using one of the following keywords:

BOTH — Both designs. This is the default value.

CURRENT — The design specified by **-design**.

REFERENCE — The design specified by **-refdesign**.

For example:

```
NewCell* both
oldCell reference
```

See “[Handling Hierarchy Changes in DBdiff](#)” on page 74 for an illustration and example.

- `-h`

An optional argument that specifies to display all available DBdiff options.

- `-hierarchyonly`

An optional argument that specifies to generate only hierarchical differences. Shape differences in cells are not reported.

- `-ignoreemptycells`

An optional argument that specifies to ignore instances of a cell when calculating differences if the cell is empty in both designs. Empty cells are reported in the summary section of the report file.

DBdiff ignores zero-length paths and zero-area polygons in GDS and OASIS format input designs unless the [-comparezerowidthpaths](#) or [-comparezerowidthpolygons](#) are applied.

- `-ignoreboxrecords`

An optional argument that specifies to ignore BOX and BOXTYPE records in the GDS input.

Note



It is not recommended to specify this option for FastXOR runs because the DBdiff option value can be different from the Layout Process Box Record SVRF statement that is applied during the Calibre portion of the FastXOR run.

- `-include_layer filename`

An optional parameter set that specifies only those layers that are listed in the *filename* are compared. The *filename* contains lines of this format:

```
layer_constraint datatype_constraint
```

The *layer_constraint* and *datatype_constraint* use the same syntax as the constraints in the [Layer Map](#) SVRF specification statement, but whitespace in a constraint string may not be used.

The *filename* is a path to a file that contains a list of layers and datatypes. Layers that meet the constraints are included in the run. If no datatype is given, a datatype of 0 is assumed.

Here is an example of an include file:

```
2 0
3 0
4 1
5 >2<5
```

DBdiff only compares shapes on layer 2 datatype 0, layer 3 datatype 0, and layer 4 datatype 1. The final line with constraint notation indicates layer 5 datatypes 3 and 4 are also compared.

The `-exclude_layer` argument may not be specified with `-include_layer`.

- `-layermap layermap_file`

An optional parameter set that specifies a layer mapping file. The layer mapping file maps database layer numbers and datatypes to layer names. When `-layermap` is specified, the layer names are output instead of the database layers. If a database layer cannot be mapped to an entry in the rule file, that layer is output as-is.

DBdiff supports two types of layer maps, a GDS or OASIS format and a Place and Route format. If your database is a third-party format, it is recommended to first translate your design to GDS or OASIS using the `fdi2gds` or `fdi2oasis` utilities.

The GDS or OASIS layer map file has the following columns: layer number constraint, datatype number constraint, and layer name. The constraints use a similar syntax to the constraints in the [Layer Map](#) SVRF specification statement; however, whitespace between the characters within a constraint specification is not allowed in the `-layermap` format. Here is an example:

Figure 4-3. Layermap Format for GDS and OASIS

#Layer Number	Datatype	Layer Name
>2<=5	>0<4	MET1
>5<7	<5	MET2
8	0	MET3

The layers in the specified layer and datatype ranges are assigned the names MET1, MET2, and MET3, respectively, in the output report or rule file. If multiple layers are mapped to the same name, then that name is used for all the mapped layers.

The Place and Route format has the following columns: layer name, purpose name, layer number, and datatype. For example:

Figure 4-4. Layermap Format for Place and Route (OpenAccess)

#Layer Name	Purpose Name	Layer Number	Datatype
METAL1	DRAWING	1	0
METAL2	DRAWING	2	0
METAL3	DRAWING	5	0

The layers 1.0, 2.0, and 5.0 are assigned the names METAL1.DRAWING, METAL2.DRAWING, and METAL3.DRAWING respectively in the ASCII report and RDB output.

All column values are required for the place and route layer map format. This format cannot be combined with the GDS or OASIS column format. If any line contains more than four columns and `-write_xor_rules` is not specified, then any remaining columns are ignored.

- `-max_cell_diff_count N`

An optional parameter set that specifies a per-cell limit for the number of listed violations in DBdiff. *N* represents the maximum number of listed violations. All types of differences are considered. These differences include missing shapes, new shapes, text on all layers, missing instances, and new instances. The following warning is issued in the ASCII report and RDB output.

```
WARNING: Maximum difference count of N exceeded in cell <cellname>.
```

DBdiff also outputs the following note in the ASCII report.

```
NOTE: Maximum difference count of N exceeded in cell <cellname>.
```

- `-maxdiff {N | auto}`

An optional parameter set that specifies to automatically stop a FastXOR run and restart it as a standard XOR run when a certain number of total differences is exceeded. In cases where the number of differences between two designs is large, it is more efficient to enable a maximum difference threshold to switch from FastXOR mode to standard XOR mode automatically. You can specify a number manually or enable the tool to choose automatically. The arguments behave as follows:

- *N* — An integer that represents the maximum number of differences between two designs.
- `auto` — A keyword that instructs the tool to set an internal default for the maximum number of differences.

Apply this option before a FastXOR run using the SVRF DBDiff Options statement in your XOR rule file, for example:

```
SVRF DBDIFF OPTIONS "-maxdiff 5.0e+06"
```

You can also set this option with the CALIBRE_FX_DB_DIFF_OPTIONS environment variable as follows:

```
setenv CALIBRE_FX_DB_DIFF_OPTIONS "-maxdiff 5.0e+06"
```

In this example, the FastXOR run halts at five million violations and restarts the run as a standard XOR comparison.

See also “[Management of Large Numbers of Differences](#)” on page 44.

- -nowait

An optional argument that causes DBdiff not to wait to acquire substitute licenses. This option is equivalent to specifying “-wait 0”.

- -optimizerepetitions

An optional parameter that handles degenerate repetitions in GDS and OASIS files to improve performance. This option removes repeated array structures from the comparison if one or more identical array objects that are placed at the same coordinates. This option is enabled by default in the Fast XOR template.

- -passthroughlayer

An optional argument that instructs DBdiff to ignore exclusive layers. An exclusive layer is a layer that exists in one layout but not the other. If there is at least one unique layer (deleted or added) between compared designs, consider setting this option to improve performance. If you do not set this option, DBdiff counts all objects on the layer as different.

Note



If you specify -comparetext and -passthroughlayer, DBdiff does not ignore text objects on the layer, but it does ignore all other objects.

- -rdb *filename* [FC | FP | FT] [-sortlayer]

An optional parameter set that specifies RDB output is written to *filename*. You can view RDB results and cross-probe between the reference and current designs using Calibre RVE. This argument can be used without specifying a primary cell name in the DBdiff invocation and supports designs that contain multiple top-level cells. If multiple top cells are detected and no top cells are specified, the results are generated within the context of each cell.

If there are no differences between two designs, an empty RDB file is generated.

The FC, FP, and FT keywords are filtering options that control which items appear in the results database. One or more of the keywords may be applied. They are defined as follows:

FC — filter that excludes differences in cells from the results database.

FP — filter that excludes differences in polygons from the results database.

FT — filter that excludes differences in text from the results database.

For example, if you do not wish to see results from text differences between two GDS layouts, the FT option should be applied as follows:

```
dbdiff -system GDS -design 2.gds -refdesign 1.gds -rdb  
out_notext.rdb FT
```

The -sortlayer option specifies to generate output on a sorted layer basis in order to view the differences in Calibre RVE by layer.

See “[Generate a Calibre RVE Results Database \(RDB\) File with DBdiff](#)” on page 73.

All design differences are written in the RDB format as the following checks:

- Missing shapes
- New shapes
- Missing instances
- New instances
- Missing text
- New text

The DBdiff application accounts for shape differences by adding the following properties to the RDB:

- CN *cellName*: Indicates the cell in which the shape difference exists.
- LN *layerNo layerName*: Indicates the layer number and layer name in which the shape difference exists.
- NN *netName*: Indicates the net name of the differing shape. This output is generated only when the net name information is present in the database. The NN property is supported for OpenAccess databases only.

- -removeduplicateinsts

An optional parameter that handles duplicate cell references in OASIS files to improve performance for files that contain large numbers of duplicate references. This option removes repeated cell instances in OASIS files if one or more identical cells are placed at the same coordinates. This option must only be applied for input layouts that contain large numbers of duplicate references, otherwise performance can be impacted. This option only handles single cell references and rectilinear OASIS arrays (the OASIS format supports eleven types of arrays, where seven of them are not rectilinear and are not handled by this option).

- `-report filename`

An optional parameter set that specifies the file to which the difference report is written. By default, the report is written to STDOUT. This report is an ASCII file that lists all of the differences between the two databases including automatched cells. Information in the report file also includes compared top-level cells in each design if a primary cell is not specified. Bounding boxes of differences are reported with respect to the parent cell. See [“Generate an ASCII Comparison Report with DBdiff”](#) on page 72.

- `-template filename`

An optional parameter set that specifies a file containing the command line options for DBdiff. Environment variables can be used in the *filename* parameter.

The file can contain any or all of the DBdiff command line options. Options specified on the command line take precedence over the template file if there is a conflict. The following requirements apply to the format of the template file:

- Only one option can be entered per line.
- An option and parameter must be separated by whitespace.
- Lines beginning with '#' are ignored.

The following example shows a segment of a template file:

```
# this is a comment
-system GDS
-design designA TOP
-refdesign designB TOP
-automatch ./automatch.txt -multimatch
-cellmap ./cellmap.txt
-rdb out.rdb
-report out.rep
```

In this example, the `-cellmap` option forces a cell equivalence between two cells whose names differ, based on the contents of the *cellmap.txt* file, and compares them. The `-automatch` option uses the input name patterns in the *automatch.txt* file to try to geometrically match cells in each design. The `-multimatch` option enables matching of multiple geometrically-equivalent cells based on the input name patterns.

- `-transform {[auto_offset | x_offset y_offset] [R0 | R90 | R180 | R270] [MX | MY]}`

An optional parameter set that specifies transformations to apply to the **-design** or **-lef** and **-def** layout. The transformation is performed at the primary cell level. At least one of these transformations must be specified:

- `auto_offset` — Specifies to automatically adjust the current and reference layouts to use the same origin during a comparison. To use this option in DBdiff, you must also specify the top cells for both the current and reference layout. If the top cells are not specified, an error message is printed. The top cell restriction does not apply to FastXOR. The `-turbo` option is not supported when you apply this option. If you apply `-turbo` and `-transform auto_offset`, the `-turbo` argument is ignored and a warning is printed.

- *x_offset y_offset* — Specifies a translation in the x- and y-directions, respectively. The arguments must be floating-point numbers in user units.
- R0 | R90 | R180 | R270 — Specifies a counter-clockwise rotation in degrees corresponding to the integer after the R. Only one rotation may be specified.
- MX — Specifies to reflect (mirror) across the horizontal axis. May not be specified with MY.
- MY — Specifies to reflect (mirror) across the vertical axis. May not be specified with MX.

If more than one transformation is specified, the transformations are applied in this order: reflection, rotation, then offset.

- -turbo [*N*]

An optional parameter set that causes DBdiff to read the current design and the reference design in parallel. The optional *N* argument specifies the number of processors to acquire for the DBdiff run.

This option is only supported for GDS and OASIS design formats.

- -turbo_all

An optional parameter that stops the run if the tool cannot obtain the exact number of CPUs specified with the -turbo option.

- -version

An optional argument that prints the version of the DBdiff application.

- -wait *N*

An optional parameter set that specifies the maximum amount of time in minutes for DBdiff to queue for a license. If the license is unavailable after queuing for the specified time, DBdiff attempts to acquire any substitute licenses or exits if no suitable substitutions are defined. The maximum value for *N* is 45000 minutes.

Refer to the section “[Using License Substitution With Calibre Classic Licensing](#)” in the *Calibre Administrator’s Guide* for more information on license substitution.

In the following example, the -wait option queues on a license for five minutes:

```
dbdiff -system GDS -design d1.gds -refdesign d2.gds -wait 5
```

If a license does not become available within five minutes, the application exits with the following message:

```
// License request failed. The maximum queue duration has elapsed.
```

- -write_xor_rules *rule_file* [*prefix*]

See “[Write XOR Rules Command Line Syntax](#)” on page 98.

Description

DBdiff has two syntaxes. This section describes the syntax for layout comparison. The syntax used for XOR rule file generation is discussed under “[Write XOR Rules Command Line Syntax](#)” on page 98.

DBdiff ignores zero-length paths and zero-area polygons in GDS and OASIS format input designs unless you use the `-comparezerowidthpath` and `-comparezerowidthpolygons` options. DBdiff prints a warning when such structures are found.

The DBdiff application uses a tolerance of 0.001 degrees when comparing rotation of instances. This means that angles that fall around 0, 90, 180, and 270 degrees within the given tolerance are treated as 0, 90, 180, and 270 degrees. Tolerance can be reset for GDS databases by using the environment variable `MGC_DBDIFF_GDSREAD_FLOAT_PRECISION`. Setting `MGC_DBDIFF_GDSREAD_FLOAT_PRECISION` to 4 sets the tolerance to 0.0001, while setting the environment variable to 5 specifies a tolerance value of 0.00001.

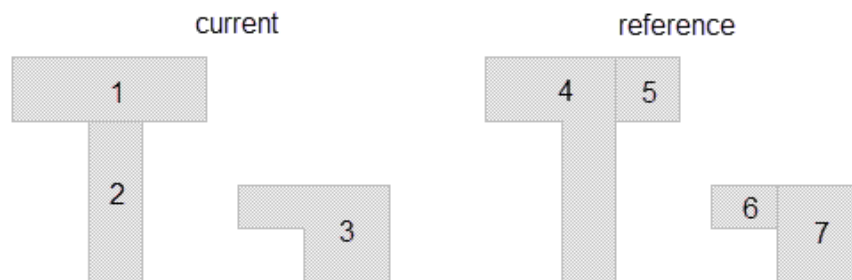
The DBdiff application treats an AREF of 1×1 and an SREF as similar types by default. If `MGC_DBDIFF_STRICT_DIFF` is set to 1 then these types are considered as different.

By default, DBdiff prints a maximum of 20 warnings per message (for each design read) and then suppresses further warnings after printing a message that warnings are being suppressed. You can change the default behavior and print all warnings by unsetting the `FDI_DBDIFF_SUPPRESS_VALUES` environment variable and setting `CALIBRE_DBDIFF_WARNING_ALL` to 1.

Compare Differences as Merged Shapes

When compared databases contain unmerged shapes that are identical from the merged point of view, DBdiff reports any differences by default. This figure illustrates the situation:

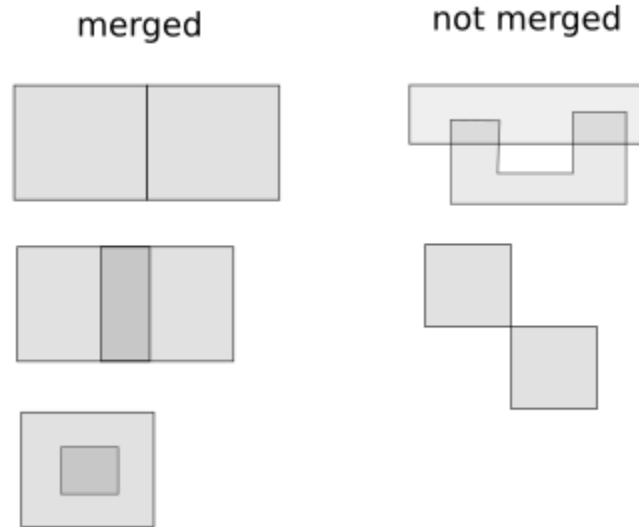
Figure 4-5. Comparing Unmerged Shapes with DBdiff



All of the shapes in the preceding figure are different in the unmerged point of view. By default, shapes 1 through 3 in the current design are reported as new, while shapes 4 through 7 in the reference design are reported as missing.

The `-comparemergeddiffshape` option instructs DBdiff to merge difference shapes per layer when comparing layout databases. The next figure illustrates configurations that are merged and not merged when `-comparemergeddiffshape` is used.

Figure 4-6. `-comparemergeddiffshape` Behavior



Shapes that abut or overlap without forming a hole are merged (with certain exceptions discussed later). Shapes that form a hole or share only a single point are not merged.

If `-comparemergeddiffshape` is used, then the shapes in [Figure 4-5](#) are identical after merging, and no difference is reported.

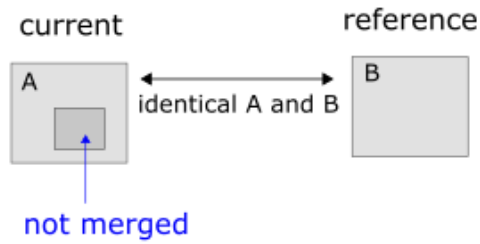
These are the details of the merging process for the `-comparemergeddiffshape` option:

1. Compare shapes and construct initial difference shapes.
2. Convert difference shapes to polygons. All the new and missing rectangles and paths are converted to polygons.
3. Merge difference shapes. All new shapes are merged to form merged new shapes, and all missing shapes are merged to form merged missing shapes.

The merging process has these restrictions:

- Polygons that would form a hole after being merged are not merged and a warning is issued.
- Polygons that are touching only at single points (corner to corner or corner to edge) are not merged.
- If the original and reference designs contain identical unmerged shapes A and B, respectively, but there are overlapping shapes that do not create any new shape

outside of A or B, then the overlapping shapes are not merged. The following figure illustrates this situation:



This causes differences to be reported by DBdiff. If these differences are undesirable, then the Fast XOR flow may be used instead.

- A merged polygon cannot have more than 200 vertices by default. This limit can be changed by setting the environment variable `MGC_DBDIFF_MAX_VERTEXCOUNT`. For example:

```
setenv MGC_DBDIFF_MAX_VERTEXCOUNT 300
```

Write XOR Rules Command Line Syntax

The `-write_xor_rules` DBdiff command syntax is used for automatic generation of XOR rules.

Usage

```
dbdiff -system { GDS | OASIS | LEFDEF | {OA [version]} }  
  {{ -design {library | filename} [cell] } |  
  { -lef filename [filename ...] -def filename [cell] }}  
  {{ -refdesign filename [cell] } |  
  { -reflef filename [filename ...] -refdef filename [cell] }}  
-write_xor_rules rule_file [prefix]  
[-base_layers layer1 [layer2 layer3...]]  
[-common_only]  
[-comparetext]  
[-cto]  
[-enhanced]  
[-exclude_layer filename | -include_layer filename] [-layermap filename]  
[-include_rules filename]  
[-refsystem {GDS | OASIS | LEFDEF | OA}]  
[-resultformat [ASCII] [GDS] [OASIS]]  
[-resultmode {userMerged | pseudo | user | top | combine}]
```

Arguments

- **-write_xor_rules** *rule_file*

Required keyword and parameter that specifies the name of the generated rule file.

- *prefix*

An optional string used for the root names of the output DRC Results Database, DRC Summary Report, and DRC Check Map statements. The prefix then appears as the root name in the corresponding output files in a subsequent Calibre nmDRC run. By default, the *prefix* is the *rule_file* name.

- **-base_layers** *layer1* [*layer2 layer3 ...*]

An optional parameter set that specifies base layers during XOR rule generation. This option can only be specified with the `-write_xor_rules` argument. If a layer does not exist in the design, then dbdiff creates layer mapping for this layer and defines it using the [Layout Base Layer](#) specification statement. The *layer* arguments must only be layer numbers and datatypes. Layer and datatype combinations must be separated by a period, for example:

```
-base_layers 1 2.0 4 9.76
```


DBdiff issues the following warning if you do not specify `-base_layers`:

```
WARNING: [FDI2043] LAYOUT BASE LAYER statement was not generated. It  
is highly recommended to generate XOR rules with -base_layers and  
the device creation layer.datatypes for polysilicon, diffusion and  
contacts.
```

You can disable this warning with the following environment variable setting:

```
setenv FDI_DBDIFF_SUPPRESS_VALUES "FDI2043 0"
```

Note

 If you specify this option with the `-layerMap` option, and your layer map contains statements that map multiple layers to the same layer name, then all of the layers must be either base layers or not base layers.

- `-common_only`

An optional parameter that generates XOR rules only for the layers that appear in both input databases. This means both layer numbers and datatype numbers match, respectively. Layers that do not appear in both input databases are called “exclusive layers.” No XOR rules are generated for these layers. A list of exclusive layers appears at the end of the DBdiff transcript when `-common_only` is specified:

```
EXCLUSIVE LAYERS IN CURRENT DESIGN: <current_design_path_name>
  met2  2  0
  met2  3  0
EXCLUSIVE LAYERS IN REFERENCE DESIGN: <reference_design_path_name>
  met2  2  2
  met2  3  3
```

- `-comparetext`

An optional argument that enables text comparison statements in the XOR rule file. This enables a comparison of the text string, location, and the layer on which the text occurs. This option is enabled by default for the `-write_xor_rules` usage.

See “[Comparing Text with DBdiff](#)” on page 67 ”.

- `-cto`

An optional argument that instructs DBdiff to generate Check Text Override (CTO) comments in the generated rules file. The CTO comments apply layer highlighting in Calibre RVE using the physical and mapped layers in the design. The CTO comments are written in the following format:

```
@ RVE Show Layers  physical_layer_num.pnum mapped_layer_name
```

The following is an excerpt from a FastXOR rules file using the `-cto` option:

```
L10_D0 { @ compare layer 10 datatype 0
@ RVE Show Layers: 10.0 L10_D0
  XOR L10_D0 B_L10_D0
}
```

- `-enhanced`

An optional parameter that specifies to write a pair of NOT operations for each pair of layers appearing in an XOR operation.

- **-exclude_layer *filename***
An optional parameter set that specifies that all layers except the ones listed in the *filename* parameter are compared. This option generates [DRC Unselect Check By Layer](#) statements in the output rule file. May not be used with the -include_layer option.
- **-include_layer *filename***
An optional parameter set that specifies that only the layers listed in the *filename* parameter are compared. This option generates [DRC Select Check By Layer](#) statements in the output rule file. May not be used with the -exclude_layer option.
- **-layermap *filename***
An optional parameter set that specifies the name of a layer mapping file. By default, the generated *rule_file* uses system-generated layer names for output. Rule check names follow these system-generated names. The -layermap option causes user-specified layer names to be used in the generated *rule_file* instead. If multiple layers are mapped to the same name, then rules are generated for combining those layers and comparing them. Layers that are not mapped in the file are written in the output rule file using the system-generated names. For additional information see “[-layermap Option Details for -write_xor_rules](#)” on page 101.
- **-include_rules *filename***
An optional parameter set that specifies the name of a file containing SVRF specification statements. These specification statements are imported directly into the DBdiff-generated *rule_file*. There is no sanity check performed by DBdiff for the imported statements. Hence, you need to ensure that the imported statements are valid and do not conflict with the other SVRF statements generated by DBdiff.

See the section “[Write XOR Usage Example](#)” on page 77 for additional information on the SVRF specification statements which are written to the generated rule file.
- **-refsystem {GDS | OASIS | LEFDEF | OA}**
An optional parameter set that specifies the format of the reference design. This parameter set must be specified when the two input designs use different formats. The descriptions for the allowed formats are the same as for **-system**.
- **-resultformat [[ASCII](#)] [[GDS](#)] [[OASIS](#)]**
An optional parameter set that specifies which DRC result database formats are output. The output databases are named as follows:

ASCII — *<rule_file>.asc*
GDS — *<rule_file>.gds*
OASIS — *<rule_file>.oas*
- **-resultmode {userMerged | pseudo | user | top | combine}**
Optional argument set that specifies options for the DRC Results Database statement in the generated rule file. By default, if you do not specify the -resultmode option, then the tool does not generate a mode statement in the rule file.

For details on the modes see “[DRC Results Database](#)” in the *SVRF* manual. For example, if you specify the following DBdiff invocation arguments:

```
dbdiff ... -write_xor_rules ... -resultmode top
```

DBdiff includes the following statement in the generated XOR rule file:

```
DRC RESULTS DATABASE rules.fxor.asc ASCII TOP
```

Description

DBdiff has two syntaxes. This section describes the syntax for generating an XOR rules file. The syntax used for database comparison is discussed under “[Layout Comparison Command Line Syntax](#)” on page 79.

The `-write_xor_rules` option instructs DBdiff to generate an SVRF rule file (*rule_file*) for performing Calibre XOR operations on the two input layouts specified. This command only writes comparison rules for layers with geometries. It does not write comparison rules for layers that have only text. If you want to include comparison rules for layers with text, you must also specify the `-comparetext` option.

The `-write_xor_rules` argument cannot be used with options related to layout comparison.

The **-design** option applies to the [Layout Path](#) statement in the output rule file. The **-refdesign** option applies to [Layout Path2](#). When the Layout System is LEFDEF, then the **-lef** and **-def** options apply to the Layout Path in the output rule file. The **-reflef** and **-refdef** options apply to Layout Path2.

By default, the generated rule file specifies creation of the [DRC Summary Report](#) file and [DRC Results Database](#) files in ASCII and GDS format; the output files are named *rule_file.summary*, *rule_file.asc*, and *rule_file.gds*, respectively. You can use the `-resultformat` argument to specify which DRC result database formats are output.

-layermap Option Details for -write_xor_rules

The layer mapping file for `-write_xor_rules` can have two forms, depending on usage. The file is case insensitive. The line formats are as follows. All the arguments must be on a single line.

GDS or OASIS layout:

```
layer_number datatype layer_name [CURrent | REFerence]  
[{layout_window | layout_windel} [layer_number]]
```

Place and Route layout:

```
layer_name purpose_name  
layer_number purpose_number [CURrent | REFerence]  
[{layout_window | layout_windel} [layer_number]]
```

In the GDS or OASIS format, the columns refer to layer constraint, datatype constraint, and layer name, respectively. Layer and datatype constraints use the customary Calibre forms, but whitespace is not allowed in a constraint string. For example:

# layer	datatype	layer name
# constraint	constraint	
>2<=5	>0<4	MET1
>5<7	<5	MET2
8	0	MET3

The layers in the specified layer and datatype ranges are assigned the names MET1, MET2, and MET3, respectively, in the output rule file.

In the Place and Route format, the columns are layer name, purpose name, layer number, and datatype number. For example, if the layer map file contains this:

#Layer Name	Purpose Name	Layer Number	Datatype
METAL1	DRAWING	2	0
METAL2	DRAWING	3	0

then the output would be this:

```
// LAYOUT BUMP2 15 assumed
LAYER MAP 2 DATATYPE 0 65535
LAYER MAP 17 DATATYPE 0 65534
LAYER METAL1.DRAWING 65535
LAYER B_METAL1.DRAWING 65534
DRC CHECK MAP METAL1.DRAWING GDSII 2 0 "rules.gds"
DRC CHECK MAP METAL1.DRAWING ASCII
METAL1.DRAWING { @ compare layer METAL1.DRAWING
  XOR METAL1.DRAWING B_METAL1.DRAWING
}

LAYER MAP 3 DATATYPE 0 65533
LAYER MAP 18 DATATYPE 0 65532
LAYER METAL2.DRAWING 65533
LAYER B_METAL2.DRAWING 65532
DRC CHECK MAP METAL2.DRAWING GDSII 3 0 "rules.gds"
DRC CHECK MAP METAL2.DRAWING ASCII
METAL2.DRAWING { @ compare layer METAL2.DRAWING
  XOR METAL2.DRAWING B_METAL2.DRAWING
}
```

If the same layer name is mapped to multiple layer numbers, such as here:

```
METAL1 DRAWING 2 0
METAL2 DRAWING 3 0
METAL1 DRAWING 4 0
```

then the union of the numbered layers is taken. For this example the union of layers 2 and 4 is taken:

```
LAYER MAP 2 DATATYPE 0 65535
LAYER MAP 17 DATATYPE 0 65534
LAYER MAP 4 DATATYPE 0 65533
LAYER MAP 19 DATATYPE 0 65532
LAYER METAL1.DRAWING 65535 65533
LAYER B_METAL1.DRAWING 65534 65532
DRC CHECK MAP METAL1.DRAWING GDSII 2 0 "rules.gds"
DRC CHECK MAP METAL1.DRAWING ASCII
METAL1.DRAWING { @ compare layer METAL1.DRAWING
  XOR METAL1.DRAWING B_METAL1.DRAWING
}

LAYER MAP 3 DATATYPE 0 65531
LAYER MAP 18 DATATYPE 0 65530
LAYER METAL2.DRAWING 65531
LAYER B_METAL2.DRAWING 65530
DRC CHECK MAP METAL2.DRAWING GDSII 3 0 "rules.gds"
DRC CHECK MAP METAL2.DRAWING ASCII
METAL2.DRAWING { @ compare layer METAL2.DRAWING
  XOR METAL2.DRAWING B_METAL2.DRAWING
}
```

If multiple layer names are mapped to the same layer number and purpose number, then a warning is issued and the issue is resolved automatically. For example, assume this in the layer mapping file:

```
M6 drawing 36 40
M6 drawingz 36 40
```

This is given in the output file:

```
LAYER MAP 36 DATATYPE 40 65535
LAYER MAP 94 DATATYPE 40 65534
LAYER M6 65535
LAYER B_M6 65534
DRC CHECK MAP M6 GDSII 36 40 "rules.fxor.gds"
DRC CHECK MAP M6 ASCII
M6.drawing { @ compare layer M6 drawing drawingz
  XOR M6 B_M6
}
```

This warning is issued:

```
WARNING: Single physical layer number 36 datatype 40 is mapped to more
        than one layer M6.drawing (36 40) and M6.drawingz (36 40).
```

When -layermap option is used with either -include_layer or -exclude_layer, then the include or exclude layer file must contain the layer name instead of the layer number and datatype. The layer name is taken from the -layermap file.

In certain cases, it is necessary to specify which layers to compare from which design. For example, if layer 10 datatype 0 is mapped to metal1 in the reference design, but layer 17 datatype 0 is used in the current design, additional information is necessary to compare METAL1 between the two databases.

To handle this situation, there is an optional column that specifies a CURrent or REference keyword that corresponds to a specific design. These two keywords can be abbreviated with their first three letters. CURrent refers to a current layer to compare to a reference layer. CURrent is defined either by the **-design** parameter or the **-lef** and **-def** parameters. REference specifies the original reference layer for comparison with the current layer. REference is defined by either **-redesign** or **-reflef** and **-refdef**.

A CURrent layer must match to a corresponding REference layer; if not, a warning occurs. If CURrent or REference is specified on one layer of the layer map file, then these keywords must be specified for all layers in that file, such as here:

```
1  >=0 met1 CUR
11 >=0 met1 REF
2  >=0 met2 CUR
12 >=0 met2 REF
In this case, the output would be this:
//assume LAYOUT BUMP2 103
LAYER MAP 1 DATATYPE >=0 65535
LAYER MAP 114 DATATYPE >=0 65534
LAYER met1 65535
LAYER B_met1 65534
DRC CHECK MAP met1 GDSII 1 0 "rules.gds"
met1 { @ compare layer met1
    XOR met1 B_met1
LAYER MAP 2 DATATYPE >=0 65533
LAYER MAP 115 DATATYPE >=0 65532
LAYER met2 65533
LAYER B_met2 65532
DRC CHECK MAP met2 GDSII 2 0 "rules.gds"
met2 { @ compare layer met2
    XOR met2 B_met2
}
```

If a layer is specified as a CUR and REF layer, but is only present in one of the designs, an XOR rule is written for that layer. The layer is listed as an exclusive layer at the end of the transcript.

An error occurs if CURrent and REference are used without **-write_xor_rules**. An additional example and error information can be found in the section [“Errors when using -write_xor_rules with CURrent and REference in DBdiff”](#) on page 76.

You can also use an optional column that specifies a WINDEL_LAYER or WINDOW_LAYER keyword that corresponds to specific layer data in the layer map. The additional column provides a way to exclude or include layers in a design. Using the WINDEL_LAYER or WINDOW_LAYER keywords allows layers to be handled without an additional rule file using [Layout Windel Layer](#) or [Layout Window Layer](#).

The CUR and REF keywords can also be used along with the WINDEL_LAYER or WINDOW_LAYER keywords. For example:

```
99 0 MARKER CUR WINDOW_LAYER
```

In this case, the MARKER layer in the current design is used as a Layout Window Layer.

Optionally, an integer layer number can be specified as a destination layer for the included or excluded layers. If a number is not provided a layer number is generated by the tool automatically. For example:

```
99 0 MARKER CUR WINDOW_LAYER 990
```

In this case, the MARKER layer is mapped to layer 990.

If WINDEL_LAYER or WINDOW_LAYER is used along with -write_xor_rules, XOR rules are not generated for the specified layer.

See the sections “[Using FastXOR to Perform LVL](#)” on page 29 and “[Using Calibre nmDRC-H to Perform LVL](#)” on page 20 for procedures that use the generated rule file to perform Layout Versus Layout comparison. Also see “[FastXOR Database Comparison](#)” on page 25.

Related Topics

[Layout Comparison Command Line Syntax](#)

[Input and Output](#)

[DBdiff Messages](#)

Summary of Differences in the ASCII Report File

The ASCII report file shows differences between the compared layouts. Shape, Instance, and Text difference tables are generated, depending on the differences found.

The following are examples of the first two tables, formatted to fit the page width.

```
*****Shape difference summary begin*****
Layer  Datatype  NewObjs  MissingObjs  TotalDiffObjs  DesignObjs
16      0          0        25           25             3165
RefdesignObjs TotalObjs  Change (%)
3190          6355      0.39
*****Shape difference summary end*****
```

These are the definitions of the columns:

Layer — layer number.

Datatype — datatype number.

NewObjs — Number of objects in the current layout that are not in the reference layout.

Missing Objs — Number of objects in the reference layout that are not in the current layout.

TotalDiffObjs — Sum of NewObjs and MissingObjs columns.

DesignObjs — Number of objects on the Layer and Datatype in the current layout.

RefdesignObjs — Number of objects one the Layer and Datatype in the reference layout.

TotalObjs — Sum of the DesignObjs and RefdesignObjs columns.

Change (%) — $(\text{TotalDiffObjs} / \text{TotalObjs}) * 100$

In the instance summary table, instance count is in terms of srefs:

```
*****Instance difference summary begin*****
CellName      RefCellName      NewInsts      MissingInsts      TotalDiffInsts
top           top              39183         39182             78365
DesignInsts    RefdesignInsts TotalInsts      Change (%)
43177          43176            86353         90.75
*****Instance difference summary end*****
```

These are the definitions of the columns:

CellName — Name of the primary cell in the current layout.

RefCellName — Name of the primary cell in the reference layout.

NewInsts — Number of instances in the current layout that are not in the reference layout.

MissingInsts — Number of instances in the reference design that are not in the current layout.

TotalDiffInsts — Sum of NewInsts and MissingInsts columns.

DesignInsts — Number of instances in the current layout.

RefdesignInsts — Number of instances in the reference layout.

TotalInsts — Sum of the DesignInsts and RefdesignInsts.

Change (%) — $(\text{TotalDiffInsts} / \text{TotalInsts}) * 100$

The environment variable CALIBRE_DBDIFF_MERGE_SHAPETEXTSUMMARY can be set to 1 to output the shapes and text differences summary in a single table.

Related Topics

[Layout Comparison Command Line Syntax](#)

[Input and Output](#)

[Write XOR Rules Command Line Syntax](#)

[DBdiff Messages](#)

OASIS Format Restrictions

There are some OASIS format restrictions you must consider before running DBdiff.

The following OASIS constructs are not supported by DBdiff:

XNAME RECORD

PROPNAME RECORD

XELEMENT RECORD

PROPSTRING RECORD

XGEOMETRY RECORD

PROPERTY RECORD

For general information on Calibre support and implementation of OASIS, see the manual *Calibre for the Open Artwork System Interchange Standard (OASIS)*.

Related Topics

[Layout Comparison Command Line Syntax](#)

[Write XOR Rules Command Line Syntax](#)

[DBdiff Messages](#)

LEF/DEF Design Considerations

It is highly recommended to use fdi2gds or fdi2oasis to translate your LEF/DEF design before running any comparison. These utilities provide finer control over how your data is translated and will give you a better understanding of what objects are available for comparison.

LEF/DEF designs in their native format are handled differently and require some additional considerations in DBdiff. LEF/DEF databases contain layer names, not numbers. When comparing LEF/DEF to another layout system, a layer mapping file is needed in order to compare the appropriate layers.

When comparing LEF/DEF to LEF/DEF, the layers are compared in the order they appear in the LEF. If the **-lef** and **-reflef** files are different, a layer mapping file is needed.

Layer mapping can be specified with the -layermap option, or with either the MGC_CALIBRE_LAYERMAP_FILE or the MGC_CALIBRE_DB_READ_OPTIONS variable. The form of the layer mapping file is the same as the FDI layer map file. The layer map file format and variable usage are documented under “[Layer Map File for fdi2gds and fdi2oasis](#)” on page 172 and “[FDI Environment Variables](#)” on page 114.

Note



fdi2gds and fdi2oasis translators support a unique mapping file for LEF/DEF systems. This option is not supported in DBdiff. For more details, see “[-map](#)” in the fdi2gds and fdi2oasis usage.

While reading LEF/DEF files, DBdiff adds a suffix to via cell names. The tool may assign different names to via cells when reading the same LEF/DEF files a second time. For this reason, -automatch -multimatch is enabled by default. The suffix may not be present in the DEF file. Hence, one-to-one correspondence between via cell names in the DBdiff report file and the DEF file is not possible in this case.

For example, suppose the LEF file contains the following information for VIA2V3H_FAT:

```
VIA VIA2V3H_FAT
  RESISTANCE 1 ;
  LAYER m2-m3 ;
  RECT -0.09 -0.09 0.09 0.09 ;
  LAYER m2 ;
  RECT -0.1 -0.145 0.1 0.145 ;
  LAYER m3 ;
  RECT -0.145 -0.1 0.145 0.1 ;
END VIA2V3H_FAT

VIARULE VIA2V3H_FAT GENERATE
  LAYER m2 ;
  DIRECTION HORIZONTAL ;
  OVERHANG 0.01 ;
  LAYER m3 ;
  DIRECTION VERTICAL ;
  OVERHANG 0.01 ;
  LAYER m2-m3 ;
  RECT -0.09 -0.09 0.09 0.09 ;
  SPACING 0.53 BY 0.53 ;
  RESISTANCE 1 ;
END VIA2V3H_FAT
```

and the DEF file contains this:

```
- VIA2V3H_FAT-290-820-ALL-1-2 + PATTERNNAME VIA2V3H_FAT-290-820-I1

+ RECT m2 ( -110 -410 ) ( 110 410 )
+ RECT m3 ( -145 -365 ) ( 145 365 )
+ RECT m2-m3 ( -90 -355 ) ( 90 -175 )
+ RECT m2-m3 ( -90 175 ) ( 90 355 ) ;
```

then the report file contains this:

	Current		VIA2V3H_FAT_rule_36506394		Missing Object		Rectangle	
-110	-410	110	410			m2		
	Current		VIA2V3H_FAT_rule_36506394		New Object		Rectangle	
-100	-410	100	410			m2		

Remember, the suffixes to the via cell names are not in the DEF file.

Related Topics

[Input and Output](#)

[Layout Comparison Command Line Syntax](#)

OpenAccess Layer Mapping

If comparing OpenAccess to OpenAccess and the layer schemes differ, or if comparing OpenAccess to GDS or OASIS, then a DBdiff layer mapping file is needed.

The form of the layer mapping file is the same as the FDI layer map file, as documented under “[Layer Map File for fdi2gds and fdi2oasis](#)” on page 172:

```
<layer_name> <purpose> <output_layer> <datatype>
```

For example:

```
met1 drawing 12 0
m1 drawing 13 0
met2 drawing 14 0
m2 drawing 15 0
```

The layer mapping file can be specified with the MGC_CALIBRE_LAYERMAP_FILE or the MGC_CALIBRE_DB_READ_OPTIONS variable. Variable usage is discussed under “[FDI Environment Variables](#)” on page 114.

If a layer mapping file is specified, it applies to an OpenAccess design that is specified with the **-design** or **-refdesign** command-line arguments.

Related Topics

[Input and Output](#)

[Layout Comparison Command Line Syntax](#)

Chapter 5

Database Translation Utilities

The fdi2gds and fdi2oasis utilities translate third-party format design databases into GDSII or OASIS formats.

Third-Party Database Support	111
License Requirements	114
FDI Environment Variables	114
GDS to OASIS Translation	119
Getting Started with fdi2gds and fdi2oasis	121
Command Line Reference	130
FDI File Reference	155

Third-Party Database Support

The fdi2gds, fdi2oasis, and fdiBA utilities support LEF/DEF and OpenAccess third-party databases.

The supported database versions are listed as follows:

- LEF/DEF 5.8 and older.
- OpenAccess oa22.43p006 (default), 22.50p064, and oa22.60p025. The supported data model (DM) revisions are DM4 or older for 22.43, DM5 or older for 22.50, and DM6 and older for 22.60. Pcells are supported in OA databases created by applications using 22.43, 22.50, and 22.60.

These are proprietary formats overseen by Silicon Integration Initiative, Inc. (Si2). For questions about these formats, contact Si2.

Database Translation from the Command Line

All of these database formats can be read by Calibre tools. Processing of these layout formats is very similar to the processing of GDS and OASIS layout formats discussed under “[GDSII Layout Format](#)” and “[OASIS Layout Format](#)” in the *Calibre Verification User’s Manual*.

The third-party layout formats are specified in a rule file using the [Layout System](#) statement. The [Layout Path](#) statement needs to specify the design library for OpenAccess systems. See the *SVRF Manual* description of Layout Path for details regarding integration with third-party formats.

Database Translation from Calibre Interactive and Calibre DESIGNrev

Calibre Interactive and the Calibre layout viewers all support these layout formats for input. Calibre tools do not output these formats, however. For information on setting database options in Calibre Interactive, see “[Third Party Database Options](#)” in the *Calibre Interactive and Calibre RVE User’s Manual*.

OpenAccess Database Details

As new OpenAccess (OA) versions are released by Si2, Mentor Graphics will qualify and adopt them with approximately a one-calendar-quarter lag from the Si2 release to the public. Because Cadence provides the OA reference implementation, some Cadence tools may be released in parallel with Si2’s release, thus placing the Calibre OpenAccess library up to one-quarter out of sync with the most recent Cadence tools.

Be sure all third-party pcell evaluators are compatible with this version. Previous versions of OpenAccess (OA) may be used. FDI tools always use the library in the MGC_HOME directory, regardless of the OA_HOME setting. Use the OA_HOME environment variable to specify the path to a custom OA library that contains plug-ins required by your flow. These plug-ins are loaded from \$OA_HOME/data/plugins, but the library specified by OA_HOME is not used. Virtuoso IC615 is recommended.

OA 22.43 is the default OA version. To use OA version 22.50 or 22.60, set the MGC_FDI_OA_VERSION environment variable to “22.50” or “22.60”, respectively, or specify the OA version when applying the -system OA invocation option in fdi2gds or fdi2oasis.

Environment Variable Settings for OA databases

The following environment variables are used to control the behavior of Calibre tools with respect to OpenAccess databases:

- MGC_HOME
- CDS_INST_DIR
- CDS_ROOT
- OA_HOME
- OA_PLUGIN_PATH
- LD_LIBRARY_PATH
- MGC_FDI_OA_VERSION

Whether you set these variables depends on your specific database and Cadence tool version. The following table describes how you should set these variables:

Table 5-1. Mentor Graphics Variables

Version	Flow	MGC_HOME	MGC_FDI_OA_VERSION
OA 22.43	Virtuoso 5.x	Yes	22.43
	Virtuoso 6.1x	Yes	22.43
	Virtuoso 12.x	Yes	22.43
	Batch (NO PCELL)	Yes	22.43
	Batch (PCELL)	Yes	22.43
OA 22.50	Virtuoso 6.1x	Yes	22.50
	Virtuoso 12.x	Yes	22.50
	Batch (NO PCELL)	Yes	22.50
	Batch (PCELL)	Yes	22.50
OA 22.60	Virtuoso 6.1x	Yes	22.60
	Virtuoso 12.x	Yes	22.60
	Batch (NO PCELL)	Yes	22.60
	Batch (PCELL)	Yes	22.60

Table 5-2. Cadence Variables

Version	Flow	CDS_INST_DIR CDS_ROOT	OA_HOME OA_PLUGIN_PATH LD_LIBRARY_PATH
OA 22.43	Virtuoso 5.x	Depends on cds.lib	Depends on PCELLs
	Virtuoso 6.1x	Depends on cds.lib	Depends on PCELLs
	Virtuoso 12.x	Depends on cds.lib	Depends on PCELLs
	Batch (NO PCELL)	Depends on cds.lib	No
	Batch (PCELL)	Depends on cds.lib	Depends on PCELLs
OA 22.50	Virtuoso 6.1x	Depends on cds.lib	Depends on PCELLs
	Virtuoso 12.x	Depends on cds.lib	Depends on PCELLs
	Batch (NO PCELL)	Depends on cds.lib	No
	Batch (PCELL)	Depends on cds.lib	Depends on PCELLs

Table 5-2. Cadence Variables (cont.)

Version	Flow	CDS_INST_DIR CDS_ROOT	OA_HOME OA_PLUGIN_PATH LD_LIBRARY_PATH
OA 22.60	Virtuoso 6.1x	Depends on cds.lib	Depends on PCELLs
	Virtuoso 12.x	Depends on cds.lib	Depends on PCELLs
	Batch (NO PCELL)	Depends on cds.lib	No
	Batch (PCELL)	Depends on cds.lib	Depends on PCELLs

Related Topics

[FDI Environment Variables](#)

[GDS to OASIS Translation](#)

License Requirements

The FDI utilities check for the presence of licenses only; they do not check out a license.

In order to use these utilities, you must have one of the following Calibre licenses:

DESIGNrev	nmDRC	LFD	nmLVS	RVE
xRC	Interactive	MDPview	WORKbench	

The fdiBA utility requires a Calibre YieldServer license and checks it out during a run.

The environment variables that apply to FDI tools are listed in “[FDI Environment Variables](#)” on page 114.

Related Topics

[Third-Party Database Support](#)

[FDI Environment Variables](#)

FDI Environment Variables

Calibre uses shell environment variables to configure third-party database tools and options.

[Table 5-3](#) describes the configuration environment variables that Calibre batch tools observe in conjunction with LEF/DEF and OpenAccess databases. Calibre Interactive also observes these environment variables, except for MGC_CALIBRE_LAYERMAP_FILE and MGC_CALIBRE_CELLMAP_FILE.

The following video demonstrates how to use an environment variable to control messages during a run:



Table 5-3. FDI Environment Variables


Variable Name	Value	Description
FDI_DBDIFF_SUPPRESS_VALUES	{ <i>message_id number</i> }...	<p>Limits the number of warnings printed to the transcript from FDI utilities and DBdiff. The default maximum number of warnings for each type of message is 20.</p> <p>You can specify any number of <i>message_id</i> and <i>number</i> pairs, where the <i>message_id</i> is the identifier of the warning message and the <i>number</i> is the maximum number of messages to print.</p> <p>For example:</p> <pre>setenv FDI_DBDIFF_SUPPRESS_VALUES "FDI1002 20 FDI2005 15"</pre> <p> Note: If you want to specify a limit on each type of warning message, enter * as the <i>message_id</i>. For example, if you want to limit each type of warning messages to 5 occurrences, set the environment variable as follows:</p> <pre>setenv FDI_DBDIFF_SUPPRESS_VALUES "*" 5"</pre> <p>To remove a reasonable limitation on messages, specify the FDI_DBDIFF_SUPPRESS_VALUES with the wildcard (*) and a very large number.</p> <p>To specify no maximum limit for DBdiff messaging, unset the FDI_DBDIFF_SUPPRESS_VALUES variable and set the CALIBRE_DBDIFF_WARNING_ALL variable to 1.</p>

Table 5-3. FDI Environment Variables (cont.)

Variable Name	Value	Description
FDI_DISABLE_AUTOMAP_SUPPORT	<i>value</i>	Set to a non-null value to disable support for the -map option in fdi2gds and fdi2oasis. This enables the -layerMap option for use with LEF/DEF systems. The -map option allows you to specify a layer map file that is specifically formatted for LEF/DEF databases.
FDI_FDI2OASIS_USE_XY_RELATIVE	<i>value</i>	Set to a non-null value to enable improved compression in OASIS output files generated by fdi2oasis. This can reduce the generated file size on disk.
FDI_LEFDEF_WARNINGS_AS_ERRORS	<i>value</i>	Set to a non-null value to stop the tool when it encounters the following warnings when translating a LEF/DEF design: <div> <div>[FDI1069]</div> <div>[FDI1103]</div> <div>[FDI1111]</div> <div>[FDI1117]</div> <div>[FDI1120]</div> <div>[FDI1123]</div> <div>[FDI1133]</div> <div>[FDI1139]</div> <div>[FDI1157]</div> <div>[FDI1169]</div> </div> <div> <div>[FDI1070]</div> <div>[FDI1104]</div> <div>[FDI1112]</div> <div>[FDI1118]</div> <div>[FDI1121]</div> <div>[FDI1124]</div> <div>[FDI1135]</div> <div>[FDI1142]</div> <div>[FDI1158]</div> <div>[FDI1205]</div> </div> <div> <div>[FDI1102]</div> <div>[FDI1107]</div> <div>[FDI1114]</div> <div>[FDI1119]</div> <div>[FDI1122]</div> <div>[FDI1126]</div> <div>[FDI1136]</div> <div>[FDI1154]</div> <div>[FDI1159]</div> </div>

Table 5-3. FDI Environment Variables (cont.)

Variable Name	Value	Description
FDI_PRESERVE_ODD_WIDTH_PATHS	<i>value</i>	<p>Set to a non-null value to enable odd-width paths in the GDS output from fdi2gds. When fdi2gds encounters an odd-width path, it issues the following warning:</p> <pre>WARNING: [FDI2050] BDG_cut.def[2847]: NET VDD: A path on layer METAL1 has an odd width.</pre> <p>If you do not set this variable, fdi2gds rounds the coordinates down and issues this message:</p> <pre>WARNING: [FDI1116] file.def[line]: NET <net>: The path on layer <layer> at the point (x,y) has an odd width (w). This width will be rounded down. Ensure that all widths are even. Use FDI_PRESERVE_ODD_WIDTH_PATHS environment variable to preserve GDS path widths.</pre>
CALIBRE_FDI_USE_OA_CDBA_NAMESPACE	<i>value</i>	Set to a non-null value to enable full support of Cadence CDBA OpenAccess databases in fdi2gds and fdi2oasis. This variable does not support fdiBA.
FDIBA_DFMPROPERTY_AS_NETNAME	<i>value</i>	When this variable is set to a non-null value when running fdiBA to backannotate a DFMDB to DEF, net names are looked up in the DFM property “net_name” instead of the net connectivity info in the database.
ICC_PATH	<i>pathname</i>	Specifies the directory that includes the icc_shell executable. This environment variable is not required if icc_shell is defined in the current PATH variable.
LD_LIBRARY_PATH	<i>pathname</i>	If using Calibre 2008.3 or later on a Linux machine that is earlier than RHEL 4u6, this variable specifies the pathname of the libstdc++.so.6.0.8 compiler libraries, which are required to run the tool. This is for OpenAccess database support. Ensure that the path includes the 64-bit libraries.

Table 5-3. FDI Environment Variables (cont.)



Variable Name	Value	Description
MGC_CALIBRE_DB_READ_OPTIONS	<p>“-optionName <i>args</i> [-optionName <i>args</i> ...]”</p> <p>The following options may not be used:</p> <ul style="list-style-type: none"> -design -def -lef -outFile -system 	<p>Sets options for controlling the reading of third-party databases by Calibre tools that support them. These options are read in addition to any that may be specified on the FDI utility command lines. A list of options for batch use is shown under Layout System in the <i>SVRF Manual</i>.</p> <p>The option list should be quoted and the delimiter between multiple options is the space character.</p> <p> Note: You can also set DBdiff options using the SVRF DBdiff Options statement in an SVRF rule file. The options in the environment variable take precedence over the rule file statements.</p>
MGC_CALIBRE_LAYER_MAP_FILE	<i>pathname</i>	<p>Sets the pathname of a layer mapping file for use during third-party design database read-in by the batch tool.</p> <p>MGC_CALIBRE_DB_READ_OPTION S can perform the same function using the -layerMap option.</p> <p>MGC_CALIBRE_LAYERMAP_FILE overrides MGC_CALIBRE_DB_READ_OPTION S when both are specified.</p> <p> Note: For OpenAccess databases, if the MGC_CALIBRE_LAYERMAP_FILE environment variable is not set and the -layermap option is not specified, the FDI utilities search for a layer map file named <i>tech_lib.layermap</i> in the technology library directory, where <i>tech_lib</i> is the name of the technology library.</p>

Table 5-3. FDI Environment Variables (cont.)

Variable Name	Value	Description
MGC_CALIBRE_CELLMAP_FILE	<i>pathname</i>	Sets the pathname of a cell mapping file for use during the Layout System design database read-in by the batch tool. MGC_CALIBRE_DB_READ_OPTION S can perform the same function using the -cellMap option. MGC_CALIBRE_CELLMAP_FILE overrides MGC_CALIBRE_DB_READ_OPTION S when both are specified.
MGC_FDI_OA_VERSION	{ <u>22.43</u> 22.60 22.50}	Sets the OpenAccess version for FDI and DBdiff utilities. OA 22.43 is the default. If you specify the OA version number in the fdi2gds or fdi2oasis utilities or with Layout System and Layout System2, the environment variable value is ignored.
OA_HOME Note, if this variable is set, then the specified library must be version oa22.43p006 or later. libstdc++.so.6.0.8 must be available from the host (RHEL 4u6 and later have it by default) or through the LD_LIBRARY_PATH environment variable.	<i>pathname</i>	Calibre uses the OpenAccess library at <calibre_install_dir>/shared/pkgs/icv_oa. If additional plug-ins are needed, use the OA_HOME environment variable to specify the path to an OA library that contains the plug-ins. Note, that the library specified by the OA_HOME variable is not used. Calibre always uses the default OpenAccess library included with the Calibre installation. This variable applies only to dbdiff, fdi2gds, and fdi2oasis.

Related Topics

[Third-Party Database Support](#)

GDS to OASIS Translation

GDS to Layout translation is handled by the gds2oasis application.

The GDSII-to-OASIS translator is documented under “[gds2oasis Translator](#)” in the *Calibre for the Open Artwork System Interchange Standard*.

Related Topics

[Third-Party Database Support](#)

[License Requirements](#)

Getting Started with fdi2gds and fdi2oasis

The fdi2gds and fdi2oasis utilities required minimal input to generate GDS or OASIS outputs, but there are a number of options that allow you to customize the output to suit your flow.

The following topics cover basic procedures for these utilities.

Create and Use a LEF/DEF Mapping File	121
Translating LEF/DEF to GDS or OASIS	122
Translating a LEF Library Using Directories and Outputting to Standard Out	123
Translating LEF/DEF Properties.....	124
Translating OpenAccess to OASIS or GDS	125
Summary Tables	126

Create and Use a LEF/DEF Mapping File

It is important to understand and control how layers in a LEF/DEF database are mapped to OASIS and GDS. You can automatically generate a template mapping file and then modify that file to suit your needs.

Prerequisites

- A DEF design file or LEF macro(s) you want to convert to GDS or OASIS.
- A LEF technology file that defines the layers in the LEF/DEF design data.

Procedure

1. Automatically generate a layer map template file by calling fdi2gds or fdi2oasis without the -map option. For example:

```
fdi2gds -system LEFDEF \
-lef tech.lef macros.lef \
-def top_level.def \
-outFile top_level.gds
```

The fdi2gds and fdi2oasis utilities create a file named *fdi.map* in your working directory. Note, any existing file with that name is overwritten.

2. Rename the *fdi.map* file to *fdi_custom.map* (or any other name). This prevents your customized layer map file from being overwritten.
3. Open *fdi_custom.map* in a text editor. The file contains default mapping statements in the format described under “[LEF/DEF Map File Format](#)” on page 157.


There are four columns in the file, where each column accepts the following inputs:

```
lef_layer    lef_object_type    out_layer    out_datatype
```

The following is an example:

M1	NET	4	1
M1	SPNET	4	2
M1	LEFPIN	4	12
M1	VIA	4	31
V1	LEFPIN	5	12
V1	VIA	5	31
M2	NET	6	1
M2	SPNET	6	2
...			

Note

 The second column also supports additional subtype keywords that filter the DEF object by its properties. For example: “M1 NET:VDD 4 1” converts all VDD nets on M1 to layer 4, datatype 1.

4. Edit the mapping statements in *fdi_custom.map* as desired.
5. Translate your LEF/DEF database and apply the custom mapping file with the `-map` option:

```
fdi2gds -system LEFDEF \
-lef tech.lef macros.lef \
-def top_level.def \
-map fdi_custom.map \
-outFile top_level.gds
```

In this example, `fdi2gds` applies the mapping information from *fdi_custom.map* to the generated GDS file.

Translating LEF/DEF to GDS or OASIS

This example translates a LEF/DEF design to GDS. The procedure for OASIS is the same, except that you apply the `fdi2oasis` command.

See “[LEF/DEF FDI Usage](#)” on page 130 for options that only apply to LEF/DEF databases.

In this example, the *libs/lef* directory contains the following files:

- *tech.lef* — Technology LEF file containing layer information and LEF rules. This must be read before any other LEF file.
- *macros.lef* — Standard cell macros.

The *design/def* directory contains the following:

- *top_level.def* — Top-level chip design.
- *partitions/*.def* — DEF files for block partitions in the *top_level* design.

Procedure

1. Create a layer map file if you want to customize mapping. See [“Create and Use a LEF/DEF Mapping File”](#) on page 121 and [“LEF/DEF Mapping”](#) on page 156 for details. for details).

Note


 You can also use a standard layer mapping file with the `-layerMap` option (see [“Layer Map File for fdi2gds and fdi2oasis”](#) on page 172)

2. In a terminal window, enter the following command:

```
fdi2gds -system LEFDEF \
-lef libs/lef/tech.lef libs/lef/macros.lef \
-def design/def/partitions design/def/top_level.def \
-logFile fdi2gds.log \
-map layer_map.txt \
-outFile output/top_level.gds
```


This command reads in the LEF libraries and DEF design data and translates the chip to GDS format. If you are using a standard layer mapping file, specify the `-layerMap` option instead of `-map`.

Note

 You must read the technology LEF first.

3. Inspect the translated GDS file in Calibre DESIGNrev. Check that the layers and objects in the LEF/DEF were translated as expected. If not, modify the layer map file and re-run the translation.

Note

 PLACEMENT blockages are not currently supported for translation.

Translating a LEF Library Using Directories and Outputting to Standard Out

DEF files are not required when translating LEF macro libraries to GDS or OASIS. Outputting the layout on standard out can also be useful for passing data between flows in a single command.

In this example, the `lef_dir1` directory contains the following file:

- *tech.lef*— Technology LEF file containing layer information and LEF rules.

The `lef_dir2` directory contains the following file:

- **.lef*— LEF macros for standard cells.

The technology LEF and other LEF macros can be kept in the same directory, but you must ensure that the technology LEF is lexicographically first.

Procedure

In a terminal window, enter the following command:

```
fdi2gds -system LEFDEF -lef lef_dir1/tech.lef lef_dir2
```

The fdi2gds and fdi2oasis utilities output the translated database to standard out by default when you do not specify the -outFile option.

Results

You would most likely use this command as input to another command that expects direct GDSII input, as in a shell command like this example:

```
my_application < fdi2gds -system LEFDEF -lef lef_dir_1 lef_dir_2 \  
-def design.def
```

Translating LEF/DEF Properties

The fdi2gds and fdi2oasis utilities allow you to map properties from LEF/DEF designs to properties or text in the output GDS or OASIS files.

Procedure

Apply the -annotateProps command as follows:

```
fdi2oasis -system LEFDEF -lef files -def files ... \  
-annotateProps { {PROPERTY property ...} | TEXT }
```

The [-annotateProps](#) option is documented under “[Complete FDI Command Line Syntax](#)” on page 133.

Examples

Example 1

Translate DEF properties named NET_WIDTH to the OASIS output:

```
fdi2oasis -system LEFDEF -lef files -def files ... \  
-annotateProps PROPERTY NET_WIDTH NET_WIDTH
```

The output name for the properties is also NET_WIDTH.

Example 2

Translate the version of the DEF file as a property named DEFV, and translate the NET_WIDTH property to NTWDTH in the output.

```
fdi2oasis -system LEFDEF -lef files -def files ... \  
-annotateProps PROPERTY DEFVERSION DEFV NET_WIDTH NTWDTH
```

Example 3

Translate all properties to the OASIS output.

```
fdi2oasis -system LEFDEF -lef files -def files ... \  
-annotateProps AUTO PROPERTY NET_WIDTH NTWDTH
```

This operation renames the NET_WIDTH property to NTWDTH. All the other properties are output with their original names.

Example 4

Translate all DEF properties to text objects in the OASIS output file.

```
fdi2oasis -system LEFDEF -lef files -def files ... \  
-annotateProps AUTO TEXT 128.0 DEFVERSION DEFV
```

The DEFVERSION property is output as DEFV, (for example, “DEFV 5.8”), in the lower-left corner of the DEF design cell on the layer 128.0.

Translating OpenAccess to OASIS or GDS

The following example shows how to translate an OpenAccess (OA) design to OASIS. The procedure is the same for GDS output.

Procedure

1. Create a template file that contains options for the translation. The template file allows you to better manage the number of arguments required for translation. The following is an example:

```
# options that appear on the shell command line override these  
-design libraries/lib1 top layout  
  
# annotate nets with text objects  
-annotateNets TEXT TOP  
  
# map cell names according to the following file:  
-cellMap cellmap.txt  
  
# map layer names according to the following file:  
-layerMap layermap.txt  
  
#or you can use the OA mapping format  
#-oamap layers.map  
  
-logFile log  
  
-outFile top.oas  
  
# output blockage objects  
-outputBlockages
```

2. Enter the following command:

```
fdi2oasis -system OA 22.43 -template options.txt
```

where *options.txt* contains the remaining command line options. The output is OASIS.

Summary Tables

After performing a LEF/DEF database translation using *fdi2gds* or *fdi2oasis*, the tool outputs tables to the transcript that summarize information from the run and the input files that were read.

The following tables are output:

- An objects summary table for the LEF, DEF, and generated OASIS or GDS files.
- A summary of runtime and peak memory for parts of the translation flow.
- A count of the total messages and the printed messages in a table.
- A message containing the output layout name and precision.
- A message that reproduces the command line arguments.

Note



The used, total, and translated counts in the tables can be different depending on the design. For example, vias or macros can be defined in some files but not instantiated in the DEF, resulting in different object summary and used totals. Likewise, some layers that contain objects may not be mapped in your mapping file, resulting in more objects in the input files than the translated output.

LEFDEF Reading Summary

This table lists all LEF files that were read.

```
-----  
-----  
-----  
LEFDEF READING  
-----  
-----  
Evaluating LEF(s)  
  libs/lef/tlef/NangateOpenCellLibrary.tech.lef  
  libs/lef/std_cell_macros/NangateOpenCellLibrary.macro.lef  
  design/lef/tsv.lef  
  design/lef/ADC_5bit_sc_5.lef  
  design/lef/dac6_op2.lef  
  design/lef/myram.lef  
  design/lef/pads.lef  
LEFS EVALUATION COMPLETED. CPU TIME = 0 REAL TIME = 0 MEMORY = 21
```

LEF Layers Summary

This table displays all of the LEF layers present in all LEF files that were read, including their layer type and whether a mask name is applied.

LEF LAYERS READ		
LAYER NAME	TYPE	MASK
poly	MASTERSLICE	N/A
fpoly	MASTERSLICE	N/A
gpoly	MASTERSLICE	N/A
OD	MASTERSLICE	N/A
M1	ROUTING	N/A
V1	CUT	N/A
M2	ROUTING	N/A
V2	CUT	N/A
M3	ROUTING	N/A

DEF Objects Summary

Tables of object counts for each of the DEF files that were read. Not all objects in the DEF are translated. For example, if you do not map blockages for translation, then the objects on blockage layers are counted in the TOTAL column, but not the TRANSLATED column.

design/def/top_level.def OBJECTS SUMMARY		
TYPE	TRANSLATED	(TOTAL)
VIA	68	(133)
COMPONENT	33594	(33594)
PIN	82	(82)
FILL	0	(0)
BLOCKAGE	0	(0)
SPECIALNET	4	(4)
NET	16110	(16110)
NDR	0	(0)
ROW	0	(0)
TRACK	0	(0)

LEF Objects Summary

This table includes the total of all objects read from the input LEFs.

LEF(s) OBJECTS SUMMARY	
TYPE	(TOTAL)
VIA	(27)
VIARULE	(19)
MACRO	(156)

LEF Used Objects Summary

This table includes all used LEF objects, such as macros, that were referenced in a DEF layout. For example, if macro_invx8 was in the LEF, but not referenced in a DEF, then it will be counted in the TOTAL column, but not the Used column.

LEF USED (TOTAL) OBJECTS SUMMARY		
Objects	Used	(TOTAL)
LAYER	24	(24)
VIA	27	(27)
VIARULE	19	(19)
MACRO	59	(156)
NDR	0	(0)
SITE	0	(0)

DEF Translated Objects Summary

Similar to the LEF Used summary, this table includes the objects that were translated as compared to the total objects in the DEF file. Not all objects in the DEF are translated. For example, if you do not map blockages for translation, then the objects on blockage layers are counted in the TOTAL column, but not the TRANSLATED column.

DEF TRANSLATED (TOTAL) OBJECTS SUMMARY		
TYPE	TRANSLATED	(TOTAL)
VIA	1169	(1628)
COMPONENT	153197	(153197)
PIN	2290	(2290)
FILL	0	(0)
BLOCKAGE	0	(88)
SPECIALNET	36	(36)
NET	68917	(68917)
NDR	0	(0)
ROW	0	(0)
TRACK	0	(0)

GDS or OASIS Summary

This table includes the total count of each object type that was output to the translated GDS or OASIS layout.

----- GDS SUMMARY -----	
LAYER	84
CELL	240
PLACEMENT	770534
ARRAY	0
RECTANGLE	12602
POLYGON	373
PATH	487716
TEXT	0
PROPERTY	0

Message Summary

This table includes the displayed (printed) and total error messages output for each file that was read. See “[Calibre Utility Messages](#)” on page 213 for a description of the error messages.

----- ERROR & WARNING SUMMARY -----			
FILENAME	MESSAGE CODE	PRINTED	TOTAL

design/def/top_level.def	[FDI1204]	20	(422)

Command Line Reference

The usage and options for fdi2gds and fdi2oasis differ depending on the input layout database.

The following topics describe options unique to each database format in addition to a complete description of all options.

LEF/DEF FDI Usage	130
OpenAccess FDI Usage	131
Complete FDI Command Line Syntax.....	133

LEF/DEF FDI Usage

The fdi2gds or fdi2oasis command line utilities are used to translate LEF/DEF input files.

The minimum LEF/DEF usage is as follows:

```
fdi2gds -system LEFDEF {-lef [filename ...][directory ...]} \  
        {[-def filename] [directory ...]} -outFile file
```

or

```
fdi2oasis -system LEFDEF {-lef [filename ...][directory ...]} \  
          {[-def filename] [directory ...]} -outFile file
```

The first specified LEF file must be the technology LEF. The -def option is optional. If you do not specify DEF files, then the generated output file is a GDS or OASIS layout that contains all LEF macros translated into GDS or OASIS cells with no top cell.

The LEF/DEF system uses all of the options listed under “[Complete FDI Command Line Syntax](#)” on page 133 except the following:

```
[-abortOnEmptyPCell]  
[-instPinAsShape]  
[-layerMap]  
[-libdefs file ...]  
[-outputFills]  
[-preservePath]  
[-switchView]  
[-viewList]
```

The LEF/DEF system supports the -map option to provide correspondence between layers and objects in the LEF/DEF layout and layers and datatypes in the GDS or OASIS output. This option is highly recommended. See “[LEF/DEF FDI Usage](#)” on page 130 for details. The -objectMap, -outputBlockages, and -prBoundaryLayer options do not apply to LEF/DEF when the -map option is specified.

The LEF/DEF system is the only system that supports this option:

```
[-precision {100 | 200 | 400 | 800 | 1000 | 2000 | 4000 | 8000 | 10000  
| 20000} ]]
```

Related Topics

[OpenAccess FDI Usage](#)

[Complete FDI Command Line Syntax](#)

OpenAccess FDI Usage

The fdi2gds or fdi2oasis command line utilities are used to translate OpenAccess input files.

The minimum OA usage is as follows:

```
fdi2gds -system OA [version] {-design library cell [view]} [-oamap file]  
-outFile file
```

or

```
fdi2oasis -system OA [version] {-design library cell [view]} [-oamap file]  
-outFile file
```

The OpenAccess system uses all of the options listed under “[Complete FDI Command Line Syntax](#)” on page 133 except the following:

```
[-annotateObsSpacing]  
[-annotatePinDirection]  
[-annotatePinUse]  
[-annotateProps]  
[-flattenvias]  
[-noEmptyVias]  
[-map]  
[-outputFills]  
[-precision value]
```

The OpenAccess system is the only system to use these options:

```
[-abortOnEmptyPCell]  
[-libdefs file ...]  
[-preservePath]
```

The `-libdefs filename` option specifies the pathname of a *libs.def* library definition file explicitly. Without this option, library definitions are read in the following order:

- The tool looks for a *cds.lib* file in your working directory. If the *cds.lib* file is found, the tool uses it to generate a *lib.defs* file in the MGC_TMPDIR directory and applies it at runtime. In this case, the following message is printed:

```
INFO: cds.lib has been converted to lib.defs
```

- The tool looks for a *lib.defs* file in your working directory.

To limit the number of layer warnings printed to the transcript when reading an OA database, set the [FDI_DBDIFF_SUPPRESS_VALUES](#) environment variable to one of the supported values.

Note



OA 22.43 is the default OA version. To use OA version 22.50 or 22.60, set the MGC_FDI_OA_VERSION environment variable or specify the OA version when applying the -system OA invocation option.

Related Topics

[LEF/DEF FDI Usage](#)

[Complete FDI Command Line Syntax](#)

Complete FDI Command Line Syntax

The FDI utilities reside in the Calibre installation tree's bin directory.

Usage

```
{fdi2gds | fdi2oasis}
-system { LEFDEF | OA [version] }
{{-design library cell [view]} |
{-lef [filename ...] [directory ...] [-def [filename ...] [directory ...]]}
-outFile file
[-abortOnEmptyPCell]
[-annotateInsts {pname | pnum}]
[-annotateNets [TEXT] [PROPERTY {pname | pnum}] [NET name ...] [ALL | TOP]]
[-annotateNetType {pname | pnum} [TYPE name ...]]
[-annotateObsSpacing [TEXT] [PROPERTY {pnum | pname}]]
[-annotatePins [TEXT] [PROPERTY {pname | pnum}] [ALL | TOP]]
[-annotatePinDirection [TEXT] [PROPERTY {pname | pnum}] [ALL | TOP]]
[-annotatePinNets [TEXT] [PROPERTY {pname | pnum}]]
[-annotatePinUse [TEXT] [PROPERTY {pname | pnum}] [ALL | TOP]]
[-annotateProps {[AUTO] [PROPERTY] {[source_prop_name dest_prop_name]...}}
{[AUTO] [TEXT layer.datatype] {[source_prop_name dest_prop_name]...}}]
[-bboxlayer layerNum [datatype]]
[-cblock]
[-cellCase {PRESERVE | LOWER | UPPER}]
[-cellGDS {file ... | directory ...}]
[-cellMap file]
[-celloASIS {file ... | directory ...}]
[-convertOddWidthPaths]
[-expanddatatypes]
[-flattenvias]
[-inputException file]
[-inputExceptionList {parameter severity} ... }]
[-instPinAsShape]
[-layerMap file]
[-layerNames file]
[-libdefs file]
[-logFile file]
[-map lefdef_map_file]
[-mapInfo file]
[-mapOnly]
[-netProp file]
[-nets file]
[-noEmptyMacros]
[-noEmptyVias]
[-noPinShape]
[-noText]
[-oamap file]
[-objectMap file]
[-outputBlockages]
[-outputFills]
[-prboundarylayer layerNumber [datatype]]
[-precision {100 | 200 | 400 | 800 | 1000 | 2000 | 4000 | 8000 | 10000
| 20000} ]
[-preservePath]
[-propertyMap file]
[-switchView { destView srcView [srcView ...]} ...
[-template file]
[-version]
[-viewList viewName ...]
[-viewName viewName]
[-window llx lly urx ury ...]
```

`[-writehier file]`

Arguments

The command line arguments described in this section are grouped into two categories: required followed by optional. The optional arguments are shown alphabetically.

These arguments apply to both fdi2gds and fdi2oasis, except where stated. The arguments apply to all input layout systems unless stated otherwise.

- **-system {LEFDEF | OA [version]}**

A required parameter set that specifies the input layout system type. One of three types must be specified, as shown. The **OA** option is short for OpenAccess.

The *version* parameter applies only to **OA** and currently accepts 22.43, 22.50, or 22.60. The default version is 22.43.

Note



In order to echo the command usage line in the shell, you must specify this option. In this way, the tool decides which usage line to show.

- **-design library cell [view]**

A required parameter set used when the system is **OA**. The **library** argument specifies the library name or path, and the **cell** argument specifies the cell name. The *view* argument is optional and specifies the view name.

When using an **OA** library the **library** argument must be a library name.

- **-lef [filename ...] [directory ...]**

A required parameter set used when the system is **LEFDEF**. The **-lef** argument must be specified immediately after the **-system** argument if fdi2gds or fdi2oasis is used in the place and route flow. At least one *filename* or *directory* must be specified. Both argument types may be specified, and more than one of each type may be specified.

The *filename* argument is the name of a LEF file. The *directory* is a pathname to a directory containing one or more LEF files. The LEF files in a directory are expected to have the extension .lef. If more than one file or directory is specified, then they are read according to the order they appear on the command line. If multiple files are listed, the LEF technology file must be first. If the directory appears first, the technology file must be lexicographically first in the directory. A filename specified as a *filename* argument takes higher precedence if the same filename also resides in a specified *directory*.

- **-def [filename ...] [directory ...]**

An optional parameter set used when the system is **LEFDEF**. The *filename* is the pathname of a DEF file if reading a single DEF. If you are reading a hierarchical DEF, you must specify a list of DEF files or directories that contain the hierarchical DEF files.

The FDI utilities issue a warning and proceed to read DEF files completely when there are wildcards in the GROUPS section in the DEF file. The following warnings are issued when wildcards are encountered in route1 and route2 groups:

```
Warning: Regular expression in instance name in GROUPS section not
supported. Ignoring the entry route1/*
Warning: Regular expression in instance name in GROUPS section not
supported. Ignoring the entry route2/*
```

The GROUPS sections does not have an impact on the geometry generated by the FDI utilities, so ignoring the constructs does not alter the generated layout files.

- **-outFile *file***

A parameter set that specifies the pathname of the output database. Any existing file of the identical name to *file* is overwritten. If this option is not specified, the default output channel is STDOUT.

If you specify the file with a .gz suffix, then the output file is compressed using gzip format. If you specify the .Z suffix, then the file is compressed using the “compress” command.

- **-abortOnEmptyPCell**

Causes the Calibre application to exit when an empty Pcell appears in an OpenAccess database. This option is only valid when **-system OA** is specified.

- **-annotateInsts {*pname* | *pnum*}**

An optional parameter set that adds instance name annotations to the output file. The *pname* and *pnum* parameters specify the property name, which gets the instance name as its value. For fdi2gds the *pnum* is an integer (property names are limited to integers in GDS). For fdi2oasis the *pname* can be any string.

- **-annotateNets [TEXT] [PROPERTY {*pname* | *pnum*}] [NET *name* ...] [ALL | TOP]**

An optional parameter set that adds net name annotations to all shapes that belong to a net. The annotations can be either text objects or properties in the output file. The arguments are as follows:

TEXT — Specifies that annotations of net names are added as text objects placed on a shape object. The attachment point on the shape is chosen arbitrarily.

PROPERTY {*pname* | *pnum*} — Specifies annotations of net names are added as properties on the shape object. The *pname* and *pnum* parameters specify the property name, which gets the net name as its value. For fdi2gds the *pnum* is an integer (property names are limited to integers in GDS). For fdi2oasis the *pname* can be any string.

NET *name* — Specifies that only shapes on the specified net names receive annotations. More than one net *name* parameter may be specified. The *name* parameter is case-sensitive.

ALL | TOP — ALL specifies that objects are added in all cells and is the default. TOP specifies that objects are added in the top-level cell only.

- -annotateNetType {*pname* | *pnum*} [TYPE *name* ...]

An optional parameter set that extracts the type of net and writes it out as a property into a GDS or OASIS file. The arguments are as follows:

pname | *pnum* — Specifies the net type property name. The *pname* and *pnum* parameters specify the property name, which gets the net type as its value. For fdi2gds the *pnum* is an integer (property names are limited to integers in GDS). For fdi2oasis the *pname* can be any string.

TYPE *name* — Specifies that only shapes on the specified net types receive annotations. More than one net type *name* parameter may be specified. The *name* parameter is case-sensitive.

Table 5-5 describes the net type property values written out by the FDI utilities.

- -annotateObsSpacing [TEXT] [PROPERTY {*pnum* | *pname*}]

An optional argument for LEF/DEF systems that enables you to output obstruction spacing values as text or properties. If you specify the TEXT option, the corresponding obstruction shape's layer spacing value is created on the same layer as the obstruction (OBS) shapes. If you specify the PROPERTY {*pnum* | *pname*} option set, then the obstruction shape's property list is added as an attribute and the shape's layer spacing property is added as the value.

The -annotateObsSpacing argument set works with -map and automatic mapping in LEF/DEF translations, as well as the -oamap mapping option in LEF translation.

- -annotatePins [TEXT] [PROPERTY {*pname* | *pnum*}] [ALL | TOP]

An optional parameter set that adds pin name annotations to all pin shapes. The annotations can be either text objects or properties in the output file. The arguments are as follows:

TEXT — Specifies that annotations of pin names are added as text objects placed on a pin object. The attachment point on the pin is chosen arbitrarily but is inside the pin shape. If the layout system is LEFDEF, all shapes associated with a single pin are texted, even if they are not abutted.

PROPERTY {*pname* | *pnum*} — Specifies that annotations of pin names are added as properties on the pin shape. The *pname* and *pnum* parameters specify the property name, which gets the pin name as its value. For fdi2gds the *pnum* is an integer (property names are limited to integers in GDS). For fdi2oasis the *pname* can be any string.

ALL | TOP — ALL specifies that objects are added in all cells and is the default. TOP specifies that objects are added in the top-level cell only.

- -annotatePinDirection [TEXT] [PROPERTY {*pname* | *pnum*}] [ALL | TOP]

An optional parameter set that annotates LEF/DEF pin "DIRECTION" statements to properties or text in the generated output file. If the direction of the pin is not specified, the default direction is "input".

TEXT — Specifies that pin direction annotations are added as text objects placed on a pin object. The attachment point on the pin is chosen arbitrarily but is inside the pin shape.

PROPERTY {*pname* | *pnum*} — Specifies that pin direction annotations are added as properties on the pin shape. The *pname* and *pnum* parameters specify the property name, which gets the pin name as its value. For fdi2gds the *pnum* is an integer (property names are limited to integers in GDS). For fdi2oasis the *pname* can be any string.

ALL | **TOP** — **ALL** specifies that objects are added in all cells and is the default. **TOP** specifies that objects are added in the top-level cell only.

- **-annotatePinNets** [**TEXT**] [**PROPERTY** {*pname* | *pnum*}]

An optional parameter set that maps DEF PIN net names. When specified, the tool annotates PIN NET objects from LEF/DEF to properties in the output database. The arguments are as follows:

TEXT — Specifies that annotations of pin net names are added as text objects placed on a shape object. The attachment point on the shape is chosen arbitrarily.

PROPERTY {*pname* | *pnum*} — Specifies annotations of pin net names are added as properties on the shape object. The *pname* and *pnum* parameters specify the property name, which gets the pin net name as its value. For fdi2gds the *pnum* is an integer (property names are limited to integers in GDS). For fdi2oasis the *pname* can be any string.

```
M1 PIN:VDD 11 1
```

See “[LEF/DEF Map File Format](#)” on page 157 for more details.

- **-annotatePinUse** [**TEXT**] [**PROPERTY** {*pname* | *pnum*}] [**ALL** | **TOP**]

An optional parameter set that annotates LEF/DEF pin “USE” statements to properties or text in the generated output file. If the use of the pin is not specified, “signal” is the default.

TEXT — Specifies that pin use annotations are added as text objects placed on a pin object. The attachment point on the pin is chosen arbitrarily but is inside the pin shape.

PROPERTY {*pname* | *pnum*} — Specifies that pin use annotations are added as properties on the pin shape. The *pname* and *pnum* parameters specify the property name, which gets the pin name as its value. For fdi2gds the *pnum* is an integer (property names are limited to integers in GDS). For fdi2oasis the *pname* can be any string.

ALL | **TOP** — **ALL** specifies that objects are added in all cells and is the default. **TOP** specifies that objects are added in the top-level cell only.

- `-annotateProps {[AUTO] [PROPERTY] {[source_prop_name dest_prop_name]...}}
[AUTO] [TEXT layer.datatype] {[source_prop_name dest_prop_name]...}}`

Optional argument set used to translate properties from LEF/DEF databases to properties or text in the GDS or OASIS output. See “[Translating LEF/DEF Properties](#)” on page 124 for an example. The arguments are described as follows:

AUTO — Specifies to translate all properties to the output file. When using `fdi2gds`, the property names are translated to unused property numbers from integers between 1 and 125. When using `fdi2oasis`, the properties are translated to property name and value pairs using the original names or the names you specify with the *source_prop_name* and *dest_prop_name* arguments. If this option is not used, the only properties translated are those specified in the *source_prop_name* and *dest_prop_name* arguments.

PROPERTY — Specifies to translate properties from LEF/DEF and embed them as properties in the GDS or OASIS output.

TEXT *layer.datatype* — Specifies to translate properties from LEF/DEF and embed them as text in the GDS or OASIS output. The *layer.datatype* value is required and specifies the text layer name and datatype pair to which to write the text in the output file.

source_prop_name dest_prop_name — Specifies the source property name in the LEF/DEF database and the desired name for the property or text in the GDS or OASIS output. This argument set is optional and you can specify any number of source and destination value pairs. If you do not apply the AUTO option, only properties specified by these arguments are output.

Note

When using `fdi2gds`, the *dest_prop_name* field must be an integer between 1 and 125.

- `-bboxlayer layer_num [datatype]`

An optional parameter set that specifies the bounding box of the cell should be written out as a shape on the specified layer number and optional datatype. Both *layer_num* and *datatype* are non-negative integers. When no *datatype* is specified, a datatype of 0 is assumed.

The bounding box shape represents the bounding box of the cell as it exists in the database. The box can sometimes be different from the actual bounding box of the cell written out because the following options in `fdi2gds` and `fdi2oasis` allow you to write out only the required portion of the database cell:

- The [-layerMap](#) option selects only the layers to be written out. Some layers that contribute to the bounding may not be written out.
- The [-window](#) option selects objects that interact with the specified windows.

- -cblock

An optional parameter that enables CBLOCK compression for fdi2oasis output. CBLOCK compression can result in smaller file sizes. This option can be applied when translating LEF/DEF or OpenAccess to OASIS.

Note



This option only applies to LEF/DEF and OA layouts that are written to disk using the fdi2oasis -outFile option. The option is ignored when reading third-party layouts with direct read in SVRF, Calibre Interactive, or Calibre DESIGNrev.

- -cellCase {PRESERVE | LOWER | UPPER}

An optional parameter set that specifies how to handle the text case of cell names. This option does not affect cells specified in a -cellMap file. The arguments are as follows:

- PRESERVE — The case of cell names is preserved as found in the input database. This is the default behavior if you do not specify this option.
- LOWER — Cell names are output in lowercase letters.
- UPPER — Cell names are output in uppercase letters.

- -cellGDS {*filename* ... |*directory* ... }

An optional parameter set that defines lower-level cells and any instantiated hierarchy to read from the specified *filename* or *directory*. This option is only used for fdi2gds. Multiple files or directories are allowed. When a directory is specified, fdi2gds reads all the files in the specified directory with .gds or .gds.gz extensions. These suffixes are not case-sensitive. If the latter format is used, the gzip command must be in your environment.

If a duplicate cell name is found, the utility uses the first cell definition.

- -cellMap *filename*

An optional parameter set that specifies the pathname of a cell mapping file used during fdi2gds and fdi2oasis translation. When used, this option allows you to specify input cell names and the corresponding output cell names that you want in the GDS or OASIS file. When not specified, the input cell names are used as the output cell names.

To view the file format and syntax, see “[Cell Mapping File](#)” on page 179.

- -cellOASIS {*filename* ... |*directory* ... }

An optional parameter set that defines lower-level cells and any instantiated hierarchy to read from the specified *filename* or *directory*. This option is only used for fdi2oasis and LEF/DEF inputs and cannot be specified with -cellGDS or -cellMap. Multiple files or directories are allowed. When a directory is specified, fdi2oasis reads all the files in the specified directory with .oas or .oas.gz extensions. These suffixes are not case-sensitive. If the latter format is used, the gzip command must be in your environment.

If a duplicate cell name is found in the specified OASIS files, the utility uses the first cell definition. If you are reading hierarchical DEFs and the same cell is defined in the DEF and in the OASIS files, then the tool uses the DEF cell.

This option does *not* support the following:

- Properties from input OASIS objects.
- Text from input files that are defined by reference numbers.

Note

Repeated objects from input file are converted to separate objects. This increases the size of the OASIS output.

- **-convertOddWidthPaths**

Optional argument that specifies to enable special handling for paths with odd widths. When you specify this option, all SPECIALNET paths that have odd widths are converted to polygons that are on-grid with the original odd-width value. This argument takes precedence over the FDI_PRESERVE_ODD_WIDTH_PATHS environment variable.

The tool outputs the following message when it encounters an odd-width path and you specify this option:

```
WARNING_1227: (DEF_LINE): NET (NET): A path on layer (LAYER) at the
point ((coord),(coord) ) has an odd width ((width)) and will be
converted to polygon.
```

- **-expanddatatypes**

An optional parameter for OA that exports the datatype to the output file. For example, if you have two shapes on METAL1, one that has the datatype 10 and the other with datatype 0, then with -expanddatatypes not specified, all shapes are written to datatype 0. With -expanddatatypes specified, each shape will be on the same layer with datatypes 10 and 0, respectively.

For OpenAccess databases, if you do not set the -layerMap option and you specify -expanddatatypes, then each object type is mapped to a datatype as follows:

```
Drawing -> 0
fill -> 1
slot -> 2
OPCSerif -> 3
OPCAntiSerif -> 4
annotation -> 5
gapFill -> 6
redundant -> 7
oaAny -> 8
oaNo -> 9
oaFilloPC -> 10
oaCustomFill -> 11
```

- **-flattenvias**

An optional parameter that flattens vias when translating LEF/DEF designs to GDS or OASIS databases. The shapes defined in the via instances are flattened into the reference cell on output. Properties on the via instances, such as the net type or net name, are not preserved on the output via shapes. To flatten vias with no instance properties, specify -flattenvias as follows:

```
fdi2gds options 11 -flattenvias
fdi2oasis options 11 -flattenvias
```

When the -annotateNets or -annotateNetType options are used with the -flattenvias option, properties are attached to the shapes within the via reference as specified. To flatten vias and include net type and net name properties on the flattened via shapes, specify -flattenvias with the annotation options as follows:

```
fdi2gds options -annotateNetType PROPERTY 10 \
-annotateNets PROPERTY 11 -flattenvias
fdi2oasis options -annotateNetType PROPERTY 10 \
-annotateNets PROPERTY 11 -flattenvias
```

Note, flattening via cells negatively impacts performance and increases the database size, so only do this if absolutely necessary.

- **-inputException *filename***


An optional parameter set that specifies exception levels for certain tool behaviors. The *filename* contains the exception settings. For example, a line in the file could have this:

```
CELLGDS_PRECISION 0
```

See “[LEF/DEF FDI Usage](#)” for details.

To view the file format and syntax, see “[Input Exception File](#)” on page 188.

Note

 This option only applies to LEF/DEF for fdi2oasis. It applies to all databases for fdi2gds. The -inputExceptionList and -inputException options are mutually exclusive.

- **-inputExceptionList { {*parameter severity*} ... }**

Optional argument set that controls how errors in the LEF/DEF map file are handled.

The *parameter* argument is one of the following:

- CELLGDS_DUPLICATE_CELL
- CELLGDS_PRECISION
- WAIVER_PRECISION
- EXCLUDE_CELL_NAME
- LEFDEF_MAP_WARNINGS

- FDI1* — (All fdi2gds and fdi2oasis warning messages)

The *severity* is either 0, 1, or 2, where:

- 0 — Silently skip map file issues.
- 1 — Issue warnings and skip map file issues.
- 2 — Issue an error and terminate the run on map file issues. This is the default behavior for map file warnings.

See [Table A-3](#) in “[FDI Messages](#)” on page 213 for a list of messages. The `-inputExceptionList` and `-inputException` options are mutually exclusive.

- `-instPinAsShape`

An optional parameter that specifies all shapes belonging to an instance pin should be written out as shapes in the containing (parent) cell. This option does not apply to LEF/DEF databases.

- `-layerMap filename`

An optional parameter set that specifies the pathname of a layer mapping file used during translation of OA databases. When used, this option allows you to specify input layer names, purposes, and object types, and to map objects with those attributes to output objects with a corresponding layer and datatype.

Objects in the input database that correspond to the attributes in the mapping file can be output with the `-objectType` option. By default, all input objects are output with their input layer numbers and a datatype record of 0.

To view the file format and syntax, see “[Layer Map File for fdi2gds and fdi2oasis](#)” on page 172.

Note



For OpenAccess databases, if the `-layermap` option is not specified and the `MGC_CALIBRE_LAYERMAP_FILE` environment variable is not set, then the FDI utilities search for a layer map file named *tech_lib.layermap* in the technology library directory, where *tech_lib* is the name of the technology library.

For LEF/DEF databases, “[LEF/DEF Map File Format](#)” on page 157. This option is mutually exclusive with the `-map` option.

- `-layerNames filename`

An optional parameter set that generates an output file with layer mapping information. The file contains SVRF code using the [Layer](#) specification statement. The format for each line is this:

LAYER *name number*

The *name* argument is a layer name in the input database. The *number* argument is the corresponding layer number of the mapped layer in the GDS or OASIS layout. Here is an example:

```
LAYER METAL1 1
LAYER VIA1 2
LAYER METAL2 3
```

- **-libdefs *filename* ...**

An optional parameter set that specifies the pathname of a *libs.def* library definition file explicitly. This option applies only to the **OA** system. Without this option, library definitions are read in the following order:

- The tool looks for a *cds.lib* file in your working directory. If the *cds.lib* file is found, the tool uses it to generate a *lib.defs* file in the MGC_TMPDIR directory and applies it at runtime. In this case, the following message is printed:

```
INFO: cds.lib has been converted to lib.defs
```

- The tool looks for a *lib.defs* file in your working directory.

- **-logFile *filename***

An optional parameter set that specifies the pathname of a runtime log for the translation. No logfile is generated by default.

- **-map *lefdef_map_file***

An optional parameter set that specifies the pathname of a file that maps layers and objects from LEF/DEF databases to the translated GDS or OASIS output (see “[LEF/DEF Map File Format](#)” on page 157). This argument set only applies to LEF/DEF systems.

The -layerMap, -outputBlockages, -objectMap, and -prboundarylayer options cannot be specified with -map.

- **-mapInfo *filename***

An optional parameter set that specifies the pathname of the mapping information output file. This file shows what mappings the translator applied, including any user-specified mappings.

To view the file format and syntax, see “[Mapping Information File](#)” on page 182.

- **-mapOnly**

An optional argument that specifies to only generate an automatic layer mapping file and to not perform any translation of the input database. The -map, -layerMap, and -outFile options are mutually exclusive with this option and the option only works with LEF/DEF systems.

The MGC_CALIBRE_LAYERMAP_FILE environment variable does not support the -mapOnly option.

- **-netProp *filename***

An optional parameter set that specifies the pathname of a net type properties file. The *filename* specifies net names and the corresponding net type properties. This option is used in conjunction with a [-layerMap](#) file that contains *net_type_property* arguments that match the names used in the -netProp file. When used in this way, only the shapes that have the associated net type values are output.

To view the file format and syntax, see “[Net Properties File](#)” on page 184.

- **-nets *filename***

An optional parameter set that specifies a *filename* containing a list of net names and net types to be annotated in the output. The *filename* supports the keywords FDI_NET_NAMES and FDI_NET_TYPES. FDI_NET_NAMES indicates net names. FDI_NET_TYPES indicates net types. One or both of these parameters can be specified in the *filename*.

To view the file format and syntax, see “[Nets File](#)” on page 185.

- **-noEmptyMacros**

An optional parameter that specifies not to translate empty macros from LEF/DEF databases to the GDS or OASIS output. This option can only be used with the -system LEFDEF argument. When you apply this option, macros that do not have contents (empty macros) are not included as instances in the output layout and the tool issues the following message for each empty macro:

```
WARNING: [FDI1136] <file>.def[<line>] : COMPONENT <macro> : The  
MACRO COMPONENT <macro> could not be found in the input LEF files.
```

If you do not specify this option, which is the default behavior, then the empty instances are included in the output. In this case, the tool issues the following warning:

```
WARNING: [FDI1173] <file>.def[<line>] : COMPONENT <macro> refers to  
empty macro. Recommend to run with -noEmptyMacros option to avoid  
instantiation of empty <macro> cells.
```

- **-noEmptyVias**

An optional parameter that specifies not to translate empty vias from LEF/DEF databases to the GDS or OASIS output. This option can only be used with the -system LEFDEF argument.

- **-noPinShape**

An optional parameter that prevents the shapes associated with a pin from being written when the [-annotatePins](#) TEXT option is used. This applies to all the pin shapes for every cell in a layout. The -annotatePins PROPERTY option may not be used with -noPinShape. When both options are issued, the tools return this error:

```
ERROR: -noPinShape cannot be used with -annotatePins PROPERTY.
```

Assuming that a pin named DOUT exists in the design and there are two shapes associated with the pin, then the tool uses the following criteria for output:

- When both -noPinShape and -annotateNets are not specified, two shapes are written.
- When -noPinShape is not specified and the -annotatePins TEXT option is specified, two shapes and a text object named (DOUT) are written.
- When -noPinShape and the -annotatePins TEXT option are specified, only the text object named (DOUT) is written.

- -noText

An optional argument that specifies to ignore all text objects present in the input database. By default, text objects are read and translated.

- -oamap *file*

An optional argument set that specifies the path to an OpenAccess layer mapping file. This mapping format can be used for OA or LEF/DEF systems. For LEF/DEF systems, the -oamap format can only translate LEF files. DEF files are not supported. See “[OpenAccess Layer Mapping File](#)” on page 169 for details.

This option is mutually exclusive with the -map and -layerMap, -outputBlockages, and -prBoundaryLayer options.

The -objectmap option can only be used with the -oamap option for legacy mapping and with automatic mapping. When you specify -oamap with automatic mapping, the via instances in LEF obstructions are skipped and the tool issues the following warning message once:

```
WARNING_1209: The VIAs of LEF OBS section are not supported when
-oamap option is specified. Ignoring OBS VIAs
```

- -objectMap *filename*

An optional parameter set that specifies how blockage objects are mapped to the output. Blockage objects that are not listed in the *filename* are not output. The -objectMap argument overrides both the -outputBlockages and -prboundarylayer arguments, if specified.

For additional information see “[LEF/DEF FDI Usage](#)” on page 130.

To view the file format and syntax, see “[Object Map File](#)” on page 176.

This option is mutually exclusive with the -map option.

- -outputBlockages

An optional parameter that specifies to output shapes from blockage objects in the database. By default, these objects are not output during translation. Blockages are output to the corresponding “drawing” layer when the -outputBlockages option is used. The -objectMap argument overrides the -outputBlockages argument and can be used to specify an output layer other than the “drawing” layer.

The -map and -outputBlockages options are mutually exclusive.

- -outputFills

This option is no longer used.

- -prboundarylayer *layer_number* [*datatype*]

An optional parameter set that specifies the layer number and datatype where the OA prboundary shapes are written. This argument set only applies when using the -layermap and -objectmap mapping formats. The tool writes the prboundary for every cell as a shape on the specified *layer_number* and *datatype* in the corresponding cell. When the *datatype* is not specified it is written on datatype 0. The -objectMap argument overrides the -prboundarylayer argument.

For **OA**, it outputs the prboundary.

- -precision { 100 | 200 | 400 | 800 | 1000 | 2000 | 4000 | 8000 | 10000 | 20000 }

An optional parameter set that specifies the precision value of the output database. The precision is the number of database units per user unit. This option applies only to the **LEFDEF** system.

If a value other than those allowed is used, an error is issued.

Caution



If you run fdi2gds or fdi2oasis and specify a precision value that is different than the source DEF, the translated output may contain differences when compared to a stream-out from the place and route tool at the desired precision. This is due to the precision conversion process. In this case, the tool issues the following warning:

```
WARNING_1153: Converting input design data from <input precision> to  
<output precision>. Coordinate values may be truncated.
```

- -preservePath

An optional parameter that specifies all OpenAccess path types are converted to the matching output path type regardless of the path length and how it relates to the path width (see the third column of the table).

Paths are converted to polygons in the following situations:

Table 5-4. OA Paths to Polygons

Path Type	Converted to polygon		
	Always	Path length < 1/2 width	Angle of non-orthogonal path between two path segments is < 90
0	No	Yes	Yes
1	Yes	N/A	N/A
2	No	No	Yes

Table 5-4. OA Paths to Polygons (cont.)

Path Type	Converted to polygon		
	Always	Path length < 1/2 width	Angle of non-orthogonal path between two path segments is < 90
4	No, if the path includes extensions that contain less than 2 ³¹ polygons, otherwise yes.	Yes	Yes

When using fdi2oasis, type 1 paths are converted to type 2.

- **-propertyMap *filename***

An optional parameter set that specifies the pathname of a property mapping file to map input database properties to output properties. The contents of the property mapping file are system-dependent. By default, if you do not specify a mapping file, input database properties are not output. This option applies only to the OA system.

To view the file format and syntax, see “[Property Map File](#)” on page 187.

- **-switchView { *destView srcView [srcView ...]* } ...**

An optional parameter set that allows you to specify cell view mapping between an OpenAccess layout and GDS or OASIS output. If the *destView* in the OA database exists for the specified *srcView*, then the *destView* is written to the output. If *destView* does not exist, then the *srcView* is written to the output. This argument set takes priority over the -viewList argument set.

- **-template *filename***

An optional parameter set that specifies the pathname to a file that contains a list of command line arguments to be passed to fdi2gds or fdi2oasis. If you specify a template file, arguments specified on the command line have precedence over arguments specified in the template file.

To view the file format and syntax, see “[Template File](#)” on page 190.

- **-version**

An optional argument that displays the Calibre version banner and then quits. This option is used alone.

- **-viewList *viewName [viewName ...]***

An optional parameter set that specifies the order in which to open the views when opening lower-level cells. Each *viewName* argument is the name of a view, and multiple names may be separated by spaces or tabs. When there are multiple views of a lower-level cell, then the views are opened in the order specified. This option applies only to **OA** systems.

- For **OA**, the default `-viewList` is empty.
- `-viewName` *viewName*
An optional parameter set that specifies the design view to open. This option overrides the [-design view](#) parameter on the command line.
- `-window` *x1 y1 x2 y2 [x1 y1 x2 y2 ...]*
An optional parameter set that specifies coordinates of output windows. Only objects inside and touching the specified windows are output. Each window is specified by four user-unit coordinates in um. The delimiter between coordinates is a space or a tab. The *x1 y1* parameters specify the lower-left vertex of the rectangular window, and the *x2 y2* parameters specify the upper-right corner of the window.

Any number of windows can be specified. Windows may overlap or touch. The total number of window coordinates must be divisible by 4.
- `-writehier` *filename*
An optional parameter set that specifies a pathname of an output text file that contains only hierarchical information about the output design.

Description

When used without any of the optional parameters, the tool uses default translation mappings for output to GDS or OASIS format. To see which translations were applied, use the `-mapInfo` option. The output of this option allows you to determine which mappings you should customize.

The following is the priority for FDI command options from highest to lowest.

1. Command line.
2. [MGC_CALIBRE_LAYERMAP_FILE](#) and [MGC_CALIBRE_CELLMAP_FILE](#) environment variables.
3. [MGC_CALIBRE_DB_READ_OPTIONS](#) environment variable.
4. [SVRF FDI Options](#) statement.
5. `-template` file settings.

The environment variables that apply to FDI are listed in [Table 5-3](#) on page 115.

The FDI utilities do the following by default when translating input databases:

- Use the input cell names as the output cell names. You can control this behavior by using the [-cellMap](#) option.
- Preserve the text case of cell names. You can control this behavior by using the [-cellCase](#) option (this option has no effect upon the `-cellMap` option).

- Output all input objects (but not place and route blockages) using corresponding input layer numbers and a datatype record of 0. You can control this behavior by using the [-map](#) option for LEF/DEF or the [-layerMap](#) option for OA.
- Circles, donuts, and ellipses in the input database are converted to 64-edge polygons.

The command-line FDI utilities do not do the following by default:

- Perform net name annotation. You can control this behavior by using the [-annotateNets](#) option.
- Perform net type annotation. You can control this behavior with [-annotateNetType](#) option.
- Perform pin name annotation. You can control this behavior by using the [-annotatePins](#) option.
- Output metal blockages from OpenAccess databases or the OBS section of LEF/DEF databases. You can control this behavior by using the [-outputBlockages](#) option.
- Output properties from OpenAccess databases. You can control this behavior by using the [-propertyMap](#) option.

The tools rename cells when they encounter more than one cell with the same name from different libraries. If a cell Z is present in different libraries, then the tools rename the second cell as Z_1 and write it to the output. A note like this is written to the run transcript:

Note: Cell Z changed to Z_1 as Z appears in separate libraries.

The fdi2gds output does not support polygons with more than 8190 vertices. To overcome this limitation, the fdi2gds utility segments LEF/DEF polygons with more than 8190 vertices into polygons with fewer vertices in the translated GDS.

[-annotateNetType](#) Details

[Table 5-5](#) describes the net type property values written out by the FDI utilities. These net types can be used with the [-annotateNetType](#) option. The first three columns show the input database net types for the supported input formats. The fourth column shows the corresponding net type values written by the tool. The values in the fourth column are used with the TYPE keyword.

Table 5-5. Extracted Net Types

LEF/DEF Net Type	OA Net Type	Corresponding fdi2gds/fdi2oasis Property Value
+ USE SIGNAL/no use statement specified	oacSignalSigType	SIGNAL
+ USE POWER	oacPowerSigType	POWER
+ USE GROUND	oacGroundSigType	GROUND

Table 5-5. Extracted Net Types (cont.)

LEF/DEF Net Type	OA Net Type	Corresponding fdi2gds/fdi2oasis Property Value
+ USE CLOCK	oacClockSigType	CLOCK
Any other type	Any other type	SIGNAL

-cellGDS Details

When using the -cellGDS option, GDS files are read in based on the order specified. If a duplicate cell is found, only the first instance of the specified cell is used and a warning is issued. This behavior can be modified using the -inputException file's CELL_GDS_DUPLICATE_CELL option.

If you specify -cellGDS together with a -cellMap file that references a GDS input file, then each cell name in the -cellGDS file is checked for presence in the -cellMap GDS input file (call this the GDS cell map file). The following two cases then apply.

- **Case 1** — If a match is found between a -cellGDS cell definition and a -cellMap cell definition, then the -cellMap file's definition is used. In this case, only the matching cell definition from the cell map file is output. Any cells instantiated in the -cellMap file's matching definition *are not copied*.
- **Case 2** — If no match is found, then the -cellGDS cell definition is used. If such a cell cannot be properly instantiated in another cell (such as when Case 1 applies and a -cellGDS cell containing subcells is chosen from the cell map file because of a match, but the subcells are not matched) then the cell is output *at the top level*.

Input cells from either the **-design** option (for system OA) or the **-lef** option that are not found in either the -cellGDS file or the -cellMap GDS input file are output.

Assume the input database contains an instance of CELL1. Assume -cellGDS X.gds is specified, and X.gds contains the following hierarchy:

```
CELL1
-> CELL2
-> CELL3
```

Assume -cellMap cellmap.txt is specified and it contains the following entry:

```
CELL1 CELL1 Y.gds
```

Assume Y.gds has the following hierarchy:

```
CELL1
-> CELL4
-> CELL5
```

The final GDS hierarchy that is written contains the top-level cell from the input design with CELL1 copied from *Y.gds* (because this cell matched CELL1 in *X.gds*), along with CELL2 and CELL3 at the top level (because they cannot be properly instantiated in the CELL1 that was selected). Here is the output hierarchy:

```
TOP          (Top-level cell cannot be mapped)
-> CELL1     (from Y.gds because of the match with X.gds)
CELL2        (at the top level; this cell is not instantiated)
CELL3        (at the top level; this cell is not instantiated)
```

CELL4 and CELL5 of *Y.gds* are not output because they are child cells of a cell (CELL1 in this case) that matched a -cellGDS cell definition.

When the -cellGDS option is used and duplicate cell names are found in the -cellGDS files, fdi2gds searches the files in the order they are specified. The first cell definition encountered is used by default. Subsequent cell definitions at the same level of hierarchy as the first definition are ignored. However, subsequent cell definitions at different hierarchical levels are not ignored but are renamed. The following examples demonstrate what occurs.

Assume -cellGDS *cellA.gds cellB.gds cellC.gds* is specified. All of the cells occur at the same level of hierarchy in each file. Here are the cells in these files:

```
cellA.gds
-> AND2X2
-> AND2X4
-> INVX1
cellB.gds
-> OR2X2
-> OR2X4
-> AND2X2
cellC.gds
-> XOR2X2
-> XOR2X4
-> AND2X2
-> OR2X2
```

In this case, the AND cells and INVX1 are chosen from *cellA.gds*; subsequent definitions of the AND2X2 cell are ignored. The OR cells are chosen from *cellB.gds*; the subsequent definition of OR2X2 is ignored. The XOR cells are chosen from *cellC.gds*. The final output hierarchy is this:

```
TOP
-> AND2X2
-> AND2X4
-> INVX1
-> OR2X2
-> OR2X4
-> XOR2X2
-> XOR2X4
```


The following example describes how duplicate cell names are handled when they occur at different levels in the hierarchy. Assume `-cellGDS cellA.gds cellB.gds cellC.gds` is specified, but duplicate cell names appear at different levels of hierarchy. Here are the cells in these files:

```
cellA.gds
-> AND2X2
-> AND2X4
-> INVX1
cellB.gds
-> OR2X2
    -> INVX1
-> OR2X4
-> AND2X2
cellC.gds
-> XOR2X2
-> XOR2X4
    -> AND2X2
        -> INVX1
    -> OR2X2
```

In this case, the AND cells and INVX1 are chosen from *cellA.gds*. The OR cells are chosen from *cellB.gds*. The AND2X2 cell is ignored in *cellB.gds* because that cell occurs at the same level in *cellA.gds*. INVX1 is not ignored in *cellB.gds* but is renamed because it appears at a different level than in *cellA.gds*. The XOR cells are chosen from *cellC.gds*. The AND2X2, OR2X2, and INVX1 cells are not ignored but are renamed because they appear at different levels from cells of corresponding names selected earlier. The final output hierarchy is this:

```
TOP
-> AND2X2
-> AND2X4
-> INVX1
-> OR2X2
    ->INVX1_dbread_1
-> OR2X4
-> XOR2X2
-> XOR2X4
    ->AND2X2_dbread_1
        ->INVX1_dbread_2
    ->OR2X2_dbread_1
```

This renaming scheme for duplicate-named cells from differing levels of hierarchy is always used.

To maintain cell definitions of duplicate-named cells at the same level of hierarchy, you can use a `-inputException filename` with a setting of `CELLGDS_DUPLICATE_CELL 1`. The cell renaming scheme is similar to the one used to rename cells at different levels in the hierarchy. To view the input exception file format and syntax, see “[Input Exception File](#)” on page 188.

MASKSHIFT Cell Naming Conventions for LEF/DEF

When translating LEF/DEF to OASIS or GDS, the cell name can change due to MASKSHIFT statements in the DEF. The cell name output for DEF components that include the MASKSHIFT keyword is demonstrated with the following example:

Input DEF COMPONENTs

```
... my_cell + FIXED ( x y ) N
... my_cell + FIXED ( x y ) N + MASKSHIFT 100
... my_cell + FIXED ( x y ) N + MASKSHIFT 102
... my_cell + FIXED ( x y ) N + MASKSHIFT 010
```

Output Cell Names

```
my_cell
my_cell_MASKSHIFT_100
my_cell_MASKSHIFT_102
my_cell_MASKSHIFT_10
```

Note



Leading zeros in the MASKSHIFT statement are ignored. This is demonstrated by the fourth entry in the examples above.

Related Topics

[LEF/DEF FDI Usage](#)

[OpenAccess FDI Usage](#)

FDI File Reference

The FDI utilities accept several input files that control aspects of the flow that are specific to the design and environment, such as the layer mapping and object properties.

Table 5-6. FDI File Summary

FDI File	Description
Cell Mapping File	Maps cell names in the input design library to cells in the output GDS or OASIS library.
Input Exception File	Specifies input exceptions.
LEF/DEF Map File Format	Specifies how layers and objects in a LEF/DEF input design database are translated to the GDS or OASIS output database layers for fdi2gds and fdi2oasis.
Layer Map File for fdi2gds and fdi2oasis	Specifies how layers in the input design database correspond to the GDS or OASIS output database layers for fdi2gds and fdi2oasis.
Layer Map File for fdiBA	Specifies the layer correspondence between the fill data and the third-party layout database for fdiBA.
Layer Map File for DFM Databases in fdiBA	Maps layers for the backannotation fill input file in DFM database format.
Mapping Information File	This output file shows what mappings the FDI translators applied, including any user-specified mappings.
Net Properties File	Specifies net names and their corresponding net type properties.
Nets File	Contains a list of net names and net types to be annotated in the output.
Object Map File	Specifies how blockage objects are mapped to the output. Blockage objects that are not listed in this file are not output.
Property Map File	Maps the input database properties to output properties. The contents of the property mapping file are system-dependent. By default, if you do not specify a mapping file, input database properties are not output.
Template File	Specifies a list of command line arguments to be passed to fdi2gds or fdi2oasis. If you specify a template file, arguments specified on the command line have precedence over arguments specified in the template file.

LEF/DEF Mapping

fdi2gds and fdi2oasis use a different mapping format when translating LEF/DEF databases.

The -map command-line option specifies a mapping file that allows you to control how objects and layers are translated to GDS and OASIS.


LEF/DEF Map File Format	157
LEF/DEF Text Mapping	163
Rows and Sites	164
Example: LEF/DEF Map File.....	165

LEF/DEF Map File Format

Specified in: fdi2gds -map and fdi2oasis -map option.

Specifies how layers and objects in a LEF/DEF input design database are translated to the GDS or OASIS output database layers for fdi2gds and fdi2oasis.

Note

 If you do not have an existing mapping file, the fdi2gds and fdi2oasis utilities automatically create one from your input files if you translate your LEF/DEF database *without* the -map option. You can then customize this mapping file for future translations.

See “[Example: LEF/DEF Map File](#)” on page 165 for examples.

Format

A LEF/DEF layer mapping file must conform to the following formatting and syntax rules:

- The file is case-sensitive.
- Blank lines are skipped.
- Comments start with the hash symbol (#). For example, these comments are valid:

```
#this is a comment
METAL1 NET 12 0 # this is a comment
#this is a comment METAL2 NET 12 0
```


The file format for each mapping line consists of four arguments:

in_layer_object { ***in_layer_obj_type*** [*sub_type*] [,]... } ***out_layer_num*** ***out_data_type***

If you specify multiple ***in_layer_obj_type*** objects, then you must separate the arguments with a comma, for example:

```
M3 PIN,NET,SPNET 3 1
```

Note

 For text label mapping using the NAME keyword, see “[LEF/DEF Text Mapping](#)” on page 163.

Parameters

- ***in_layer_object***
Required argument that specifies the LEF/DEF layer object name from the input database, such as metal or via objects.
- ***in_layer_obj_type*** [*sub_type*] ...
Required argument set that specifies a list of objects to translate for the layer specified in the ***in_layer_object*** argument. At least one object must be specified. If you specify more than

one object, you must use commas between each value. For example, this statement outputs LEFPIN and VIAS of a specific size to layer 14, datatype 2:

```
via34 LEFPIN,VIA:SIZE:0.55x0.55 14 2
```

The valid values for the *in_layer_obj_type* argument are listed as follows:

Table 5-7. Objects and Subtypes

<i>in_layer_obj_type</i>	Supported <i>sub_type</i>	Equivalent LEF and DEF Object
ALL	:MASK: <i>maskvalue</i>	All LEF and DEF object types
BLOCKAGE	:MASK: <i>maskvalue</i>	DEF BLOCKAGES without FILLS
BLOCKAGEFILL	:MASK: <i>maskvalue</i>	DEF BLOCKAGES + FILLS
COMP	None	DEF COMPONENTS
DIEAREA	None	DEF DIEAREA
FILL	:MASK: <i>maskvalue</i>	DEF FILL without OPC
FILLOPC	:MASK: <i>maskvalue</i>	DEF FILL + OPC
LEFOBS	:MASK: <i>maskvalue</i>	LEF OBS
LEFOBSVIA	:SIZE: <i>dimension1xdimension2</i> :MASK: <i>maskvalue</i>	LEF OBS vias
LEFPIN	:MASK: <i>maskvalue</i>	LEF PIN
NET	:MASK: <i>maskvalue</i> :TYPE: <i>type</i> ¹ :netname	DEF NETS
PIN	:MASK: <i>maskvalue</i> :netname	DEF PINS
SPNET	:MASK: <i>maskvalue</i> :TYPE: <i>type</i> :netname	DEF SPECIALNETS
TRACK	:WIDTH: <i>width</i> :DIRECTION: <i>direction</i>	TRACK
VIA	:SIZE: <i>dimension1xdimension2</i> :MASK: <i>maskvalue</i> :TYPE: <i>type</i> :netname	LEF and DEF regular vias
VIAFILL	:SIZE: <i>dimension1xdimension2</i> :MASK: <i>maskvalue</i>	DEF FILL vias

Table 5-7. Objects and Subtypes (cont.)

<i>in_layer_obj_type</i>	Supported <i>sub_type</i>	Equivalent LEF and DEF Object
VIAFILLOPC	:SIZE: <i>dimension1xdimension2</i> :MASK: <i>maskvalue</i>	DEF FILL+OPC vias

1. The following values are supported for TYPE: ANALOG, CLOCK, GROUND, POWER, RESET, SCAN, SIGNAL, TIEOFF

Use the optional *subtype* argument to further refine the objects that are translated by their subtypes. You can specify the *subtype* option to include items such as via (cut) sizes, net names, or mask colors. The subtypes can be used on the same line.

Note



The FLOATING and VOLTAGE subtypes are not supported.

Use the following *subtype* syntax to specify the via size:

```
... VIA:SIZE:dimension1xdimension2 ...
```

where the arguments are described as follows:

VIA — Required keyword for the *in_layer_obj_type*.

:SIZE: — Required *subtype* keyword that specifies to translate vias objects of a specific size.

dimension1xdimension2 — Required values that indicate the size of the via cut in two dimensions. The two values are interchangeable (height or width) and are separated by the “x” character.

Use the following *subtype* syntax for MASK data:

```
... in_layer_obj_type:MASK:maskvalue ...
```

where the *maskvalue* argument must be an integer greater than or equal to zero.

Use the following *subtype* syntax for net names:

```
... {NET | SPNET}:netname ...
```

{NET | SPNET} — Required keyword that indicates the object is a DEF NET or DEF SPECIAL NET.

:*netname* — Required value that indicates the name of the net in the DEF file.

See the “Examples” section of this topic for details.

- ***out_layer_num***

Required number of the layer in the translated GDS or OASIS layout database. This value allows customized control of the output database. The *out_layer_num* value is a non-negative integer.

- ***out_datatype***

Required datatype number for the layer in the generated GDS or OASIS layout database. This value allows customized control of the output database. The ***out_datatype*** value is a non-negative integer.

Examples

Default Output

If you do not specify -map, then the fdi2gds and fdi2oasis utilities generate a mapping file in your working directory (named *fdi.map*) from your database with default settings. The default mapping file outputs objects to datatypes as follows:

Table 5-8. Default Object Mapping

LEF/DEF Object	Layer Datatype
NET	1
SPNET	2
PIN	11
LEFPIN	12
FILL	21
FILLOPC	22
VIAFILL	23
VIAFILLOPC	24
VIA	31
LEFOBS	41
BLOCKAGE	42
BLOCKAGEFILL	43

Note



See “[LEF/DEF Text Mapping](#)” on page 163 for details on text mapping defaults.

Mask numbers are prefixed to the datatype numbers. For example, a FILL object on layer M1 with mask number 2 is mapped by default to GDS layer 4, datatype 221. If there was no mask number, the FILL would be mapped to GDS layer 4, datatype 21.

Translate Objects on the Same Input LEF Layer to Different Output Layers

Translate all DEF NETS on LEF layer “METAL1” to the GDS or OASIS output database on layer 1, datatype 0:

```
METAL1 NET 1 0
```


Translate all DEF PINS on the same LEF layer to the GDS or OASIS output database on layer 2, datatype 0:

```
METAL1 PIN 2 0
```

Translate all LEF PINS on the same LEF layer to the GDS or OASIS output database on layer 3, datatype 0:

```
METAL1 LEFPIN 3 0
```

Translate Multiple Objects to the Same Layer

Translate all net, via, and special net objects from LEF layer “METAL2” to the GDS or OASIS output database on layer 6, datatype 0. The object types are comma-delimited.

```
METAL2 NET, VIA, SPNET 6 0
```

Translate All Objects

Translate all objects on layer “METAL1” to the GDS or OASIS output database on layer 11, datatype 0:

```
METAL1 ALL 11 0
```

Translate the Same Objects to Multiple Layers

The tool supports both many-to-one and one-to-many layer mappings. For example, the following statements translate all objects on layer M1 to layer 11, datatype 0 and layer 11, datatype 1.

```
M1 ALL 11 0  
M1 ALL 11 1
```

This effectively creates a copy of the layer. You can use the same feature to translate a subset of objects from a layer to different layer and datatypes. For example:

```
M3 NET, SPNET 3 0  
M3 PIN, NET, SPNET 3 1
```

In this example, layer 3, datatype 0 contains NET and SPNET objects from M3. Layer 3, datatype 1 includes the same elements as 3.0, but also contains PIN objects.

Translate Vias of a Specific Size

Use the *subtype* option to translate only the DEF VIA objects on LEF layer “VIA12” that have a cut size of 0.36 by 0.36 units.

```
VIA12 VIA:SIZE:0.36x0.36 5 0
```

Translate Objects by Net Name

Translate nets with the name ADDR_0 and special nets with the name VDD on layer “METAL1” to GDS or OASIS layer 11 datatype 5 and 11.6, respectively.

```
METAL1 NET:ADDR_0 11 5
```

```
METAL1 SPNET:VDD 11 6
```

Translate Mask Data

Translate mask color 1 of NET objects on the “METAL1” layer to layer 4, datatype 0 in the GDS or OASIS output.

```
METAL1 NET:MASK:1 4 0
```

You can translate vias of a specific size and on a specific mask by combining the subtype keywords in one line as follows:

```
METAL1 VIA:SIZE:0.36x0.36:MASK:1 11 7
```

Translate NAME Objects for Labels

You can translate text objects on different LEF objects to a specific layer as follows:

```
NAME metal1/PIN 0 5
```

In this example, NAME objects on pins on the metal1 layer are translated to layer 0, datatype 5. See the *in_layer_object* description in the Parameters section for more details. The following is an example for LEF pins:

```
NAME M1/LEFPIN 121 0
```

Translate DEF Tracks

Specify the TRACK object type to translate DEF tracks to the output. The WIDTH and DIRECTION subtypes are supported, where the WIDTH value is either “DEFAULT” or a floating-point number in microns. The DIRECTION value must be X or Y. For example:

```
M1 TRACK:WIDTH:0.02:DIRECTION:X 10 1
```

This statement translates track objects on the M1 layer that are 0.02 microns wide in the x-direction to layer 10, datatype 1.

Translate a Net or Via by Type

You can specify ANALOG, CLOCK, GROUND, POWER, RESET, SCAN, SIGNAL, TIEOFF for the NET or VIA subtype TYPE. The following statement maps POWER type nets to layer 5, datatype 0.

```
M1 SPNET:TYPE:POWER 5 0
```

LEF/DEF Text Mapping

Specified in: fdi2gds -map and fdi2oasis -map option.

Alternate syntax for the LEF/DEF map file that specifies how text labels in a LEF/DEF input design database are translated to the GDS or OASIS output database layers for fdi2gds and fdi2oasis.

Format

See “[LEF/DEF Map File Format](#)” on page 157 for details.

Text mapping statements are formatted as follows:

```
NAME layer_name/{LEFPIN | PIN | NET | SPNET} out_layer_num out_data_type
```

If the -map option is not specified and the -annotatePins or - annotateNets TEXT option are specified, then the default *fdi.map* mapping file also includes statements for the following NAME object types for PIN, LEFPIN/NET, SPNET object types:

Table 5-9. Default Text Mapping

LEF/DEF Object	Layer Datatype
NAME PIN	5
NAME LEFPIN	6
NAME NET	7
NAME SPNET	8

Parameters

- **NAME**
Required keyword that indicates a text label mapping operation.
- ***layer_name*/**
Required layer name in the LEF/DEF design. The trailing forward slash ‘/’ is required.
- **LEFPIN**
Translates text labels on LEF pins. You must specify the -annotatePins TEXT in order to use this option.
- **PIN**
Translates text labels on DEF pins. You must specify the -annotatePins TEXT in order to use this option.
- **NET**
Translates text labels on DEF nets. You must specify the -annotateNets TEXT in order to use this option.

- **SPNET**
Translates text labels on DEF special nets. You must specify the `-annotateNets TEXT` in order to use this option.
- ***out_layer_num***
Required number of the layer in the translated GDS or OASIS layout database. This value allows customized control of the output database. The *out_layer_num* value is a non-negative integer.
- ***out_datatype***
Required datatype number for the layer in the generated GDS or OASIS layout database. This value allows customized control of the output database. The *out_datatype* value is a non-negative integer.

Examples

You can translate text labels on different LEF/DEF objects to a specific layer as follows:

```
NAME metall/PIN 0 5
```

In this example, NAME objects on DEF pins on the metall layer are translated to layer 0, datatype 5. See the *in_layer_object* description in the Parameters section for more details. The following is an example for LEF pins:

```
NAME M1/LEFPIN 121 0
```

To translate all text labels on a specific LEF/DEF layer, do the following:

```
NAME M1/LEFPIN 1 0  
NAME M1/PIN 2 0  
NAME M1/NET 3 0  
NAME M1/SPNET 4 0
```

Rows and Sites

The automatically generated map file (created when translating LEF/DEF systems without the `-map` option) contains separate mapping statements for each row object in the DEF.

The mapping statements include unique *<Orientation>* and optional *<SiteName>* colon-delimited pairs that are properties of the row in the DEF. If no row objects are in the DEF file, then no active row mapping statements are output in the map file; instead they are written as comments. DEF row objects are generated in the map file even if the corresponding site is missing or invalid.

The following is an example:

```
#ROW ORIENTATION:N:SITEID:A 14 1
#ROW ORIENTATION:FE:SITEID:A 14 6
#ROW ORIENTATION:N:SITEID:B 14 11
#ROW ORIENTATION:N:SITEID:C 14 21
#ROW ORIENTATION:N:SITEID:D 14 31
#ROW ORIENTATION:W:SITEID:D 14 32
```

The layer number for all row objects is the same. The datatype value is generated as follows:

- The last digit of the datatype corresponds to the orientation starting from 1 and ending with 8. The tool uses the following orientation order: N, W, S, E, FN, FE, FS, FW.
- The datatype base value increases by 10 for each site name.

The mapping statements are sorted by the site name and then by the orientation value.

When manually specifying ROW mapping, there are three supported syntaxes:

```
ROW ORIENTATION:value layer datatype

ROW ORIENTATION:value:SITEID:value layer datatype

ROW ALL
```

For example:

```
#without site
ROW ORIENTATION:N 14 1

#with site
ROW ORIENTATION:FE:SITEID:A 14 6

#all rows
ROW ALL
```

Example: LEF/DEF Map File

The fdi2gds and fdi2oasis utilities output a default map file for LEF/DEF if you do not specify -map.

This file demonstrates the mapping for an example design:

M2	LEFPIN	4	12
M1	NET	4	1
M1	SPNET:VDD	4	2 # Separate VDD
M1	VIA	4	31
M1	LEFOBS	4	41
V1	VIA	5	31
V1	LEFOBS	5	41
M2	NET	6	1
M2	SPNET:VDD	6	2
M2	PIN	6	11
M2	LEFPIN	6	12
M2	VIA	6	31
M2	LEFOBS	6	41
V2	VIA	7	31
V2	LEFOBS	7	41
...			
...			
M8	NET	18	1
M8	PIN	18	11
M8	LEFPIN	18	12
M8	LEFOBS	18	41
COMP	ALL	24	0
DIEAREA	ALL	25	0

In particular, note that special nets on M1 with the netname VDD are output to a different datatype (layer 4 datatype 2) than other nets on M1.

The following example performs more advanced mapping.

```

V12 VIA:SIZE:0.36x0.36 5 0 #map V12 objects of size 0.36x0.36 to 5.0

M1 NET:ADDR_0 11 5 #map ADDR_0 nets on M1 to 11.5

NAME M1/PIN 0 5 #map DEF PIN text on M1 to 0.5

NAME M1/LEFPIN 121 0 #map LEF PIN text on M1 to 121.0

M1 NET:MASK:1 4 0 #map M1 nets on mask1 to 4.0

M3 NET,SPNET 3 0 #map M3 nets and spnets on M3 to 3.0

M3 PIN,NET,SPNET 3 1 #map DEF pins, nets and spnets on M3 to 3.1

```

OA Mapping

OpenAccess databases use a different set of mapping options and files than LEF/DEF systems in fdi2gds and fdi2oasis

Place and Route Layer Mapping	167
OpenAccess Layer Mapping File	169
Layer Map File for fdi2gds and fdi2oasis	172
Object Map File	176

Place and Route Layer Mapping

The place and route mapping format is used for OpenAccess databases. The columns refer to the layer name, purpose name, layer number, and datatype number.

Note



You can alternatively use the OA mapping file format that is sometimes exported from OpenAccess tools. See “[OpenAccess Layer Mapping File](#)” on page 169 for details.

By default, OpenAccess layer types are mapped to datatype 0. Specify the [-expanddatatypes](#) option to enable mapping according to the following table.

Table 5-10. OpenAccess Default Layer Mapping

OA Type	Name	Datatype
oacDrawingPurposeType	"drawing"	0
oacFillPurposeType	"fill"	1
oacSlotPurposeType	"slot"	2
oacOPCSerifPurposeType	"OPCSerif"	3
oacOPCAntiSerifPurposeType	"OPCAntiSerif"	4
oacAnnotationPurposeType	"annotation"	5
oacGapFillPurposeType	"gapFill"	6
oacRedundantPurposeType	"redundant"	7
oacAnyPurposeType	"oaAny"	8
oacNoPurposeType	"oaNo"	9
oacFillOPCPurposeType	"oaFillOPC"	10
oacCustomFillPurposeType	"oaCustomFill"	11

In certain cases, this default mapping can be inconsistent with the rules. If that occurs, the datatypes can be mapped using a -layermap file. This file can be specified using the MGC_CALIBRE_LAYERMAP_FILE environment variable.

Custom layer mapping can be specified as follows:

#Layer Name	Purpose Name	Layer Number	Datatype
METAL1	DRAWING	1	0
METAL2	DRAWING	2	0
METAL3	DRAWING	5	0

OpenAccess Layer Mapping File

Specified in: fdi2gds -oamap and fdi2oasis -oamap

Specifies how layers in an OpenAccess or LEF design database correspond to the GDS or OASIS output database layers for fdi2gds and fdi2oasis. The file specified by this option conforms to an industry-standard Cadence OA layer map file format.

The -oamap option is mutually exclusive with the -map, -layerMap, -outputBlockages, and -prBoundaryLayer options.

When translating LEF files, the LEFPIN and LEFVIA objects include the .PIN and .VIA suffixes, respectively.

Note



This format cannot be used for DEF files.

Format

An OA layer mapping file must conform to the following formatting and syntax rules:

- The first four fields are required on each line.
- The *qualifier*, *photomask_color*, and *color_state* fields are case-insensitive.
- One-to-many mapping is not supported. If there are multiple mapping statements for the same layer, then only the first mapping value is applied.
- Blank lines are skipped.
- Comments start with the hash symbol (#).

Each mapping line consists of the following space-separated arguments:

```
layer_name purpose_name stream_layer_number stream_datatype  
[material_type [mask_number]] [qualifier] [photomask_color] [color_state]
```

Parameters

- *layer_name*
Required argument that specifies a layer in the input OA database.
- *purpose_name*
Required argument that specifies the purpose name for the OA layer.

Note



For LEF, the purpose can only be pin, label, or other. The utility only uses pin and label mappings. All other purposes (such as drawing) are ignored with a warning.

- *stream_layer_number*

Required argument that specifies the layer number for the GDS or OASIS output. The number must be a positive integer or a range of positive integers. You can use a comma or a hyphen to specify ranges, for example:

- 5-9
- 1,6
- 1,5-9,106

Note



The OA mapping format does not support one-to-many mapping. If you specify multiple layers, the tool only uses the first layer.

- *stream_datatype*

Required argument that specifies the datatype number for the GDS or OASIS output. The number must be a positive integer or a range of positive integers (see *stream_layer_number* for examples).

- *material_type*

Optional argument that specifies the material type. This argument is ignored during translation.

- *mask_number*

Optional argument that specifies the mask number. This argument is ignored during translation but it must be a positive integer. This argument can only be specified if *material_type* is also specified.

- *qualifier*

Optional argument that specifies how the shapes are handled. This argument is case-insensitive. This argument is ignored for LEF translation. For OA systems, you must specify one of the following qualifier values:

- **pin** — Specifies that the shape is a pin.
- **floating** — Checks whether the specified shape is connected to any other shape for the purpose of fill and fillOPC. This qualifier value is ignored during translation.
- **cutsizes** — Ensures that the bbox value of the shape matches the specified cutsizes. The cutsizes format is given as follows:

cutSize:x:y

Where *x* and *y* are floating-point numbers that represent the size of the cut shape.

- **ignoreLPP** — Ignores the specified LPPs during mapping and does not display a missing LPP warning for them. This qualifier value is not valid with any other optional field.

- **WSPRegionType** — Enables the translation of Width Spacing Pattern (WSP) regions. These values are ignored during translation.
- *photomask_color*
Optional argument that specifies a color value for the layer. The color is specified using the string “maskXColor”, where X is an integer from 0 to 8 (for example, mask2Color). If this argument is specified, it must be followed by the *color_state* argument. This argument is case-insensitive.
- *color_state*
Optional argument that specifies whether the shape is locked. The value must be “locked” or “unlocked.” This argument is case-insensitive. This option is ignored for LEF translation.

Examples

To apply an OA mapping file, invoke the fdi2gds or fdi2oasis utilities with the -oamap argument as follows:

```
fdi2oasis -system OA -design multi_color_MASK4 test1 layout \  
-outfile out.oas -oamap all_colors.map
```

Where the *all_colors.map* map file contains statements similar to the following:

```
M4 drawing 21 0  
M4 pin 21 1 metal 4 mask1Color locked
```

Layer Map File for fdi2gds and fdi2oasis

Specified in: fdi2gds -layerMap and fdi2oasis -layerMap

Specifies how layers in the input design database correspond to the GDS or OASIS output database layers for fdi2gds and fdi2oasis.

Note



LEF/DEF designs use a different mapping file format. See “[LEF/DEF Mapping](#)” on page 156 for details.

Note



For OA databases, you can alternatively use the OA mapping file format that is sometimes exported from OpenAccess tools. See “[OpenAccess Layer Mapping File](#)” on page 169 for details.

Format

A layer mapping file must conform to the following formatting and syntax rules:

- The file is case-sensitive.
- Blank lines are skipped.
- Lines that start with # are skipped.

The file format for each mapping line is in one of these two forms:

```
input_layer_name [purpose] output_layer_number datatype  
[object_type | net_type_property]  
[mask_number | mask<N>Color {locked | unlocked}]
```

or

```
input_layer_name [purpose] output_layer_number datatype  
CUT [CUTSIZE:height:width] [mask_number | mask<N>Color {locked | unlocked}]
```

The second syntax form is used only for the CUT (via) material type. In that syntax, the *purpose* is used only for systems **LEFDEF** (if not using the -map option) or **OA**.

Parameters

- *input_layer_name*

Required name of the layer in the input layout database. By default, output shapes generated from an *input_layer_name* and optional *purpose* are mapped using their input layer numbers and a datatype record of 0.

- *purpose*

Optional purpose of the layer in the input layout database. The *purpose* corresponds to the customary purpose names in OA databases. LEF/DEF does not require a *purpose* column, but you can specify “drawing” if desired.

Note



LEF/DEF only requires the *purpose* column if you are mapping fill layers. Apply the “opcfill” layer purpose name if you are mapping FILL or FILL OPC DEF objects.

- *output_layer_number*

Required layer number of the layer in the translated GDS or OASIS layout database. This value allows customized control of the output database. The *output_layer_number* is a non-negative integer.

- *datatype*

Required datatype number of the layer in the generated GDS or OASIS layout database. This value allows customized control of the output database. The *datatype* is a non-negative integer.

- *object_type*

Optional type of object in the generated GDS or OASIS database. The object type cannot be specified with the *net_type_property*. You can control mapping based upon object type, which is specified with the optional *object_type* parameter. The supported object types are as follows:

- FILL
- NET
- PIN
- TEXT

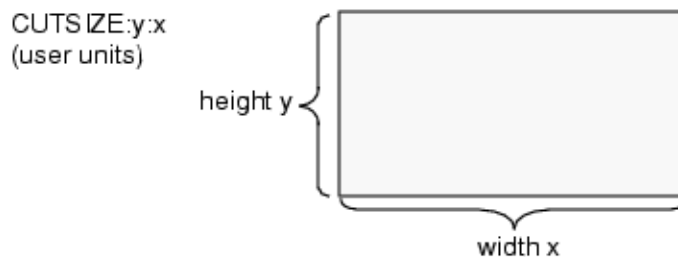
These objects are output to the layer with the “drawing” purpose and null *object_type* by default. Only the objects that satisfy the specified layer mapping are output, so it is critical to have a “drawing” entry without an object type so that all objects on a given layer that are not a specified object type are still returned as output.

- *net_type_property*

Optional type of net property in the generated GDS or OASIS database. The object type cannot be specified with the *object_type*. The *net_type_property* argument is used with -netProp. This argument acts as a filter such that only the shapes that have the specified *net_type_property* get mapped for output. If you use the -netProp option, you must annotate your -layerMap file by adding the *net_type_property* argument at the end of a line where you want net type property filtering to occur. These arguments should match the names used in the [Net Properties File](#).

- *mask_number*
Optional argument used for LEF/DEF and OpenAccess multi-patterning only. This argument is a non-negative integer less than four corresponding to the “color” of a split mask. The value 0 is not used in production flows but is accepted.
- *mask<N>Color {locked | unlocked}*
Optional argument used for LEF/DEF and OpenAccess multi-patterning only. This argument provides an alternative format to the *mask_number* syntax, where *<N>* is an integer that represents the mask number. For example, *mask1Color* and *mask2Color*. If you use this format, then you must also include the *locked* or *unlocked* keywords.
- *CUT*
Specifies that the material is a CUT material type. This syntax is supported for all input layout systems.
- *CUTSIZE:height:width*
The CUTSIZE column is optional. If CUTSIZE is not specified, then that row maps all vias on the input layer that are not mapped by some other row that contains a CUTSIZE parameter set. The optional *CUTSIZE:y:x* parameter set defines height (*y*) and width (*x*) in user units and maps only vias of the specified dimensions (see [Figure 5-1](#)).

Figure 5-1. CUTSIZE:y:x Dimensions



If an incorrect object type is used with the CUT syntax form, this warning is issued:

```
WARNING: Object type:<object> can't be specified with material  
type:CUT in layer mapping file <filename>. Ignoring the entry.
```

Examples

Example 1

Here is an example layer mapping file for system OA:

```
#layer name      purpose      output layer      datatype      object type
metall1          drawing      1                  1
via1             drawing      2                  1
metal2           drawing      3                  3

#Blockages are output to layer 1 datatype 1 when -outputBlockages is used.
metall1          1          1

#All metall1 shapes that belong to a pin are mapped to 31,102
metall1          drawing      31                102          PIN

# All metall1 text is mapped to 31,201
metall1          drawing      31                201          TEXT
```

For the preceding example, the -outputBlockages option is necessary to output the metall blockages to layer 1 datatype 1. To change the layer and datatype combination for blockages, use the -objectMap option.

Example 2

There is a separate syntax for the CUT material type. This syntax is supported for all input layout systems. Here is a corresponding layer mapping file:

```
# layer name      output layer      datatype      object      CUTSIZE:y:x
via1              10               100          CUT
via1              10               200          CUT          CUTSIZE:2:2
via1              10               300          CUT          CUTSIZE:3:3
```

In the preceding layer mapping file, all 2×2 user unit vias on layer via1 are mapped to layer 10 datatype 200. All 3×3 user unit vias on layer via1 are mapped to layer 10 datatype 300. Any via on layer via1 having dimensions that do not correspond to a row containing a CUTSIZE parameter are mapped to layer 10 datatype 100.

Object Map File

Specified in: `fdi2gds -objectMap` and `fdi2oasis -objectMap`

Specifies how blockage objects are mapped to the output. Blockage objects that are not listed in this file are not output.

The object map file is specified with the `-objectMap` argument. The object map file contents override both the `-outputBlockages` and `-prboundarylayer` arguments, if specified.

The object map file provides specialized control over the types of blockages output as compared to using `-outputBlockages` with the `-layerMap` option. When both `-outputBlockages` and `-objectMap` are specified, then `-outputBlockages` is ignored and the following warning is issued:

```
WARNING: When -outputBlockages and -objectMap are used -outputBlockages
option is ignored
```

Format

The object map file must conform to the following formatting and syntax rules:

- Blank lines are skipped.
- Lines that start with the `#` character are skipped.

The format of the lines in the object map file is as follows:

```
object_type subtype [layer_name] layer_number datatype
[mask_value | mask<N>Color]
```

Parameters

- *object_type subtype*

The allowed values for *object_type* and each associated *subtype* for LEF/DEF and OpenAccess are listed in [Table 5-11](#).

Table 5-11. LEF/DEF and OA object_type and subtype Values

object_type	subtype
Boundary	PR
	Snap
	Area
	Cluster
AreaBlockage	placement

Table 5-11. LEF/DEF and OA object_type and subtype Values (cont.)

object_type	subtype
LayerBlockage	Routing
	viaRouting
	feedthru
	Fill
	Pin
	Screen
	Slot
	Wiring
AreaHalo	placement

- *layer_name*
The *layer_name* column is optional and is not needed when the *object_type* is Boundary, AreaBlockage, or AreaHalo.
- *layer_number*
The GDS or OASIS layer number.
- *datatype*
The GDS or OASIS datatype number.
- *mask_value*
Optional mask number for double or triple patterning lithography. This allows you to specify different layer mapping based on the object's mask value.
- mask<N>Color
Optional argument used for LEF/DEF and OpenAccess multi-patterning only. This argument provides an alternative format to the *mask_value* syntax, where <N> is an integer that represents the mask number. For example, mask1Color and mask2Color.

Examples

Assuming that the input design has LayerBlockage Fill, LayerBlockage Routing and Boundary PR in the database and you specify the following object map file, then only the LayerBlockage Fill and the Boundary PR are output.

#	Type	SubType	LayerName	LNum	DType
	LayerBlockage	Fill	METAL1	31	121
	Boundary	PR		100	1

A separate section is added to the [Mapping Information File](#) to specify the mapping information for object map files. An example is shown here.

#	LayerNum	DataType	objType:subType	optionalLayerName
	31	121	LayerBlockage:Fill	METAL1
	100	1	Boundary:PR	

Cell Mapping File

Specified in: `fdi2gds -cellMap` and `fdi2oasis -cellMap`

Maps cell names in the input design library to cells in the output GDS or OASIS library.

Format

A cell mapping file must conform to the following formatting and syntax rules:

- The cell mapping file is case-sensitive and is not affected by the `-cellCase` option.
- Blank lines are skipped.
- Lines that start with the `#` character are skipped.

For the OA system, the format for each line is as follows:

```
library input_cell view_name output_cell [gds_file]
```

For the LEFDEF system, the file format is as follows:

```
input_cell output_cell [gds_file]
```

Parameters

- *library*
Specifies the input library name. For example, `/design/analog_libs`. This option does not apply for LEFDEF systems.
- *input_cell*
Specifies a cell from the input *library*. For example, `NAND2_X1`.
- *view_name*
Specifies the view name of the *input_cell*. This option does not apply for a LEFDEF system.
- *output_cell*
Name of the translated output cell in the GDS or OASIS database. For example, `nand2_x1`.
- *gds_file*
Optional argument that specifies a filename from which to read the GDS cell. This allows you to read cells from GDS files other than the file you specify in the `-design` option (for OA) or the `-lef` and `-def` options. The specified GDS file may be a `.gz` or `.Z` compressed file.

If you specify a cell mapping file with the *gds_file* argument, and you also specify the `-cellGDS` option, then cells in the *gds_file* that match cells in the cell mapping file are chosen for output. The matching cell in the file specified by `-cellGDS` is not output in this case.

Examples

Example 1

The following is an example file for system OA:

#	library name	input name	view name	output name
	Des_lib	nand2.1	layout	nand2x1
	sdes_lib	nand2.2	layout	nand2x2

Example 2

The following is an example file for system LEF/DEF:

#	input name	output name
	a1001	nand1_1
	a2001	nand2_1

Example 3

The following is an example for system OA with the *gds_file* option:

#	library	input name	view	output name	gds file
	Des_lib	nand2.1	layout	nand2x1	top.gds
	sdes_lib	nand2.2	layout	nand2x2	design.gds

Layer Names File

Output by: fdi2gds -layerNames and fdi2oasis -layerNames.

Used to view SVRF layer mapping generated by fdi2gds and fdi2oasis.

Format

The file contains SVRF code using the [Layer](#) specification statement. The format for each line is this:

```
LAYER name number
```

Parameters

- **LAYER**
SVRF statement.
- *name*
Layer name in the input database.
- *number*
Corresponding layer number of the mapped layer in the GDS or OASIS database.

Examples

```
LAYER METAL1 11  
LAYER VIA1 12  
LAYER METAL2 13
```

Mapping Information File

Specified in: `fdi2gds -mapInfo` or `fdi2oasis -mapInfo`

This output file shows what mappings the FDI translators applied, including any user-specified mappings.

Format

The output file is organized by the type of mapping performed.

Parameters

- The file includes the following sections:
 - **Cell mapping** — Contains the information about the output cells and the corresponding input database design name, cell name, and view name. The [-cellMap](#) affects this mapping. The entries are as follows:

```
output_cell_name library_name input_cell_name view_name
```

- **Layer mapping** — Contains the information about how the output layers are mapped to the input layers. The [-layerMap](#) option affects this mapping. The entries are as follows:

```
output_layer_number datatype input_layer_name purpose_name
```

- **Object mapping** — Contains information about object mapping. The [-objectMap](#) option affects this mapping:

```
layer_number datatype object_type:subtype [layer_name]
```

- **Property mapping** — Contains information on how the properties in the output database are translated from the input database properties. The [-propertyMap](#) option affects this mapping. The entries are as follows:

```
output_stream_ID input_object_type property_name
```

Examples

```
fdi2gds ... -mapInfo applied_mapping.txt

more applied_mapping.txt
...
MappedName          CellName
ADC_5bit_sc_5       ADC_5bit_sc_5
AND2_X1             AND2_X1
...
VialArray-4_SDA_UNIQ_1  VialArray-4_SDA_UNIQ_1
VialArray-4_SDA_UNIQ_11 VialArray-4_SDA_UNIQ_11

VLayerNum    DataType    LayerName    PurposeName
4            0           M1           drawing
22           0           M10          drawing
6            0           M2           drawing
8            0           M3           drawing
10           0           M4           drawing

LayerNum      DataType    objType:subType    optionalLayerName
...

StreamId      ObjectName    PropertyName
...
```

Net Properties File

Specified in: `fdi2gds -netProp` or `fdi2oasis -netProp`

Specifies net names and their corresponding net type properties.

This file is used in conjunction with a -layerMap file that contains *net_type_property* arguments that match the names used in the net properties file. When used in this way, only the shapes that have the associated net type values are output.

Format

The net properties file must conform to the following formatting and syntax rules:

- Blank lines are skipped.
- Lines that start with the # character are skipped.
- The file is case-sensitive

The format of the lines in the net properties file is as follows:

```
net_path net_type_property
```

Parameters

- *net_path*
Name of a net in the design, for example TIMINGNET.
- *net_type_property*
Property of the net, for example ADDR_net.

Examples

The following is an example of a net properties file:

```
#net_path      net_type_property

net1           TIMINGNET
net2           CRITICALPATH
x1/net3        TIMINGNET
```

So, if your layer map file has an entry like this:

```
metal1        1        2        CRITICALPATH
```

then only metal1 objects that belong to net2, and that have the CRITICALPATH net type property, are mapped to layer 1 datatype 2. See [Table 5-5](#) in “[Complete FDI Command Line Syntax](#)” on page 133 for a list of net type property values.

Nets File

Specified in: fdi2gds -nets and fdi2oasis -nets

Contains a list of net names and net types to be annotated in the output.

The list of net names specified by the nets file supersedes net names specified by [-annotateNets](#) with the NET option. The list of type names specified by the nets file supersedes type names specified by [-annotateNetType](#) with the TYPE option.

The nets file supports the FDI_NET_NAMES and FDI_NET_TYPES keywords. FDI_NET_NAMES indicates net names. FDI_NET_TYPES indicates net types. One or both of these parameters can be specified in the nets file.

If you specify -nets *filename*, you must also specify -annotateNets or -annotateNetType. If either of these options are not applied, the tool issues a warning and the nets file is ignored.

Format

The nets file must conform to the following formatting and syntax rules:

- Blank lines are skipped.
- Lines that start with the # character are skipped.
- Net names and types are case-sensitive.
- The wildcard character “*” may be used within net and type names.
- Using “*” alone is not supported.

The form of the lines in the nets file is as follows:

```
FDI_NET_NAMES
net_name
...

FDI_NET_TYPES
POWER | GROUND | SIGNAL | CLOCK
...
```

Parameters

- **FDI_NET_NAMES**
Keyword that indicates the start of a list of net names.
- *net_name*
Specifies a net name. Each net must appear on a new line below the FDI_NET_NAMES keyword.

- **FDI_NET_TYPES**

Keyword and list of net types. The net types appear on new lines and are applied to the nets under the FDI_NET_NAMES keyword in the order that they are written. The following net types are supported:

- POWER
- GROUND
- SIGNAL
- CLOCK

Examples

```
FDI_NET_NAMES  
NET278  
NET284
```

```
FDI_NET_TYPES  
POWER  
GROUND
```

Property Map File

Specified in: `fdi2gds -propertyMap` and `fdi2oasis -propertyMap`

Maps the input database properties to output properties. The contents of the property mapping file are system-dependent. By default, if you do not specify a mapping file, input database properties are not output.

This file is specified with `-propertyMap` option and applies only to OA.

If you specify a property mapping file, the translator creates properties with *object_type* and *property_name* combinations found in the mapping file by using the specified *stream_ID*. Properties not found in the map file are not output.

Format

The property map file must conform to the following formatting and syntax rules:

- Blank lines are skipped.
- Lines that start with the `#` character are skipped.

The format of the lines in the property map file is as follows:

```
stream_ID object_type property_name
```

Parameters

- *stream_ID*
Number of the stream ID in the OA layout.
- *object_type*
The valid *object_type* values are these:

AnyObject	Polygon
ArrayInst	Rect
Instance	ScalarInst
Path	Text

- *property_name*
Name of the property to map.

Examples

#	stream_ID	object_type	property_name
	1	Rect	instP1
	2	Polygon	poly1
	3	Path	sigName

Input Exception File

Specified in: `fdi2gds -inputException`

Specifies input exceptions.

If a duplicate cell name is found but resides at a different level of hierarchy from a selected cell, the cell with the duplicate name is used and receives the suffix: `_dbread_<integer>`. The *integer* starts at 1 and increases by one each time a duplicate name is found at a different hierarchical level. Duplicate-named cells from differing levels of hierarchy are always renamed. See [-cellGDS](#) for a complete discussion of this behavior.

Format

The input exception file must conform to the following formatting and syntax rules:

- Blank lines are skipped.
- Lines that start with the `#` character are skipped.

The format is as follows:

```
CELLGDS_DUPLICATE_CELL severity
CELLGDS_PRECISION severity
```

Parameters

- `CELLGDS_DUPLICATE_CELL severity`

Controls which cell definition is chosen when duplicate cell names are found at the same level of hierarchy. By default, the first cell definition encountered in the `-cellGDS` files is chosen. These files are processed in the order they are specified.

Severities:

- `0` — Use the first cell definition encountered and warn. This is the default.
- `1` — Quietly use the first cell definition encountered.
- `2` — Pick the first cell definition and rename duplicate cells at the same level of hierarchy using the suffix: `_dbread_<integer>`. The integer is a serial number starting with 1 and increasing by 1 for each duplicate. A note is issued.
- `3` — Fatal error with the following message:

```
ERROR: Duplicate cell definition for cell present in file and
file.
```

- `CELLGDS_PRECISION severity`

Controls how a difference in database precisions is handled. The primary input database precision is always used. By default, `fdi2gds` places the `-cellGDS` layout and scales it appropriately.

Severities:

- 0 — Place the -cellGDS design with no magnification and warn.
- 1 — Magnify the -cellGDS design by the factor: database precision / -cellGDS file precision and issue a message describing the magnification. This is the default.
- 2 — Fatal error.

Examples

```
CELLGDS_DUPLICATE_CELL 3  
CELLGDS_PRECISION 0
```

Template File

Specified in: `fdi2gds -template` and `fdi2oasis -template`

Specifies a list of command line arguments to be passed to `fdi2gds` or `fdi2oasis`. If you specify a template file, arguments specified on the command line have precedence over arguments specified in the template file.

Format

The template file must conform to the following formatting and syntax rules:

- Each argument and its value(s) must appear on a single line.
- Each token of an argument must be separated by either a space or a tab
- Blank lines are skipped.
- Lines that start with the `#` character are skipped.

The format of the lines in the template file is as follows:

```
fdi_command_argument value  
...
```

Parameters

- *fdi_command_argument*
An argument from the list of FDI arguments described under “[Complete FDI Command Line Syntax](#)” on page 133.
- *value*
The value(s) associated with the *fdi_command_argument*.

Examples

```
# fdi2gds arguments  
-logFile fdi2gds.log  
-cellMap Cell.map  
-layerMap layer.map  
-propertyMap Prop.map
```


Chapter 6

Via and Fill Backannotation

The fdiBA utility performs backannotation of fill data or vias stored in a specified input database.

Application of fdiBA is outlined as part of the “[Layout Enhancement](#)” in the *Calibre YieldAnalyzer and YieldEnhancer User’s and Reference Manual*.

Note

 LEF/DEF versions 5.5 and older do not support polygons in the DEF SPECIALNETS section. As a result, fdiBA converts them to path objects. Enhancement shapes with even widths and lengths could result in a 1 dbu snap during back-annotation with fdiBA.

This is not an issue for LEF/DEF versions 5.6 and newer.

Directly Backannotating to DEF Using SVRF	191
Getting Started with fdiBA	195
fdiBA Command Line Syntax	199
Mapping Files for Backannotation	204

Directly Backannotating to DEF Using SVRF

The Calibre DEF backannotation flow enables you to generate geometries (such as redundant vias, line extensions, and fill) for a LEF/DEF layout in Calibre and then export the new geometries to a full or incremental DEF file. You can read the generated DEF file into your place and route tool using to combine the enhanced geometries with your existing design.

This procedure demonstrates how to perform backannotation using a LEF/DEF design with the [DFM RDB DEF](#) statement.

Note


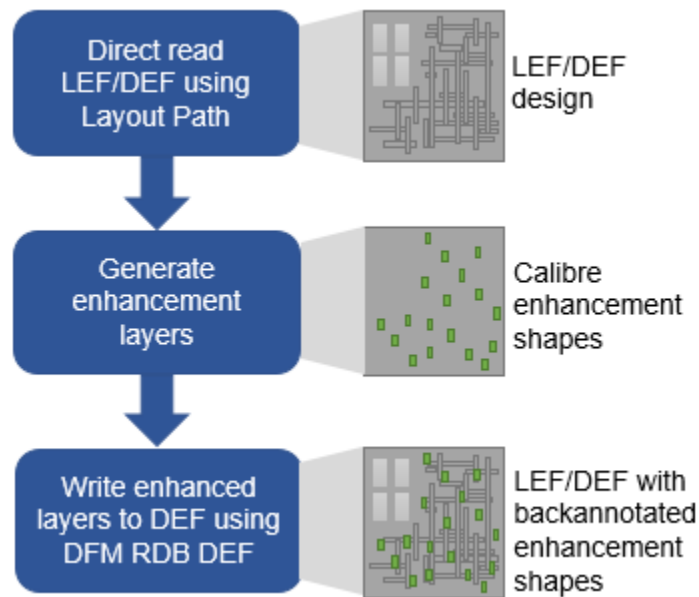
 The DFM RDB DEF statement uses a Calibre RealTime Digital license.

Figure 6-1. Workflow for Direct Backannotation to DEF



Prerequisites

- A design in LEF/DEF format.
- An SVRF rule file that generates enhancement shapes, such as vias, line extensions, or fill.

Procedure

1. In your place and route tool, export your design to a LEF/DEF database.
2. Ensure that your rule file contains the Layout System statement and that the Layout Primary statement specifies your top cell name.

The rule file should also include statements that generate enhancement geometries and a DFM RDB.

3. To generate a new incremental DEF (*out.def*) that contains enhancement shapes, add the DFM RDB DEF statement to your rule file within a rule check:

```
rule_check_name {  
    DFM RDB DEF out.def <list_of_layers>  
}
```

4. Map all layers in the input design to your DEF design using the DFM RDB DEF LAYERMAP option. For example:

```
rule_check_name {  
    DFM RDB DEF out.def list_of_layers  
  
    LAYERMAP METAL1 NET:VIA_BOT METAL1__  
    ...  
}
```


This is the minimum usage. See the [DFM RDB DEF](#) statement in the *SVRF Manual* for complete information on all optional arguments.

5. Identify the layers the comprise the via stacks. For example:

```
rule_check_name {  
  DFM RDB DEF out.def <list_of_layers>  
  
  LAYERMAP METAL1 NET:VIA_BOT METAL1__  
  ...  
  VIASTACK METAL1 VIA12 METAL2  
}
```

6. Create a summary report of the run:

```
rule_check_name {  
  DFM RDB DEF out.def <list_of_layers>  
  
  LAYERMAP METAL1 NET:VIA_BOT METAL1__  
  ...  
  VIASTACK METAL1 VIA12 METAL2  
  
  SUMMARYREPORT summary.txt  
}
```

7. Run Calibre using the rule file:

```
calibre -drc -hier -turbo -hyper $rules |& tee run.log
```

Results

The Calibre run creates an output DEF file that contains enhancement shapes generated by your Calibre rules. The tool has three output modes:

- **Enhancement** — By default, or if you specify the DEFMODE ENH argument set, the tool outputs an incremental DEF that you can import back into your place and route tool using incremental DEF read.
- **Merged DEF** — To merge enhancement shapes with an existing DEF design, specify your LEF/DEF layout with the Layout Path statement and include the DEFMODE FULL option in the DFM RDB DEF statement. The new DEF file includes both the enhancement shapes and all DEF objects from the original file.
- **Engineering Change Order DEF** — To generate an incremental DEF that only contains NET, VIA, and SPECIALNET annotations, set the DEFMODE to ECO.

If your input DEF files use different precisions than the output DEF, use the OUTPUTDEFPRECISION option to explicitly define the precision.

Examples

```
BA_FLOW {
  DFM RDB DEF out.def M1_CAL VIA12_CAL M2_CAL VIA23_CAL M3_CAL
    VIA34_CAL M4_CAL

  NEWNETMODE specialnets
  VIAMODE LEF
  DEFMODE FULL
  OUTPUTLEF out.lef

  LAYERMAP METAL1 NET:VIA_BOT M1_CAL
  LAYERMAP VIA12 NET:VIA_CUT VIA12_CAL
  LAYERMAP METAL2 NET:VIA_TOP M2_CAL
  LAYERMAP VIA23 NET:ROUTING VIA23_CAL
  LAYERMAP METAL3 NET:ROUTING M3_CAL
  LAYERMAP VIA34 NET:ROUTING VIA34_CAL
  LAYERMAP METAL4 NET:ROUTING M4_CAL
  VIASTACK METAL1 VIA12 METAL2

  NETPROPERTYNAME "net"
  SUMMARYREPORT sum.log
}
```

Getting Started with fdiBA

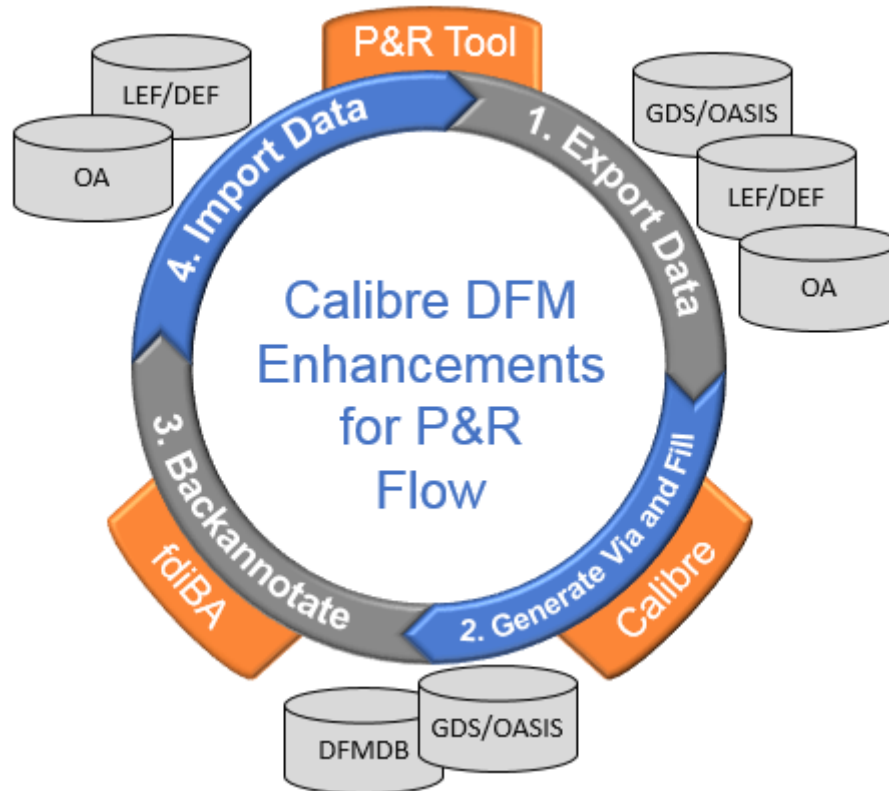
The fdiBA utility, when used with the fdi2gds and fdi2oasis utilities, can provide a closed loop flow between Calibre layout enhancements and third-party design development.

Note

The fdiBA utility requires a Calibre YieldServer license and checks it out during a run.

The following figure shows the general steps for integrating Calibre enhancement shapes into a place and route environment. This could be performed at any stage of physical implementation.

Figure 6-2. Backannotating Calibre DFM Enhancements to P&R with fdiBA



Backannotating Fill and OPC Fill to GDS or OASIS.....	195
Backannotating Line End Extensions and Vias to DEF.....	197
fdiBA Examples	197

Backannotating Fill and OPC Fill to GDS or OASIS

This procedure demonstrates how to use the fdiBA utility to backannotate fill objects to DEF.

Prerequisites

- An SVRF rule file that generates fill.

Procedure

1. Convert your LEF/DEF design to GDS.

```
fdi2gds \  
-system LEFDEF \  
-def DEF/top.def \  
-lef LEF/tech.lef LEF/*.lef \  
-outfile top.gds \  
-outputblockages \  
-outputfills \  

```

2. Generate fill shapes using Calibre.

```
calibre -drc -hier -turbo ./rules/fill.svrf
```

For this example procedure, the results are output in a layout named *fill.gds*.

3. Create a layer mapping file for fdiBA with the following format:

```
lef_layer_name [obj_type] in_layer_number in_layer_dtype [out_mask_num]
```

The object type for normal LEF/DEF FILL is not required. However, FILLOPC must include the “opcfill” object name. Mask numbers are optional and are only required to map objects to a specific mask number. The following is an example of an fdiBA mapping file:

```
M3          6 100 1  
M3 opcfill  6 101 1  
M3          6 200 2  
M3 opcfill  6 201 2
```

4. Backannotate the fill data to DEF with fdiBA:

```
fdiBA -system LEFDEF -gds fill.gds -layermap ./map/ba.map \  
-design DEF/top.def -defout filled_top.def -logfile ba.log
```

5. (Optional) Convert the backannotated DEF to GDS for viewing in Calibre DESIGNrev:

```
fdi2gds \  
-system LEFDEF \  
-def filled_top.def \  
-lef LEF/tech.lef LEF/*.lef \  
-outfile filled_top.gds \  
-outputblockages \  
-outputfills
```

6. Open the GDS with fill in Calibre DESIGNrev and verify the operation:

```
calibredrv filled_top.gds
```

Backannotating Line End Extensions and Vias to DEF

You can use fdiBA to backannotate DFM shapes that require net names to your DEF layout.

Prerequisites

- DEF layout.
- SVRF rules to stamp enhancement shapes with net information from the annotated GDS.

Procedure

1. Set the following environment variable to enable backannotation of net properties from DFM database to DEF:

```
setenv FDIBA_DFMPROPERTY_AS_NETNAME 1
```

2. Convert your DEF layout to an annotated GDS using the following fdi2gds command options:

```
$MGC_HOME/bin/fdi2gds \  
-system LEFDEF \  
-def ./DEF/top_level.def \  
-lef ./LEF/NangateOpenCellLibrary.tech.lef ./LEF/*.lef \  
-outfile top_level.gds \  
-annotateNets PROPERTY 1 \  
-layermap ./maps/fdi2gds.map \  
-flattenvias -logfile ./logs/fdi.log
```

The output of this command is *top_level.gds*. The `-annotateNets PROPERTY 1` instructs fdi2gds to annotate nets in the GDS with properties from the DEF.

3. Stamp net properties to the enhancement shapes:

```
$MGC_HOME/bin/calibre -dfm -hier -turbo -hyper ./rules/dfm.rul
```

4. Run fdiBA to backannotate the enhancement shapes with properties as net objects (from the generated DFM database in step 3) to a new DEF:

```
$MGC_HOME/bin/fdiBA \  
-system LEFDEF \  
-design ./DEF/top_level.def \  
-dfmdb dfmdb/ \  
-layermap ./maps/fdiba.map \  
-logfile ./logs/ba.log \  
-defout new.def
```

fdiBA Examples

The fdiBA utility backannotates to LEF/DEF or OpenAccess third-party databases.

These examples are typical of **-dfmdb** and **-gds** option usage.

```
# fdiBA with -dfmdb specified

fdiBA -system LEFDEF -design ./test_DFM_fdiBA.def \
-defout ./test_DFM_FILL_fdiBA.def -dfmdb dfmdb.fill \
-layermap fdiBA_lefdef_layer.map -logFile fdiBA_fill_bano.log

# fdiBA with -system OA specified
# backannotate GDS fill shapes to OA databases

fdiBA -system OA -design two_inv.oa two_inv -gds two_inv_fill.gds \
-layermap gds2oa.lmap
```

fdiBA Command Line Syntax

The fdiBA tool resides in the Calibre tree's bin directory.

Usage

```
fdiBA -system {OA [version] | LEFDEF}
-design {{design_name top_cell_name [view_name]} | def_file_name}
{{-dfmdb file_name}
  | {-gds {file_name | directory_name} [-topcell top_cell_name] ...}
  | {-oasis {file_name | directory_name} [-topcell top_cell_name] ...} }
-layermap layer_map_file
[-criticalnetsfile criticalnet_file_name]
{[-createdef def_file [-version n.n] [-topcell cell] [-units precision]
  [-template def_file]] | [-defout defout_file_name]}
[-excludecells file_name]
[-fillhierarchy {0 | 1}]
[-filllayers layer1 [layer2 ...]]
[-groupviapath {0 | 1}]
[-includecells file_name]
[-libdefs file]
[-logfile log_file_name]
[-netlayers layer1 [layer2 ...]]
[-preserveformat {0 | 1}]
{[-vialayers layer1 layer2 layer3 [layer4 layer5 layer6] ...]
  [-addviatoleffile techFile]}
```

Arguments

- **-system** {OA [version] | LEFDEF}

A required parameter set that specifies the output format to which the input data is backannotated — OpenAccess or LEF/DEF, respectively.

The *version* parameter applies only to **OA** and currently accepts 22.43, or 22.50. The default version is 22.43.

- **-design** {{*design_name top_cell_name* [view_name]} | *def_file_name*}

A required parameter set that specifies the input layout database. The *design_name* and *top_cell_name* parameters must be specified when **-system** is set to **OA**; the *def_file_name* parameter must be specified when **-system** is set to **LEFDEF**. If the input DEF filename ends with the .gz extension, then the file is handled as compressed gunzip file.

- **-dfmdb** *file_name* | {-gds {*file_name* | *directory_name*} [-topcell *top_cell_name*] ...} | {-oasis {*file_name* | *directory_name*} [-topcell *top_cell_name*] ...}

A required parameter set that specifies a backannotation fill input file. Either **-dfmdb**, **-gds**, or **-oasis** must be selected.

When **-dfmdb** is specified, the *file_name* is a DFM database.

When **-gds** or **-oasis** is specified, then either a single layout file is specified (*file_name*) or a directory containing layout files is specified (*directory_name*). The specified *file_name* or *directory_name* contains the fill information to be added to the database. When **-gds** or **-oasis** is used, the specified files must have the same layout format as the option. The

-system type can be **LEFDEF** or **OA**. Also, the input supports hierarchical and flat fill for GDS and OASIS. Compressed files (.gz and .Z) are allowed. The parameters **-gds** and **-oasis** use a Calibre RVE license. If an Calibre RVE license is not found, a Calibre nmDRC/Calibre nmDRC-H license is used.

You can optionally specify the top cell in the GDS or OASIS layout using the **-topcell** *top_cell_name* argument set.

Note



The *top_cell_name* argument must match the top cell specified in the DEF output.

- **-layermap** *layer_mapping_file*

A required parameter set that specifies the name of a layer mapping file. The layer mapping file maps layer names in the backannotation input file to those in the input layout database.

When **-dfmdb** is used, the *layer_mapping_file* format of the follows what is shown in the section “[Layer Map File for DFM Databases in fdiBA](#)” on page 209.

When **-gds** or **-oasis** is used, the *layer_mapping_file* follows the format under “[Layer Map File for fdiBA](#)” on page 205.

- **-libdefs** *filename ...*

An optional parameter set that specifies the pathname of a *libs.def* library definition file explicitly. This option applies only to the **OA** system. Without this option, library definitions are read in the following order:

- The tool looks for a *cds.lib* file in your working directory. If the *cds.lib* file is found, the tool uses it to generate a *lib.defs* file in the MGC_TMPDIR directory and applies it at runtime. In this case, the following message is printed:

```
INFO: cds.lib has been converted to lib.defs
```

- The tool looks for a *lib.defs* file in your working directory.

- **-criticalnetsfile** *criticalnet_file_name*

An optional parameter set that specifies the critical nets filename. Backannotations are not performed on the critical nets specified in this file. The critical nets file should only contain one net name per line.

- **-createdef** *def_file*

An optional parameter set that instructs fdiBA to create a new DEF file at the specified path. The DEF file contains the data from the database specified by the **-dfmdb** argument. This option is mutually exclusive with the **-defout** and **-design** options. The **-version**, **-topcell**, **-units**, and **-template** options only apply to the **-createdef** option.

- **-version** *n.n*

An optional parameter set that specifies the version of the new DEF file exported with the **-createdef** parameter. The default version is 5.8. This option only applies to the **-createdef** command.

- `-topcell cell`

An optional parameter set that specifies the name of the DESIGN section of the DEF. The default name is the top cell of the specified DFMDB. This option only applies to the `-createdef` command.

- `-template def_file`

An optional parameter set that specifies a template for a DEF file. This option only applies to the `-createdef` command. If you do not specify this option, fdiBA uses the following DEF template by default:

```
VERSION 5.7 ;
DIVIDERCHAR "/" ;
BUSBITCHARS "[]" ;
DESIGN my_design ;
UNITS DISTANCE MICRONS 2000 ;

VIAS 0 ;
END VIAS

NETS 0 ;
END NETS

SPECIALNETS 0 ;
END SPECIALNETS

END DESIGN
```

Note



The `-units`, `-topcell`, and `-version` parameters take precedence over the values specified in the DEF template file.

- `-units precision`

An optional parameter set that specifies the DEF units for the exported DEF file. The default precision is 1000. This option only applies to the `-createdef` command.

- `-defout defout_file_name`

An optional parameter set that specifies the name of the output DEF file. This argument must be used with **-system LEFDEF**. If this argument is not specified, the output DEF file is named `<def_file_name>.calibre.def`. This option is mutually exclusive with the `-createdef` parameter. If the output DEF filename ends with the `.gz` extension, then the file is compressed in gunzip format.

- `-excludecells file_name`

An optional parameter set that specifies the name of a file containing names of cells to exclude from the run. To use this option, you must also set the `-fillhierarchy` option to 1, otherwise the tool issues a warning message. The *file_name* contains a newline-delimited list of cell names. All cells that are not in the list are used in the run. The top cell cannot be included in the file. This option is only used with OA databases. The `-excludecells` option cannot be specified with the `-includecells` option.

- **-fillhierarchy {0 | 1}**

An optional argument that backannotates fill shapes based on the hierarchy present in the DFM database (Calibre can generate fill shapes in the top cell in addition to any lower level cells). When 0 is specified (the default), fill shapes are flattened and then backannotated into the top-level cell. When 1 is specified, the fill shapes are backannotated hierarchically. The 1 option is not applicable to LEF/DEF output (**-system LEFDEF**). For LEF/DEF output, all fill shapes are flattened, then added to the FILLS section in the DEF file.

- **-filllayers *layer1* [*layer2* ...]**

An optional parameter set that specifies the names of DFM database layers to backannotate as fill layers.

- **-groupviapath {0 | 1}**

An optional parameter set used with **-system LEFDEF** that controls via grouping. If 1 is specified, paths are grouped with vias to form a valid route, which is then written to the DEF file. If 0 is specified, paths are not grouped. The default is 1.

If you set the **-addConnectingPath** option to 0 and the **-groupviapath** option to 1, the **-addConnectingPath** option is ignored

- **-includecells *file_name***

An optional parameter set that specifies the name of a file containing names of cells to use during the run. The *file_name* contains a newline-delimited list of cell names. Only cells in the list are used in the run. The **-includecells** option cannot be specified with the **-excludecells** option. This option is only used with OA databases.

- **-logfile *log_file_name***

An optional parameter set that specifies the name of an output log file. Information related to the backannotated shapes is written to the log file. A log file generated with **-gds** or **-oasis** has a different output than a log file created in a **-dfmdb** run.

- **-netlayers *layer1* [*layer2* ...]**

An optional parameter set that specifies the names of DFM database layers to backannotate as layers with connectivity.

- **-preservedefformat {0 | 1}**

An optional parameter set that controls how the DEF file is written. If 1 is specified (the default), the contents from the original DEF file are first copied to the new file, then the backannotation information is added. If 0 is specified, the content from the original DEF file is not preserved in the new file. This option only applies to LEF/DEF systems. The **-preservedefformat** option cannot be specified with **-createdef**.

- **-vialayers {*layer1 layer2 layer3* [*layer4 layer5 layer6*] ...} [-addviatoleffile *techFile*]**

An optional parameter set that specifies the names of DFM database layers (in sets of three) to backannotate as vias.

The optional `-addviatoleffile techFile` parameter set writes any new via definitions added by fdiBA to a file named *techFile.calibre.lef*. The *techFile* must be a technology LEF file. The file is only created when new via definitions are added. If `-addviatoleffile` is not specified when `-system` is LEFDEF, then new via definitions are not written out. For example, when this is specified:

```
-addviatoleffile BDG_via
```

then new via definitions are written to a file named *BDG_via.calibre.lef*.

Related Topics

[fdiBA Examples](#)

Mapping Files for Backannotation

The fdiBA utility uses its own layer mapping file.

Layer Map File for fdiBA 205

Layer Map File for DFM Databases in fdiBA..... 209

Layer Map File for fdiBA

Specified in: fdiBA -layerMap

Specifies the layer correspondence between the fill data and the third-party layout database for fdiBA.

OpenAccess and LEF/DEF version 5.8 supports annotating fill layers with +OPC to indicate that the fill shapes on these layers need OPC correction during the OPC phase. fdiBA back-annotates the DFM database layers with the +OPC construct. To enable FILL or FILL OPC backannotation to OA and LEF/DEF, you must specify the purpose of the layer as “oaFillOPC” or “opcfill”, respectively, in the layer map file.

Both file formats also support adding vias as part of the FILLS section. fdiBA adds vias into the FILLS section. When a new via placement has to be added and via definition is not present in a DEF file, fdiBA adds the via definition to the VIAS section of the DEF file.

When vias that are added to the FILLS section form an asymmetric via, they are split into a symmetric via and equivalent rectangles. The symmetric via is added in the FILLS section and the additional rectangles are added as rectangles in the FILLS section.

By default, fdiBA creates a default temporary file location in `$CALIBRE_HOME/tmp`. You can change the temporary file location by setting the environment variable `MGC_TMPDIR` to a file pathname.

Format

A layer mapping file must conform to the following formatting and syntax rules:

- The file is case-sensitive.
- Blank lines are skipped.
- Lines that start with the # character are skipped.

The contents of the layer mapping file are system-dependent. The file format for each mapping line is in one of these two forms:

```
input_BA_layer_name [out_purpose] out_layer_number out_datatype  
[mask_number]
```

or

```
input_BA_layer_name [out_purpose] out_layer_number out_datatype  
CUT [CUTSIZE:height:width] [mask_number]
```

The second syntax form is used only for the CUT (via) material type. In that syntax, the *purpose* is used only for systems **LEFDEF** or **OA**.

Parameters

- *input_BA_layer_name*

Required layer from the backannotation input file. By default, shapes from an *input_BA_layer_name* and optional *purpose* are mapped using their input layer numbers and a datatype record of 0. The *input_layout_layer_number* and *datatype* arguments are non-negative integers and allow customized control of the output.

- *out_purpose*

Optional purpose of the layer in the backannotation input file. The *out_purpose* corresponds to the customary purpose names in OA databases.

Note



LEF/DEF only requires the *purpose* column if you are mapping fill layers. Apply the “opcfill” layer purpose name if you are mapping FILL or FILL OPC DEF objects.

- *out_layer_number*

Required layer number of the layer for the output layout database. This value allows customized control of the output. The *out_layer_number* is a non-negative integer.

- *out_datatype*

Required datatype number of the layer for the output layout database. This value allows customized control of the output. The *out_datatype* is a non-negative integer.

- *mask_number*

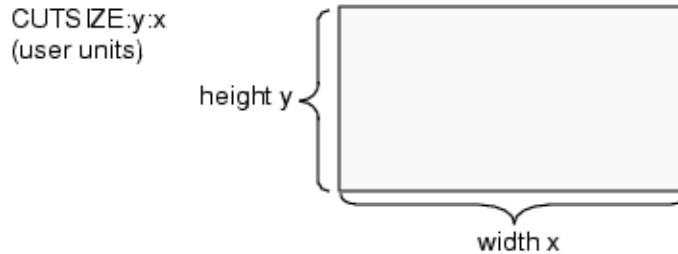
Optional argument used for OpenAccess multi-patterning only. This argument is a non-negative integer less than four corresponding to the “color” of a split mask. This number is associated with shapes on the backannotated layer. The value 0 is not used in production flows, but is accepted.

- CUT

Specifies that the material is a CUT material type. This syntax is supported for all input layout systems.

- CUTSIZE:*height:width*

The CUTSIZE column is optional. If CUTSIZE is not specified, then that row maps all vias on the input layer that are not mapped by some other row that contains a CUTSIZE parameter set. The optional CUTSIZE:*y:x* parameter set defines height (*y*) and width (*x*) in user units and maps only vias of the specified dimensions (see [Figure 6-3](#)).

Figure 6-3. CUTSIZE:y:x Dimensions for -layermap in fdiBA

If an incorrect object type is used with the CUT syntax form, this warning is issued:

```
WARNING: Object type:<object> can't be specified with material
type:CUT in layer mapping file <filename>. Ignoring the entry.
```

Examples

Example 1

Here is an example layer mapping file for system **LEFDEF** or **OA**:

```
#BA layer name    purpose    output layer    datatype    object type
metall1           drawing    1               1           1
via1              drawing    2               1           1
metal2            drawing    3               3           3
metall1           1         1               1           1

#All metall1 shapes that belong to a pin are mapped to 31,102
metall1           drawing    31              102         PIN

# All metall1 text is mapped to 31,201
metall1           drawing    31              201         TEXT
```

Example 2

There is a separate syntax for the CUT material type. This syntax is supported for all input layout systems. Here is a corresponding layer mapping file:

```
#BA layer name    output layer    datatype    object    CUTSIZE:y:x
via1              10             100         CUT        CUTSIZE:2:2
via1              10             200         CUT        CUTSIZE:2:2
via1              10             300         CUT        CUTSIZE:3:3
```

In the preceding layer mapping file, all 2×2 user unit vias on layer via1 are mapped to layer 10 datatype 200. All 3×3 user unit vias on layer via1 are mapped to layer 10 datatype 300. Any via on layer via1 having dimensions that do not correspond to a row containing a CUTSIZE parameter are mapped to layer 10 datatype 100.

Example 3

This example demonstrates how to backannotate OPC fill to a GDS output database.

Create an fdiBA layermap file that maps fill layers to layer numbers and datatypes:

```
MET1 997 0
MET1 opcfill 1 0
```

Call fdiBA with the appropriate options and specify the layer map file. The fdiBA utility creates a DEF file that contains OPC fill in the FILLS section as follows:

```
FILLS 2 ;
- LAYER MET1
RECT ( 16480 13380 ) ( 16600 13500 )
...
- LAYER MET1 + OPC
RECT ( 8480 488 ) ( 10736 1488 )
RECT ( -232 2128 ) ( 3480 5736 )
RECT ( 3128 -152 ) ( 4128 848 )
RECT ( 0 920 ) ( 1000 1000 ) ;

END FILLS
```


Layer Map File for DFM Databases in fdiBA

Specified in: fdiBA -dfmdb

Maps layers for the backannotation fill input file in DFM database format.

Format

```
dfm_database_layer layout_database_layer [layer_type] [spacing_value]
[mask_number]
```

Parameters

- *dfm_database_layer*
Specifies the name of a layer in the DFM database. DFM naming conventions like “name::<1>” are supported for this argument.
- *layout_database_layer*
Specifies the name of a layer in the database to which information is backannotated.
- *layer_type*
Optionally specifies the type of layer. Allowed values for *layer_type* are these:
 - **blockagelayer** — Blockage layer.
 - **blockagefillslayer** — Blockage fill layer.
 - **blockagespacinglayer** — Spacing layer.
 - **blockagefillsspacinglayer** — Spacing fill layer.
 - **filllayer** — Fill layer.
 - **fillopcplayer** — Fillopc layer. The string +OPC is added to the layer header in the FILLS section.
 - **fillvialayer** — Fillvia layer. A new via is added into the FILLS section. When specifying the layers, they are always specified as a set of three (lower metal layer, upper metal layer, and via layer) similar to the vialayer type.
 - **fillviaopplayer** — Fillvia layer that is also a fillopcplayer. The string +OPC is added to layer header in the FILLS section.
 - **netlayer** — Layer is a line-end extension layer.
 - **vialayer** — Layer is a via layer. Via layers are always specified in sets of three (lower metal layer, upper metal layer, and via layer).
 - **COMPRESSEDFILLLAYER** — This keyword is used in conjunction with the [DFM Fill](#) COMPRESS and HLayer COMPRESS options to backannotate compressed fills. When multiple fill layers are present, then compressed fill generation has an option of generating a single HLayer to contain the hierarchy

for all the fill layers at the same time. The value of the HLayer needs to be specified through this environment variable:

`CALIBRE_BACK_ANNOTATE_HIER_LAYER=hlayer_name`

When this option is used, the compressed fills summary is written as follows:

```
Processing compressed fill layer HIER_LAYER
-----
  Layer statistics for compressed fillvialayer  HIER_LAYER
-----
Shapes backannotated                      0
Instances backannotated                   3518872
Total number of objects backannotated     3518872
-----
Processing compressed fill layer M1_FILL_COMPRESSED
-----
  Layer statistics for compressed fillvialayer  M1_FILL_COMPRESSED
-----
Shapes backannotated                      3
Instances backannotated                   0
Total number of objects backannotated     3
-----
Processing compressed fill layer M2_FILL_COMPRESSED
-----
  Layer statistics for compressed fillvialayer  M2_FILL_COMPRESSED
-----
Shapes backannotated                      4
Instances backannotated                   0
Total number of objects backannotated     4
-----
Processing compressed fill layer M3_FILL_COMPRESSED
-----
  Layer statistics for compressed fillvialayer  M3_FILL_COMPRESSED
-----
Shapes backannotated                      3
Instances backannotated                   0
Total number of objects backannotated     3
-----
```

- *spacing_value*

Optional positive floating-point number that specifies the spacing for certain blockage layers. This option only applies when the *layer_type* argument is set to *blockagespacinglayer* or *blockagefillspacinglayer*.

- *mask_number*

Optional argument used for multi-patterning. This argument is a non-negative integer less than four corresponding to the “color” of a split mask. This number is associated with shapes on the backannotated layer. The value 0 is not used in production flows, but is accepted.

Examples

The following is a layer map file used with a DFM Database input:

```
# dfmdblayerName  layout_layer  layer_type

# Example if via layer triplets
OAM1_V1           M1           vialayer
OAM2_V1           M2           vialayer
OAV1              VIA1         vialayer
OAM2_V2           M2           vialayer
OAM3_V2           M3           vialayer
OAV2              VIA2         vialayer

# Example of via enclosure layer
OEAM1_V1          M1           netlayer
OEAM2_V1          M2           netlayer

# Example of normal fill layers
DM1Fx             M1           filllayer
DM2Fx             M2           filllayer
DM3Fx             M3           filllayer

# Example of fill opc layers
DM1FOPCx          M1           fillopclayer
DM2FOPCx          M2           fillopclayer

# Example of fill vias
OAFILLM1_V1       M1           fillvialayer
OAFILLM2_V1       M2           fillvialayer
OAFILLV1          VIA1         fillvialayer

# Example of fillviaopc layers specification
OAFILLM2_V2       M2           fillviaopclayer
OAFILLM3_V2       M3           fillviaopclayer
OAFILLV2          VIA2         fillviaopclayer
```

It is recommended that you use a layer mapping file instead of specifying layers individually using the -filllayers, -vialayers, and -netlayers options.

Appendix A

Calibre Utility Messages

Error and warning messages are issued by the fdi2gds, fdi2oasis, fdiBA, DBdiff, FastXOR, and Calibre View utilities at runtime.

FDI Messages	213
DBdiff Messages	237
FastXOR Messages	256

FDI Messages

This section describes messages issued by fdi2gds, fdi2oasis, fdiBA, and Calibre View.

Error and warning messages take the following form:

```
ERROR: <message>
WARNING: [<key>] <message>
```

where the key is a four-digit unique identifier.

By default, the tool prints each message twenty times and then suppresses additional messages. When messages are suppressed, the following informational message appears in the transcript:

```
INFO: Additional warning messages have been filtered. Use the
FDI_DBDIFF_SUPPRESS_VALUES environment variable to customize global or
individual message output.
```

To change the default messaging behavior, use the FDI_DBDIFF_SUPPRESS_VALUES environment variable. See “[FDI Environment Variables](#)” on page 114 for details.

The FDI_LEFDEF_WARNINGS_AS_ERRORS environment variable controls the behavior of the tool when it encounters certain warning messages. When it is set to a non-null value, the tool stops when it encounters the following warnings when translating a LEF/DEF design:

```
[FDI1069] [FDI1070] [FDI1102] [FDI1103] [FDI1104] [FDI1107] [FDI1111]
[FDI1112] [FDI1114] [FDI1117] [FDI1118] [FDI1119] [FDI1120] [FDI1121]
[FDI1122] [FDI1123] [FDI1124] [FDI1126] [FDI1133] [FDI1135] [FDI1136]
[FDI1139] [FDI1142] [FDI1154] [FDI1157] [FDI1158] [FDI1159] [FDI1169]
[FDI1205]
```

Use the [-inputExceptionList](#) argument set to control how the fdi2gds and fdi2oasis tools handle errors in the LEF/DEF map file.

See the following tables for all warning messages:

- [FDI Error Messages](#)
- [fdi2gds and fdi2oasis Warning Messages](#)
- [fdi2gds and fdi2oasis Map File Messages](#)
- [fdiBA Warning Messages](#)
- [Calibre View Messages](#)

Table A-1. FDI Error Messages

Message ID (key)	Message Text	Possible Causes and Solutions
FDI100	Invalid argument count for option.	Incorrect arguments or files specified on the command-line.
FDI101	File does not exist or does not have a read access.	
FDI102	Argument is mandatory.	
FDI103	Option is not supported.	
FDI1000	TOP and ALL are mutually exclusive.	Incorrect options specified on the command-line.
FDI1001	PROPERTY should be followed by <i>prop_name</i> .	
FDI1002	The <i>pnum</i> in the option should be an integer.	
FDI1003	The option should be followed by file name.	
FDI1004	The option is not supported without an environment switch.	
FDI1005	Map file could not be opened for reading.	
FDI1006	The option(s) are mutually exclusive.	
FDI1007	The option(s) are not supported, when automap is enabled.	

Table A-2. fdi2gds and fdi2oasis Warning Messages

Message ID (key)	Message Text	Possible Causes and Solutions
FDI101	File does not exists or does not have a read access.	There may be an issue with the file specified by the -cellMap option.
FDI102	Invalid entry in the reset properties list. Ignoring the entry.	Occurs when a property list contains an entry with incorrect syntax. The format must be <i>propName=propValue</i> .
FDI105	Option is not yet supported.	The option you specified may be reserved for a future release.
FDI1000	Copying data from waiver files can cause duplicate cell definitions for some cells.	Warns that the generated GDS file can contain duplicated cells. It can occur when the -waiverGDS option of the fdi2gds flow is specified.
FDI1001	Copying data from waiver files can cause final GDS to have duplicate cell definitions for some cells.	Warns that the generated GDS file can contain duplicated cells. This is caused when the -cellGds and -waiverGds options contain cells with same name.
FDI1002	Layer name is not present.	Occurs when there is a layer in the input design that is not present in layermap file or there is a layer in layermap file that is not present in input design.
FDI1003	Cell already picked up, ignoring the cell.	Occurs when the same cell exists in both the -cellGDS and -waiverGDS files.
FDI1004	Path with extension value <i>value</i> found in cell <i>cell</i> will be converted to polygon.	fdi2gds and fdi2oasis will convert paths that have extension values equal or greater than 2^{31} to polygons unless you specify the -preservePath option.
FDI1005	Empty pcell is found, an empty cell definition is added.	A pcell without geometry was found in the specified OpenAccess database.
FDI1006	Cell referenced but not defined, empty cell used.	Occurs when a cell in an input database is referenced but not defined. For an undefined cell empty cell will be used.
FDI1007	Unable to open cell in design.	Occurs when a referenced cell does not exist in the input database.
FDI1008	Renaming cell is not allowed when -cellGDS is specified or a GDS file is specified in cellMap.	Cell renaming disabled to prevent cellName conflicts.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1009	Renaming cell, as another cell is mapped in the cell map file.	Occurs when the -cellMap option is specified and there is a cell name conflict due to a cellMap file. The tool automatically resolved the conflict by renaming the cell.
FDI1010	-nets can be used only with -annotateNets or -annotateNetType. Ignoring -nets.	The -nets option was applied without specifying either -annotateNets or -annotateNetType.
FDI1011	When <i>option</i> and <i>option</i> are used, <i>option</i> is ignored.	Occurs when -objectMap is specified either with the -outputBlockages or -prboundarylayer options. The -objectMap option takes precedence.
FDI1012	<i>option</i> is supported only for <i>option</i> .	The -preservePath and -abortOnEmptyPCell options are only supported for OpenAccess databases.
FDI1013	Ignoring cellGDS value specified by environment variable as -cellGDS is already specified.	Occurs when a -cellGDS file is specified with both the environment variable and the command line parameter. In this case, the command line has higher priority.
FDI1014	Wrong format of file.	Occurs when one of the following files is incorrectly formatted: propertyMap, cellMap, layerMap, netProp, objectMap.
FDI1015	An entry already exists. Ignoring the new entry.	Occurs when checking the validity of the -objectMap file. If there is more than one entry in the file with the same object type and subtype, then the subsequent entries are ignored.
FDI1017	An entry already exists. Ignoring the new entry.	There are multiple entries with same layer name, object type, and subtype.
FDI1018	AreaBlockage does not need the layerName. Ignoring the layerName.	Occurs if there is an AreaBlockage object type and layer name specified in the Object Map file. In this case, the layer name is ignored.
FDI1020	Ignoring the entry in DIES file.	An entry of the -inputException file has incomplete field(s).
FDI1021	Entry is not valid value for DATABASE INPUT EXCEPTION SEVERITY.	The second column of the -inputException file is incorrect. For CELLGDS_DUPLICATE_CELL the entry must be 0, 1, 2, or 3. For CELLGDS_PRECISION, it must be 0, 1, or 2.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1022	Entry is not valid for DATABASE INPUT EXCEPTION SEVERITY.	The first column of the -inputException file is unsupported. The supported parameters are listed as follows: CELLGDS_DUPLICATE_CELL, CELLGDS_PRECISION, EXCLUDE_CELL_NAME, and WAIVER_PRECISION.
FDI1023	The input stream id in property map file should be an integer.	Unused warning message.
FDI1024	The environment variable <i>var</i> ignored as <i>option</i> is already set.	Occurs when the -cellMap or -layerMap options are applied on the command line when the MGC_CALIBRE_CELLMAP_FILE or MGC_CALIBRE_LAYERMAP_FILE environment variables are already defined. The command line options take precedence over the environment variables.
FDI1025	Not a valid argument.	Occurs when you specify an invalid option.
FDI1026	Not a valid argument, please check the template file.	An entry of a template file only contains the “-” character.
FDI1027	System is mentioned twice, ignoring the latter.	Occurs when the -system option is specified twice.
FDI1028	The option specified through <i>var</i> allowed only through command line. Ignored.	Some options are not supported inside a template file or with the MGC_CALIBRE_DB_READ_OPTIONS environment variable. The following options can only be specified with command line: -system, -design, -outFile, -template, -lef, -def, -64.
FDI1029	The option specified through <i>var</i> already set. Ignored.	The same option is specified in multiple locations. For example, the same option appears in the MGC_CALIBRE_DB_READ_OPTIONS environment variable or a template file and the command line.
FDI1030	Unable to open directory . Ignoring the directory.	A directory specified by the -cellGds or -waiverGds options cannot be opened.
FDI1031	Unable to open file. Ignoring the file.	A file specified by the -cellGds or -waiverGds options is not readable.
FDI1032	Version not <i>version</i> - taking version as <i>version</i> .	Occurs for OA systems when the OA version value is not 22 or 2.2.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1033	Non a-string for TEXTSTRING at location in cell on layer.	A string at the specified location contains an invalid symbol. This warning occurs only for OASIS databases.
FDI1034	Non a-string for TEXTSTRING at location in cell on layer in input file.	A string at the specified location contains an invalid symbol. This warning occurs when DBdiff reads an OASIS text record.
FDI1035	Non a-string for TEXTSTRING record in input file at record offset.	Occurs when DBdiff reads an OASIS file and the string specified by a text record is not an a-string.
FDI1036	Option is supported only for OA.	The -preservePath and -abortOnEmptyPCell options are only used for OpenAccess databases.
FDI1037	GDS file specified in the cellMapFile are ignored.	A cellMap file entry contains gds file.
FDI1038	Cell referenced but not defined, empty cell used.	The input GDS or OASIS file contains a reference to an undefined cell.
FDI1039	Found an unsupported orient for a row. Ignoring the ROW.	Unused warning message.
FDI1040	Unable to access units from the database using default values.	Unused warning message.
FDI1041	Reserved key word not allowed. The entry is ignored.	The net properties file has a specified net property of type "TEXT" or "PIN". These are reserved keywords.
FDI1042	Automatch pattern file is empty.	DBdiff issues this message when the file specified by the -automatch option is empty.
FDI1043	EXCLUDE CELL name is not located within input layout database.	DBdiff issues this message when the file specified by the -exclude_cell option contains a cell name that is absent from the input database.
FDI1044	Vertex text object in cell on layer, datatype dropped in input file.	DBdiff issues this message if there is a multi-vertex text object in the input database
FDI1045	Object at location in cell on layer datatype dropped in input file.	DBdiff issues this message if there is a zero sized or degenerate (vertex count = 1) path object in the input design.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1046	Found non-standard aref record of cell in the parent cell at offset in input file . Original ROWS= <i>val</i> COLUMNS= <i>val</i> interpreted as ROWS= <i>val</i> COLUMNS= <i>val</i>	<p>Occurs when DBdiff encounters an array structure in a GDS file that contains mismatched coordinates and row/column values. DBdiff calculates the correct array properties and uses those values for translation. This may be an unintended array structure.</p> <p>A GDS array structure contains a field that defines the number of rows and columns in the array. The array structure also contains coordinates that specify the following properties in the array:</p> <ul style="list-style-type: none"> • The origin point of the array. • The X distance from the origin point (inter_column_spacing x num_columns). • The Y distance from the origin point (inter_row_spacing x num_rows) <p>In some cases, when DBdiff reads an array structure with an angle of 90 or 270 degrees, it swaps rows to columns and columns to rows. In these cases, if the angle does not correspond to the dimensions defined by the x and y coordinates, then this warning occurs. This can also occur when the source array structure results in completely overlapping geometry (A 1xN array is defined but it results in 1x1 array).</p> <p>In both these cases, DBdiff translates the array structure with the original intent of the array in the GDS, but it may be useful to verify that the array in the original GDS file is correct.</p>
FDI1047	Non-finite or non-positive PLACEMENT magnification in input file at record offset ignored.	DBdiff issues this message when reading a GDS and the magnification of an AREF or SREF is non-positive.
FDI1048	Ignoring invalid box record at offset in input file.	DBdiff issues this message if there is a degenerate object in the input design.
FDI1049	Degenerate (vertex count) boundary at location in cell on layer dropped in input file.	DBdiff issues this message if there is a zero sized or degenerate path object in the input design.
FDI1050	Ignoring invalid boundary record at offset in input file.	DBdiff issues this message when reading a GDS database if it encounters a polygon with invalid coordinates.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1051	Cell not defined in input file.	DBdiff issues this message when reading a GDS database and there is an undefined cell in the input file.
FDI1052	Ignoring GDS_NODE structure found at offset in input file.	DBdiff issues this message when reading a GDS database. The message alerts you to a GDS_NODE structure in the GDS file. A GDS_NODE structure is a single point and it is ignored because it is not a valid geometrical object.
FDI1053	GDS_NODE structure found. Ignored.	Unused warning message. See 1052.
FDI1054	Property with empty attribute ignored in input file.	Unused warning message. DBdiff issues this message when reading OASIS files if there is a property with an empty attribute.
FDI1055	<i>Object</i> at location in cell on layer, datatype dropped in input file.	DBdiff issues this message when reading OASIS files if the input design contains a zero-radius circle.
FDI1056	<i>Object</i> at location in cell on layer, datatype ignored in input file.	DBdiff issues this message when reading OASIS files if the input design contains an XGEOMETRY record. This record is ignored, as it is not supported by DBdiff.
FDI1057	<i>Object</i> record seen and ignored in input file.	DBdiff issues this message when reading OASIS files if the input design contains an Xname, Xname_reference, Xelement record. The records are ignored, as they are not supported by DBdiff.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1069	MASKSHIFT statement used without COMPONENTMASKSHIFT MASKSHIFT is ignored...	Occurs when a MASKSHIFT statement is used without COMPONENTMASKSHIFT. The COMPONENTMASKSHIFT statement defines the layers of a component that can be shifted from the original mask colors in the LEF. The specified layers must have a LEF MASK statement to indicate that the layer is either a two or three mask layer. The MASKSHIFT statement specifies shifting of the master cell masks used in double or triple patterning for specific layers of an instance of the master cell. As a result, the MASKSHIFT statement cannot be used without COMPONENTMASKSHIFT statement. The COMPONENTMASKSHIFT statement is ignored by fdi2gds/oasis if it contains a layer that has no MASK statement in input LEF file.
FDI1070	MASKSHIFT's length is greater than defined COMPONENTMASKSHIFT length. Ignoring MASKSHIFT.	Occurs when the MASKSHIFT's length is greater than the COMPONENTMASKSHIFT's length.
FDI1071	Cell is missing.	Unused warning message.
FDI1072	Cell already defined in file, ignoring cell definition.	Occurs when multiple definitions of the same cell are found in LEF/DEF databases.
FDI1073	PINS Section will be dumped before COMPONENTS Section.	fdiBA issues this message when writing the PINS section of the DEF.
FDI1074	Net is not present in DEF file. Net is ignored...	fdiBA issues this message when the DEF file does not contain the net.
FDI1075	<i>Value</i> can't be specified with <i>value</i> in layer mapping file. Ignoring the entry...	Occurs when there are mutually exclusive records specified in the layer map file. For example, when object type is specified with material type.
FDI1076	Invalid value specified in layer mapping file. Ignoring the entry...	Occurs when invalid material types or object types are specified in the layer mapping file,
FDI1077	Invalid qualifier keyword specified in layer mapping file. Ignoring the entry...	Occurs when an incorrect object type is used with the CUT syntax form. For example when the height = 0.0 or width = 0.0.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1078	Invalid net entry ignored.	fdi2gds and fdi2oasis issue this message when a net name in the nets file contains only '*' characters.
FDI1079	Invalid entry in without tag FDI_NET_NAMES/ FDI_NET_TYPES ignored...	Occurs if the FDI_NET_NAMES or FDI_NET_TYPES keywords are not specified in the nets file.
FDI1080	Value specified but not present in the design.	Occurs when the net or net type is specified but is not present in the design.
FDI1081	File is empty.	Occurs if the nets file is empty (the file does not contain net names and net types).
FDI1083	Overriding the file with file.	fdiBA issues this message when the -includecells and -exclude cells options are specified together.
FDI1084	The units in database is different from the units in DEF file using a magnification factor.	Unused warning message.
FDI1085	Via definition does not exist in the database a new via definition created.	Unused warning message.
FDI1091	Unknown datatype.	Occurs for GDS files when there is an invalid data type.
FDI1092	Unknown record type.	Occurs for GDS files there is an invalid record type.
FDI1093	Database Units in file is different from database units of the database.	Occurs when the database units of the waiverGDS or cellGDS file is different from the database units of the input design.
FDI1094	Database Units file is different from database units of the database magnification will be applied.	Occurs when the database units of the waiverGDS or cellGDS file is different from the database units of the input design.
FDI1095	Duplicate cell definition present.	Occurs when the cell was already defined in an input GDS file.
FDI1096	Path with odd width value is detected.	Unused warning message.
FDI1101	Found an unsupported construct while processing net.	Occurs when found an unsupported construct was found while processing the net in a DEF file.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1102	COMPONENTMASKSHIFT statement was already defined. Using the first definition.	Printed when COMPONENTMASKSHIFT statement was already defined in the LEF.
FDI1103	UNITS DATABASE: The value is different from the technology value. The data was scaled to the technology value, and the scaling might cause a loss of precision.	Occurs when the UNITS DATABASE statement in a DEF file is greater than the UNITS DATABASE in the technology LEF file.
FDI1104	Ignoring duplicate macro definition for macro.	Occurs when duplicated MACRO statements are detected.
FDI1105	<file>.def[<line>]: Zero-sized rectangle is not supported. Zero-sized rectangle at location (<x>,<y>) on layer <layer> was ignored.	Occurs when zero sized rectangles are detected. Zero-width objects can result from converting a high precision layout in the place and route tool to a lower precision in DEF.
FDI1106	LAYER: An overlap layer already exists. This new definition was ignored. Verify that the MACROs use the correct OVERLAP layer.	Occurs when the overlap layer already exists.
FDI1107	NONDEFAULTRULE: A rule with this name already exists. This new definition was ignored.	Occurs when more than one NONDEFAULTRULE rule with the same name is specified in the input design.
FDI1108	VIARULE GENERATE: The via rule has no cut layer defined. This rule was ignored. Ensure that the via rule is correctly defined.	Occurs when VIARULE GENERATE statement is defined without a cut layer definition in the LEF file.
FDI1109	VIA: The via has no cut layer defined. This via was ignored. Ensure that the vias are correctly defined.	Occurs when a VIA is defined without cut layer.
FDI1110	VIA: The via is missing top or bottom routing layers. This via was ignored. Ensure that the vias are correctly defined.	Occurs when VIA is defined without the top or bottom layers in the input LEF/DEF file.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1111	VIA: The via contains multiple cut layers. This via was ignored. Ensure that the vias are correctly defined.	Occurs when the VIA has multiple cut layers defined in the LEF file.
FDI1112	VIA: The master design for this via was not found. This via was ignored. Ensure that the vias are correctly defined in the technology database.	Occurs when FDI utilities detect an undefined via.
FDI1113	VIA: The via has no 3 layers defined. This via was ignored. Ensure that the vias are correctly defined.	Occurs when the count of the layers of a via is not three.
FDI1114	Too big mask shift value is specified for layer. Ignoring mask value.	Occurs when the mask shift value of a layer is greater than its mask value.
FDI1115	NET: Zero length two points path is not allowed. This path segment was ignored. Ensure that all lengths are non zero.	Occurs when invalid coordinates of path objects are detected by FDI utilities.
FDI1116	NET: The path on layer at the point has an odd width. This width will be rounded down. Ensure that all widths are even.	Occurs when the number of points in a path is odd. To preserve odd-width paths in fdi2gds, set the FDI_PRESERVE_ODD_WIDTH_PATHS environment variable to a non-null value. Odd-width paths are not supported in OASIS.
FDI1117	Layer specified in COMPONENTMASKSHIFT Ignoring COMPONENTMASKSHIFT statement.	Occurs when reading a DEF and the COMPONENTMASKSHIFT layer does not exist or has no MASK statement.
FDI1118	The VIA is not correctly defined, please double-check the DEF file. Ignoring the VIA instance.	Occurs when reading a DEF and a VIA is not correctly defined.
FDI1119	The layer was not found. The geometry was ignored. Verify that the technology data contains the correct layer definitions.	Occurs when an undefined layer has been detected. An undefined layer may be caused by reading a LEF macro or DEF file before reading the technology LEF. The technology LEF contains layer definitions and must be read first.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1120	The via was not found. The via shapes were ignored. Ensure that the via is defined in the technology database.	Occurs when a via was not found when reading a DEF file.
FDI1121	MASK value is too large for layer. Ignoring MASK value.	Occurs when a MASK value is too large for the layer specified in a LEF/DEF database.
FDI1122	Too large maskshift is specified for layer. Ignoring MASKSHIFT statement.	Occurs when the maskshift value is too large for the specified layer in a LEF/DEF database.
FDI1123	The via is used but not defined. Ignoring.	Occurs when an undefined via has been detected.
FDI1124	Ignoring the component for which master cell name is not present.	Occurs when the master cell name is not present for the component.
FDI1125	Rounded end path encountered, converting it into square end path.	Unused warning message.
FDI1126	UNITS DATABASE: The value is different from the technology value. The technology value was used.	Occurs when the UNITS DATABASE in a DEF file is different from the UNITS DATABASE in the technology LEF file.
FDI1127	VIARULE: The via rule's via is not already defined. Ensure that the via rule is correctly defined.	Occurs when the via is not defined in VIARULE statement in the technology LEF file.
FDI1128	A via with the same name already exists. This new definition is being used.	Unused warning message.
FDI1129	Can not generate GDS property number greater than 125. 125 will be used for the property.	Occurs when a GDS property number is greater than 125.
FDI1130	Bounding box values incorrect. Outputting text properties for the cell to (0, 0) point.	Occurs when the bounding box value is incorrect, (too large or small).

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1131	Property value for GDS output should be less than 126 characters long. Truncating the string to 126 characters.	Occurs when the length of the property value in a GDS file is greater than 126 characters. The maximum length is 126 characters.
FDI1132	Bus net with base name already exists.	Occurs when fdi2gds or fdi2oasis reads a LEF and the bus net and base name are already defined.
FDI1133	UNITS DATABASE <i>value</i> defined in <i>file</i> is not one of the supported values (100 200 400 800 1000 2000 4000 8000 10000 20000). Using value of <i>value</i> . Output precision may not be accurate.	Occurs when the value of the UNITS DATABASE is not one of the supported precisions in the DEF file. In this case, your translated output may not exactly match the input database.
FDI1134	Invalid bus syntax.	Occurs when fdi2gds or fdi2oasis encounters a bus with incorrect syntax.
FDI1135	COMPONENT: The instance references its own design. This instance was ignored. Ensure that there is no recursion in instance masters.	Occurs when fdi2gds or fdi2oasis finds an instance name that is the same as its own design name in a LEF/DEF file.
FDI1136	< <i>file</i> >.def[< <i>line</i> >]: COMPONENT < <i>macro</i> >: The MACRO COMPONENT < <i>macro</i> > could not be found in the input LEF files.	Occurs when the master cell cannot be found for an instance. Check to ensure that the LEF macro for the instance in the DEF file was read. By default, empty macros are instantiated without their contents. Apply the fdi2gds or fdi2oasis -noEmptyMacros option to exclude the empty macro from the layout.
FDI1137	NET: Invalid bus syntax.	Occurs when the bus syntax is incorrect for a NET in a DEF file.
FDI1138	NET: The via was not found. The routing for this net was ignored. Check whether the vias used in this design are defined in the technology database.	Occurs when a via was not found in the DEF file.
FDI1139	VIA: The via uses more than three layers. This via was ignored. Ensure that the vias are correctly defined.	Occurs when a via has more than three layers.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1140	VIA: The via has no geometries defined. This via was ignored. Ensure that the via geometries are correctly defined.	Occurs when the via has no geometry.
FDI1141	VIA: The via has fewer than three layers. This via was ignored. Ensure that the vias are correctly defined.	Occurs when a via has less than three layers.
FDI1142	Undefined via layer.	Occurs when the cut, lower, or upper layers are not defined in a LEF file.
FDI1143	The file is used as layermap file.	Occurs in FDI utilities if the -layermap option is not specified for LEF/DEF files. In this case, the utilities use the techLib/techLib.layermap file as the layer map (if it exists).
FDI1144	Invalid line, skipping this line.	Occurs when there is an invalid line in the input file specified by the -map option.
FDI1145	Invalid non-numeric or non-positive character encountered in positive numeric field, skipping this line.	Occurs when a layer number or datatype is specified as a non-numeric or non-positive character in a map file.
FDI1146	Object Type specified with layer is not legal, skipping this object type.	Occurs when the specified layer object type is not legal.
FDI1147	Only Layer Object Type 'ALL' can be specified with, skipping this line.	Occurs when the object type is not "ALL" for either the "DIEAREA" or "COMP" layer types in the -map file for fdi2gds and fdi2oasis.
FDI1148	Invalid subtype, skipping this line.	Occurs when invalid MASK or SIZE subtypes are detected in the -map file.
FDI1149	Unsupported subtype, skipping this line.	Occurs the map file contains unsupported subtypes.
FDI1150	SIZE subtype can be specified only with VIA object type, skipping this line.	Occurs when the object type is not a VIA and the SIZE subtype is specified.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1153	Converting input design data from <i>input_precision</i> to <i>output_precision</i> . Coordinate values may be truncated.	If you run fdi2gds or fdi2oasis and specify a precision value that is different than the source DEF UNITS DATABASE statement, the translated output may contain differences when compared to a stream-out from the place and route tool at the desired precision. This is due to the precision conversion process.
FDI1151	Net name subtype can be specified only with NET and SPNET object types.	Occurs when the subtype is a net name and the object type is not NET or SPNET.
FDI1154	NONDEFAULTRULE <i>name</i> used in net <i>name</i> not defined in the DEF file.	There is a missing definition for a non-default rule.
FDI1172	The "FDI_DISABLE_AUTOMAP_SUPPORT" environment variable is ignored as "-map" option is specified.	The FDI_DISABLE_AUTOMAP_SUPPORT variable enables support for the legacy -layerMap mapping format for LEF/DEF, which is mutually exclusive with -map. This variable should not be set if using the -map option.
FDI1173	<file>.def[<line>]: COMPONENT <macro> refers to empty macro. Recommend to run with -noEmptyMacros option to avoid instantiation of empty <macro> cells.	A macro was read from a DEF file that did not match any master cell read from a LEF file. By default, the macro is instantiated empty unless you apply the -noEmptyMacros option to exclude the macro from the output.
FDI1200	Reached the maximum number of warnings. Further warnings are suppressed.	Occurs when the number of warnings exceeds 1000.
FDI2050	<file>.def[<line>]: NET VDD: A path on layer <layer> has an odd width.	This warning occurs when the FDI_PRESERVE_ODD_WIDTH_PATHS environment variable is set and the fdi2gds utility encounters an odd-width path. The utility keeps the odd-width path instead of rounding it down.

Table A-2. fdi2gds and fdi2oasis Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1228	The value <value> of “EXCLUDE_CELL_NAME” parameter of “-inputExceptionList” option is not valid.	See -inputExceptionList for a list of supported values.
FDI1229	The “EXCLUDE_CELL_NAME” parameter of “-inputExceptionList” option is not correct.	
FDI1230	The value of “message” parameter of “-inputExceptionList” option is empty.	

Table A-3. fdi2gds and fdi2oasis Map File Messages

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1144	Invalid line, skipping this line.	Review the “Format” section under “ LEF/DEF Map File Format ” on page 157.
FDI1145	Invalid non-numeric or non-positive character <value> encountered in positive numeric field, skipping this line.	One of the options has an invalid argument value.
FDI1146	Object Type <object> specified with layer <layer> is not valid, skipping this object type.	
FDI1147	Only Layer Object Type 'ALL' can be specified with DIEAREA, skipping this line.	
FDI1148	Invalid SIZE subtype, skipping this line.	See Table 5-7 under “ LEF/DEF Map File Format ” on page 157.
FDI1150	SIZESubtype can be specified only with VIA, VIAFILL, VIAFILLOPC object types, skipping this line.	SIZESubtype is only supported for VIA objects.

Table A-3. fdi2gds and fdi2oasis Map File Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1151	Net name and Net TYPE subtypes can be specified only with NET, SPNET and VIA object types.	
FDI1155	Invalid text line, skipping this line.	See “ LEF/DEF Text Mapping ” on page 163 for text-related mapping support.
FDI1170	The MASK subtype is already specified, ignoring the line.	
FDI1171	Incorrect NET TYPE <type> is specified.	
FDI1215	Mask number is out of supported range of [1; 9].	
FDI1220	Mask number is out of supported range of [1; 9], skipping the NET object type.	
FDI1221	Invalid SIZE subtype, skipping the VIA object type.	See Table 5-7 under “ LEF/DEF Map File Format ” on page 157.
FDI1222	The SIZE subtype can be specified only with VIA, VIAFILL, VIAFILLOPC and LEFOBSVIA object types, skipping the PIN object type.	
FDI1223	Net name and Net TYPE subtypes can be specified only with NET, SPNET and VIA object types, skipping the PIN object type.	
FDI1224	Incorrect NET TYPE <type> is specified, skipping the NET object type.	
FDI1225	The SIZE subtype can not be specified with NET object type, skipping the SIZE subtype for the NET object type.	

Table A-3. fdi2gds and fdi2oasis Map File Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI1226	Net name and Net TYPE subtypes can not be specified with PIN object type, skipping the NET name subtype for the PIN object type.	
FDI1231	DIRECTION subtype can be specified only with TRACK object type.	
FDI1232	The value a of WIDTH subtype is not correct, the value should be positive number or "DEFAULT" keyword.	
FDI1233	The value <value> of DIRECTION subtype is not correct, the value should be "X" or "Y".	

Table A-4. fdiBA Warning Messages

Message ID (key)	Message Text	Possible Causes and Solutions
FDI4000	The file does not exist or does not have a read access. Ignoring file.	Occurs in fdiBA when a file specified by the -gds or -oasis options does not exist or cannot be read.
FDI4001	Unable to open directory. Ignoring the directory.	Occurs in fdiBA when a directory specified by the -gds or -oasis options cannot be opened.
FDI4002	Ignoring the invalid option.	An argument specified for fdiBA is not valid.
FDI4003	-turbo could not be licensed sufficiently. Ignoring -turbo.	Occurs in fdiBA when the -turbo option is not allowed.
FDI4004	Path objects will be ignored.	Occurs in fdiBA when there are path objects in the input GDS or OASIS files.
FDI4005	Ignoring below entry in layer map file.	Occurs in fdiBA when a layer map file contains an entry with incorrect syntax.
FDI4006	Ignoring the name specified in the layer map file.	Occurs in fdiBA when a layer map file contains an entry with an invalid purpose name.
FDI4007	Unable to get at idx.	The fdiBA utility is unable to retrieve the rectangle or polygon at the specified index during back-annotation.

Table A-4. fdiBA Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI4008	Failed to create in database.	Occurs in fdiBA when it is not possible to create a rectangle or polygon in the output database.
FDI4009	Failed to color annotate in OA database.	Occurs in fdiBA when it is not possible to color a polygon or rectangle in the OA database.
FDI4010	Created cell in library.	Occurs in fdiBA when new cells are created in the OA library.
FDI4011	Layer datatype was mapped. Remapping to <i>value</i> .	Occurs in fdiBA when the mapping for the layer number and data type already exist in the layer map file.
FDI4012	Layermap entry for layer datatype not found in input.	Occurs in fdiBA when a layer specified in the layer map file does not exist in the input GDS or OASIS file.
FDI4013	Geometries present on layer, datatype are ignored as its not specified in layer map file.	Occurs in fdiBA when a layer exists in the input GDS or OASIS file, but is not specified in the layer map file.
FDI4014	Unable to access the units from the database, ensure that the precision units in the input file and the database are same.	Occurs when fdiBA cannot read the units value from the input database.
FDI4017	View name was set, resetting.	Occurs in fdiBA when the specified database is OpenAccess and the view name is not layout. In this case, the fdiBA flow resets the view name value to layout.

Table A-5. Calibre View Messages

Message ID (key)	Message Text	Possible Causes and Solutions
FDI3009	Unsupported parasitic type <i><type></i> received.	Occurs when an unsupported parasitic type is specified in a netlist file.
FDI3013	Unable to connect instance <i><inst></i> to net <i><net></i> .	Occurs when it is not possible to connect an instance to the net.
FDI3014	Could not find cell mapping for device <i><dev></i> . Ignoring instance <i><inst></i> .	Occurs when there is no cell mapping for a device in the cellMap file.
FDI3017	Could not find schematic view of cell <i><cell></i> to copy properties from.	Occurs when it is not possible to open the schematic view of a cell.

Table A-5. Calibre View Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI3018	Library <i><lib></i> not found for device <i><dev></i> . Ignoring instance <i><inst></i> .	Occurs when the library is not found for a device.
FDI3020	Could not find symbol or schematic for cell <i><cell></i> . All terminals in view <i><view></i> will be of type inputOutput.	Occurs when it is not possible to open symbol or schematic views for a cell.
FDI3021	Creating an inputOutput terminal <i><term></i> in (<i><cell></i> <i><view></i>) since matching terminal was not found on <i><symView_or_schView></i> .	Occurs when the -createUnmatchedTerminals option is specified.
FDI3022	Not creating terminal <i><term></i> in (<i><cell></i> <i><view></i>) since matching terminal was not found on <i><symView_or_schView></i> .	Occurs when the corresponding terminal does not exist in the symbol view and the -createUnmatchedTerminals option is not set.
FDI3023	Additional entry for parasitic in cellmap. ignoring parameter <i><param></i> .	Occurs when an additional entry is defined for the parasitic devices in a cellmap file.
FDI3024	Can't find layermap entry for layer <i><layer></i> in cellmap file.	Occurs when a layer from the netlist file does not exist in the cellmap file.
FDI3025	Invalid layer-purpose pair <i><layer></i> <i><purpose></i> .	Occurs when an invalid layer-purpose pair is specified in the cellmap file.
FDI3026	lpe construction <i><expr></i> is not supported.	Occurs when the syntax of the lpe construction is not correct (its value is non-numeric).
FDI3027	lpe value for device <i><dev></i> is not specified in netlist file.	Occurs when the lpe value is not specified for the device in the Calibre View netlist file.
FDI3028	Failed to open cellview <i><lib>/<cell>/<view></i> because cellView database does not exist.	Occurs when the cellview database does not exist.
FDI3029	Failed to open schematic cellview <i><lib>/<cell>/<view></i> from SKILL.	Occurs when it is not possible to open a schematic cellview from SKILL.
FDI3030	Could not evaluate property <i><prop></i> for the instance: <i><inst></i> .	Occurs when a property value is not specified for the instance.

Table A-5. Calibre View Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI3031	Cellview <i><cell></i> (<i><view></i>) does not exist in library <i><lib></i> .	Occurs when the cellview does not exist in the library.
FDI3032	Could not create instances of parasitic device " <i><dev></i> ": Cell mapping information not specified.	Occurs when the cell mapping information is not specified for an instance of a parasitic device.
FDI3033	Schematic instance <i><inst></i> not found.	Occurs when a schematic instance from the netlist file is not found.
FDI3034	Schematic instance <i><inst></i> not found, use found instance <i><inst></i> instead.	Occurs when the schematic instance is not found.
FDI3035	Terminal <i><term></i> does not exist on cells: (<i><cells></i>). Nets will be added as <i><prop></i> property values.	Occurs if the terminal does not exist in the cell but exists in the CDF properties list.
FDI3036	Terminal <i><term></i> does not exist on cells: (<i><cells></i>).	Occurs when the cell views of the matched schematic instances and generated instances are different.
FDI3037	Copying properties from schematic instance of <i><schLib:schCell:schView></i> to instance of <i><lib:cell:view></i> .	Occurs when copying properties from a schematic view to a view and the master cells are different.
FDI3038	Wrong cellmap syntax detected. Block is partly ignored.	Occurs when an unknown cellmap syntax is detected. In this case, the entire cell block is ignored.
FDI3039	Failed to evaluate expression <i><expr></i> . Leaving value <i><val></i> .	Occurs when an expression evaluation fails and the value from the netlist file is mapped as is.
FDI3044	Failed to create mapping for device <i>deviceName</i> . Placement is skipped.	Occurs when the auto-mapping process cannot determine a mapping. For example, if case-sensitive name mapping only is specified and the Calibre View netlist pins are ("A" "B" "C" "D"), while the Cadence pins are ("E" "B" "A" "D"), auto-mapping fails.

Table A-5. Calibre View Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI3045	Netlist for <i>deviceName</i> has fewer pins than schematic view. The following pins are missing and will be connected to noConn device: <i>pin_list</i> .	Occurs if auto-mapping fails because the pin count in the Calibre View netlist is less than the pin count in the schematic. The extra pins are connected to the noConn device.
FDI3046	Failed to create mapping for device <i>deviceName</i> . Netlist for <i>deviceName</i> has more pins than schematic view.	Occurs when auto-mapping fails because the pin count in the Calibre View netlist is greater than the pin count in the master for the matching schematic instance.
FDI3047	Failed to create mapping for device <i>deviceName</i> . More than one netlist pins mapped to one pin from schematic view.	Occurs when auto-mapping fails because more than one Calibre View netlist pin maps to the same pin in the Cadence schematic view.
FDI3048	Incorrect mapping of bus pins detected in cellmap file. Block is ignored.	Occurs if the order of bus pins in the cellmap file is different or the number of bus pins is not the same.
FDI3050	NET GEOMETRY: <i>file_name_line_number</i> : Invalid statement <i>statement_name</i> .	Occurs when the query_results file has an invalid specification statement.
FDI3051	The attached technology library was not found.	Occurs when the design has no attached technology file.
FDI3052	NET GEOMETRY: Could not find layer <i>layer_name</i> . Skipping objects on Calibre layer <i>calibre_layer_name</i> in the generated layout in Calibre View.	Occurs when the mapped layer is missing from the technology library.
FDI3053	NET GEOMETRY: Calibre layer <i>layer_name</i> is mapped to nil. Skipping objects on that layer.	Occurs when the Calibre layer is mapped to nil in the cellmap file.
FDI3054	NET GEOMETRY: No default purpose set. Skipping objects on Calibre layer <i>layer_name</i> .	Occurs when the technology library does not have default purpose.

Table A-5. Calibre View Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
FDI3056	NET GEOMETRY: Could not find layer-purpose <i>purpose_name</i> >. Skipping objects on Calibre layer <i>layer_name</i> .	Occurs when the Calibre layer's purpose from the cellmap file does not exist in the technology library.
FDI3059	NET GEOMETRY: Could not find net <i>net_name</i> .	Occurs when the net name in the <i>mgc_rve_net</i> statement does not exist in the generated view.
FDI3060	NET GEOMETRY: Could not find “query_results” file.	Occurs when running MaskLayout with Net Geometry flow if the query_results file has not been created within one minute of invoking the flow.
FDI3061	File < <i>file_name</i> > specified by “mgc_point_to_point” statement could not be opened for reading.	Occurs when the file specified by a “mgc_point_to_point” statement does not exist or is not readable.
FDI3062	The device < <i>device_name</i> > with pins_count pins is already defined in the cellmap file. Ignoring the previous definitions.	Occurs when the callmap contains multiple definitions of the device with the same pin count.
FDI3063	Terminal with name “< <i>terminal_name</i> >” already exists in the design.	Occurs when a “mgc_rve_cell_start” statement has more than one terminal with the same name.
FDI3064	Failed to open cellview lib/cell/view from SKILL.	Occurs when the generating view fails to open from SKILL.
FDI3065	Failed to close cellview from SKILL.	Occurs when the generating view fails to close from SKILL.
FDI3066	Failed to create pCell instances on SKILL side.	Occurs when creating pCell instances using SKILL procedures fails.
FDI3067	Failed to get pCell schematic instance < <i>instance_name</i> > from SKILL.	Occurs when retrieving the pCell schematic instance using SKILL procedures fails.
FDI3068	< <i>file_name_line_number</i> > Invalid Point2Point statement: < <i>statement_text</i> >.	Occurs when a RESISTANCE statement is not valid.

Related Topics

[FDI Environment Variables](#)

DBdiff Messages

Use these error and warning summary tables to debug errors in DBdiff.

Numbered Messages

Numbered messages are generated in the following format:

```
ERROR: <message>  
WARNING: [<key>] <message>
```

where the key is a four-digit unique identifier.

Table A-6. Numbered DBdiff Warning Messages

Message ID (key)	Message Text	Possible Causes and Solutions
2000	No cell definition	Occurs when the input GDS file contains a reference to an undefined cell.
2001	<i>value</i> is not valid value for DATABASE INPUT EXCEPTION SEVERITY.	The second column of the input exception file is incorrect. <ul style="list-style-type: none">• For CELLGDS_DUPLICATE_CELL, the entry must be 0, 1, 2, or 3.• For CELLGDS_PRECISION, the entry must be 0, 1, or 2.
2002	<i>value</i> is not valid value for DATABASE INPUT EXCEPTION SEVERITY.	The first column of the input exception file is incorrect. The following values are supported: CELLGDS_DUPLICATE_CELL, CELLGDS_PRECISION, EXCLUDE_CELL_NAME, and WAIVER_PRECISION.
2003	Ignoring the entry in -inputException file.	Occurs if an entry of the inputException file has an incomplete field.
2004	The format with two arguments per line, found in layermap file, is deprecated.	Occurs if an entry with two arguments per line is found in the layermap file. The entry is ignored.
2005	The format with one argument per line, found in file is deprecated.	Occurs if you incorrectly specify only one argument per line in the include layer file. You must specify two arguments.
2006	More than two columns found in file at line number.	Occurs when an entry with more than two arguments per line is found in the include layer file. You must specify two arguments.

Table A-6. Numbered DBdiff Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
2007	-turbo could not be licensed sufficiently. Ignoring -turbo.	Occurs when the value of the -turbo option is not allowed. The value can be different for each Linux system.
2009	Option is not supported when <i>value</i> is applied. Ignoring.	Occurs when -turbo and -transform auto_offset are specified together. -turbo is ignored.
2010	<i>option</i> is supported only for -system GDS, OASIS or OA. Ignoring.	
2011	OA layermap file is not set. Set it through either MGC_CALIBRE_CURRENT_DESIGN_LAYERMAP_FILE, MGC_CALIBRE_LAYERMAP_FILE or MGC_CALIBRE_DB_READ_OPTIONS.	Occurs if the OpenAccess layermap file is not set through MGC_CALIBRE_LAYERMAP_FILE, MGC_CALIBRE_DB_READ_OPTIONS, or MGC_CALIBRE_CURRENT_DESIGN_LAYERMAP_FILE.
2012	OA layermap file is not set. Set it through either MGC_CALIBRE_CURRENT_DESIGN_LAYERMAP_FILE, MGC_CALIBRE_LAYERMAP_FILE or MGC_CALIBRE_DB_READ_OPTIONS.	Occurs if the OpenAccess layermap file is not set through MGC_CALIBRE_REFERENCE_DESIGN_LAYERMAP_FILE, MGC_CALIBRE_DB_READ_OPTIONS, or MGC_CALIBRE_LAYERMAP_FILE.
2013	Version not 22 - Taking version as 22.	Occurs for OA systems when the OA version value is not 22, 22.43 or 22.50. Specify the OA version on the command line or with the environment variable.
2014	Some merged polygons are forming holes hence not merged.	Occurs when FastXOR encounters unmerged polygons.
2015	Current design precision does not match reference design precision. Scaling input data to precision for compare. Output result match the current design precision.	Occurs if the precisions of the current and reference designs are different.

Table A-6. Numbered DBdiff Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
2016	Maximum difference count exceeded in cell.	Occurs when the difference count per cell has exceeded the value specified by the <code>-max_cell_diff_count</code> option.
2017	Cell definition not found.	Occurs when the cell definition does not exist in the current (or reference) design, but exists in the reference (or current) design.
2018	Cell design specified in file is empty, hence not <i>value</i> .	Occurs when a cell of the current or reference design that is specified in the checkcell or cellmap file is empty.
2019	Entry not found in design.	Occurs when a layer from the layermap file or when a cell from the cellmap, checkcell, or exclude_cell files is not found in current or reference design.
2020	Cell is not flattened in design as cell is not used.	Occurs when a cell specified in the flattencell file does not exist in the current or reference design.
2021	Cell can not be flattened in design as this is topcell.	Occurs when the top cell has been detected in the flattencell file.
2022	Flattencell file has no cells that are flattened.	Occurs when the flattencell file has no defined cells.
2023	Flattencell file is empty.	Occurs when the flattencell file is empty.
2024	File is empty.	Occurs if the checkcell, exclude_cell, cellmap, or layermap file is empty.
2025	Layer was mapped to <i>value</i> . Remapping.	Occurs when the layer name already exists in the layermap file.
2026	File has no layers.	Occurs when the include_layer or exclude_layer file has no layers.
2027	No matching layer present in the input designs for layer number, data type, specified in file.	Occurs when a layer specified in the include_layer or exclude_layer file is not present in the input designs.
2028	No matching layer present in the input designs for specified in file.	Occurs when a layer specified in the include_layer or exclude_layer file is not present in the input designs.
2032	Reference design layer name <i>name</i> for layer <i>layer</i> conflicts with current design layer name <i>name</i> . Using <i>name</i>	Occurs when a layer is named differently in the current and reference designs.

Table A-6. Numbered DBdiff Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
2033	Layer name found in file is not present in layermap file, ignoring.	Occurs when the layer name specified in the include_layer or exclude_layer file does not exist in the layermap file.
2034	Layer datatype specified in layermap file cannot be mapped to layer number. Mapping to <i>value</i> .	Occurs when the -write_xor_rules option is specified and there is illegal mapping of WINDOW or WINDEL layers in the layermap file. A layer number is generated by the tool automatically.
2035	Rules cannot be generated for layer present in layermap file, as it is specified as WINDOW/WINDEL layer.	Occurs when -write_xor_rules is specified and there is a WINDOW_LAYER or WINDEL_LAYER layer type in the input layermap file.
2036	Mapping for layer (%1) in layer mapping file (%2) must be mapped for both the current and reference designs.	Occurs when the -write_xor_rules option is specified and there is a layer is specified only for the current or reference design in the input layermap file.
2037	Rules cannot be generated for layer present in layermap file, as it is not present in both the current and reference designs.	Occurs when the -write_xor_rules option is specified and there is a layer present in the layermap file for both the current and reference designs, but it is not present in the input designs.
2038	Rules cannot be generated for layer number datatype present in design, as it is not present in layermap file.	Occurs when the layer exists in the current or reference design but is not specified in the layermap file.
2039	Single physical layer number datatype is mapped to more than one layer.	Occurs when the -write_xor_rules option is specified and there is a layer number, datatype value mapped to more than one layer in the input layermap file.
2041	Error while querying design layers.	Occurs when it is not possible to find layers in current or reference design.
2042	Single physical layer number datatype is mapped to more than one layer. Ignoring.	Occurs when the -write_xor_rules option is specified and there is a a layer number and a datatype that already exists in the layermap file. The last mapping is ignored.

Table A-6. Numbered DBdiff Warning Messages (cont.)

Message ID (key)	Message Text	Possible Causes and Solutions
2048	Found duplicate instance of cell. Current offset is <i>value</i> in file.	Occurs when more than one instance of the same cell exactly overlaps. The dbdiff -removeduplicateinsts option removes duplicate instances. See “-removeduplicateinsts” for details.

Categorized Messages

The following table shows possible GDS file format errors.

Table A-7. Input GDS File Errors for DBdiff

Error message	Description
ERROR: Detected a circular reference between cell <cellname> and cell <cellname> in input file <designName>.	Two cells referenced each other.
ERROR: GDZip format read not supported yet.	GDzip is unsupported.
ERROR: Invalid GDS File <designName> !!!	There was a problem with the GDS format of the file.
ERROR: There are no cells in the GDS file. Invalid GDS File <designName> !!!	The GDS file had no cell definition and references.
ERROR: Empty GDS file <fileName>.	The file was empty.

The following table shows possible OASIS file format errors.

Table A-8. Input OASIS File Errors for DBdiff

Error message	Description
ERROR: Design has invalid ctrapezoid type in input file.	There was an invalid CTRAPEZOID record.
ERROR: Invalid OASIS file <DesignName>. Exiting.	There was a problem with the OASIS format of the file.
ERROR: cellname record for <cellnamerecord> encountered twice in input file <DesignName>.	A cell name was duplicated.
ERROR: Design has negative magnification in input file <DesignName>. Exiting.	Magnification can only be positive.
ERROR: Design has invalid property value at offset (<number>) in input file <designName>. Exiting.	There was an invalid property value at the indicated location.

Table A-8. Input OASIS File Errors for DBdiff (cont.)

Error message	Description
ERROR: Design has invalid polygon info byte at offset (<number>) in input file <designName>.	There was an invalid byte at the indicated location.
ERROR: Empty OASIS file <fileName>.	The indicated file was empty.
ERROR: 64-bit coordinate not handled in input file <fileName>. Exiting.	64-bit coordinates are not supported by DBdiff.
ERROR: compression scheme in CBLOCK compression in <fileName> is not DEFLATE CDF. Not supported.	The CBLOCK compression scheme used in an input layout file was not supported. Change the compression scheme to DEFLATE CDF.
ERROR: Non b-string of zero length at offset <number> in <filename>.	A string has zero length at the indicated location.

Table A-8. Input OASIS File Errors for DBdiff (cont.)

Error message	Description
ERROR: Unable to assign modal layer number at offset <number> in design.	A modal variable was specified incorrectly or was invalid. Refer to the latest OASIS specification to define a valid value.
ERROR: Unable to assign modal datatype at offset <number> in design.	
ERROR: Unable to assign modal repetition at offset <number> in design.	
ERROR: Unable to assign modal geometry h at offset <number> in design.	
ERROR: Unable to assign modal geometry w at offset <number> in design.	
ERROR: Unable to assign modal polygon pointlist at offset <number> in design.	
ERROR: Unable to assign modal path pointlist at offset <number> in design.	
ERROR: Unable to assign modal path half width at offset <number> in design.	
ERROR: Unable to assign modal ctrapezoid type at offset <number> in design.	
ERROR: Unable to assign modal circle radius at offset <number> in design.	
ERROR: Unable to assign modal text layer number at offset <number> in design.	
ERROR: Unable to assign modal text type at offset <number> in design.	
ERROR: Unable to assign modal text string at offset <number> in design.	
ERROR: Unable to assign modal placement cell at offset <number> in design.	
ERROR: Unable to assign modal property name at offset <number> in design.	
ERROR: Unable to assign modal property value at offset <number> in design.	

Table A-8. Input OASIS File Errors for DBdiff (cont.)

Error message	Description
ERROR: wrong layer type encountered while processing layertype record in input file <filename>.	An incorrect or unspecified layer type or datatype was detected in the layer map file.
ERROR: wrong layer datatype encountered while processing layertype record in input file <filename>.	
ERROR: Both TEXTSTRING type 5 and 6 in design <designname>.	Mixed TEXTSTRING types 5 and 6 are not allowed.
ERROR: Multiple entry for TEXTSTRING <textstring> in design.	There was an invalid TEXTSTRING record in the file.
ERROR: Multiple entry for TEXT_REFNUM <textReferenceNumber> in design.	A text reference number was repeated in the design.
ERROR: cellname record for <record> encountered twice in input file <DesignName>.	A cell name was duplicated in the input OASIS file.
ERROR: No TEXTSTRING record corresponding to text string reference number <RefNum> in design.	A TEXTSTRING record was missing for the given reference number in the file.
ERROR: No CELLNAME record corresponding to cell name reference number <RefNum> in design.	A CELLNAME record was missing for the given reference number in the file.
ERROR: Invalid CTRAPEZOID type found <trapezoidType> in cell <OasisCellName> on layer <layer> datatype <purpose> in design.	There was an invalid CTRAPEZOID record at the indicated location.
ERROR: Degenerate CTRAPEZOID at location (<x,y>) in cell <OasisCellName> on layer <layer> datatype <purpose> in design.	There was a degenerate CTRAPEZOID record at the indicated location.
ERROR: Both square and height bits are present in the rectangle record at offset <number> in design.	Both square and height bits in the rectangle record are not allowed.
ERROR: Invalid implicit closure for type <Type> POLYGON point list at file offset <number> in file.	There was an invalid POLYGON record at the indicated location.
ERROR: CBLOCK compress format error in input file <fileName>.	A CBLOCK record has improper form.

Table A-8. Input OASIS File Errors for DBdiff (cont.)

Error message	Description
ERROR: FATAL INTERNAL ERROR.	An internal error caused the DBdiff run to terminate.

The following table shows possible LEF/DEF file format errors for DBdiff.

Table A-9. Input LEF/DEF File Errors for DBdiff

Error message	Description
ERROR: -system LEFDEF must be specified with -def.	The -def option must be used with -system LEFDEF.
ERROR: -def should be followed by a DEF file.	The specified file is not a DEF file.
ERROR: -system LEFDEF must be specified with -lef.	The -lef option must be used with -system LEFDEF.
ERROR: -lef should be followed by a list of LEF files or directory.	The -lef option only supports at least one LEF file or a directory that contains the LEF files.
ERROR: -refsystem LEFDEF must be specified with -refdef.	The -refdef option must be used with -system LEFDEF.
ERROR: -refdef should be followed by a DEF file.	The specified file is not a DEF file.
ERROR: -refsystem LEFDEF must be specified with -reflef.	The -refdef option must be used with -system LEFDEF.
ERROR: -reflef should be followed by a list of LEF files or directory.	The -reflef option only supports at least one LEF file or a directory that contains the LEF files.
ERROR: Unable to create named pipe <pipe_name>:<error_message>.	This error occurs when a temporary named pipe cannot be created when reading a LEF/DEF database. Check your read/write permissions to MGC_TMPDIR.
ERROR: MGC_TMPDIR=<mgc_tmpdir_value>:<error_message>.	This error occurs when a temporary named pipe cannot be created when reading a LEF/DEF database. Check your read/write permissions to MGC_TMPDIR.

The following table shows possible DBdiff comparison errors.

Table A-10. DBdiff Comparison Errors

Error message	Description
ERROR: Cell <refTopcellname> is not present in the <design_name> design.	The specified cell does not exist in the current design.

Table A-10. DBdiff Comparison Errors (cont.)

Error message	Description
ERROR: Wrong Current Cell <curTopCellName>.	An incorrect current cell was specified upon invocation. Another possible cause of this error might be that the specified cell does not exist in the design.
ERROR: Database precisions for input designs are not identical.	There was a precision value mismatch between the reference and current design.

The following table shows possible errors when DBdiff is invoked.

Table A-11. DBdiff Invocation Errors

Error messages	Description
ERROR: The products could not be licensed sufficiently.	There may not be enough licenses available for the execution of DBdiff.
ERROR: System name can only be “GDS”, “OA”, or “OASIS”.	An invalid database was specified. Only GDS, OA, LEF/DEF, or OASIS can be specified with DBdiff.
ERROR: -system should be followed by appropriate system value.	
ERROR: <optionName> is not a valid option.	An invalid option was specified. For more information on invoking DBdiff, see the Layout Comparison Command Line Syntax .
ERROR: Framework not specified.	GDS, OA, LEF/DEF, or OASIS must be specified with the -system argument when DBdiff was invoked.
ERROR: Invalid database type <database> specified, valid types are GDS OASIS OA.	
Missing or Incorrect Invocation Arguments	These errors indicate that an argument was missing from a DBdiff invocation. For more information on invoking DBdiff, see the Layout Comparison Command Line Syntax .

Table A-11. DBdiff Invocation Errors (cont.)

Error messages	Description
ERROR: -refDesign should be followed by designName and topCellName.	
ERROR: -rdb should be followed by output File name.	
ERROR: -report should be followed by output File name.	
ERROR: -cellmap should be followed by cellMap file name.	
ERROR: -automatchreport should be followed by automatch report file name.	
ERROR: -include_layer should be followed by file name.	
ERROR: -include_layer and -exclude_layer can not be used together.	
ERROR: -exclude_layer should be followed by file name.	
ERROR: -flattencell should be followed by flattened Cell file name.	
ERROR: -checkcell should be followed by file name containing cells to be checked.	
ERROR: -design should be followed by designName and topCellName.	
ERROR: -template should be followed by templateFile.	
ERROR: -transform should be followed by arguments.	
ERROR: Option -transform cannot be used when the -design and -refdesign files are same.	The specified layout files must be different to use -transform.
ERROR: Option -transform not valid without top cell.	The top cell cannot be determined and must be specified.
ERROR: Top cells should be specified when auto_offset is applied.	You can only use the -transform auto_offset option in DBdiff if you specify the top cells for the current and reference layouts.
ERROR: <arg> not a valid argument.	The -transform option has an invalid argument.

Table A-11. DBdiff Invocation Errors (cont.)

Error messages	Description
ERROR: <arg> not a valid argument; more than two translations specified.	The -transform option only takes x- and y-translations.
ERROR: <arg> not a valid argument; rotation already specified.	The -transform option takes only one rotation argument.
ERROR: <arg> not a valid argument; mirroring already specified.	The -transform option takes only one reflection argument.
ERROR: -write_xor_rules should be followed by output rule file name.	
ERROR: -resultformat should be followed by correct output DB.	
ERROR: -rules should be followed by rule file name.	
ERROR: only -report output option can be used with -nocompare.	The -report option is the only legal output option when -nocompare is specified; you cannot use the -rdb option.
ERROR: Repeating RDB output filter: <filter_name>.	This error occurs if you have incorrectly specified more than one of the same type of filtering option with the -rdb switch. For example, you may have invoked dbdiff with two text filtering options for the results database: dbdiff ... -rdb results1.rdb FT FT FP Remove the repeated filter to correct this error (FT in this case).
ERROR: Unknown RDB output filter: <wrong_filter_name>.	You have specified an incorrect filtering option with the -rdb switch. The allowed filtering options are FT, FP, and FC.
-turbo Specification Errors	
ERROR: -turbo <number_of_processors> must be greater than 0.	A positive integer must be specified with the -turbo argument.
ERROR: -turbo_all option is invalid when -turbo is not specified.	-turbo must be specified with -turbo_all.
ERROR: <required_cpu_count> CPUs were requested and <number_of_cpus_on_system> were available.	More CPUs were specified than currently available.
ERROR: Could not get requested number of CPUs and -turbo_all was specified.	The total number of CPUs requested specified was unavailable with -turbo_all. Consider using -turbo <number> for fewer processors.

Table A-11. DBdiff Invocation Errors (cont.)

Error messages	Description
Design Specification Errors	
ERROR: Current design not specified.	These errors indicate that a current design or a reference design was specified incorrectly from a DBdiff invocation. For more information on invoking DBdiff, see the Layout Comparison Command Line Syntax .
ERROR: Current design not specified. Use -design option.	
ERROR: Reference design not specified.	
ERROR: Reference design not specified. Use -refdesign option.	
Cell and Layer Specification Errors	
ERROR: -automatch [filename] required with -multimatch.	The -automatch option must be specified with -multimatch or -nocompare.
ERROR: -automatch [filename] required with -nocompare.	
ERROR: -checkcell is not a valid option with -automatch/multimatch.	The -checkcell option cannot be specified with the -automatch or -multimatch options.
ERROR: layerMapfile <layerMapfile> doesn't exist.	The specified file was invalid or did not exist. Verify the correct location of the specified file.
ERROR: checkCell file <cellCheckfile> doesn't exist.	
ERROR: flatCell file <cellFlatfile> doesn't exist.	
ERROR: cellMapfile <cellMapfile> doesn't exist.	
ERROR: autoMatch pattern file <autoMatchPatternfile> doesn't exist.	
ERROR: include_layer file <processLayerfile> doesn't exist.	
ERROR: exclude_layer file <processLayerfile> doesn't exist.	
ERROR: -sortlayer is not valid without -rdb argument.	-rdb must be specified with the -sortlayer argument.
ERROR: Top cell specified in one reference/current design, specify top cell in both or none.	Only one top cell was specified in either the reference design or current design. A top cell should be specified for both designs or not at all.
ERROR: -include_layer and -exclude_layer cannot be used together.	The -include_layer and -exclude_layer options must not be used in the same DBdiff invocation.

Table A-11. DBdiff Invocation Errors (cont.)

Error messages	Description
ERROR: Wrong number of arguments found in include/exclude file <filename> at line number <line number>. Layer name format is expected.	When -layermap is used with either -include_layer or -exclude_layer the include or exclude file format requires that layer names be used.
ERROR: Wrong number of arguments found in layer mapping file <filename> at line number <line number>.	All lines of the layer map file must have the same number of arguments.
ERROR: Layer map format for CUR/REF keywords in layer mapping file <filename> are reserved for XOR rule generation.	Layer maps can only contain CURrent or REference when -write_xor_rules is used.
Returned Result Errors	
ERROR: -rdb cannot be created as design(s) contain more than one topcell. Use single top cell and rerun.	DBdiff only completes the run and generates an RDB if both designs have a single top cell detected.
ERROR: -hierarchyonly cannot be used along with -ignoreemptycells argument.	An invalid argument was specified with -hierarchyonly.
ERROR: -write_xor_rules can not be used with other compare/output options.	An invalid argument was specified with -write_xor_rules.
ERROR: Rules can not be created as design(s) contain more than one topcell. Use single top cell and rerun.	-write_xor_rules could not generate a rule file because one or both of the designs contained more than one top cell. Specify a single top cell and rerun DBdiff.
ERROR: rule file <RuleFileName> could not be created.	An XOR rule file could not be generated.
ERROR: -resultformat should be preceded by -write_xor_rules.	The -resultformat argument was specified out of order. The -write_xor_rules argument must be specified first.
ERROR: File <ReportFile> could not be created.	A report file could not be generated.
Input File Errors	
ERROR: Unable to open template file <templateFile>. Exiting.	The template file may already be used by another process or might still be open for editing. Read access of the template file might also be disabled.
ERROR: Environment variable <env_var>, used in template file is not specified.	An environment variable was not previously specified before using the template file for a DBdiff invocation.

Table A-11. DBdiff Invocation Errors (cont.)

Error messages	Description
ERROR: Number of arguments not two at line number <lineCount> in file <ProcessLayerFile>.	An invalid value was specified in the layer file for the -include_layer or -exclude_layer arguments.
ERROR: Invalid layer number expression at line number <lineCount> in file <ProcessLayerFile>.	
ERROR: Invalid layer number constraint at line number <lineCount> in file <ProcessLayerFile>.	
ERROR: Invalid data type constraint at line number <lineCount> in file <ProcessLayerFile>.	
ERROR: Invalid data type expression <expression> at line number <lineCount> in file <ProcessLayerFile>.	
ERROR: In layermap file at line number <lineCount>.	An invalid argument was specified in the layer map file.
ERROR: Invalid first argument <first_argument> in layermap file at line number <lineCount>.	
ERROR: Invalid layer number expression at line number <lineCount> in layer map file.	
ERROR: Invalid second argument <tokens[1]> in layermap file at line number <lineCount>.	
ERROR: Invalid data type expression at line number <lineCount> in layer map file.	
ERROR: In Cell Map File ... <strNewName> AND <strOldName> Exiting.	Invalid strings were detected in the cell map file.
ERROR: Unable to open current library <curLibName>.	<p>The specified library does not have read access or the specified primary cell does not exist in the specified library. When the primary cell does not exist, these two additional error messages are issued:</p> <p>ERROR: Cell <cellname> is not present in the current design.</p> <p>ERROR: Unable to open current library <LibName>.</p>
ERROR: Unable to open reference library <refLibName>.	

The following table contains various error messages.

Table A-12. Miscellaneous Errors for DBdiff

Error messages	Description
ERROR: Insufficient memory.	There was not enough memory to complete the job.
ERROR: Cell <currTopCell> is not present in the current design.	A specified primary cell was not detected in the current design. Verify that the primary cell was specified correctly and that the cell exists in the current design.
ERROR: Cell <refTopCell> is not present in the reference design.	A specified primary cell was not detected in the reference design. Verify that the primary cell was specified correctly and that cell exists in the reference design.

The following table contains possible warnings that may appear when DBdiff is invoked.

Table A-13. DBdiff Warnings

Warning messages	Description
WARNING: Reference cell definition not found <refCellName>.	The specified reference cell was not found.
WARNING: Current cell definition not found <curCellName>.	The specified current cell was not found.
WARNING: Version not 22 - Taking version as 22.	The specified OpenAccess file is not version 22. The system is read in as version 22.
WARNING: AREF placements are treated as SREF placements when system is OASIS.	AREF placements are treated as SREF placements when the specified system is OASIS.
WARNING: -turbo could not be licensed sufficiently. Ignoring -turbo.	More CPUs were specified for -turbo execution than available licenses. The -turbo argument is ignored.
WARNING: Option -turbo is not supported when auto_offset is applied. Ignoring -turbo.	The -turbo option is not compatible with the -transform auto_offset option. If you specify both -turbo and -transform auto_offset, the -turbo argument is not applied.
WARNING: The format with two arguments per line, found in layermap file, will be deprecated.	The layer map file contained two arguments.
WARNING: Some merged polygons are forming holes hence not merged.	Some merged polygons were forming holes that were not merged.
WARNING: Reference design layer name <refName> for layer <layerNum>.	A specified reference design layer was detected for a layer number.

Table A-13. DBdiff Warnings (cont.)

Warning messages	Description
WARNING: cellmap file <CellMapFile> is empty.	The specified cell map file was empty.
WARNING: include_layer file <LayerFile> has no layers.	The specified layer file for -include_layer did not contain a list of layers.
WARNING: exclude_layer file <LayerFile> has no layers.	The specified layer file for -exclude_layer did not contain a list of layers.
WARNING: layermap file <LayerMapFile> is empty.	The specified layer map file did not contain layer mappings.
WARNING: checkcell file <CheckFile> is empty.	No cells were detected in the check cell file specified.
WARNING: flattencell file <FlatFile> is empty.	No cells from <FlatFile> were flattened.
WARNING: flattencell file <FlatFile> has no cells that are flattened.	
WARNING: Cell <cellname> is not flattened in reference design as cell is not used.	The specified cell was not present in the reference design so flattening was not carried out.
WARNING: Cell <cellname> is not flattened in current design as cell is not used	
WARNING: cellmap entry for reference cell <cellName> not found in reference design.	The specified cell in the cell map file does not exist in the reference design.
WARNING: cellmap entry for current cell <cellName> not found in current design.	The specified cell in the cell map file does not exist in the current design.
WARNING: checkcell entry for cell <cellName> not found in current design.	The specified cell in the check cell file does not exist in the current design.
WARNING: layermap entry for layer <LayerName> not found in current design.	The specified layer in the layer map file does not exist in the current design.
WARNING: cell <cellName> of current design specified in check cell file is empty, hence not checked.	The specified cell in the check cell file was empty and ignored.
WARNING: cell <cellName> of reference design specified in check cell file is empty, hence not checked.	
WARNING: cell <cellName> of current design specified in cell map file is empty, hence not used.	

Table A-13. DBdiff Warnings (cont.)

Warning messages	Description
WARNING: cell <cellName> of reference design specified in cell map file is empty, hence not used.	The specified cell in the cell map file of the reference design was empty and ignored.
WARNING: Cell <cellname> referenced in <designname> but not defined, empty cell used.	The specified cell was not found in the design but an empty cell is used for this DBdiff run.
WARNING: Ignoring GDS_NODE structure found at offset <offsetnumber> in input file <designName>.	A GDS_NODE structure was detected and ignored at the specified offset and design.
WARNING: Cell <cellname> not defined in input file <designName>.	The specified cell was not defined or did not exist in the layout database.
WARNING: Ignoring invalid boundary record at offset <Offset> in input file <designName>.	An invalid BOUNDARY record was detected and ignored.
WARNING: Ignoring invalid box record at offset <boxFileOffset> in input file <designName>.	An invalid BOX record has been detected and ignored.
WARNING: Ignoring invalid path record at offset <pathFileOffset> in input file <designName>.	An invalid PATH record has been detected and ignored.
WARNING: Degenerate (vertex count = <numOfPoints>) boundary at location (<bBox.left>,<bBox.top>) in cell <cellName> on layer <layer> dropped in input file <designName>.	A zero-area polygon has been detected and ignored.
WARNING: Degenerate (vertex count = 1) path at location (<X,Y>) in cell <cellname> on layer <layer> datatype <datatype> dropped in input file <DesignName>.	A single-point path has been detected and ignored.
WARNING: Path of zero width at location (<X,Y>) in cell <cellName> on layer <layer> datatype <purpose> dropped in input file <DesignName>.	A zero-width path has been detected and ignored.
WARNING: <numOfPoints>-vertex text object <textBuffer> in cell <cellName> on layer <layer> datatype <purpose> dropped in input file <designName>.	A text object did not have exactly one point associated with it.

Table A-13. DBdiff Warnings (cont.)

Warning messages	Description
WARNING: Non a-string for TEXTSTRING record in input file <DesignName> at record offset <offsetnumber>.	A TEXTSTRING record has been detected that does not use an a-string. The record is ignored.
WARNING: Xname record seen and ignored in input file <DesignName>.	An XNAME record has been detected and ignored.
WARNING: Xname_reference record seen and ignored in input file <DesignName>.	An XNAME_REFERENCE record has been detected and ignored.
WARNING: Xelement record seen and ignored in input file <DesignName>.	An XELEMENT record has been detected and ignored.
WARNING: XGEOMETRY at location (<x,y>) in cell <OasisCellName> on layer <layer> datatype <purpose> ignored in input file <DesignName>.	An XGEOMETRY record has been detected and ignored.
WARNING: Circle of radius zero at location (<x,y>) in cell <OasisCellName> on layer <layer> datatype <purpose> dropped in input file <DesignName>.	A zero-radius circle has been detected and ignored.
WARNING: GDS_NODE structure found at <offsetnumber>. Ignored.	A GDS_NODE structure has been detected and ignored.
WARNING: Mapping for layer <layername> in layer mapping file <filename> must be mapped for both the current and reference designs.	A layer must be mapped in both current and reference design files.
WARNING: Single physical layer number <n> datatype <n> is mapped to more than one layer <layer>.drawing (<m> <n> and <layer>.drawing (<m> <n>).	A layer/datatype combination is mapped to more than one layer in a -layermap file. This is handled automatically by the tool.
WARNING: Layer name <layername> found in include file <filename> is not present in layermap file <filename>, ignoring.	Unused entries in an -include_layer or -exclude_layer file.
WARNING: OA layermap file is not set. Set it through either MGC_CALIBRE_LAYERMAP_FILE or MGC_CALIBRE_DB_READ_OPTION.	When comparing an OpenAccess database to a database of a different format, you must specify a valid layer mapping file.

Related Topics

[Input and Output](#)

Layout Comparison Command Line Syntax

FastXOR Messages

Use these error and warning message descriptions to debug issues during a run.

The following are fatal error messages that occur in a calibre -fx run:

Table A-14. FastXOR Error Messages

Message	Description
ERROR: A layer cannot be paired with more than one layer during FastXOR, when the -fx option is specified. Layer <layer> is paired with layer <layer> in rule <rule> and also paired with layer <layer>.	A layer is specified in a rule check together with some other layer but does not appear in a corresponding XOR rule check together with that other layer.
ERROR: Error SPC1 on line <line_number> of <file_name> - superfluous specification statement: svrf dbdiff options.	The SVRF DBdiff Options statement can only be specified once in your rule file. Only the first statement is used and the rest are ignored.
ERROR: Error SPC1 on line <line_number> of <file_name> - superfluous specification statement: svrf fdi options.	The SVRF FDI Options statement can only be specified once in your rule file. Only the first statement is used and the rest are ignored.
ERROR: Layers from same design cannot be compared during FastXOR, when the -fx option is specified. Layer <layer> and layer <layer_number> in check <check> are from same design.	Layers that originate from the same design cannot be compared to each other. Check the Layer statements for a conflict.
ERROR: Layers from same design cannot be compared during FastXOR, when the -fx option is specified. Layer <layer> datatype <datatype> and layer <layer> datatype <datatype> in check <check> are from same design.	Layers that originate from the same design cannot be compared to each other. Check the Layer Map statements for a conflict.
ERROR: LAYOUT PRIMARY[2] cannot be * when invoked with -fx.	The Layout Primary[2] specification statement cannot use a bare wildcard.
ERROR: LAYOUT PRIMARY[2] is not specified.	The Layout Primary[2] specification statement must appear in the rule file.
ERROR: LAYOUT SYSTEM[2] can only be GDS, OASIS, OA or LEFDEF when invoked with -fx.	Only the four layout formats are supported.
ERROR: LAYOUT SYSTEM[2] is unspecified.	The Layout System[2] specification statement must appear in the rule file.

Table A-14. FastXOR Error Messages (cont.)

Message	Description
ERROR: Only one file is allowed in LAYOUT PATH[2] when invoked with -fx.	The Layout Path [2] statement cannot specify more than one layout design.
ERROR: Same layer cannot be used in more than one check during FastXOR, when the -fx option is specified. Layer <layer> used in check <check_name> and check <check_name>.	The same layer is used in two rule checks. There is a problem with a Layer statement or a rule check statement.
ERROR: Same layer/datatype combination cannot be used in more than one check during FastXOR, when the -fx option is specified. Layer <layer> datatype <datatype> used in check <check_name> and check <check_name>.	A layer and datatype combination is referenced by layers in two rule checks. There is a problem with a Layer Map statement.
ERROR: Unsupported layer map specification statement in check <rule check> at line <N> in <rule file>. Layer <N> datatype <range> can not have more than one source layer and source type specified when using the -fx option.	One or more Layer Map statements map more than one layer or datatype onto the same target layer.
ERROR: Unsupported layer specification statement in check <check_name> at line <line_num> in <filename>. Layer <layer_name> may have only one original layer during FastXOR, when the -fx option is specified.	A Layer specification statement uses two original layers, which is unsupported.
ERROR: Unsupported operation during FastXOR, when the -fx option is specified in check <check_name> at line <line_num> in <filename>.	An XOR operation specifies a layer that represents an original layer group. Layer groups are not supported.
ERROR: Unsupported operation in check <check_name> at line <line_num> in <filename>. <operation> operation is not supported during FastXOR, when the -fx option is specified.	The specified operation may not appear in the rule file when the -fx option is used.
ERROR: Unsupported operation in check <check_name> at line <line_num> in <filename>. Only one operation is permitted within rulechecks during FastXOR, when the -fx option is specified.	Rule checks having multiple layer operations in them are not supported.
ERROR: Unsupported operation in check <check_name> in <filename> file. Only 1 NOT operation between 2 original layers is allowed in one rule check.	NOT operations only support original layer inputs.

The following are warning messages that occur in a calibre -fx run

Table A-15. FastXOR Warning Messages

Message	Description
WARNING: As of version 2009.4 environment variable <env_variable> is deprecated. Use environment variable <env_variable>.	An environment variable is no longer valid.
WARNING: EXCLUDE CELL name <name> is not located within input layout database <file>.	An Exclude Cell statement parameter cannot be located in the input design. This message is a warning by default, but can be an error when Layout Input Exception Severity EXCLUDE_CELL_NAME 2 is set.
WARNING: OA layermap file is not set. Set it through either MGC_CALIBRE_LAYERMAP_FILE or MGC_CALIBRE_DB_READ_OPTIONS.	When comparing an OpenAccess database to a database of a different format, you must specify a valid layer mapping file.
WARNING: Value specified by deprecated environment variable <env_variable> is ignored. Value specified by environment variable <env_variable> is used.	A value for an environment variable has changed.
WARNING: Missing LAYOUT BASE LAYER and LAYOUT TOP LAYER specification statements from rule deck. These statements are highly recommended for all hierarchical applications	Defining base layers can significantly improve performance. Use the dbdiff -write_xor_rules -base_layers option or include the SVRF Layout Base Layer statement in your rule file.

— Symbols —

`[]`, 17

`{}`, 17

`|`, 17

— B —

Bold words, 17

— C —

Calibre DESIGNrev, 112

Calibre Interactive, 112

CALIBRE_FX_DB_DIFF_OPTIONS, 43

Cell renaming, 150

Command lines

 DBdiff, 80

 Fast XOR, 34

Command syntax, 17

Compare two layouts, 25

Courier font, 17

— D —

Database format support, 111

DBdiff

 AREF and SREF types, 95

 comparing designs with pcells, 74

 instance rotation, 95

 introduction, 63

 licensing, 63

 syntax, 80

 zero length paths and polygons, 95

DBdiff template file, 43

Double pipes, 17

DRC

 XOR, 14

Dual database comparisons

 hierarchical semantics, 23

 layer concatenation, 22

 rule file statements, 19

 supported formats, 19

— E —

Environment variables

 MGC_CALIBRE_CELLMAP_FILE, 119

Environment variables, 115

 CALIBRE_BACK_ANNOTATE_HIER_ LAYER, 210

 CALIBRE_DBDIFF_MERGE_SHAPETE XT SUMMARY, 107

 CALIBRE_FX_DB_DIFF_OPTIONS, 60

 CALIBRE_FX_DISABLE_OPS, 60

 CALIBRE_FX_PASS_THROUGH_LAY ER_ENABLE, 60

 CALIBRE_FX_TRANSCRIPT_ENABLE , 60

 CALIBRE_FX_VERBOSE, 60

 CALIBRE_FXOR_GDS_MODE, 60

 FastXOR, 60

 ICC_PATH, 117

 LD_LIBRARY_PATH, 117

 MGC_CALIBRE_DB_READ_OPTIONS, 118, 149

 MGC_CALIBRE_LAYERMAP_FILE, 118

 MGC_TMPDIR, 61

 OA_HOME, 119

Examples

 fdiBA -dfmdb option, 198

 fdiBA -gds option, 198

 LEF/DEF to GDS conversion, 123

 OA to OASIS conversion, 125

Exclude Cell statement, 20

— F —

FastXOR, 25

 command line, 34

 disallowed layer operations, 48

 troubleshooting guidelines, 50

FDI comma nd option priority, 149

FDI default options, 149

FDI non-default o ptions, 150

FDI translators, 241

fdi2gds, 130

fdi2oasis, 130

Files, 155

cell mapping file, 179

DFM database file, 209

input exception file, 188

layer mapping file for backannotation, 205

layer mapping file for database translation,
172

mapping information file, 182

net properties file, 184

nets file, 185

object map file, 176

property map file, 187

template file, 190

Font conventions, 17

Foreign Database Interface (FDI), 15, 16

— G —

GDS hierarchy, 152

GDS to OASIS, 119

— H —

Heavy font, 17

— I —

Inside Cell operation, 23

Italic font, 17

— L —

Layer group, 37

Layer Map statement, 20

Layer statement, 20

Layout Bump2 statement, 19, 21

Layout databases

translating LEF/DEF and OpenAccess
databases, 241

Layout Path2 statement, 19

Layout Primary2 statement, 19

Layout Rename Cell statement, 20

Layout System2 statement, 19

Layout Text statement, 20, 23

Layout-versus-layout, 19

LEF/DEF database format, 111

License requirements, 114

LVL, *see* Layout-versus-layout and Dual
database comparison

— M —

Member layer, 37

Minimum keyword, 17

— N —

Net type properties, 150

— O —

OpenAccess database format, 111

— P —

Parentheses, 17

Pipes, 17

Primitive layers, 22

— Q —

Quotation marks, 17

— S —

Simple layer, 22, 37

Slanted words, 17

Specification statements

dual-database comparisons, 19

Exclude Cell, 20

Layer, 20

Layer Map, 20

Layout Bump2, 19, 21

Layout Path2, 19

Layout Primary2, 19

Layout Rename Cell, 20, 23

Layout System2, 19

Layout Text, 20, 23

Square parentheses, 17

— T —

Translate layout databases, 15, 16

Translation

LEF/DEF, 130

OpenAccess, 131

— U —

Underlined words, 17

Usage

dbdiff, 80

FastXOR, 34

Usage syntax, [17](#)

Utilities

DBdiff error messages, [241](#), [245](#)

Third-Party Information

Details on open source and third-party software that may be included with this product are available in the *<your_software_installation_location>/legal* directory.

