

SIEMENS EDA

# Calibre® Interactive™ (New GUI) User's Manual

Software Version 2021.2  
Document Revision 3

SIEMENS

Unpublished work.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](http://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](http://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](http://support.sw.siemens.com)

Send Feedback on Documentation: [support.sw.siemens.com/doc\\_feedback\\_form](http://support.sw.siemens.com/doc_feedback_form)

# Revision History

---

Revision	Changes	Status/ Date
3	<p>Modifications to improve the readability and comprehension of the content. Approved by Lucille Woo.</p> <p>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.</p>	Released April 2021
2	<p>Modifications to improve the readability and comprehension of the content. Approved by Lucille Woo.</p> <p>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.</p>	Released January 2021
1	<p>Modifications to improve the readability and comprehension of the content. Approved by Lucille Woo.</p> <p>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.</p>	Released October 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

<b>Chapter 1</b>		
<b>Introduction to Calibre Interactive.....</b>		<b>21</b>
Calibre Interactive Overview.....		21
Interactive Workflow for Calibre.....		22
Requirements for Calibre Interactive .....		24
Integration to Design Tools .....		25
GUI Customization in Calibre Interactive.....		28
Tips for Using the Calibre Interactive GUI.....		28
Syntax Conventions.....		30
<b>Chapter 2</b>		
<b>Basic Calibre Interactive Usage.....</b>		<b>33</b>
About Calibre Interactive Runsets.....		34
Using Calibre Interactive Runsets .....		34
Methods to Load a Calibre Interactive Runset .....		35
Converting Classic Calibre Interactive Runsets .....		37
Environment Variables for Calibre Interactive Runsets.....		38
Populating the Load Runset Files Dialog Box and Recent Files Lists.....		39
Using Multiple Calibre Interactive Runsets to Combine Different Sets of Options .....		43
About Control Files .....		44
Control File in Calibre Interactive .....		44
Control Filenames.....		46
Creating and Viewing the Control File Without Starting a Run.....		46
Running Batch Calibre with a Calibre Interactive Control File .....		46
About Rule Files .....		50
Rule File Operations in Calibre Interactive .....		50
How Rule Files Relate to Control Files .....		51
Environment Variables in Calibre Interactive GUI Fields .....		52
Using Simplified Views.....		54
<b>Chapter 3</b>		
<b>Running Calibre Interactive .....</b>		<b>57</b>
Invoking the Calibre Interactive GUI .....		58
License Timeout in Calibre Interactive .....		60
Calibre Interactive Command Line .....		61
Connecting to a Design Tool in Calibre Interactive .....		64
Setting Layout and Schematic Export in Calibre Interactive .....		67
Methods for Populating the Calibre Interactive GUI.....		69
Populating the Calibre Interactive GUI .....		69
Automatically Loading a Rule File.....		70

Managing Rule File and Calibre Interactive GUI Precedence .....	71
Providing OpenAccess Input with Calibre Interactive .....	73
Providing LEF/DEF Input with Calibre Interactive .....	75
Setting Up and Starting a Run .....	76
Starting a Basic Batch Run in Calibre Interactive .....	78
Specifying Layout and Schematic Export for a Calibre Interactive Batch Run .....	79
Calibre Interactive Operation with a TVF Rule File .....	80
Check Selection in Calibre Interactive .....	82
Check Selection Recipe .....	82
Specifying a Check Selection Recipe .....	82
Creating a Custom Check Selection Recipe .....	83
Basic Editing of Custom Check Selection Recipes .....	84
Advanced Editing of Check Selection Recipes in Calibre Interactive .....	86
Running Only Checks Selected in the Rule File with a Custom Check Selection Recipe .....	88
Recipe Editor Page .....	90
Viewing and Editing Environment Variables .....	98
Configuration Files for Startup Preferences .....	98
Templates for File Naming .....	100
Example for File Naming Template .....	101
Disabling Templates So That Saved Runset Values Are Used .....	102
Export from Viewer Preferences .....	103
<b>Chapter 4</b> <b>Calibre Interactive GUI Reference .....</b>	<b>105</b>
Options Page in Calibre Interactive: Common Options .....	109
Database Page in Calibre Interactive .....	110
OA/LEFDEF and OPENACCESS Pages in Calibre Interactive .....	112
Environment Page in Calibre Interactive .....	116
Run Control Page in Calibre Interactive .....	118
Triggers Page in Calibre Interactive .....	122
Templates Page in Calibre Interactive .....	124
Preferences Page in Calibre Interactive .....	126
Runset Preferences .....	127
Rules Preferences .....	128
Disable Preferences .....	130
Miscellaneous Preferences .....	131
Search Toolbar in Calibre Interactive .....	132
Layer Derivations Tree .....	134
<b>Chapter 5</b> <b>Calibre Interactive nmDRC .....</b>	<b>137</b>
Setting Up a Calibre Interactive nmDRC Run .....	137
Specifying the Checks to Execute in a Calibre Interactive nmDRC Run .....	140
Running with Area DRC .....	141
Creating a Check Text Override (CTO) File With Calibre Interactive nmDRC .....	143
Creating a DRC HTML Report from Calibre Interactive nmDRC .....	145
Reporting DRC Results in Local Cell Space or Top Cell Space .....	148
Limiting the DRC Results Count .....	148

## Table of Contents

---

FastXOR Database Comparison in Calibre Interactive . . . . .	149
Running FastXOR Database Comparison in Calibre Interactive . . . . .	149
Fast XOR Page . . . . .	152
DRC-Specific Pages in Calibre Interactive . . . . .	156
Options Page in Calibre Interactive nmDRC . . . . .	157
Outputs Page in Calibre Interactive nmDRC . . . . .	158
 <b>Chapter 6</b>	
<b>Calibre Auto-Waivers in Calibre Interactive . . . . .</b>	<b>161</b>
Introduction to Calibre Auto-Waivers in Calibre Interactive . . . . .	161
Calibre Auto-Waivers Requirements for Calibre Interactive . . . . .	162
Generating Waiver Shapes with Calibre Interactive . . . . .	163
Starting a Waiver Run in Calibre Interactive . . . . .	167
 <b>Chapter 7</b>	
<b>Calibre Interactive nmLVS . . . . .</b>	<b>173</b>
Setting Up a Calibre Interactive nmLVS Run . . . . .	173
Supplying Hcell Data . . . . .	176
Executing Electrical Rule Check (ERC) Operations in Calibre Interactive nmLVS . . . . .	177
Running Short Isolation in Calibre Interactive nmLVS . . . . .	178
Changing Trace Property Statements . . . . .	179
Adding and Changing LVS Box Statements in Calibre Interactive . . . . .	181
Performing Hcell Analysis in Calibre Interactive . . . . .	184
Creating and Analyzing an H-Cell List With Calibre Interactive . . . . .	184
H-Cells Analysis Results . . . . .	186
Device Signatures: Using and Creating in Calibre Interactive nmLVS . . . . .	187
Generating Device Signatures Using Calibre Interactive . . . . .	187
Using a Device Signature File in a Calibre Interactive nmLVS Run . . . . .	190
Edit Parameters for Device Dialog Box in Calibre Interactive . . . . .	191
Layers File for Device Signature Creation With Calibre Interactive . . . . .	194
LVS-Specific Pages in Calibre Interactive . . . . .	197
 <b>Chapter 8</b>	
<b>Calibre Interactive PEX . . . . .</b>	<b>199</b>
Introduction to Calibre Interactive PEX . . . . .	199
Inputs for a PEX Run . . . . .	202
Setting Up a Calibre Interactive PEX Run . . . . .	202
Using Iterative Extraction to Create a Mixed Parasitic Network . . . . .	205
Setting Cell Extraction Options for Blocks . . . . .	206
Setting Probe Points . . . . .	208
Specifying Extracted Netlist Annotation of Net Geometries to Calibre View . . . . .	210
Naming the Extracted Parasitic Netlist Based on the Extraction Type and Format . . . . .	211
Controlling PEX Step Execution . . . . .	213
PEX-Specific Pages in Calibre Interactive . . . . .	214
LVS Page in Calibre Interactive PEX . . . . .	215

---

<b>Chapter 9</b>	
<b>Calibre Interactive PERC .....</b>	<b>219</b>
Setting Up a Calibre PERC Topological Run with Calibre Interactive.....	219
Setting Up a Calibre PERC LDL Run with Calibre Interactive .....	221
Specifying Waivers for a Calibre PERC Run .....	224
Generating a Calibre PERC Rule File with High Level Checks.....	224
Using the Calibre PERC Cell-Based Flow .....	226
<b>Chapter 10</b>	
<b>GUI Customization for Calibre Interactive .....</b>	<b>229</b>
Loading a Configuration File in Calibre Interactive .....	230
Loading a Classic Tcl Customization File in Calibre Interactive .....	232
Customizing the GUI With the Configuration Editor .....	233
Saving GUI Settings from Custom Controls .....	237
Configuration File Reference.....	239
Configuration File Format for Calibre Interactive GUI Customization.....	240
Global Object Definitions.....	243
Arrays and Maps.....	244
Tool Properties .....	247
Option Properties .....	250
GUI Signals.....	252
Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive.....	252
Finding Option Names to Use in the Configuration File .....	254
Function Reference for GUI Customization .....	256
General Configuration File Functions .....	257
addGroup .....	259
addHeader.....	261
addLabel.....	263
addMaster .....	265
addMasterSelect .....	268
addMastersWithExpr .....	269
addMastersWithFunction .....	270
addMutuallyExclusiveGroup .....	272
addPage.....	274
addSeparator.....	275
addTable .....	276
envVars.....	281
FileIO .....	283
getCalibreVersion.....	286
getProcessID.....	287
hasOwnProperty .....	288
removePage .....	289
setItemOrder.....	290
transcript.....	292
Configuration Commands to Execute Functions and Commands .....	293
connect .....	294
launchCommand.....	298
launchCommandReturnString .....	301

## Table of Contents

---

Configuration File Commands to Add #DEFINE and #UNDEFINE Statements . . . . .	303
addChoicesDefine. . . . .	305
addDefine . . . . .	310
addInputDirTextDefine . . . . .	313
addInputFileTextDefine . . . . .	316
addIntegerDefine . . . . .	320
addOutputDirTextDefine . . . . .	324
addOutputFileTextDefine . . . . .	327
addRealDefine . . . . .	330
addTextBoxDefine . . . . .	334
addTextDefine . . . . .	338
Configuration File Functions to Add Variable Statements. . . . .	342
addChoicesVariable . . . . .	344
addListVariable . . . . .	348
addInputDirTextVariable . . . . .	351
addInputFileTextVariable. . . . .	354
addIntegerVariable . . . . .	358
addMultiChoicesVariable. . . . .	361
addOutputDirTextVariable. . . . .	365
addOutputFileTextVariable . . . . .	368
addRealVariable . . . . .	371
addTextBoxVariable. . . . .	375
addTextVariable . . . . .	379
Configuration File Commands to Add Buttons . . . . .	382
addAction . . . . .	383
addChoicesAction. . . . .	386
Configuration File Commands for Trigger Functions . . . . .	390
Trigger Parameters in Calibre Interactive . . . . .	390
add ... Trigger . . . . .	393
Guide and Examples for Converting Tcl Customization Files to New Configuration Files .	397
Correspondence of Tcl Customization Commands to New Configuration Commands . . .	397
Adaptation Example: #DEFINE and Variable Statements . . . . .	401
Adaptation Example: Move and Set GUI Options . . . . .	403
Adaptation Example: Callback Function . . . . .	407
Troubleshooting Configuration Files for the Calibre Interactive GUI . . . . .	411
<b>Chapter 11</b>	
<b>Trigger Functions in Calibre Interactive . . . . .</b>	<b>413</b>
Trigger Overview . . . . .	413
Internal Trigger Execution in Calibre Interactive . . . . .	415
Internal Triggers in Calibre Interactive . . . . .	417
Trigger Parameters for Internal Triggers in Calibre Interactive. . . . .	417
Setting Calibre Interactive Internal Triggers . . . . .	419
Trigger Access to the Design Tool Environment . . . . .	422
Calibre Interactive Internal Trigger Examples . . . . .	424
Internal Trigger for Calling a UNIX Utility. . . . .	424
Internal Trigger for Calling a Script. . . . .	424
Internal Trigger to Set an SVRF Variable . . . . .	424

Internal Trigger Executed in Calibre DESIGNrev to Count Layout Cells . . . . .	426
Applying Triggers According to the Calibre Interactive Application . . . . .	427
External Triggers in Calibre Interactive . . . . .	429
External Trigger Types . . . . .	429
External Trigger Functions and File Locations . . . . .	430
External Trigger Definitions in Calibre DESIGNrev . . . . .	430
External Trigger Definitions in Cadence Virtuoso . . . . .	431
External Trigger Definitions in Synopsys IC Compiler . . . . .	432
Calibre Interactive External Trigger Examples . . . . .	433
External Trigger in Tcl . . . . .	433
External Trigger in SKILL for Cadence Virtuoso . . . . .	433
<b>Chapter 12</b>	
<b>Run Control and Licensing Setup . . . . .</b>	<b>435</b>
Specifying the Host Processor and Run Options . . . . .	435
Configuring for Distributed (MTflex) Execution . . . . .	438
Running With Spectrum LSF . . . . .	440
Configuring the Remote Environment . . . . .	443
Setting Licensing Options in Calibre Interactive . . . . .	445
Reference for Run Control Options . . . . .	447
Hyperscaling Mode . . . . .	447
Hyperscale Compare Option in Calibre Interactive nmLVS . . . . .	447
Hyperscale Pathchk Option . . . . .	447
Remote Data Server (RDS) Option . . . . .	448
Hyperscale Remote Option . . . . .	448
Backup Data Option . . . . .	448
<b>Appendix A</b>	
<b>Interfacing Calibre Interactive to Layout and Schematic Viewers . . . . .</b>	<b>449</b>
Basic Interface . . . . .	450
Calibre Interactive . . . . .	450
Setting the Socket Port in Calibre Interactive . . . . .	451
Setting the Calibre Interactive Socket Port for a Layout Viewer . . . . .	451
Setting the Calibre Interactive Socket Port for a Schematic Viewer . . . . .	452
Custom Menu Items in Design Tools . . . . .	453
Creating Custom Menus in Design Tools . . . . .	454
Menu Customization File . . . . .	455
Menu Customization Commands . . . . .	456
mgc_calibre_get_viewer_name . . . . .	457
mgc_calibre_add_menu_item . . . . .	458
mgc_calibre_delete_menu_item . . . . .	462
Calibre Integration Menu Labels and Backward Compatibility for Menu Customization Files . . . . .	462
Menu Customization Example . . . . .	463
Calibre DESIGNrev and WORKbench . . . . .	465
Setting Socket Connections with Calibre DESIGNrev or Calibre WORKbench . . . . .	465
Calibre Interactive with Calibre DESIGNrev or WORKbench . . . . .	466

## Table of Contents

---

IC Station/Pyxis . . . . .	467
Setting Socket Connections with Pyxis . . . . .	467
Using Calibre Interactive with Pyxis . . . . .	467
Siemens Tanner . . . . .	468
Cadence Virtuoso . . . . .	469
Creating an Interface to Cadence Virtuoso. . . . .	469
The Calibre Menu in Cadence Virtuoso . . . . .	471
Layout Export in Cadence Virtuoso . . . . .	474
Calibre Layout Export Setup Dialog Box in Cadence Virtuoso. . . . .	475
Using a Template File for Layout Export with Cadence Virtuoso. . . . .	478
SKILL Functions for Layout Export . . . . .	480
Direct Read of Cadence Virtuoso OpenAccess Database. . . . .	482
Automatically Saving the Express Pcell Cache . . . . .	482
Netlist Export in Cadence Composer . . . . .	484
Calibre Netlist Export Setup Dialog Box in Cadence Composer. . . . .	485
SKILL Functions for Netlist Export . . . . .	488
Using the Calibre - Virtuoso Interface . . . . .	489
Setting Socket Connections with Cadence Virtuoso . . . . .	491
Calling SKILL Functions from Calibre Interactive . . . . .	492
Creating a Calibre View . . . . .	493
What You Need Before Creating a Calibre View . . . . .	494
What a Calibre View Produces. . . . .	495
Automatic OpenAccess Version Detection in Calibre View . . . . .	495
Device Property Calculation in Calibre View Generation . . . . .	496
Handling of Nets With Inherited Connectivity . . . . .	497
Bus Delimiter in Calibre View . . . . .	498
Identifying Devices with a Cellmap File . . . . .	498
Creating the Cellmap File. . . . .	498
Syntax for Cellmap Statements . . . . .	500
Cellmap Examples . . . . .	510
Setting the Calibre View Cellmap File in Calibre Interactive . . . . .	517
Importing Schematic Properties for Calibre View . . . . .	518
Auto-Mapping Behavior, Control, and Debug in Calibre View Generation . . . . .	519
Calibre View Setup Dialog Box . . . . .	520
Creating a Post-LVS Calibre View . . . . .	523
Creating a Post-Parasitic Extraction Calibre View . . . . .	526
Creating a Gate-Level Calibre View . . . . .	529
Creating Calibre Views in a Multiple Corner Extraction Run . . . . .	532
Creating a SPECTRE Netlist from the Calibre View Netlist. . . . .	533
Creating a CalibreView Setup File . . . . .	534
Showing Parasitic Polygons in Calibre View. . . . .	535
CalibreView Setup File Format . . . . .	536
SKILL Functions for Creating a Calibre View . . . . .	540
Global Variables for Calibre View Generation . . . . .	542
Creating a Calibre View in Batch Mode . . . . .	543
Calibre View and IC Manage Revision Control. . . . .	544
Calibre View Generation Warnings . . . . .	544
Setting the Bus Delimiter for Schematic Netlists . . . . .	550
Convert Bus Name Delimiter During Schematic Netlist Export . . . . .	550

Automatically Convert Bus Name Delimiter for Calibre Interactive and Calibre RVE . . . . .	550
Mapping Schematic Nets to Calibre View Nets for Simulation and Plotting . . . . .	552
Mapping Schematic Nets to Calibre View Nets with a Single Extracted View for Simulation and Plotting. . . . .	552
Mapping Schematic Nets to Calibre View Nets with Multiple Extracted Views for Simulation and Plotting. . . . .	553
Annotating Point to Point Resistance in Calibre View . . . . .	554
Backannotation of Parasitic Values to the Schematic in Cadence Virtuoso. . . . .	555
Automating Calibre Interactive Runsets in the Cadence Design Environment . . . . .	556
Cadence Composer . . . . .	558
Cadence Encounter . . . . .	559
Creating an Interface to Cadence Encounter . . . . .	559
Importing saveNetlist Parameters for Verilog Export in Cadence Encounter . . . . .	560
Cadence Innovus . . . . .	562
Creating an Interface to Cadence Innovus . . . . .	562
Synopsys IC Compiler and Synopsys IC Compiler II. . . . .	564
Calibre Flow with Synopsys IC Compiler . . . . .	565
Calibre Interface to Synopsys IC Compiler . . . . .	566
Loading the Calibre Interface to Synopsys IC Compiler . . . . .	566
Loading the Calibre Interface to Synopsys IC Compiler II . . . . .	567
README File for the Calibre-IC Compiler Integration . . . . .	569
Calibre Menu in Synopsys IC Compiler . . . . .	569
Running Calibre Interactive with Synopsys IC Compiler . . . . .	570
Setting Socket Connections with Synopsys IC Compiler and IC Compiler II . . . . .	573
Troubleshooting the Calibre Interface with Synopsys IC Compiler. . . . .	574
Silvaco Expert . . . . .	576
Creating the Interface to Silvaco Expert. . . . .	576
Setting Socket Connections with Silvaco Expert . . . . .	576
Calibre Menu in Silvaco Expert . . . . .	576
Other Viewer Integrations . . . . .	577
<b>Appendix B</b>	
<b>Environment Variables . . . . .</b>	<b>579</b>
Environment Variables for Calibre Interactive . . . . .	579
Environment Variables for Calibre View . . . . .	585
<b>Appendix C</b>	
<b>Runset Options . . . . .</b>	<b>589</b>
Calibre Interactive Runset File Format . . . . .	590
Common Runset Options. . . . .	591
DRC Runset Options . . . . .	624
LVS Runset Options . . . . .	650
PERC Runset Options . . . . .	684
PEX Runset Options . . . . .	711
Trigger Functions in the Calibre Interactive Runset . . . . .	798
Template Definitions . . . . .	799
Waiver Setup Runset Options . . . . .	805

## **Table of Contents**

---

**Glossary**

**Third-Party Information**



# List of Figures

---

Figure 1-1. Interactive Verification Flow .....	23
Figure 2-1. Relating User Input to Control Files.....	45
Figure 2-2. Relating Control Files to Rule Files .....	52
Figure 3-1. Design Tool Setting on the Run Control Page .....	66
Figure 3-2. Adding a Recipe Expression.....	87
Figure 3-3. Check Selection Recipe Editor Basic Controls.....	90
Figure 3-4. Check Selection Recipe Editor Advanced Controls .....	91
Figure 3-5. Defining Templates for Generated Filenames .....	102
Figure 3-6. Defining Export Preferences in the Templates Page.....	104
Figure 4-1. Environment Page.....	116
Figure 4-2. Triggers Page .....	122
Figure 4-3. Templates Page .....	124
Figure 4-4. Layer Derivations Tree .....	134
Figure 6-1. Waivers Tab for Waiver Creation.....	164
Figure 7-1. Changing Trace Property Statements .....	181
Figure 7-2. H-Cells Analysis Result Tabs .....	186
Figure 7-3. Edit Parameters for Device Dialog Box .....	191
Figure 8-1. Generic PEX Flow .....	201
Figure 8-2. Blocks Page in Calibre Interactive PEX .....	208
Figure 8-3. Calibre Interactive PEX Probes Page .....	209
Figure 10-1. Configuration Editor .....	234
Figure 10-2. hyperlink Property Example .....	253
Figure 10-3. #DEFINE and Variable Custom Configuration Example.....	403
Figure 10-4. Moving GUI Options with the Configuration File .....	406
Figure 10-5. Configuration File Example with Callback Function .....	409
Figure 11-1. Trigger Pre- and Post-Execution Order .....	414
Figure 11-2. Internal Trigger Timing for Parasitic Extraction Runs .....	416
Figure 11-3. Calibre Interactive Internal Triggers.....	420
Figure A-1. Custom Menu Example for Calibre DESIGNrev.....	454
Figure A-2. Cadence Technology File.....	516
Figure A-3. Interactive Calibre Flow with Synopsys IC Compiler .....	565
Figure A-4. Calibre Menu Contents in Synopsys IC Compiler .....	570
Figure A-5. Calibre Menu in Synopsys IC Compiler .....	571



# List of Tables

---

Table 1-1. Calibre Interactive Features .....	22
Table 1-2. Calibre Integrations .....	25
Table 1-3. Tips for Using the Calibre Interactive GUI .....	28
Table 1-4. Syntax Conventions .....	30
Table 2-1. Loading, Saving, and Using Calibre Interactive Runsets .....	35
Table 2-2. Loading Calibre Interactive Runsets .....	36
Table 3-1. Overwrite File Dialog Box for Layout and Schematic Export .....	68
Table 3-2. Check Selection Recipe Editor — Action Buttons .....	91
Table 3-3. Check Selection Recipe Editor — Basic Controls .....	92
Table 3-4. Check Selection Recipe Editor — Advanced Controls .....	93
Table 3-5. Expression Categories for Check Selection Recipes .....	94
Table 3-6. Replaceable Parameters in Template Names .....	100
Table 3-7. Export Preference Settings in the Setup Preferences Dialog .....	104
Table 4-1. Calibre Interactive GUI Items .....	106
Table 4-2. Read Options .....	112
Table 4-3. Mapping Files Options .....	113
Table 4-4. Additional Files Options .....	114
Table 4-5. Other Options .....	114
Table 4-6. OpenAccess Environment Options .....	115
Table 4-7. Derivation Tree Listings .....	134
Table 4-8. Layer Derivation Buttons .....	135
Table 5-1. Area DRC Options .....	142
Table 5-2. Template Options for DRC HTML Report Creation .....	146
Table 5-3. Format Options for DRC HTML Report Creation .....	147
Table 5-4. Format Options for FastXOR Rule File Generation .....	153
Table 5-5. Mapping Files Options for FastXOR Rule File Generation .....	154
Table 5-6. Options for FastXOR Runs .....	154
Table 5-7. DRC-Specific Pages in Calibre Interactive .....	156
Table 6-1. Calibre Auto-Waivers Information Sources .....	162
Table 7-1. Changing Trace Property Statements in Calibre Interactive nmLVS .....	180
Table 7-2. Device Parameter Fields for Create Device Signature .....	192
Table 7-3. Layers File for Device Signature Creation Contents .....	194
Table 7-4. LVS-Specific Pages in Calibre Interactive .....	197
Table 8-1. Files Required by the Calibre Interactive PEX GUI .....	200
Table 8-2. Inputs Pane for PEX .....	202
Table 8-3. Replaceable Parameters for Output PEX Netlist .....	212
Table 8-4. PEX-Specific Pages in Calibre Interactive .....	214
Table 10-1. Loading a Configuration File .....	230
Table 10-2. Loading a Classic Tcl Customization File .....	232
Table 10-3. Tool Properties .....	247

Table 10-4. Common Option Properties . . . . .	250
Table 10-5. Signals from the Calibre Interactive GUI . . . . .	252
Table 10-6. Object Properties for addTable . . . . .	277
Table 10-7. Component Functions for addTable . . . . .	277
Table 10-8. Functions for FileIO Component . . . . .	284
Table 10-9. Object Properties for launchCommandReturnString . . . . .	301
Table 10-10. Object Properties for addChoicesDefine . . . . .	306
Table 10-11. Object Properties for addDefine . . . . .	311
Table 10-12. Object Properties for addInputDirTextDefine . . . . .	314
Table 10-13. Object Properties for addInputFileTextDefine . . . . .	317
Table 10-14. Object Properties for addIntegerDefine . . . . .	321
Table 10-15. Object Properties for addOutputDirTextDefine . . . . .	325
Table 10-16. Object Properties for addOutputFileTextDefine . . . . .	328
Table 10-17. Object Properties for addRealDefine . . . . .	331
Table 10-18. Object Properties for addTextDefine . . . . .	335
Table 10-19. Object Properties for addTextDefine . . . . .	339
Table 10-20. Object Properties for addChoicesVariable . . . . .	345
Table 10-21. Object Properties for addListVariable . . . . .	349
Table 10-22. Object Properties for addInputDirTextVariable . . . . .	352
Table 10-23. Object Properties for addInputFileTextVariable . . . . .	355
Table 10-24. Object Properties for addIntegerVariable . . . . .	359
Table 10-25. Object Properties for addMultiChoicesVariable . . . . .	362
Table 10-26. Object Properties for addOutputDirTextVariable . . . . .	366
Table 10-27. Object Properties for addOutputFileTextVariable . . . . .	369
Table 10-28. Object Properties for addRealVariable . . . . .	372
Table 10-29. Object Properties for addTextVariable . . . . .	376
Table 10-30. Object Properties for addTextVariable . . . . .	380
Table 10-31. Object Properties for addAction . . . . .	383
Table 10-32. Control Functions for addAction . . . . .	384
Table 10-33. Object Properties for addChoicesAction . . . . .	387
Table 10-34. Control Functions for addChoicesAction . . . . .	388
Table 10-35. Replaceable Trigger Parameters . . . . .	390
Table 10-36. Run Modes for Trigger Functions . . . . .	394
Table 10-37. Correspondence for Customization and Configuration File Commands . . . . .	398
Table 11-1. Triggers in Calibre Interactive . . . . .	413
Table 11-2. Internal Trigger Parameters . . . . .	418
Table 11-3. File Locations for External Trigger Definitions in Calibre DESIGNrev . . . . .	430
Table 11-4. File Locations for External Trigger Definitions in Cadence Virtuoso . . . . .	431
Table 11-5. File Locations for External Trigger Definitions in Synopsys IC Compiler . . . . .	432
Table 12-1. Run Calibre On Options on the Run Control Page . . . . .	436
Table 12-2. Run Calibre Execution Mode . . . . .	436
Table 12-3. Run Calibre Execution Options . . . . .	437
Table 12-4. Resource Option Variables for Spectrum LSF . . . . .	442
Table A-1. Supported Design Tools for Custom Menu Items . . . . .	453
Table A-2. Return Value for mgc_calibre_get_viewer_name . . . . .	457

## List of Tables

---

Table A-3. Verification Menu in Calibre DESIGNrev and WORKbench .....	465
Table A-4. Calibre Menu in Cadence Virtuoso .....	472
Table A-5. Calibre Setup Menu in Cadence Virtuoso .....	473
Table A-6. Calibre Layout Export Setup Dialog Box Contents .....	476
Table A-7. Calibre Netlist Export Setup Dialog Box Contents .....	485
Table A-8. Defining Properties in the Cellmap File .....	504
Table A-9. MGC_FDI_PIN_AUTOMAP_METHOD Values .....	519
Table A-10. Calibre View Setup Dialog Box Contents .....	521
Table A-11. CalibreView Setup File Keywords and Values .....	536
Table A-12. Global Variables for Calibre View Generation .....	542
Table A-13. Calibre View Warnings .....	544
Table A-14. Calibre Menu in Cadence Encounter .....	560
Table A-15. Calibre Menu in Cadence Innovus .....	563
Table A-16. Calibre Menu in Silvaco Expert .....	577
Table B-1. Calibre Interactive Environment Variables .....	579
Table B-2. Calibre View Environment Variables .....	586
Table C-1. General Runset Options for Calibre Interactive .....	591
Table C-2. DRC Runset Options .....	624
Table C-3. LVS Runset Options .....	650
Table C-4. PERC Runset Options .....	684
Table C-5. PEX Runset Options .....	711
Table C-6. Replaceable Parameters in Template Values for Calibre Interactive .....	804
Table C-7. Waiver Setup Runset Options .....	806



# Chapter 1

## Introduction to Calibre Interactive

---

New Calibre® Interactive™ includes support for running Calibre® nmLVS™, Calibre® nmDRC™, Calibre® PERC™, and Calibre® xRC™. The GUI includes the following new features:

- A search option to easily locate controls
- More flexible GUI customization, including the ability to create new GUI pages and relocate options
- Simplified views for the DRC and LVS applications. The simplified view places the most commonly used options together on one application page.

Many features from classic Calibre Interactive are available, such as:

- Trigger functions
- The ability to set environment variables within the GUI
- Integration with Calibre® DESIGNrev™, Cadence® Virtuoso®, Cadence® Encounter®, Cadence® Innovus®, and all design tools supported by classic Calibre Interactive

The topics in this chapter are enough to get you started with the GUI. See the remaining chapters for detailed usage.

<b>Calibre Interactive Overview .....</b>	<b>21</b>
<b>Interactive Workflow for Calibre .....</b>	<b>22</b>
<b>Requirements for Calibre Interactive .....</b>	<b>24</b>
<b>Integration to Design Tools.....</b>	<b>25</b>
<b>GUI Customization in Calibre Interactive .....</b>	<b>28</b>
<b>Tips for Using the Calibre Interactive GUI.....</b>	<b>28</b>
<b>Syntax Conventions .....</b>	<b>30</b>

## Calibre Interactive Overview

Calibre Interactive is a GUI interface to Calibre nmDRC, Calibre nmLVS, Calibre parasitic extraction tools, and Calibre PERC. You can easily adjust rule file settings and run-time options without editing the rule file, then submit the Calibre job on a local, remote, or distributed computer resource.

Calibre Interactive is integrated to most standard industry design tools, as discussed in the section “[Integration to Design Tools](#)” on page 25. When Calibre Interactive is connected to an integrated design tool you can make changes in the layout or schematic, then run Calibre on the modified version; the current layout or schematic is exported to Calibre prior to the start of the run.

Calibre Interactive includes the features listed in the following table.

**Table 1-1. Calibre Interactive Features**

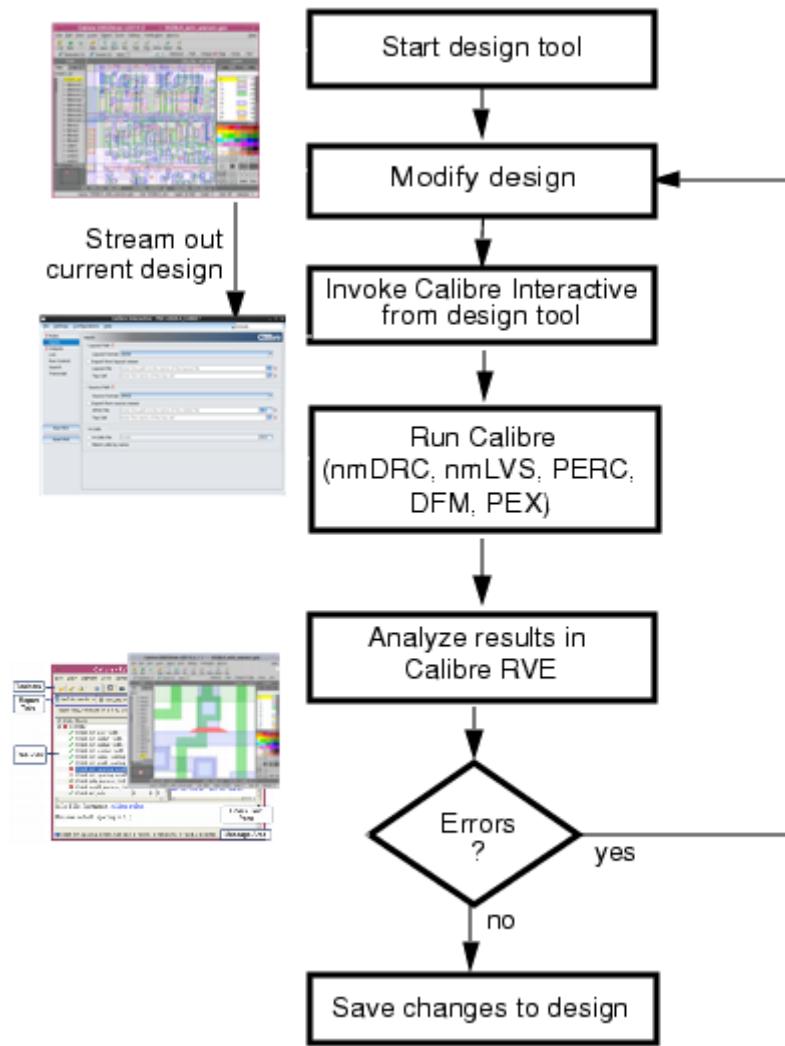
Feature	Reference
Customize rule file settings and run-time options without editing the rule file	<a href="#">“Running Calibre Interactive” on page 57</a>
Submit jobs on local, remote, or distributed computer resources	<a href="#">“Run Control and Licensing Setup” on page 435</a>
Specify scripts and commands to run before and after the Calibre job	<a href="#">“Trigger Functions in Calibre Interactive” on page 413</a>
Run without opening the GUI (Calibre Interactive batch mode)	<a href="#">“Starting a Basic Batch Run in Calibre Interactive” on page 78</a>
Customize the GUI to control certain rule file and GUI settings	<a href="#">“GUI Customization for Calibre Interactive” on page 229</a>
Set environment variables for the run	<a href="#">“Environment Variables” on page 579</a>
Save GUI settings in a “runset” and load them in a later Calibre Interactive session	<a href="#">“Using Calibre Interactive Runsets” on page 34</a>
Add custom menu items to the Calibre integration menu in supported design tools	<a href="#">“Custom Menu Items in Design Tools” on page 453</a>

## Interactive Workflow for Calibre

Calibre® Interactive™ and Calibre® RVE™ are essential components of an interactive workflow using Calibre physical verification tools. Calibre Interactive and Calibre RVE are integrated to most standard industry design tools,

The following figure shows how you can use Calibre Interactive and Calibre RVE in an interactive design process.

**Figure 1-1. Interactive Verification Flow**



The preceding figure shows a general, high-level flow diagram; for more detailed flows, refer to the following topics:

- “[Calibre Interactive nmDRC](#)” on page 137
- “[Calibre Interactive nmLVS](#)” on page 173
- “[Calibre Interactive PEX](#)” on page 199
- “[Calibre Interactive PERC](#)” on page 219

For information on Calibre RVE, see the [Calibre RVE User’s Manual](#).

# Requirements for Calibre Interactive

Several requirements must be met to use Calibre Interactive.

- Using Calibre 2020.4 or later on a platform running the AOI executable.  
The AOI executable supports platforms running Red Hat Enterprise Linux 6 (RHEL 6), RHEL 7, SLES 11 Service Pack 2, or SLES 11 Service Pack 3. See the *Calibre Administrator's Guide* for more information.
- The required variable is defined for the Calibre Interactive application. For example, set the environment variables as follows using csh:

```
setenv CALIBRE_ENABLE_NEW_CI_LVS 1
setenv CALIBRE_ENABLE_NEW_CI_DRC 1
setenv CALIBRE_ENABLE_NEW_CI_PERC 1
setenv CALIBRE_ENABLE_NEW_CI_XRC 1
```

- The Calibre Interactive GUI uses the XKEYBOARD extension. If you notice that keyboard input to the GUI fields is not correct, make sure your interface to the Linux machine running Calibre uses the XKEYBOARD protocol extension.

The following system requirements also must be met:

- **Platform support** — Calibre Interactive is available on all supported platforms found in the *Calibre Administrator's Guide*.
- **Licensing** — Calibre Interactive requires a Calibre Interactive product license. Further license requirements depend upon the Calibre application that is run.

See the *Calibre Administrator's Guide* for complete licensing information.

See “[License Timeout in Calibre Interactive](#)” on page 60 for information on setting the license timeout period.

- **Environment variable for software tree** — Before invoking Calibre Interactive, you must set an environment variable to the location of your Calibre software tree. See “[Setting the CALIBRE\\_HOME Environment Variable](#)” in the *Calibre Administrator's Guide* for details.
- **Layout Design Tool (Optional)** — Access to a layout editor or viewer with a Calibre integration is optional. Calibre Interactive can export the current design from a connected viewer.

Integrated layout viewers include Calibre DESIGNrev, Siemens Sierra Olympus/Pinnacle, Siemens Tanner L-Edit, Cadence Virtuoso, and Synopsys IC Compiler. See “[Integration to Design Tools](#)” on page 25 for a complete list of integrated viewers.

- **Schematic Design Tool (Optional)** — Access to a schematic viewer with a Calibre integration is optional. Some applications, such as Calibre nmLVS, Calibre PERC, and

PEX, require the source schematic as input. Calibre Interactive can export the current schematic from a connected schematic viewer.

Integrated schematic viewers include Siemens IC Station/Pyxis Schematic, Siemens Tanner S-Edit, and Cadence Composer. See “[Integration to Design Tools](#)” on page 25 for a complete list of integrated viewers.

## Integration to Design Tools

Calibre Interactive and Calibre RVE can establish connections to most standard industry design tools. This section describes where to find setup and operation information.

**Table 1-2** lists the design tools that support a Calibre integration. The Setup and Operation Information column indicates whether the integration is automatic or requires configuration.

The Calibre Interfaces web page has additional information about the integration to different design tools:

[www.mentor.com/products/ic\\_nanometer\\_design/calibre-integration/](http://www.mentor.com/products/ic_nanometer_design/calibre-integration/)

**Table 1-2. Calibre Integrations**

Design Tool	Setup and Operation Information	Support by
<b>Custom Design Tools</b>		
IC Station®, Pyxis™ Layout, and Pyxis™ Schematic	No setup required. See <a href="#">IC Station/Pyxis</a> for integration information.	Siemens Digital Industries Software
L-Edit and S-Edit	Setup required. See “ <a href="#">Tanner AMS and MEMS Flows</a> ” on <a href="http://mentor.com">mentor.com</a> for general product information.	Siemens Digital Industries Software
Cadence® Virtuoso®	See <a href="#">Creating an Interface to Cadence Virtuoso</a> and <a href="#">Cadence Virtuoso</a> .	Siemens Digital Industries Software
Synopsys Laker™ (layout only)	No setup required. See “ <a href="#">Synopsys</a> ” on the Calibre Interfaces web page for additional information.	Synopsys
Synopsys® Custom Compiler™	See “ <a href="#">Synopsys</a> ” on the Calibre Interfaces web page.	Synopsys
Keysight	See “ <a href="#">Keysight</a> ” on the Calibre Interfaces web page for additional information.	Agilent

**Table 1-2. Calibre Integrations (cont.)**

Design Tool	Setup and Operation Information	Support by
PhoeniX OptoDesigner	No setup required. See “ <a href="#">PhoeniX Software</a> ” on the Calibre Interfaces web page for additional information.	Phoenix Software
Silvaco Expert	No setup required. See <a href="#">Silvaco Expert</a> in this manual for more information.  Also see “ <a href="#">Silvaco</a> ” on the Calibre Interfaces web page for additional information.	Silvaco
<b>Place and Route Tools</b>		
Siemens Olympus-SoC™	No setup required.	Siemens Digital Industries Software
Synopsys® IC Compiler™ and Synopsys IC Compiler II (ICC2)	See <a href="#">Synopsys IC Compiler and Synopsys IC Compiler II</a> .	Siemens Digital Industries Software
Cadence® Encounter®	See <a href="#">Cadence Encounter</a> .	Siemens Digital Industries Software
Cadence® Innovus®	See <a href="#">Cadence Innovus</a> .	Siemens Digital Industries Software
<b>Layout Editors and Viewers</b>		
Calibre® DESIGNrev™ and Calibre® WORKbench™	No setup required. See <a href="#">Calibre DESIGNrev and WORKbench</a> for operational information.	Siemens Digital Industries Software
Cadence® QuickView Signoff Data Analysis Environment	No setup required. The integration is to Calibre RVE only. See “ <a href="#">Cadence QuickView</a> ” in the <i>Calibre RVE User’s Manual</i> .	Cadence
TOOL Corporation LAVIS	See “ <a href="#">TOOL</a> ” on the Calibre Interfaces web page.  Supports the interface to Calibre RVE; no setup required.	TOOL

**Table 1-2. Calibre Integrations (cont.)**

<b>Design Tool</b>	<b>Setup and Operation Information</b>	<b>Support by</b>
AnaGlobe Technology	See “ <a href="#">AnaGlobe Technology</a> ” on the Calibre Interfaces web page.  Supports the standard interfaces to Calibre Interactive and Calibre RVE. No setup required; on by default.	AnaGlobe Technology
Jedat a-SX Ismo	See “ <a href="#">Jedat</a> ” on the Calibre Interfaces web page.  Supports the standard interfaces to Calibre Interactive and Calibre RVE. No setup required.	Jedat
<b>Schematic Viewers and Editors</b>		
Siemens DFT Schematic Viewer in DFTInsight™	No setup required.	Siemens Digital Industries Software
Cadence® Composer	<a href="#">Creating an Interface to Cadence Virtuoso</a> and <a href="#">Cadence Composer</a> .	Siemens Digital Industries Software
<b>Analysis Tools</b>		
Ansys® Totem™	See “ <a href="#">Ansys</a> ” on the Calibre Interfaces web page.	Ansys
CWS SiPEX™	See “ <a href="#">CWS</a> ” on the Calibre Interfaces web page.	CWS
Helic® VeloceRF™	See “ <a href="#">Helic</a> ” on the Calibre Interfaces web page.	Helic
Integrand Software, Inc.	See “ <a href="#">Integrand Software</a> ” on the Calibre Interfaces web page.	Integrand
Lorentz Solution, Inc.	See “ <a href="#">Lorentz Solutions</a> ” on the Calibre Interfaces web page.	Lorentz Solution
NI AWR Analog Office® and Microwave Office®	See “ <a href="#">NI (formerly AWR)</a> ” on the Calibre Interfaces web page.	NI
Magwel NV	See “ <a href="#">Magwel NV</a> ” on the Calibre Interfaces web page.	Magwel NV

## GUI Customization in Calibre Interactive

GUI Customization is done with a configuration file that uses configuration API commands and the JavaScript<sup>TM</sup><sup>1</sup> language for scripting. The configuration file changes the Calibre Interactive GUI, rather than creating a Customizations Setting dialog box, as with classic Calibre Interactive. As with classic Calibre Interactive, you can specify #DEFINE and Variable statements, and read and set GUI options. The new configuration file enables you to create new GUI pages, move GUI controls, and add GUI buttons that call a user-defined function.

[“GUI Customization for Calibre Interactive” on page 229](#) has all the information you need to develop and use custom configuration files, including a guide and examples for adapting existing Tcl-based customization files to the new configuration file format.

You can write your own configuration file, or create a configuration file with an easy to use GUI. See [“Customizing the GUI With the Configuration Editor” on page 233](#).

You can use a simplified view of the GUI without creating your own customization file. See [“Using Simplified Views” on page 54](#).

## Tips for Using the Calibre Interactive GUI

A search toolbar, rule file loading, and other features make the Calibre Interactive GUI easy to use.

**Table 1-3. Tips for Using the Calibre Interactive GUI**

To do this ...	Do the following:
Search the GUI	Enter a search term in the entry field at the top right of the GUI. You can enter runset option names (including classic options) to find the corresponding control. Use Ctrl-f or Ctrl-F to change focus to the search entry box. <a href="#">See “Search Toolbar in Calibre Interactive” on page 132.</a>
Hide and show pages	Choose <b>Settings &gt; Show Pages</b>
View the control file	Click the <b>Run &lt;app&gt;</b> button while pressing the Ctrl key, or choose <b>File &gt; View Control File</b> . The control file is opened in the Files page and the command line is displayed on the Transcript page.
Load the rule file	Click the  button on the Rules page. Note: The rule file settings overwrite current GUI settings, including the path to the input design and the top cell name.

---

1. JavaScript is a registered trademark of the Oracle Corporation.

**Table 1-3. Tips for Using the Calibre Interactive GUI (cont.)**

To do this ...	Do the following:
Specify additional rule files	<p>On the <b>Options</b> page, check “Include Rules Files,” and enter the path to the additional rule file(s).</p> <p>Choose <b>Settings &gt; Show Pages</b> if the Options page is not available.</p>
Specify additional layout or schematic input files	<p>On the <b>Database</b> page, expand the Library area and specify files.</p> <p>Choose <b>Settings &gt; Show Pages</b> if the Database page is not available.</p>
Use wildcards for GUI input	<p>You can use the glob style wildcard characters in the fields for layout, source, and v2lvs input files, and for included rule files.</p> <ul style="list-style-type: none"> <li>* — Matches multiple characters.</li> <li>? — Matches a single character.</li> <li>[<i>chars</i>] — Matches any character in the set <i>chars</i>. If a sequence of the form a-z appears in <i>chars</i>, then any character between a and z, inclusive, is matched.</li> </ul> <p>If no matches are found, a red ! is displayed next to the field. The literal input is saved in the runset; for example “*.gds”. The control file uses the found matches; for example “layoutA.gds layoutB.gds”.</p>
Start Calibre® RVE™ after the run	Click <b>Run Control</b> on the left panel, expand the “RVE Options” area, and enable “Show Results in RVE.”
Load a runset	<p><b>File &gt; Load Runset</b> or <b>File &gt; Recent Runsets</b></p> <p>Also see “<a href="#">About Calibre Interactive Runsets</a>” on page 34.</p>
Save a runset	<p><b>File &gt; Save Runset</b>, or use Ctrl-r to save a runset.</p> <p>Only non-default options are saved. To save all options, set the environment variable MGC_CALIBRE_SAVE_ALL_RUNSET_VALUES to 1 before invoking Calibre Interactive.</p>
Use a simplified view	For DRC and LVS, you can use a simplified view that only displays basic controls. See “ <a href="#">Using Simplified Views</a> ” on page 54.
Change the font size for the GUI	<b>Settings &gt; Change Font</b>

**Table 1-3. Tips for Using the Calibre Interactive GUI (cont.)**

To do this ...	Do the following:
View and edit text files	<b>File &gt; Open Text File</b> , <b>File &gt; Recent Text Files</b> , or click the open file icon (  ) next to an input file such as the rule file. See “ <a href="#">Populating the Load Runset Files Dialog Box and Recent Files Lists</a> ” on page 39. Choose <b>File &gt; Save Text File</b> to save a file that you edit.

## Related Topics

[Invoking the Calibre Interactive GUI](#)

[Setting Up and Starting a Run](#)

# Syntax Conventions

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

You should enter literal text, that which is not in italics, exactly as shown.

**Table 1-4. Syntax Conventions**

Convention	Description
<b>Bold</b>	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.
[ ]	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.

**Table 1-4. Syntax Conventions (cont.)**

Convention	Description
<b>Example:</b> <code>mgc_calibre_add_menu_item -type menu -label “label”</code> [-view_type {layout   schematic}] [-after “label”   -before “label”] [-parent “label”]	



# Chapter 2

## Basic Calibre Interactive Usage

---

This chapter describes basic usage, such as starting a run, connecting to a design tool, and searching for controls.

<b>About Calibre Interactive Runsets.....</b>	<b>34</b>
Using Calibre Interactive Runsets .....	34
Methods to Load a Calibre Interactive Runset .....	35
Converting Classic Calibre Interactive Runsets .....	37
Environment Variables for Calibre Interactive Runsets.....	38
Populating the Load Runset Files Dialog Box and Recent Files Lists .....	39
Using Multiple Calibre Interactive Runsets to Combine Different Sets of Options .....	43
<b>About Control Files .....</b>	<b>44</b>
Control File in Calibre Interactive .....	44
Control Filenames.....	46
Creating and Viewing the Control File Without Starting a Run.....	46
Running Batch Calibre with a Calibre Interactive Control File .....	46
<b>About Rule Files.....</b>	<b>50</b>
Rule File Operations in Calibre Interactive .....	50
How Rule Files Relate to Control Files .....	51
<b>Environment Variables in Calibre Interactive GUI Fields .....</b>	<b>52</b>
<b>Using Simplified Views .....</b>	<b>54</b>

## About Calibre Interactive Runsets

A runset is a text file created by Calibre Interactive to store the settings you specify in the GUI. You can load a runset into Calibre Interactive to restore the GUI interface to the saved settings. You also use a runset to run Calibre Interactive in batch mode.

Runsets provide two important benefits:

- They reduce your work time. You no longer need to repeatedly enter all the information necessary to run Calibre.
- They help you manage your design environment. You can create a different runset for each design situation (process, cells, block-level verification, full-chip verification, and so on).

A Calibre Interactive runset includes all the information in the GUI that is not a default value, however, you can set an environment variable to save all GUI options to the runset. Choose **File > Save Runset** or **File > Save Runset As** to save a runset. Information saved in the runset includes:

- All GUI settings.
- Settings in the **Preferences** page (**Settings > Show Pages > Preferences**), except the **Startup** section.

There are various methods for specify and load a Calibre Interactive runset. You can specify the runsets that are displayed in the Load Runset Files dialog box and the recent files list.

<b>Using Calibre Interactive Runsets .....</b>	<b>34</b>
<b>Methods to Load a Calibre Interactive Runset.....</b>	<b>35</b>
<b>Converting Classic Calibre Interactive Runsets.....</b>	<b>37</b>
<b>Environment Variables for Calibre Interactive Runsets.....</b>	<b>38</b>
<b>Populating the Load Runset Files Dialog Box and Recent Files Lists .....</b>	<b>39</b>
<b>Using Multiple Calibre Interactive Runsets to Combine Different Sets of Options ...</b>	<b>43</b>

## Using Calibre Interactive Runsets

Calibre Interactive runsets save the settings in the GUI so you can easily return to the same setup. There are preference settings and environment variables that affect loading and saving runsets.

**Table 2-1. Loading, Saving, and Using Calibre Interactive Runsets**

Action	Method and Description
Load a runset from the GUI	<b>File &gt; Load Runset</b> <b>File &gt; Recent Runsets</b> <b>i Tip:</b> You can pre-populate the recent runset list; see “ <a href="#">Populating the Load Runset Files Dialog Box and Recent Files Lists</a> ” on page 39.
Automatic runset loading	You can load a runset from the command line, with an environment variable, or using a startup preference setting. See “ <a href="#">Methods to Load a Calibre Interactive Runset</a> ” on page 35.
Save a runset	<b>File &gt; Save Runset</b> Only non-default options are saved to the runset. To save all options, set the environment variable MGC_CALIBRE_SAVE_ALL_RUNSET_VALUES to 1.
Set preferences for saving runsets	Choose <b>Settings &gt; Show Pages &gt; Preferences</b> to view the Preferences page and expand the Runset option area. The preferences behave as described. <ul style="list-style-type: none"> <li>• On Close Runset:           <ul style="list-style-type: none"> <li>• Prompt to save changes before closing runset</li> <li>• Save changed settings before closing runset</li> <li>• Do not save changed settings</li> <li>• Save runset each time Calibre is run</li> </ul> </li> </ul>
Specify runset type and information	Choose <b>Settings &gt; Show Pages &gt; Preferences</b> to view the Preferences page, expand the Runset option area, and specify Type and Info.

## Methods to Load a Calibre Interactive Runset

You can load Calibre Interactive runsets from the command line, from the GUI, by defining an environment variable, or by setting a startup preference.

**Table 2-2. Loading Calibre Interactive Runsets**

Action	Method and Description
Load runset(s) from the GUI	<b>File &gt; Load Runset</b> To load multiple runsets, click a second runset in the list of recent runsets, or click the + icon (  ) to add a new runset entry. Use the up and down icons to change the runset order.
Automatic runset loading, given in order of priority:	
<ul style="list-style-type: none"> <li>Load a runset or multiple runsets from the command line (Priority 1, highest)</li> </ul>	-runset command line option: -runset my_runset -runset runset1 -runset runset2 The -runset_options_display command line option can be used in batch operation.
<ul style="list-style-type: none"> <li>Load a runset or multiple runsets with an environment variable (Priority 2)</li> </ul>	Define the environment variable MGC_CALIBRE_<app>_RUNSET_FILE with the runset(s), where <app> is replaced with DRC, LVS, PEX, or PERC. Enclose multiple runsets in quotation marks.
<ul style="list-style-type: none"> <li>Load a runset on startup (Priority 3, lowest)</li> </ul>	1. Choose <b>Show Pages &gt; Preferences</b> . 2. Expand the Startup option. 3. For “Runset Loading,” select Load Runset Files and specify a runset. To specify multiple runsets, position cursor in first runset entry and press Enter on the keyboard.

If you load multiple runsets, they are loaded in the order specified. If any settings conflict, the setting in the most recently loaded file is used. When you save a runset after loading multiple runsets, you are prompted to enter a new runset filename. The saved runset includes settings from both runsets that were loaded, and any changes that were made during the Calibre Interactive session.

---

**Note**

 Configuration files are loaded first and in the order specified, then runsets are loaded in the order specified. If configuration files are specified in a runset, the configuration files are loaded before the runset settings. If any settings conflict, the setting in the most recently loaded file is used. For example, if a configuration file and a runset both specify a value for the same GUI setting, the value in the runset is used.

---

# Converting Classic Calibre Interactive Runsets

Calibre Interactive can read in classic Calibre Interactive runsets. When you save a runset it is automatically saved in the new runset format. Conversion issues are noted in the transcript.

## Prerequisites

- “[Requirements for Calibre Interactive](#)” on page 24

## Procedure

1. Invoke the GUI as instructed in “[Invoking the Calibre Interactive GUI](#)” on page 58.

If you are invoking the GUI using the command line, you can specify a classic runset with the -runset command-line option.

If you are running in batch mode (with the -batch command line option), you can specify classic runset options on the command line, as with the classic Calibre Interactive command-line invocation. You can also specify to save the classic runset as a new converted runset; see “[Calibre Interactive Command Line](#)” on page 61 for more information.

2. If you did not specify a runset on the command line, choose **File > Load Runset** and select a classic Calibre Interactive runset.

If there were issues reading the classic runset, the view automatically switches to the Transcript page, where warning, informational, and error messages are reported. If you ran in batch mode, the messages are printed to the transcript.

3. If the Transcript page or transcript contains messages related to runset options, handle them as follows:

Message	Problem and Action
WARNING ... <runset_option>: Value of <value> is not supported ...	The option value is not supported. Supply a valid value in the classic runset and reload the runset, or correct the problem in the GUI.
WARNING ... Conversion for <runset_option> not found -- (skipping)	The classic runset option does not have a conversion. It may be an old, unused option, or a new and unrecognized option. Confirm the validity of the option and contact your program representative if needed.
WARNING ... Keyword <runset_option> not recognized -- (skipping)	The classic runset option is not recognized. Confirm the validity of the option and contact your program representative if needed.
INFO ... Conversion for <runset_option> currently not supported -- (skipping)	The classic runset option is not supported. Contact your program representative if you have questions about support for the option.

4. Look for  icons in the left panel of the GUI, indicating an incorrect or missing setting.  
Click the page name with the error, locate the problem option, hover over the  to view a message about the problem, and correct as needed.
5. View each GUI page and search for the  icon. The icon indicates a warning, such as an option setting that conflicts with another option. Hover over the icon to view a message about the problem and correct as needed. Warning icons are not displayed by the page name in the left panel of the GUI.
6. Choose **File > Save Runset As** to save the runset in the new format. You are not allowed to overwrite the classic runset.

## Environment Variables for Calibre Interactive Runsets

You can specify runsets to load automatically on invocation with an environment variable. You can also specify that all runset options are saved rather than only non-default options.

### Loading Runsets

The following environment variables specify a runset that is loaded automatically when you invoke Calibre Interactive. You can specify multiple runsets as a space-separated list.

MGC\_CALIBRE\_LVS\_RUNSET\_FILE

MGC\_CALIBRE\_DRC\_RUNSET\_FILE

MGC\_CALIBRE\_PERC\_RUNSET\_FILE

MGC\_CALIBRE\_PEX\_RUNSET\_FILE

The environment variable name indicates the corresponding application. If multiple runsets are specified, they are loaded in order (left to right) and the settings loaded last take precedence.

The -runset command-line option has precedence over the environment variable.

For example, in csh:

```
setenv MGC_CALIBRE_DRC_RUNSET_FILE "runset1_path runset2_path"
```

The specified runsets are loaded in order when Calibre Interactive nmDRC is invoked, unless the -runset command-line option is used.

**Note**

 Configuration files are loaded first and in the order specified, then runsets are loaded in the order specified. If configuration files are specified in a runset, the configuration files are loaded before the runset settings. If any settings conflict, the setting in the most recently loaded file is used. For example, if a configuration file and a runset both specify a value for the same GUI setting, the value in the runset is used.

---

## Populating the Recent Runsets List

You can pre-populate the Recent Runsets list; see “[Populating the Load Runset Files Dialog Box and Recent Files Lists](#)” on page 39.

## Saving Non-Default Options

By default, only non-default options are saved when you save a runset. Set the environment variable MGC\_CALIBRE\_SAVE\_ALL\_RUNSET\_VALUES to 1 to save all options.

# Populating the Load Runset Files Dialog Box and Recent Files Lists

You can pre-populate the Load Runset Files dialog box with a list of runsets. You can also populate the recent files lists for runset and text files. The list of runsets and text files is given in a text file that is specified with an environment variable.

The runset list file is specified by one of the following environment variables:

MGC_CALIBRE_DRC_RUNSET_LIST	MGC_CALIBRE_PEX_RUNSET_LIST
MGC_CALIBRE_LVS_RUNSET_LIST	MGC_CALIBRE_XACT_RUNSET_LIST
MGC_CALIBRE_PERC_RUNSET_LIST	

The runset list file can be in the basic or advanced format. You cannot mix formats within the same file.

**Basic Format** — Used to specify single runsets:

```
runset1
runset2
runset3
...
```

**Advanced Format** — Used to specify single runsets, runset groups, runset type and info, and text files.

```
<app>_RS paths: runset [runset ...] [runsetType: type] [runsetInfo: info]
<app>_RS paths: runset [runset ...] [runsetType: type] [runsetInfo: info]
...
TXT paths: text_file [text_file ...]
TXT paths: text_file [text_file ...]
...
```

where file entries are defined as follows:

- *<app>\_RS paths: runset\_file ...* — Keyword and parameter specifying one or more runset files. If more than one runset file is specified, they are grouped together in the Load Runset Files dialog box. *<app>* is one of DRC, LVS, PERC, PEX, or XACT.  
Include one line for each runset file or runset group you want to list in the Load Runset Files dialog box and Recent Runsets list.
- *runsetType: type* — Optional entry with “*<app>\_RS paths:*” that specifies information to display in the Type column of the Load Runset Files dialog box.
- *runsetInfo: info* — Optional entry with “*<app>\_RS paths:*” that specifies information to display in the Info column of the Load Runset Files dialog box.
- *TXT paths: text\_file ...* — Keyword and parameter specifying one or more text files. Text files are listed one per line in the Recent Text Files list.

Both relative and absolute filenames can be used. Relative filenames are evaluated relative to the Run Directory specified on the Rules page. Environment variables can be used in the filenames. Examples are given after the procedure.

## Prerequisites

- “[Requirements for Calibre Interactive](#)” on page 24
- A text file in the basic or advanced format, described previously

## Procedure

1. Specify the runset list with the environment variable appropriate for the application (see previous list).

For example, with Calibre Interactive nmDRC:

```
setenv MGC_CALIBRE_DRC_RUNSET_LIST drc_runset_list
```

If the file referenced by the environment variable is a relative pathname, the tool searches for it in the directory from which you invoke Calibre Interactive.

You can define the environment variable as a list of multiple files:

```
setenv MGC_CALIBRE_XACT_RUNSET_LIST "xact_list1 xact_list2"
```

Different files can have different formats (advanced or basic), but you cannot mix formats within the same file.

2. (Optional) Specify the default selected runset with the environment variable MGC\_CALIBRE\_\*\*\*\_RUNSET\_FILE.

For example, with Calibre Interactive nmDRC:

```
setenv MGC_CALIBRE_DRC_RUNSET_FILE drc_runset_file
```

When MGC\_CALIBRE\_\*\*\*\_RUNSET\_LIST is defined, the runsets specified by MGC\_CALIBRE\_\*\*\*\_RUNSET\_FILE are selected if they are in the list of predefined runsets.

3. Invoke Calibre Interactive.

See “[Invoking the Calibre Interactive GUI](#)” on page 58.

## Results

When you choose **File > Load Runset**, the dialog box is populated both the listed runsets and recent runsets. The Load Runset Files dialog box includes a dropdown selection to view “Predefined Runsets” (listed runsets) or “Recent Runsets”.

When you choose **File > Recent Runsets**, the list is pre-populated with the listed runsets.

If you used the advanced format runset list and specified text files, when you choose **File > Recent Text Files**, the list is pre-populated with the listed text files.

If you open other runsets or text files during the Calibre Interactive session, those files are added to the appropriate recent files list.

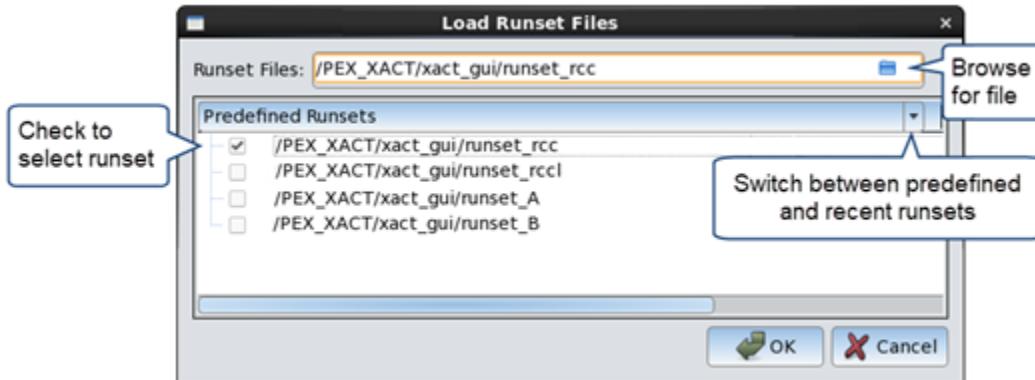
## Examples

### Basic Runset List

This basic runset list is for Calibre Interactive xACT, and specified with MGC\_CALIBRE\_XACT\_RUNSET\_LIST.

```
runset_rcc
runset_rccl
runset_A
runset_B
```

The dialog box is populated with the listed runsets.



When you choose **File > Recent Runsets**, that list is also pre-populated with the listed runsets.

#### Advanced Runset List

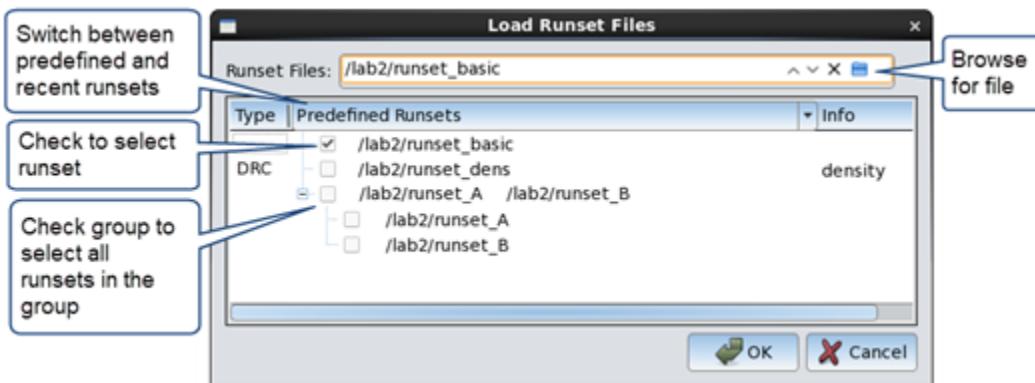
This is an example of an advanced format runset list file for Calibre Interactive nmDRC. The file is specified with MGC\_CALIBRE\_DRC\_RUNSET\_LIST:

```
DRC_RS paths: runset_basic
DRC_RS paths: runset_dens runsetType: DRC runsetInfo: density
DRC_RS paths: runset_A runset_B
TXT paths: $PROJECT/README_Project
```

This example specifies two single runsets, *runset\_basic* and *runset\_dens*, with relative filenames; these files must exist in the Run Directory specified on the Rules page. The optional runsetType and runsetInfo entries are provided for the runset *runset\_dens*.

The runsets *runset\_A* and *runset\_B* are specified as a group. They appear as a group in the Load Runset Files dialog box, and are both loaded if selected as a group. The two runsets are listed separately in the Recent Runsets list.

This is a view of the Load Runset Files dialog box:



The text file, *\$PROJECT/README\_Project*, is specified with an absolute path and uses an environment variable for the directory path. It is listed in the **File > Recent Text Files** list (not shown).

## Using Multiple Calibre Interactive Runsets to Combine Different Sets of Options

When multiple runsets are loaded in Calibre Interactive, they are loaded in order and the settings loaded last take precedence. You can use multiple runsets to combine runsets that include different sets of options. For example, you can combine a basic runset with a second runset that contains settings for output options or host processor control.

Suppose you want to combine a basic DRC runset with a runset which only contains options for creating the DRC HTML report. You can create the following runset named *runset\_HTML*:

```
*drcCreateDRCReport: 1  
*drcDRCReportConfigFile: /user/drc_html.ini  
*drcDRCReportConfigTemplate: Custom Template
```

To load *runset\_HTML* with a runset named *runset\_DRC*, issue the following on the command line:

```
% calibre -gui -drc -runset "runset_DRC runset_HTML"
```

If the runsets specify any of the same options, the settings in *runset\_HTML* take precedence because it is loaded last.

For other methods to load multiple runsets, see “[Methods to Load a Calibre Interactive Runset](#)” on page 35. You can load multiple runsets with a startup preference, or with an environment variable.

You can also load multiple runsets from the GUI. Select **File > Load Runset** and select multiple runsets in the Load Runset Files dialog box.

---

### Note

---

 If you load multiple runsets and the setting “Save runset to run directory each time Calibre is run” (**Setting > Show Pages > Preferences**, in the **Runset** section) is enabled, the runset is saved as *runset.autosaved*.

---

## About Control Files

---

The control file in Calibre Interactive contains the settings in the GUI.

<b>Control File in Calibre Interactive.....</b>	<b>44</b>
<b>Control Filenames .....</b>	<b>46</b>
<b>Creating and Viewing the Control File Without Starting a Run .....</b>	<b>46</b>
<b>Running Batch Calibre with a Calibre Interactive Control File.....</b>	<b>46</b>

## Control File in Calibre Interactive

The Calibre Interactive control file provides all the information Calibre needs to process data, based on the settings you define in the user interface.

These settings include the following:

- The rule file pathname
- The input file pathnames
- The output file pathnames
- Special settings for controlling the run

The control file contains valid SVRF statements or TVF code if the main rule file is a TVF file. Environment variables are expanded in SVRF statements that specify a pathname. Environment variables for primary cell names are also expanded.

---

### Caution

 You should *not* modify control files generated by Calibre Interactive because this can cause unexpected conflicts with rule files that appear in Include statements. The statement DRC ICSTATION YES is internal to Calibre Interactive and should not be used in any rule file except the control file that Calibre Interactive internally generates.

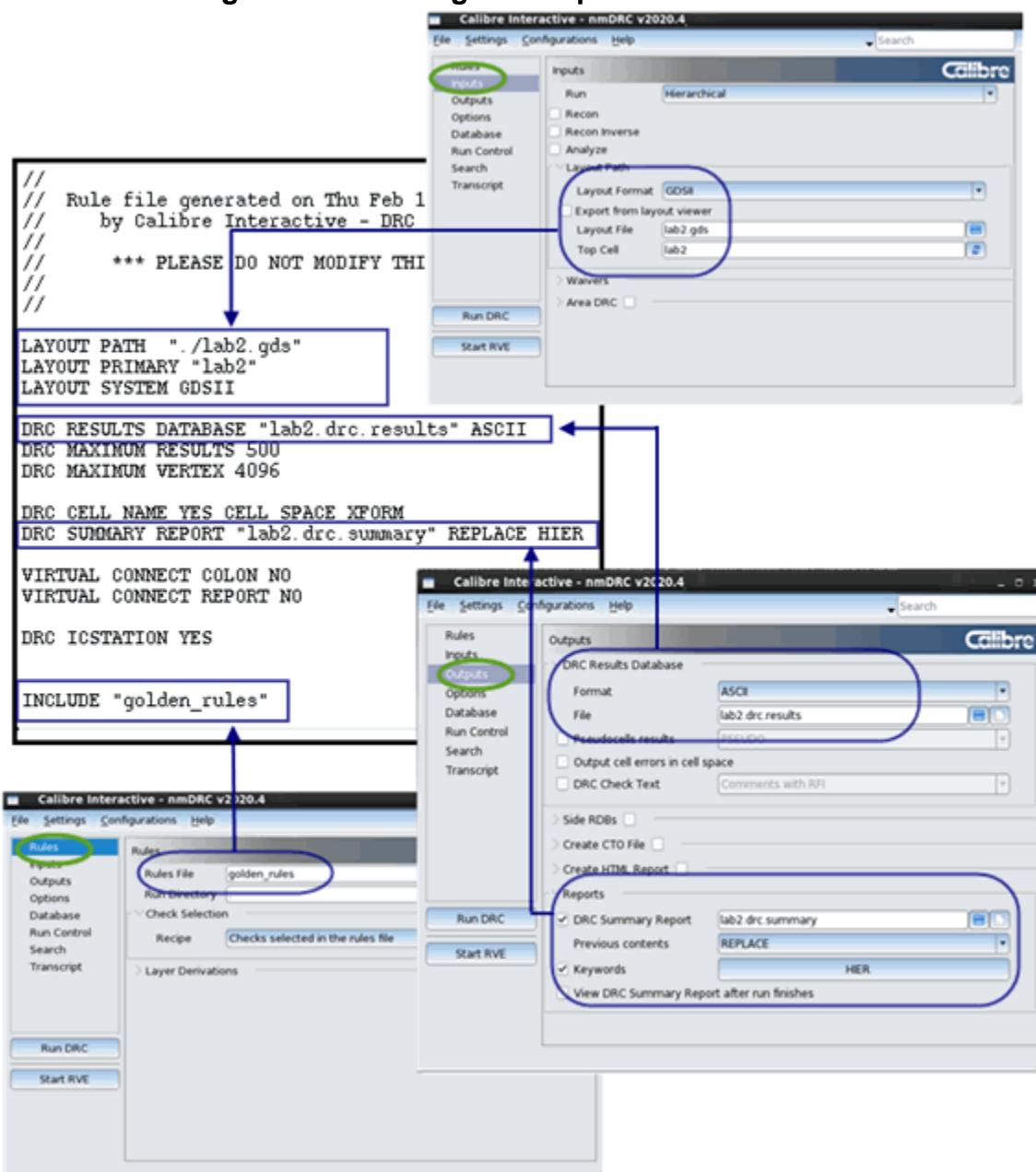
---

The control file is a special type of rule file. Control files differ from standard rule files in the following ways:

- **Who creates them** — Calibre Interactive creates the control files based on the settings you define in the GUI; rule files are created separately, typically by the foundry or the design team.
- **File precedence** — Control files take precedence over rule files. Where there are possible conflicts, the control file settings are used for the run.

Figure 2-1 shows a Calibre Interactive control file and how the entries correspond to settings in the GUI. Also see “[How Rule Files Relate to Control Files](#)” on page 51 for more information.

**Figure 2-1. Relating User Input to Control Files**



**Tip**

- i** You can instruct Calibre Interactive to create a control file that can be used in a command line Calibre run. The setting “Write control file suitable for command line run” is in the **Rules** section of the **Preferences** page, as described in “[Rules Preferences](#)” on page 128. See “[Running Batch Calibre with a Calibre Interactive Control File](#)” on page 46.

## Control Filenames

Calibre Interactive writes the control file to the run directory with a name that is based upon the loaded rule file, with preceding and succeeding underscores (\_).

\_<ruleFileName>\_

for example:

\_testrun3A\_

This control file is overwritten each time you run your job using the same rule file.

You can specify a different name for the control file with the setting “Specify control file name” in the Rules section of the **Preferences** page.

## Creating and Viewing the Control File Without Starting a Run

You can instruct Calibre Interactive to create a control file based on the current GUI settings. This is useful if you want to view the control file before starting the run.

### Procedure

1. Make all required GUI settings in Calibre Interactive.
2. Do one of the following:
  - Choose **File > View Control File** from the main menu.
  - Hold down the Ctrl key and click the **Run** button on the left panel of Calibre Interactive.

### Results

Calibre Interactive generates the control file and opens it in a text viewer. In addition, the Transcript pane displays the Calibre command line.

## Running Batch Calibre with a Calibre Interactive Control File

Calibre Interactive can create a control file that you can use as the rule file for a command line Calibre run. The control file includes GUI settings from your Calibre Interactive setup, such as check selection, top cell name, and other settings.

The control file does not include the following settings from Calibre Interactive:

- Export from layout or schematic viewer

Export from a design tool is only possible when using Calibre Interactive. You can run Calibre Interactive in batch mode and export the design from an open design tool. See “[Specifying Layout and Schematic Export for a Calibre Interactive Batch Run](#)” on page 79.

- Database read options for OpenAccess and LEF/DEF direct read

If the layout database format is OPENACCESS or LEFDEF, then the database read options must be set with the `MGC_CALIBRE_DB_READ_OPTIONS` environment variable; this is described in the following procedure.

- Environment variables defined in Calibre Interactive with the [Environment Page in Calibre Interactive](#)

Environment variables defined within a Calibre Interactive runset are not included in the control file and must be defined in your shell environment prior to a command line Calibre run. This is described in the following procedure.

- Trigger settings

Trigger functions defined within Calibre Interactive are not included in the control file; the trigger functions must be called separately. See “[Trigger Functions in Calibre Interactive](#)” on page 413 for more information on trigger functions.

## Procedure

1. Set up Calibre Interactive with the desired options or load a runset with the desired options.
2. Specify a control file for a command line run as follows:
  - a. Choose **Settings > Show Pages > Preferences** to open the **Preferences** page.
  - b. In the Rules section, enable “Write control file suitable for command line run.”
  - c. (Optional) Enable “Specify control file name” and enter a filename in the text field.
3. Determine if environment variables are defined within Calibre Interactive as follows:
  - a. Choose **Settings > Show Pages > Environment** to open the **Environment** page.
  - b. Adjust the filter to search for environment variables. A green check mark in the Runset Value column indicates that the environment variable is defined in the Calibre Interactive runset. Make a note of any environment variables defined in the runset and the corresponding runset value so the environment variable can be defined in the shell environment later.

4. Determine if trigger functions are defined as follows:
  - a. Choose **Settings > Show Pages > Triggers** and navigate to the **Triggers** page. This page displays internal triggers.
  - b. Review the section “[External Trigger Functions and File Locations](#)” on page 430, then determine if any external triggers are defined within your design tool environment. Make a note of any external trigger functions.

You may want to call any external and internal trigger functions from a shell script. Consult the chapter “[Trigger Functions in Calibre Interactive](#)” on page 413 to determine the correct timing of the trigger functions in relation to the Calibre run.

5. Choose **File > View Control File** to open a text viewer with the control file.
6. If your layout format is OPENACCESS or LEFDEF in the Layout Path section of the **Inputs** page, find lines similar to the following at the beginning of the control file:

```
// MGC_CALIBRE_DB_READ_OPTIONS to:  
//  
// -rundir ./data -abortOnEmptyPCell
```

Make a note of the line beginning with “-rundir”—this is used later to set an environment variable with the database read options for the run. The database read options are determined from the settings on the OA/LEFDEF page.

7. Choose **File > Save As** to give the control file a new name or close the control file to keep the same filename.

The name of the control file is *\_rules\_*, where *rules* is the rule file on the Rules pane, or the filename specified in Step 2.c. The control file is overwritten each time you run Calibre Interactive, so you may want to save this control file to a different filename.

8. Click the **Transcript** button on the left panel of Calibre Interactive to view the Transcript pane. Make a note of the command line for the run, which is displayed in the Transcript pane.
9. Exit Calibre Interactive.
10. If runset environment variables were defined in Calibre Interactive (Step 3), define them in your shell environment.
11. If your layout format is OpenAccess or LEF/DEF, define the environment variable **MGC\_CALIBRE\_DB\_READ\_OPTIONS** as found in Step 6, using quotes around the option string. For example:  

```
setenv MGC_CALIBRE_DB_READ_OPTIONS "-rundir ./data -abortOnEmptyPCell"
```
12. Execute any external pre-execution trigger functions found in Step 4, then internal pre-execution trigger functions.

13. Execute Calibre with the options used on the command line viewed on the Transcript pane (Step 8) and use the control file saved in Step 7 as the rule file. If you renamed the control file in Step 7, then the command line on the Transcript pane will not have the correct filename.
14. Execute any internal post-execution trigger functions found in Step 4, then external post-execution trigger functions.

## About Rule Files

Rule files define the processing Calibre performs during a run. You specify the rule file for a Calibre Interactive run on the Rules pane.

Calibre Interactive uses rule files specified in the following ways:

- **The primary rule file defined on the Rules pane** — This serves as the rule file on which the Calibre run is based.
- **Rule files included through the primary rule file** — These are included in any Calibre run regardless of whether Calibre is invoked from the command line or Calibre Interactive. Loading the primary rule file in Calibre Interactive causes included rule files to be loaded as well. (For more information, refer to the description of the [Include](#) command in the *Standard Verification Rule Format Manual*).
- **Rule files included through the GUI** — These are defined on the [Include](#) tab on the Options pane. These rule files act as included files called by the GUI. They allow you to specify included rule files outside of the primary rule file. They are listed in the control file within include statements.

<b>Rule File Operations in Calibre Interactive .....</b>	<b>50</b>
<b>How Rule Files Relate to Control Files .....</b>	<b>51</b>

## Rule File Operations in Calibre Interactive

When working with rule files in Calibre Interactive you can view and edit the rule file, check the rule file syntax, select the rule checks to execute, and load rule file settings into the Calibre Interactive GUI.

The following actions are available:

- **View and edit rule files** — You can view and edit any rule file using the Calibre Interactive text editor. Choose **File > Open Text File** and select the rule file. Choose **File > Save Text File** to save the file that you edit.  
  
You can also view the Layer Derivations tree near the bottom of the Rules pane. Click the  icon to open the tree and continue to expand the tree as needed. See “[Layer Derivations Tree](#)” on page 134 for more information.
- **Specify additional rule files and rule file statements** — In the **Options** page (**Settings > Show Pages > Options**), enable Include Rules Files and Include Rules Statements and specify rule files and statements.

- **Check rule file syntax**— Calibre Interactive causes your rule file to be compiled and reports any errors. Rule file compilation occurs as a preliminary step in each of the following tasks:
  - Loading a rule file ( on the Rules page).
  - Opening the [Recipe Editor Page](#).
  - Choosing **Settings > Show Pages > Options** in Calibre Interactive PEX if the **Options** page is not already displayed.
  - Selecting the **Probes** page (**Settings > Show Pages > Probes**) in Calibre Interactive PEX.
- **Choose which checks to run** — The [Check Selection Recipe](#) specifies the checks that are executed during a Calibre Interactive nmDRC run or ERC checks that are executed during a Calibre nmLVS run. You can use built-in check recipes or define your own custom check recipe. See “[Check Selection in Calibre Interactive](#)” on page 82 for details.
- **Load values from the rule file into the GUI** — The  button on the **Rules** page causes Calibre Interactive to populate the GUI with information from the rule file.

Two preference settings affect how rule file values are loaded:

  - **Automatically load rules file** — In the **Preferences** page (**Settings > Show Pages > Preferences**), under the Rules section. This setting instructs Calibre Interactive to automatically load a rule file, as described in “[Automatically Loading a Rule File](#)” on page 70.
  - **Read Inputs/Outputs fields when loading rules file** — In the **Preferences** page under the Rules section. When this setting is enabled, Calibre Interactive reads information for the Inputs and Outputs panes from the rule file. This setting is enabled by default.

## How Rule Files Relate to Control Files

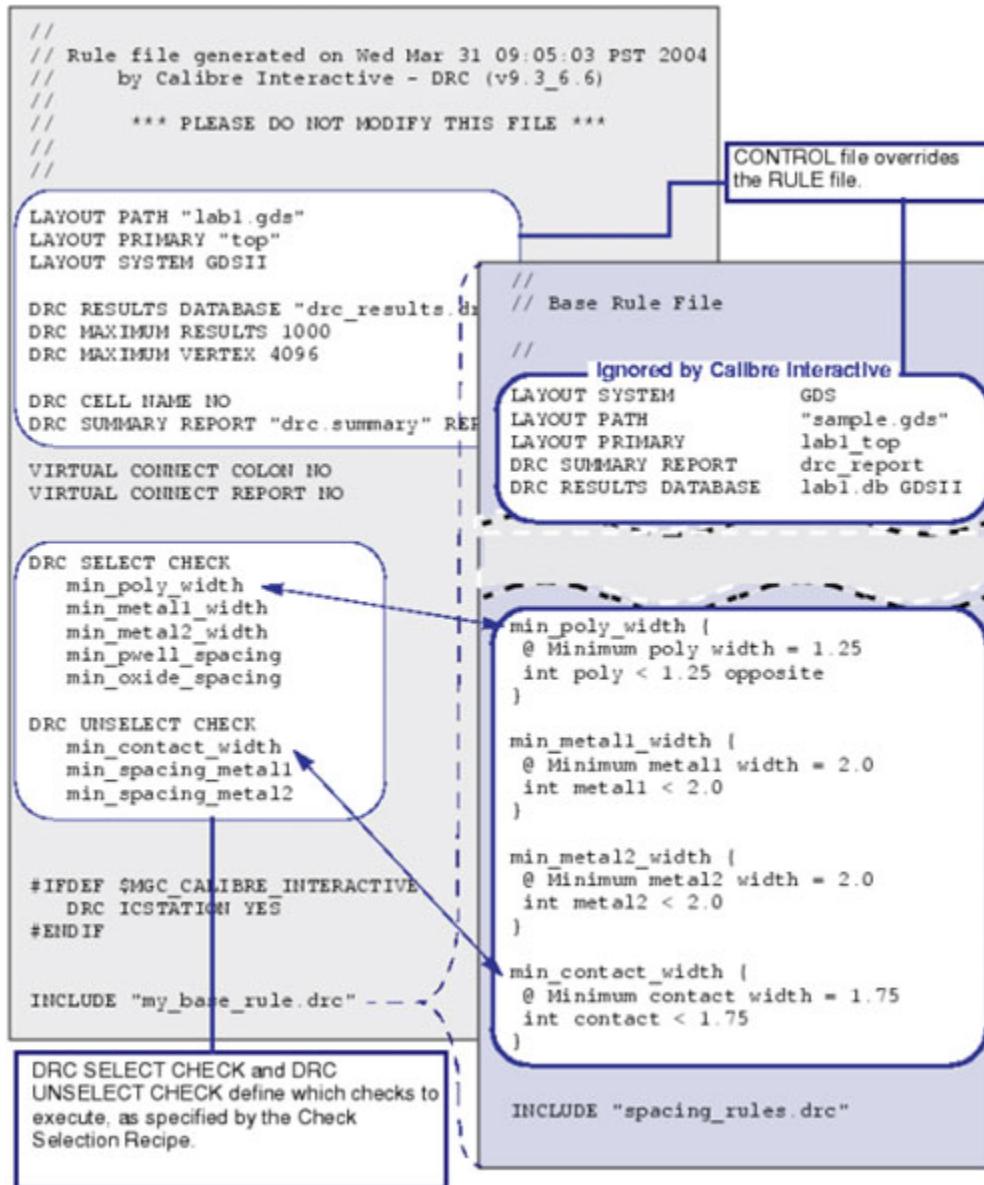
Calibre uses the control file to define the run. The primary rule file that you define in the Rules pane is included in the Calibre run as an Include file in the Calibre Interactive control file. When Calibre compiles a control file, the settings in the control file take precedence over any corresponding settings in the rule file. In other words, the GUI settings at the time the job is initiated control how the job is run.

Because the rule file is included as an Include file in the control file, current settings in the rule file are always used each time Calibre Interactive starts a run. However, if a GUI setting differs from a rule file setting, the GUI setting takes precedence.

Also see the preference setting “Automatically load rules file” in “[Rules Preferences](#)” on page 128.

[Figure 2-2](#) shows a control file and a rule file for a simple DRC run.

**Figure 2-2. Relating Control Files to Rule Files**



## Environment Variables in Calibre Interactive GUI Fields

You can use environment variables in Calibre Interactive fields and runsets with some restrictions.

The following restrictions apply:

- Environment variables are supported for the following fields and associated runset variables:
  - **Pathname fields** — Any field that gives a pathname or filename. For example, the layout input file and output database (drcLayoutPaths and drcResultsFile for a DRC runset).
  - **Top cell name for layout and source** — For example, drcLayoutPrimary in a DRC runset.

Environment variables are not allowed for any other fields.

- Environment variable names are delimited by the following characters:

/      "      #      .      \$

end of string (white space)

surrounding pair of braces: \${ENV\_VAR}

The tool recognizes the end of a variable name when it comes to one of the specified delimiter characters or the end of the string. For example, if \$DESIGN1/\$TOP.gds is entered into the GUI field for the layout file, Calibre Interactive interprets the “/” and “.” characters as delimiters and expects to find environment variables named “DESIGN1” and “TOP”.

Environment variables may also be surrounded by braces, as in \${PROC}. This provides a way to join an environment variable with text without using one of the recognized delimiter characters.

## Example: Environment Variable Usage in Calibre Interactive GUI

For example, suppose the following environment variables are defined:

```
setenv PROC 90nm
setenv TOP a_top
setenv DESIGN mem_b
```

The following examples illustrate correct usage of these environment variables:

\${PROC}_lvs.rul	resolves as 90nm_lvs.rul
\$DESIGN/\$TOP.gds	resolves as mem_b/a_top.gds

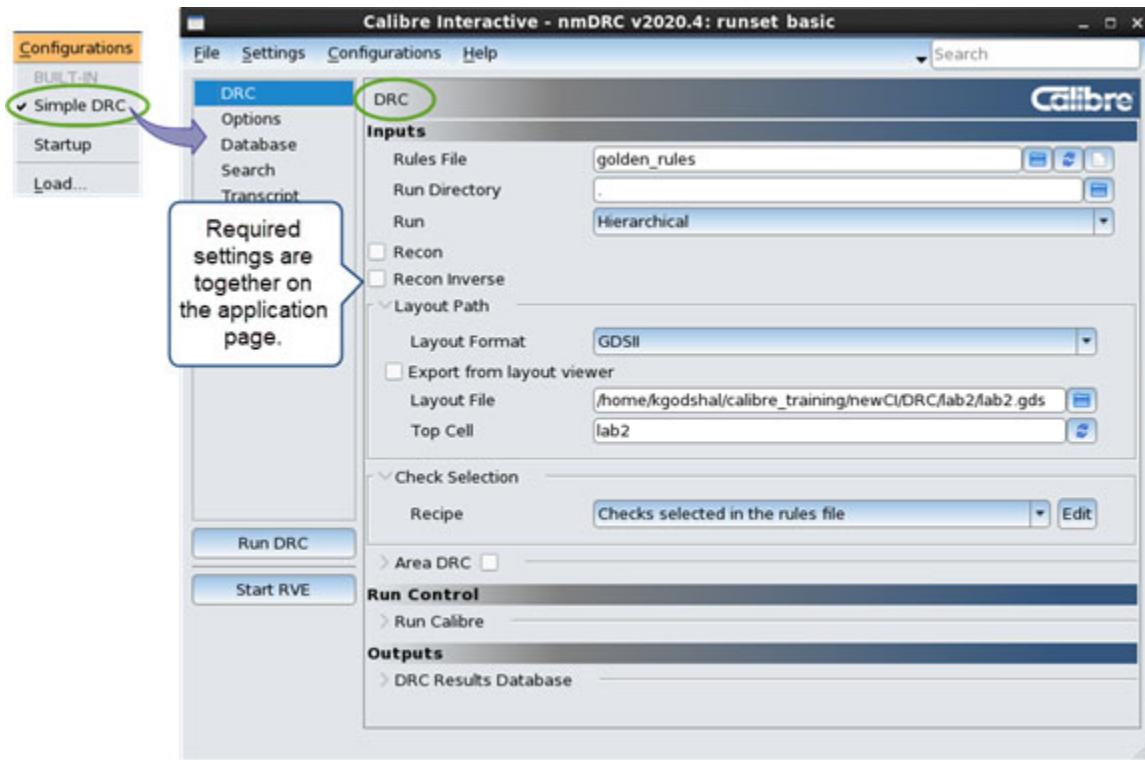
The following examples do not resolve correctly:

\$PROC_lvs.rul	not resolved, variable PROC_lvs is not defined
\$DESIGNA/\$TOP.gds	not resolved, variable DESIGNA is not defined

## Using Simplified Views

You can work with a simplified view of the Calibre Interactive GUI if you only need to change a basic set of options.

Simplified views are available for Calibre Interactive nmDRC and nmLVS. The basic settings from the **Rules**, **Inputs**, and **Outputs** pages are combined onto one application page.



### Prerequisites

- Save your runset.

The runset is closed before applying the simplified view, and any unsaved changes to the GUI will be lost.

- Calibre Interactive nmDRC or Calibre Interactive nmLVS is open.

### Procedure

1. Choose **Configurations > Simple DRC**, **Configurations > Simple LVS**, or **Configurations > Recon SI** from the main menu. The Recon SI option is available when Calibre Interactive nmLVS is open.

The application page is displayed by default, with the most commonly used settings.

2. Configure settings as desired.

3. (Optional) Choose **Settings > Show Pages** to display additional pages, such as the **Options**, **Database**, or **Preferences** pages, and make desired settings.
4. Save the runset with **File > Save Runset**.

The current view is saved with the runset, and applied the next time you load the runset.

---

**Tip**

 You can choose **Configurations > Startup** at any time to return to the default view. However, if you have made changes to the GUI settings, you will be prompted to save the runset.

---



# Chapter 3

## Running Calibre Interactive

---

Some options and setup procedures pertain to all of the Calibre Interactive applications.

<b>Invoking the Calibre Interactive GUI .....</b>	<b>58</b>
<b>License Timeout in Calibre Interactive .....</b>	<b>60</b>
<b>Calibre Interactive Command Line .....</b>	<b>61</b>
<b>Connecting to a Design Tool in Calibre Interactive .....</b>	<b>64</b>
<b>Setting Layout and Schematic Export in Calibre Interactive .....</b>	<b>67</b>
<b>Methods for Populating the Calibre Interactive GUI .....</b>	<b>69</b>
Populating the Calibre Interactive GUI .....	69
Automatically Loading a Rule File .....	70
Managing Rule File and Calibre Interactive GUI Precedence .....	71
<b>Providing OpenAccess Input with Calibre Interactive .....</b>	<b>73</b>
<b>Providing LEF/DEF Input with Calibre Interactive .....</b>	<b>75</b>
<b>Setting Up and Starting a Run .....</b>	<b>76</b>
<b>Starting a Basic Batch Run in Calibre Interactive .....</b>	<b>78</b>
<b>Specifying Layout and Schematic Export for a Calibre Interactive Batch Run .....</b>	<b>79</b>
<b>Calibre Interactive Operation with a TVF Rule File .....</b>	<b>80</b>
<b>Check Selection in Calibre Interactive .....</b>	<b>82</b>
Check Selection Recipe .....	82
Specifying a Check Selection Recipe .....	82
Creating a Custom Check Selection Recipe .....	83
Basic Editing of Custom Check Selection Recipes .....	84
Advanced Editing of Check Selection Recipes in Calibre Interactive .....	86
Running Only Checks Selected in the Rule File with a Custom Check Selection Recipe .....	88
Recipe Editor Page .....	90
<b>Viewing and Editing Environment Variables .....</b>	<b>98</b>
<b>Configuration Files for Startup Preferences .....</b>	<b>98</b>
<b>Templates for File Naming .....</b>	<b>100</b>
Example for File Naming Template .....	101
Disabling Templates So That Saved Runset Values Are Used .....	102
Export from Viewer Preferences .....	103

# Invoking the Calibre Interactive GUI

You invoke Calibre Interactive from a design tool or from the command line. Calibre DESIGNrev and Cadence Virtuoso are supported.

## Prerequisites

- “[Requirements for Calibre Interactive](#)” on page 24
- (Optional) Set environment variables to specify runsets and configuration files to load on invocation, as described in the following:
  - “[Environment Variables for Calibre Interactive Runsets](#)” on page 38
  - “[Loading a Configuration File in Calibre Interactive](#)” on page 230
- (Optional) Pre-populate the Recent Runsets and Recent Text Files lists as described in “[Populating the Load Runset Files Dialog Box and Recent Files Lists](#)” on page 39.

## Procedure

1. Set the required environment variable(s). For example, using csh:

```
setenv CALIBRE_ENABLE_NEW_CI_LVS 1
setenv CALIBRE_ENABLE_NEW_CI_DRC 1
setenv CALIBRE_ENABLE_NEW_CI_PERC 1
setenv CALIBRE_ENABLE_NEW_CI_XRC 1
```

2. Use one of the following methods to start the GUI:

Invoke from ...	Method
Calibre DESIGNrev	<b>Verification &gt; Run &lt;app&gt;</b> Where <app> is one of nmDRC, nmLVS, PERC, or PEX
Most other design tools	<b>Calibre &gt; Run &lt;app&gt;</b> Where <app> is one of nmDRC, nmLVS, PERC, or PEX
Command line	<code>calibre -gui { -drc   -lvs   -perc   -pex } [-runset runset]... [-config config_file]... [-batch]</code> See “ <a href="#">Calibre Interactive Command Line</a> ” on page 61 for more information.
Calibre Interactive palette	<code>calibre -gui</code> 

The -runset and -batch command line options are the same as for other Calibre Interactive applications. You can load a classic Calibre Interactive runset, as explained in “[Converting Classic Calibre Interactive Runsets](#)” on page 37.

For consistency with classic Calibre Interactive applications, the `-runset` switch is optional, but it is recommended in all cases. In particular, the `-runset` switch should be used when using “`-config config_file`” to specify a custom configuration file, to avoid parsing errors. It is also possible to specify multiple runsets as a space-separated list of runsets. However, best practice is to specify the `-runset` switch with each runset.

The “`-config config_file`” switch and argument option loads one or more custom configuration file(s); see “[GUI Customization for Calibre Interactive](#)” on page 229.

---

### Note

---

Configuration files are loaded first and in the order specified, then runsets are loaded in the order specified. If configuration files are specified in a runset, the configuration files are loaded before the runset settings. If any settings conflict, the setting in the most recently loaded file is used. For example, if a configuration file and a runset both specify a value for the same GUI setting, the value in the runset is used.

---

## Results

If you invoke the GUI from a layout tool, the open design is automatically used as the layout input. See “[Templates Page in Calibre Interactive](#)” on page 124.

The initial view of the GUI is similar to that shown in the following figure. A indicates an incomplete or incorrect field. A (not shown) indicates a warning, such as for a setting that conflicts with another option setting.



## Related Topics

[Calibre Interactive Command Line](#)

## License Timeout in Calibre Interactive

There are several options for setting up the timeout of your Calibre Interactive license.

The application looks at timeout settings in the following order of priority to determine when to release the license:

1. The server-side FLEXnet TIMEOUT option serves as top priority for when to release your Calibre Interactive application license. Refer to the *FLEXnet License Administration Guide* for information on setting the TIMEOUT option. See the section “[Mentor Standard Licensing \(MSL\)](#)” in the *Calibre Administrator’s Guide* for information on where to find the FLEXnet licensing documentation.
2. The environment variable, MGC\_CALGUI\_RELEASE\_LICENSE\_TIME follows the FLEXnet setting in priority for setting license timeout.

The default time unit is hours. You can specify the unit of time as hours, minutes, or seconds by appending h, m, or s to the time; a space between the time and the unit indicator is optional. The unit indicator may be a word—only the first character is used, therefore “sec”, “seconds”, and “s” all indicate seconds as the time unit. The ‘h’ to indicate hours is optional, as the default time unit is hours.

The following examples all set a license timeout of 18 minutes:

```
setenv MGC_CALGUI_RELEASE_LICENSE_TIME 0.3
setenv MGC_CALGUI_RELEASE_LICENSE_TIME 18m
setenv MGC_CALGUI_RELEASE_LICENSE_TIME "18 min"
```

The license timeout for the Calibre PERC Rule Generator GUI (**Setup > Generate PERC Rules**) is set only by MGC\_CALGUI\_RELEASE\_LICENSE\_TIME.

3. The environment variable, MGC\_RVE\_RELEASE\_LICENSE\_TIME follows in priority for setting license timeout. The value is set as described for MGC\_CALGUI\_RELEASE\_LICENSE\_TIME.
4. The “Release license after: X inactive hours” option in the **Disable** tab of the Setup Preferences dialog box has the lowest priority. This choice stores your preference in the sunset file and governs when your license is released if none of the other license timeout settings are specified.

If the time-out period is set with a higher precedence setting, that value is displayed in the GUI and can not be changed.

If a license timeout is not set with one of the above methods, then there is no license timeout. License timeout values are set in hours; the timeout value must be greater than five minutes. The default timeout period is 500 hours.

# Calibre Interactive Command Line

The Calibre Interactive command line has options to run in interactive mode or batch mode. You can also specify one or more runsets to open.

## Usage

### Interactive operation:

**calibre -gui**

**calibre -gui { -drc | -lvs | -perc | -pex } [-runset *runset*]... [-config *config\_file*]...  
[-custom *custom\_file*]**

### Batch operation:

**calibre -gui { -drc | -lvs | -perc | -pex } { -runset *runset* | -runset\_options\_display *runset*}...  
[-config *config\_file*]... [-customgui *custom\_file*]  
[-output\_runset *new\_runset*]  
-batch [-platform offscreen]**

### Calibre PERC Rule File Generator GUI:

**calibre -gui -perc\_rule\_gen [-prgfile *rulegen\_setup.prg*]**

### Configuration Editor:

**calibre -gui { -drc | -lvs | -perc | -pex } -config\_editor  
[-config *config.cfg* ...] [-custom *custom.tcl* ...]**

## Arguments

- **-gui**

A required argument that specifies the Calibre Interactive application. When specified with -drc, -lvs, -perc, or -pex, the respective user interface window opens. When specified alone, the Calibre Interactive Palette opens, as shown in “[Invoking the Calibre Interactive GUI](#)” on page 58. You can open the various application-specific GUIs from the Calibre Interactive Palette.

- **-drc | -lvs | -perc | -pex**

An argument used with -gui to specify the Calibre Interactive application. The interfaces are Calibre nmDRC, Calibre nmLVS, Calibre PERC, and PEX, respectively.

- **-runset *runset***

An argument set specifying the pathname of a runset. When specified, the runset is automatically loaded when the GUI opens. A runset is required for batch operation (the -batch switch); it is optional for interactive operation.

You can load multiple runsets by specifying the switch and argument multiple times. The runsets are loaded in the order specified and the settings loaded last take precedence.

**Note**

 For consistency with the classic Calibre Interactive applications, the -runset switch is optional, but it is recommended in all cases. In particular, the -runset switch should be used when using “-config *config\_file*” to specify a custom configuration file, to avoid parsing errors.

It is also possible to specify multiple runsets as a space-separated list of runsets. However, best practice is to specify the -runset switch with each runset.

---

- **-runset\_options\_display *runset***

An argument set that specifies the pathname of a runset and also instructs the tool to display the runset options in the transcript. This option is only valid for batch operation (the -batch switch).

**Note**

 When -runset\_options\_display is used, all the runsets specified on the command line must be of the same type—all classic runsets or current-version runsets.

---

You can load multiple runsets by specifying the switch and argument multiple times. The runsets are loaded in the order specified and the settings loaded last take precedence.

You can combine the -runset\_options\_display and -runset switches. Runsets are loaded in the order given in the command line; options in runsets occurring later in the command line override options in runsets occurring earlier on the command line. Only options in runsets specified by -runset\_options\_display are echoed to the transcript. If an option in a -runset\_options\_display runset is overridden by a later runset specified with the -runset option, then the option is not echoed to the transcript.

- **-config *config\_file***

An optional argument set that specifies a configuration file to load. See “[GUI Customization for Calibre Interactive](#)” on page 229.

- **-custom *custom\_file***

An optional argument set specifying the pathname of a classic Calibre Interactive Tcl customization file. The controls in the Tcl customization file are converted to equivalent configuration controls and placed on a page named Custom.

This switch overrides the MGC\_CALIBRE\_CUSTOMIZATION\_FILE environment variable and customization files set in the runset, if one is loaded.

See “[Loading a Classic Tcl Customization File in Calibre Interactive](#)” on page 232 for more information.

- **-customgui *custom\_file***

An optional argument set specifying the pathname of a classic Calibre Interactive Tcl customization file. This option is only available in batch mode.

The controls in the Tcl customization file are converted to equivalent configuration controls and placed on a page named Custom. When a batch run is started, a GUI with only the Custom page and the Run button is displayed. You can make changes on the Custom page. Click the Run button to proceed with the batch run using the settings on the Custom page.

This switch overrides the MGC\_CALIBRE\_CUSTOMIZATION\_FILE environment variable.

- **-batch**

A required argument in batch invocation. Calibre Interactive runs in batch mode, without opening the GUI. See “[Starting a Basic Batch Run in Calibre Interactive](#)” on page 78 for more information.

- **-output\_runset *new\_runset***

An optional argument set specifying the name of a new runset file that is generated using runset. This option is useful for converting classic runsets, or for combining multiple runsets into one. When multiple runsets are specified, the last runset’s settings takes precedence.

- **-platform offscreen**

An optional argument in batch invocation. Enable this argument when the DISPLAY environment variable is not defined in the shell.

- **calibre -gui -perc\_rule\_gen [-prgfile *rulegen\_setup.prg*]**

An argument set used to invoke the standalone Calibre PERC Rule Generator GUI.

- **-gui -perc\_rule\_gen** — Required argument that specifies to open the Calibre PERC Rule Generator GUI.
- **-prgfile *rulegen\_setup.prg*** — An optional switch and argument specifying the path to a binary Calibre PERC Rule Generator setup file (.prg file). The setup file is used to populate settings in the GUI.

See “[Running the Rule File Generator GUI](#)” in the *Calibre PERC User’s Manual* for more information.

You can also invoke the Calibre PERC Rule Generator GUI from Calibre Interactive PERC; see “[Generating a Calibre PERC Rule File with High Level Checks](#)” on page 224.

- **-config\_editor [-config *config.cfg* ...] [-custom *custom.tcl* ...]**

An argument set used to invoke the Calibre Interactive Configuration editor. See “[Customizing the GUI With the Configuration Editor](#)” on page 233 for details.

**-config\_editor** — Required argument that specifies to open the configuration editor.

**-config *config.cfg*** — Optional argument and one or more existing configuration files to load in the configuration editor.

**-custom *custom.tcl*** — Optional argument and one or more classic Calibre Interactive Tcl customization files to load in the configuration editor.

## Examples

Invoke with a runset:

```
calibre -gui -drc -runset runset_drc
```

Invoke with two runsets and a custom configuration file:

```
calibre -gui -drc -runset rs_drc -runset rs_drc_opt -config drc.cfg
```

Invoke in batch mode:

```
calibre -gui -lvs -runset runset_lvs -config lvs.cfg -batch
```

# Connecting to a Design Tool in Calibre Interactive

The connection to a design tool is made automatically when you invoke Calibre Interactive from the design tool. You can also make the connection to the design tool manually after the GUI is invoked from the command line.

Connection Method	Directions
Automatic connection	Invoke GUI from: <ul style="list-style-type: none"><li>• Calibre DESIGNrev: <b>Verification &gt; Run &lt;app&gt;</b></li><li>• Cadence, Synopsys, and most other design tools: <b>Calibre &gt; Run &lt;app&gt;</b></li></ul>
Manual connection	See the following procedure.

See the “Results” section after the procedure for notes on verifying the connection.

---

### Note

 You can set the socket port with the environment variables MGC\_CALIBRE\_LAYOUT\_SERVER and MGC\_CALIBRE\_SCHEMATIC\_SERVER. See “[Setting the Socket Port in Calibre Interactive](#)” on page 451 for more information.

---

## Prerequisites

- “[Requirements for Calibre Interactive](#)” on page 24

## Procedure

1. Open your design tool to start the socket connection and note the socket number:
  - Calibre DESIGNrev

Select **Verification > RVE/CI Setup**. In the Server area of the dialog box, make a note of the Socket Number and change it if needed. If the Status icon is not green, click the **Start Server** button.

- Cadence Virtuoso

Select **Calibre > Setup > Socket**. Make a note of the Socket Number and change it if needed.

- Other tools

See “[Interfacing Calibre Interactive to Layout and Schematic Viewers](#)” on page 449 or the design tool documentation.

2. Invoke the Calibre Interactive GUI; see “[Invoking the Calibre Interactive GUI](#)” on page 58.
3. Click **Run Control** on the left panel.
4. Expand the “Design Tool Settings” area.
5. In the “Layout Design Tool” area, enter the socket number from Step 1 and click the **Connect** button.
6. Make settings in the “Schematic Design Tool” area as follows:

Schematic Tool Connection	“Use the same connection as the layout tool” setting	Further Action
No connection	Unchecked	None
Same as layout tool (such as for Cadence Virtuoso)	Checked	None
Different than layout tool	Unchecked	Enter the socket number for the schematic tool and click <b>Connect</b> .

7. (Optional) Specify a Highlight Data File. This file is used when sending highlight data to the design tool. The default value is *query\_results*.
8. Click **Inputs** on the left panel and specify design export as follows:
  - a. For layout export, expand “Layout Path” and enable “Export from layout viewer.”
  - b. For schematic export, expand “Source Path” and enable “Export from schematic viewer.”

## Results

If the connection is successful, the Status is reported as “Connected” on the Run Control page in the “Design Tool Settings” area. In addition, the “Tool Information” display appears below the Status line with the name of the connected design tool.

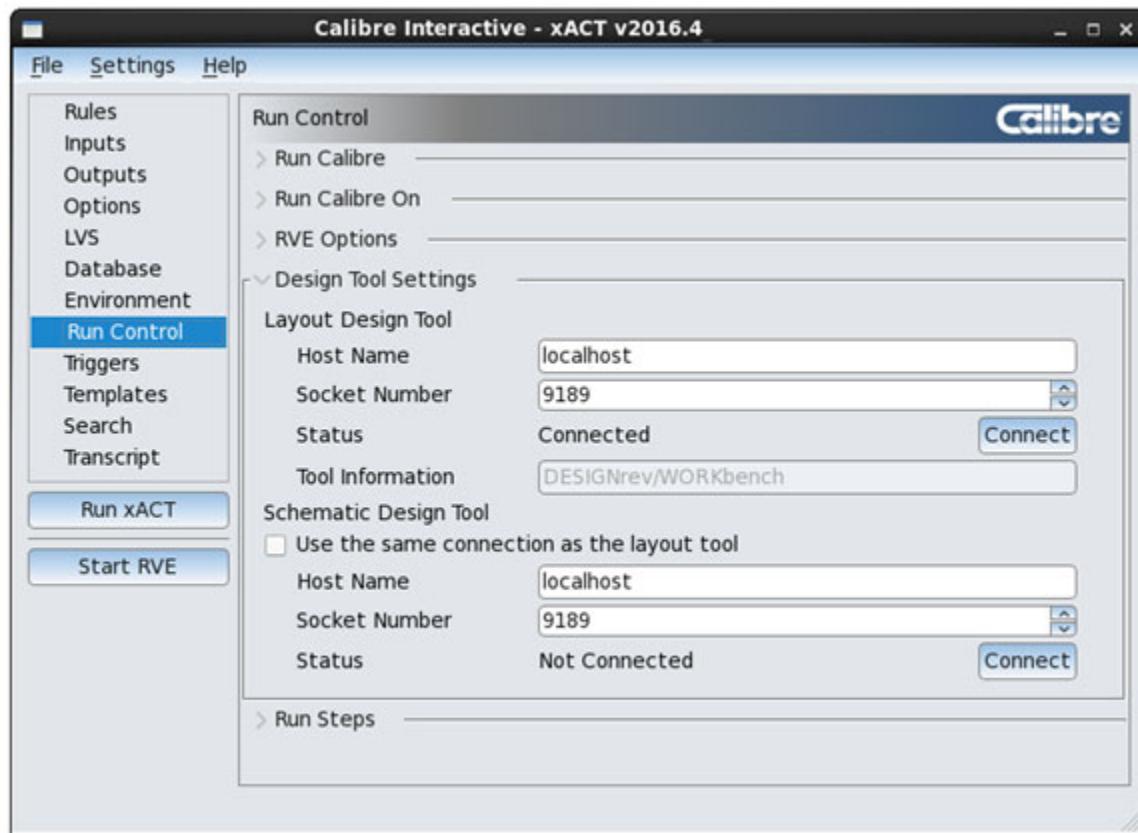
### Note

 If “Use the same connection as the layout tool” is enabled and a connection attempt is made to a layout design tool that does not include a schematic tool (such as Calibre DESIGNrev), a message is displayed that the schematic connection is not supported and the setting “Use the same connection as the layout tool” is disabled.

---

The following image shows a layout design tool connection to Calibre DESIGNrev/WORKbench and no connection to a schematic design tool.

**Figure 3-1. Design Tool Setting on the Run Control Page**



# Setting Layout and Schematic Export in Calibre Interactive

You can use the open design in a layout or schematic viewer as input for your Calibre run; this is referred to as layout and schematic export. Layout and schematic export can also be used in a batch mode Calibre Interactive run with an additional setting.

## Prerequisites

- The design is open in a supported layout or schematic viewer.
- Calibre Interactive is open to the desired application.

## Procedure

1. If exporting a layout from a connected layout viewer:
  - a. Enable “Export from layout viewer” on the **Inputs** page, Layout Path section of Calibre Interactive.

Layout data comes from the active cell in the layout editor.

  - b. Verify the filename given to the extracted layout database in the Layout File field.

When “Export from layout viewer” is selected, the extracted layout database is saved to the filename in the File field. Calibre Interactive generates the filename based on the file naming template defined on the Inputs section of the **Templates** page (**Settings > Show Pages > Templates**). By default, the extracted layout filename is based upon the layout cell name. You can enter a different filename if you want. See “[Templates for File Naming](#)” on page 100.

If the filename in the File field is red, the file does not exist and Calibre Interactive will create it.

If the filename in the File field is green, the file exists and Calibre Interactive will overwrite the file. Enter another filename if you want to keep the existing file.

---

### Caution

 When “Export from layout viewer” is enabled, a layout file is extracted and saved. Do not use the extracted layout file named in the File field for tapeout or any other use.

---

2. If exporting a source schematic from a connected schematic viewer (for Calibre nmLVS, PEX, and Calibre PERC):
  - a. Enable “Export from source viewer” in the Source Path section of the **Inputs** page.

Source netlist data comes from the active cell in the schematic editor.

  - b. Verify the filename given to the extracted source netlist in the File field.

When “Export from source viewer” is selected, the extracted netlist is saved to the filename in the File field. Calibre Interactive generates the filename based on the file naming template defined on the Inputs section of the **Templates** page

(**Settings > Show Pages > Templates**). By default, the extracted netlist filename is based upon the schematic cell name. See “[Templates for File Naming](#)” on page 100.

If the filename has a , the file does not exist and Calibre Interactive will create it.

Otherwise, the file exists and Calibre Interactive will overwrite the file. Enter another filename if you want to keep the existing file.

- c. Verify the name of the top cell in the “Top Cell” field.
- d. (Optional) If the runset will be used for a batch mode Calibre Interactive run, do the following:

In the **Preferences** page (**Settings > Show Pages > Preferences**), select the Misc section, and disable “Ignore layout and source export in batch mode.”

3. Make remaining Calibre Interactive settings for the run.

## Results

When you click the **Run** button, Calibre Interactive instructs the design tool to export the design.

If the file for the extracted database already exists, Calibre Interactive displays the Overwrite File dialog box with the following options:

**Table 3-1. Overwrite File Dialog Box for Layout and Schematic Export**

Button	Action
Overwrite	Overwrite the existing file.
Don't Overwrite	Keep the existing file and continue with the run. The Calibre run uses the existing extracted database. Use this selection if your design has not changed since the last export operation.
Cancel	Cancel the run.

# Methods for Populating the Calibre Interactive GUI

---

There are several methods for populating the fields in the Calibre Interactive GUI.

<b>Populating the Calibre Interactive GUI .....</b>	<b>69</b>
<b>Automatically Loading a Rule File.....</b>	<b>70</b>
<b>Managing Rule File and Calibre Interactive GUI Precedence .....</b>	<b>71</b>

## Populating the Calibre Interactive GUI

The settings in the Calibre Interactive GUI can come from several sources, including direct user input, a Calibre Interactive runset, and template settings.

Calibre Interactive settings come from the following sources:

- **Default values** — Default values are programmed into Calibre Interactive. You see the default values when you open Calibre Interactive without a runset and not connected to a design tool. The default settings vary between applications.
- **Runset** — A Calibre Interactive runset is a text file that stores the Calibre Interactive GUI settings. You can load a runset in a later Calibre Interactive session to restore the GUI settings from a previous session. See “[About Calibre Interactive Runsets](#)” on page 34 for methods to save and load a runset.
- **System-generated parameters based on template settings** — When Calibre Interactive is invoked from a layout or schematic viewer, the default behavior is to overwrite certain GUI settings with values derived from the open database. For example, the output results database in Calibre Interactive nmDRC is automatically named `<top_cell>.drc.results`, where `<top_cell>` is the name of the primary layout cell in the open design. See “[Templates for File Naming](#)” on page 100.
- **Rule file** — Settings from the rule file populate the GUI when you click the **Load** button on the Rules pane. You can also instruct Calibre Interactive to load the rule file when a runset is loaded, as described in “[Automatically Loading a Rule File](#).”
- **GUI Customization** — When customizing the GUI, you can set values in the Calibre Interactive GUI using runset control statements in the configuration file, as described in the “[GUI Customization for Calibre Interactive](#)” chapter. Doing this also adds statements to the Calibre Interactive control file. See “[Configuration File Format for Calibre Interactive GUI Customization](#)” on page 240 for an example.
- **User input** — Direct user input overrides any preset value or setting in the Calibre Interactive GUI.

The settings present in the Calibre Interactive GUI at the time of the run always take precedence over rule file settings if there is a conflict.

## Automatically Loading a Rule File

You can instruct Calibre Interactive to automatically load a rule file. The rule file is loaded when a runset is loaded and reloaded at runtime if the rule has been modified. You can choose whether or not to have a confirmation prompt displayed when the rule is about to be loaded.

### Procedure

1. Choose **Settings > Show Pages > Preferences**.
2. Choose the **Rules** section.
3. Enable “Automatically load rules file.”

The rule file is automatically loaded when you load a runset with a rule file defined, or when you start a run if the rule file has changed since it was last loaded.

4. If desired, disable the “Show prompt” setting, which is on by default.

**Enabled** — Show a confirmation dialog before automatically loading the rule file.

**Disabled** — Do not show a confirmation dialog before automatically loading the rule file. If “Show prompt” is disabled, the setting for “Automatically load rules file” applies during Calibre Interactive batch runs.

5. (Optional) Select a setting for “Read Inputs/Outputs fields when loading rules file.”

This setting is enabled by default, and causes Calibre Interactive to read settings for the Inputs and Outputs panes from the rule file.

### Results

When “Automatically load rules file” is enabled, Calibre Interactive loads the rule file at these times:

- When you load a runset with a rule file defined.
- When you start a Calibre run if the rule file has changed since it was last loaded.

If “Show prompt” is enabled (the default), a confirmation dialog is displayed before loading the rule file; otherwise, the rule file is loaded without notification.

Loading or not loading the rule file has this effect:

- Rule file loaded — GUI fields are updated from the rule file. This may overwrite runset and user-defined GUI settings.
- Rule file not loaded — GUI fields are not updated from the rule file. The current rule file is used in the Calibre run, but GUI settings take precedence over rule file settings where there is a conflict.

**Note**

 When you exit Calibre Interactive, you will be prompted to save the runset. Be aware that rule file settings may have overwritten the original runset settings.

---

## Managing Rule File and Calibre Interactive GUI Precedence

The settings present in the Calibre Interactive GUI at the time of the run always take precedence over rule file settings if there is a conflict. This behavior allows you to customize your Calibre Interactive run without editing the rule file. However, in some cases you may want to insure that rule file settings are used for the run, or disable changes in portions of the Calibre Interactive GUI. Several features are available to manage requirements like this.

You can use the following features in Calibre Interactive to help you manage the precedence of the rule file and the GUI settings:

[Load the Rule File](#)

[Use Customization File](#)

[Use Runsets](#)

[Run Calibre Interactive in Batch Mode](#)

[Disable GUI Settings](#)

### Load the Rule File

You can do the following to help insure that the rule file settings are used for the run:

- In the **Preferences** page (**Settings > Show Pages > Preferences**), click the Rules section, and enable “Automatically load rules file.” This setting causes Calibre Interactive to load the rule file when it loads a runset. This setting applies in Calibre Interactive batch runs if the “Show prompt” preference setting is disabled.

If you operate Calibre Interactive connected to a design tool and use [Templates for File Naming](#), certain filenames and design-related parameters are determined from the open design database. Templates are applied after the rule file is automatically loaded as part of loading a runset, so template settings override the rule file settings if “Apply Templates” on the **Templates** page (**Settings > Show Pages > Templates**) is enabled.

- Click **Load** on the Rules pane immediately before a run. This loads the rule file settings and overwrites any GUI settings that have been made since the last load of the rule file.

If you perform this step and use [Templates for File Naming](#), you may want to disable “Read Inputs/Outputs fields when loading rules file” in the Rules section on the **Preferences** page to avoid overwriting template settings with rule file settings.

## Use Runsets

You can use Calibre Interactive runsets to specify the settings in the GUI, as discussed in “[About Calibre Interactive Runsets](#)” on page 34. You can define a runset for each allowed configuration of Calibre Interactive.

If you operate Calibre Interactive connected to a design tool, you can specify [Templates for File Naming](#) so that certain filenames and design-related parameters are determined from the open design database rather than the runset or rule file.

## Disable GUI Settings

Calibre Interactive includes several settings which disable portions of the GUI interface. These features are useful if you want to limit the changes made in the Calibre Interactive GUI. See “[Disable Preferences](#)” on page 130 for more information.

In addition, Calibre Interactive nmLVS, Calibre Interactive PERC, and Calibre Interactive PEX have specific options for specifying rule file settings related to device reduction and filtering. The following options are located in the **Options** page (**Settings > Show Pages > Options**):

- **Gate Reduction** — When these options are enabled, the rule file settings for LVS Reduce Split Gates, LVS Reduce Parallel MOS, and LVS Short Equivalent Nodes are used; GUI settings are ignored.
- **LVS Filter Unused Option** — When this checkbox is enabled, the rule file settings for LVS Filter Unused Option are used; GUI settings are ignored.

## Use Customization File

You can also use a configuration file to control access to the Calibre Interactive GUI. This file can be edited to only allow changes to certain Calibre Interactive GUI settings. See “[GUI Customization for Calibre Interactive](#)” on page 229 for more information.

## Run Calibre Interactive in Batch Mode

When you run Calibre Interactive in batch mode, the GUI does not open; the settings for the run are controlled by the runset and an optional customization file. This mode of operation makes it easy to control the run parameters and still use the important features of Calibre Interactive, such as runsets, customization files, and export from layout.

See these topics for additional information on running Calibre Interactive in batch mode:

- “[Starting a Basic Batch Run in Calibre Interactive](#)” on page 78
- “[Specifying Layout and Schematic Export for a Calibre Interactive Batch Run](#)” on page 79

# Providing OpenAccess Input with Calibre Interactive

You provide OpenAccess database input on the Inputs page in Calibre Interactive. You specify database conversion options on the OA/LEFDEF page. The page name is OPENACCESS for Calibre Interactive PEX.

## Caution

-  Runs started from Calibre Interactive do not use the statement [SVRF FDI Options](#) to set third-party database read options in the rule file. Make sure database conversion options are set correctly in the GUI or with MGC\_CALIBRE\_DB\_READ\_OPTIONS or the database may not be converted correctly.
- 

## Prerequisites

- For Cadence Virtuoso users, make sure CDS\_INST\_ROOT or CDS\_INST\_DIR is set. CDS\_INST\_ROOT takes precedence if both CDS\_INST\_ROOT and CDS\_INST\_DIR are defined.
- Make sure LD\_LIBRARY\_PATH includes the libstdc++.so.6.0.8 compiler libraries. This is true by default for Linux machines with RHEL 4u6 or later.

For example with Cadence Virtuoso 6.15.500:

```
setenv LD_LIBRARY_PATH ${CDS_INST_DIR}/share/oa/lib/  
linux_rhel40_gcc44x_64/opt:  
${CDS_INST_DIR}/tools/cdsskillPcell/lib/64bit:  
${CDS_INST_DIR}/tools/lib/64bit:${LD_LIBRARY_PATH}
```

The actual path depends on the tool version.

- Make sure environment variables for database read are set correctly. See “[FDI Environment Variables](#)” in the *Calibre Layout Comparison and Translation Guide*.

In particular, if your design tool requires OpenAccess version 22.50, set the environment variable MGC\_FDI\_OA\_VERSION to 22.50.

You can use the environment variable MGC\_CALIBRE\_DB\_READ\_OPTIONS to set database read options. The settings are displayed on the OA/LEFDEF page in Calibre Interactive and the GUI controls are made inactive.

## Procedure

1. Click Inputs on the left panel of Calibre Interactive.
2. In the Layout Path area, set Format to OPENACCESS.
3. Enter the path to the library definitions file in the “Library Path” field. This is typically *lib.defs* or *cds.lib* (if OA\_HOME is set).
4. Enter the name of the design library in the “Library Name” field.

**Tip**

 You can click the **Browse** button to open a library browser. The library browser enables you to select the library, cell, and view, then automatically fills in the correct GUI fields.

The “Library Path” field must be correctly defined for the library browser to work correctly.

---

5. Specify the Top Cell and View Name if these have not been filled in by the library browser.
6. Click OA/LEFDEF on the left panel to set database conversion options. The page name is OPENACCESS for Calibre Interactive PEX.
7. For Cadence Virtuoso users, do the following to automatically set certain Cadence environment variables:
  - a. Click OpenAccess Environment.
  - b. Enable “Set OA\_HOME when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”
  - c. Enable “Set CALIBRE\_READDB\_LD\_LIBRARY\_PATH when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”
  - d. Enable “Set PATH when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”
  - e. If you are using Express Pcells, enable “Set CDS\_EXP\_PCELL\_DIR when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”
  - f. If you are using Express Pcells, enable “Set CDS\_EXP\_ALL\_PARAMS when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”
  - g. If you are using Express Pcells, enable “Set CDS\_ENABLE\_EXP\_PCELL when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”
8. Set other database conversion options as needed on the OpenAccess/LEFDEF page. For complete details, see “[OA/LEFDEF and OPENACCESS Pages in Calibre Interactive](#)” on page 112.

These options control what is read from the OpenAccess database and sent to Calibre as input.

**Tip**

 You can define a file naming template for the library and view. Click Templates on the left panel, expand the Database area, and specify templates in the “Layout Library” and “Layout View” fields. See “[Templates Page in Calibre Interactive](#)” on page 124 for general information.

---

# Providing LEF/DEF Input with Calibre Interactive

You can read LEF/DEF input from design files using Calibre Interactive.

## **Caution**

 Runs started from Calibre Interactive do not use the statement [SVRF FDI Options](#) to set third-party database read options in the rule file. Make sure database conversion options are set correctly in the GUI or with MGC\_CALIBRE\_DB\_READ\_OPTIONS or the database may not be converted correctly.

---

## Prerequisites

- You are using Calibre Interactive nmDRC, nmLVS, or PERC.
- Make sure environment variables for database read are set correctly. See “[FDI Environment Variables](#)” in the *Calibre Layout Comparison and Translation Guide*.

You can use the environment variable MGC\_CALIBRE\_DB\_READ\_OPTIONS to set database read options. The settings are displayed on the OA/LEFDEF page in Calibre Interactive and the GUI controls are made inactive.

## Procedure

1. Click Inputs on the left panel of Calibre Interactive.
2. In the Layout Path area, set Format to LEFDEF.
3. Provide DEF Input as follows:
  - **Export from layout viewer** — When checked, the layout in a connected design tool is exported in DEF format.  
The design streamed out by the design tool is written to the first file listed in the DEF Files field; the file is overwritten if it already exists. If additional files are listed in the DEF Files field or using the DEF Directories field, they are considered additional input files.
  - **DEF Files** — One or more DEF files. At least one file must be specified if “Export from layout viewer” is checked.
  - **DEF Directories** — DEF directories.  
At least one of the entries must be filled in for DEF input.  
Multiple files or directories are entered on separate lines.
4. Provide LEF Input as follows:
  - **LEF Technology Files** — One or more LEF technology files.

- **LEF Files** — One or more LEF files.
- **LEF Directories** — LEF directories.

At least one of the entries must be filled in for LEF input. The LEF Technology Files entry is optional as long as the technology information is one of the LEF file or directory entries. If the LEF Technology File entry is not used, then the LEF technology file should be first in the list of LEF files.

5. Fill in the Top Cell field.
6. If you did not set MGC\_CALIBRE\_DB\_READ\_OPTIONS, set database conversion options as needed on the OA/LEFDEF page. For complete details, see “[OA/LEFDEF and OPENACCESS Pages in Calibre Interactive](#)” on page 112.

## Setting Up and Starting a Run

To set up the GUI for a run, click each of the page names on the left panel of the GUI and fill in the required fields.

Errors and warnings in the GUI settings are indicated as follows; hover over the icon to view a message about the problem.



A missing or incorrect setting. A red exclamation point (!) is also displayed by the page name if any settings have an error.



A warning, such as for a setting that conflicts with another option setting. The run can proceed with warnings. The warning icon is not displayed next to the page name.

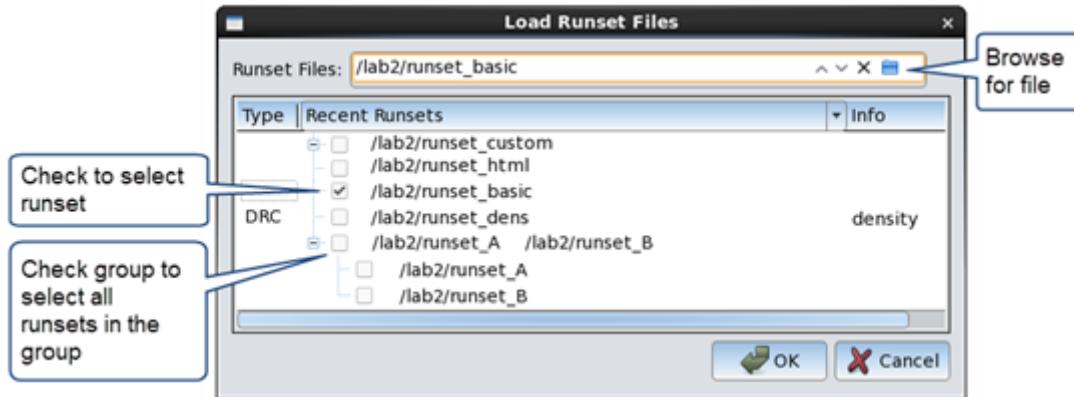
### Prerequisites

- The Calibre Interactive GUI is open. See “[Invoking the Calibre Interactive GUI](#)” on page 58.

### Procedure

1. (Optional) Select **File > Load Runset** to load a saved runset.

Click a checkbox to select the runset or a group of runsets. Check multiple individual runsets to load them as a group. When multiple runsets are loaded, the runsets are loaded in order, with options loaded last having precedence if there is a conflict.



You can load a classic Calibre Interactive runset; see “[Converting Classic Calibre Interactive Runsets](#)” on page 37.

2. Click **Rules** on the left panel and specify the rule file and run directory.

If desired, click the  button to load the settings from the rule file into the GUI.

3. Click **Inputs** on the left panel and specify the required input. Source (schematic) input is available for Calibre Interactive nmLVS and PERC runs. The information is filled in automatically if you invoked the GUI from a design tool.

Note that other settings in the GUI can affect the controls displayed on the Inputs page.

To specify additional layout or schematic input files, click **Database** on the left panel and expand the Library area. (Choose **Settings > Show Pages** if the page is not present.)

4. (Optional) If you invoked the GUI from a design tool, check that “Export from layout viewer” and “Export from source viewer” are set as desired in the “Layout Path” and “Source Path” areas. When enabled, the active cell in the design tool is used as the layout or source input.

#### **Tip**

 You can view the design tool connection information on the Run Control page in the “Design Tool Settings” area. If you invoked the GUI from a design tool, the design tool connection is made automatically. If you need to make a manual connection to a design tool, see “[Connecting to a Design Tool in Calibre Interactive](#)” on page 64.

5. Click each of the other page names on the left panel and set options as needed.

---

**Tip**

 To quickly find a specific option, enter a search string in the search field at the top right of the GUI. Matches are placed on the Search page. Changes made in the Search page are reflected when you view the control in its original page.

---

6. If any GUI pages have a  by their name, provide the required input.
7. (Recommended) Choose **File > Save Runset** to save the GUI settings.
8. Click the **Run <app>** button to start the run.

During the run, you can click the **Stop <app>** button to open a dialog box with options to stop or suspend the run.

---

**Tip**

 You can generate and view a control file without running Calibre by holding down the Ctrl key and clicking **Run**. In addition, the calibre command line for the run is written the Transcript pane.

---

## Results

The run transcript is displayed in the Transcript page, with any errors and warnings listed in the bottom portion of the page.

To save the Transcript report, right-click and select “Save Errors and Warnings As”.

## Related Topics

[Calibre Interactive nmDRC](#)

[Calibre Interactive nmLVS](#)

[Calibre Interactive PERC](#)

[Calibre Interactive PEX](#)

[Connecting to a Design Tool in Calibre Interactive](#)

# Starting a Basic Batch Run in Calibre Interactive

You can run Calibre Interactive in batch mode without opening the GUI. All Calibre Interactive options are set in the runset.

## Procedure

1. Set up Calibre Interactive as described in “[Setting Up and Starting a Run](#)” on page 76, but do not start a run.

2. If you want Calibre RVE to start after the batch run and open the results database, navigate to the RVE Options section on the **Run Control** page and enable “Start RVE after a batch run.”
3. Save a runset with **File > Save Runset**.
4. Exit Calibre Interactive.
5. Start the batch run with the following command:

```
calibre -gui -app -runset_options_display my_runset -batch
>& batch.log
```

where *app* is one of drc, lvs, perc, or pex, and *my\_runset* is the runset saved in Step 3.

The option *-runset\_options\_display* causes the runset options to be echoed to stdout. This is useful so you can determine what Calibre Interactive settings might override rule file settings. If you do not want the runset options echoed, use the option *-runset* instead.

## Specifying Layout and Schematic Export for a Calibre Interactive Batch Run

You can export the design from an open design tool for use as input for the Calibre Interactive batch run.

### Procedure

1. Open the design in your design tool to the desired cell and view.
2. Make sure the socket connection is started and uses the default socket number (9189 for layout export, 9199 for schematic export).

To do this in Calibre DESIGNrev, choose **Verification > RVE/CI Setup**, and click **Start Server** to start the socket connection. See “[Interfacing Calibre Interactive to Layout and Schematic Viewers](#)” on page 449 for instructions for other design tools.

If your design tool is not using the default socket numbers, you need to specify the socket number. This can be done with an environment variable; see “[Setting the Calibre Interactive Socket Port for a Layout Viewer](#)” on page 451 and “[Setting the Calibre Interactive Socket Port for a Schematic Viewer](#)” on page 452.

3. Open Calibre Interactive and make the following settings:
  - a. Enable “Export from layout viewer” in the Layout Path section of the **Inputs** page.
  - b. (Optional) Enable “Export from source viewer” in the Source Path section of the **Inputs** page if you are running Calibre Interactive nmLVS, PERC, or PEX.
  - c. Disable “Ignore layout and source export in batch mode” in the Misc section of the **Preferences** page (**Settings > Show Pages > Preferences**).

- d. Check that all other Calibre Interactive settings for your run are correct.
4. Save the runset with **File > Save Runset**.
5. Exit Calibre Interactive.
6. Start the Calibre Interactive batch run. The following command is for a Calibre nmDRC run:

```
calibre -gui -drc -runset drc_runset -batch
```

---

**Tip**

 For supported design tools, you can add a custom menu item to the integration menu of the design tool. The new menu item can be defined to start the Calibre Interactive batch run. See “[Creating Custom Menus in Design Tools](#)” on page 454.

---

## Calibre Interactive Operation with a TVF Rule File

You specify a TVF rule file in Calibre Interactive on the Rules pane just as you do for an SVRF rule file. Some additional options are available when running with a TVF rule file. Debugging methods can be slightly different with a TVF rule file.

The following sections are included:

- [Specify the TVF Rule File](#)
- [Include TVF Statements with the Calibre Interactive Run](#)
- [Save the Generated SVRF Rule File](#)
- [Debugging Errors in a Calibre Interactive Run with a TVF Rule File](#)

### Specify the TVF Rule File

Click the **Rules** button on the left panel on Calibre Interactive and specify the TVF rule file in the Rules File field.

### Include TVF Statements with the Calibre Interactive Run

You can use Calibre Interactive to add TVF statements to a run that uses a TVF rule file as the main rule file on the Rules pane.

Select the **Options** page (**Settings > Show Pages > Options**). Enable “Include Rule Statements” and choose TVF in the Type dropdown menu. Enter statements in the text area; the statements are added to the control file when you start the run. If you choose SVRF in the button dropdown list the statements are wrapped in a TVF::VERBATIM block.

## Save the Generated SVRF Rule File

You can enable a preference setting to save the translated SVRF code to a file. This option corresponds to the calibre -E command line option.

In the **Preferences** page (**Settings > Show Pages > Preferences**), in the Rules section, enable “Save TVF Rules to SVRF before running Calibre.” The saved file is named `_<tvf_rule_file>_.svrf`, where `<tvf_rule_file>` is replaced with the name of the TVF rule file specified on the Rules pane. The generated SVRF file includes statements from the Calibre Interactive control file.

## Debugging Errors in a Calibre Interactive Run with a TVF Rule File

If the TVF rule file or the SVRF code generated from it has a compilation error the run is stopped. The generated SVRF code is still saved if “Save TVF Rules to SVRF before running Calibre” is enabled on the in the **Preferences** page *and* the run is on localhost. In such a case the generated SVRF file *does not* include statements from the Calibre Interactive control file.

If the generated SVRF code contains a compilation error, a pop-up message similar to the following is displayed:

```
Error while compiling rules file <tvf file>
Error xxx on line xxx of SVRF generated from TVF ...
```

where the line number refers to the line number in the generated SVRF code. If “Save TVF Rules to SVRF before running Calibre” is enabled you can examine the generated SVRF code to debug the error.

If the error occurs in the TVF code, an error similar to the following occurs:

```
Error while compiling rules file <tvf file>
Error xxx ...
(file <tvf file> line xxx)
```

where the line number refers to the TVF rule file.

## Check Selection in Calibre Interactive

The method for the checks that are executed in a Calibre Interactive run depends on which application you are running.	
<b>Check Selection Recipe .....</b>	<b>82</b>
<b>Specifying a Check Selection Recipe .....</b>	<b>82</b>
<b>Creating a Custom Check Selection Recipe .....</b>	<b>83</b>
<b>Basic Editing of Custom Check Selection Recipes .....</b>	<b>84</b>
<b>Advanced Editing of Check Selection Recipes in Calibre Interactive .....</b>	<b>86</b>
<b>Running Only Checks Selected in the Rule File with a Custom Check Selection Recipe</b>	<b>88</b>
<b>Recipe Editor Page.....</b>	<b>90</b>

## Check Selection Recipe

A Check Selection Recipe is a set of expressions for selecting the checks to execute during a Calibre run. You can use built-in check recipes or create user-defined check recipes tailored for specific cases. Check Selection Recipes are used in Calibre Interactive nmDRC and Calibre Interactive nmLVS for ERC checks. The recipes are saved to the runset.

Calibre Interactive defines the following two built-in check recipes:

- **Checks selected in the rules file** — Execute the same set of checks that would be executed in a batch Calibre run. This is the only built-in recipe that obeys [DRC \[Un\]Select Check](#) and [ERC \[Un\]Select Check](#) statements in the rule file.
- **All checks** — Executes *all* checks in the rule file except those excluded by pre-processor directives such as #IFDEF. DRC [Un]Select Check and ERC [Un]Select Check statements are *not* considered for the “All checks” recipe.

Custom Check Selection Recipes, also called user recipes, are useful for specific tasks or designs. User-defined check recipes allow you to tailor the rule checks that are executed to the type of design work you are doing. For example, you can define a custom recipe to only run checks for metal layers, or run all checks except those that include connectivity. User-defined check recipes and the currently active check recipe are saved to the Calibre Interactive runset.

You can disable selecting and editing the check recipe with the settings “Disable check selection” and “Disable check selection recipe editing” in the Disable section of the [Preferences page \(Settings > Show Pages > Preferences\)](#).

## Specifying a Check Selection Recipe

The Check Selection Recipe specifies the checks that are executed in a Calibre Interactive nmDRC or nmLVS run. You can use a built-in check recipe or define your check recipe.

## Prerequisites

- Calibre Interactive nmDRC or Calibre Interactive nmLVS is running.
- “Disable check selection” is not enabled in the Disable section of the Preferences page (**Settings > Show Pages > Preferences**).

## Procedure

1. Click the **Rules** button on the left panel to display the Rules page.
2. Specify a rule file.
3. Choose a check recipe using the dropdown list for Recipe in the Check Selection section.

# Creating a Custom Check Selection Recipe

You can create a new custom check recipe or copy an existing recipe from the Check Selection Recipe Editor dialog box. Custom check recipes (also called user recipes) allow you to control which Calibre rule checks are executed during a Calibre nmDRC or Calibre nmLVS run. Check recipe definitions are saved in the Calibre Interactive runset, along with the currently active check recipe.

## Prerequisites

- Calibre Interactive nmDRC or Calibre Interactive nmLVS is running.
- A rule file is available.
- “Disable check selection” is not enabled in the Disable section of the Preferences page (**Settings > Show Pages > Preferences**).

## Procedure

1. Click the **Rules** button on the left panel of Calibre Interactive to view the Rules page.
2. Specify a rule file. You cannot open the check recipe editor without specifying a rule file.
3. In the Check Selection section, click the **Edit** button to the right of the Recipe entry.

A view of the recipe editor is shown in “[Recipe Editor Page](#)” on page 90.

4. Do one of the following:

To do this ...	Take these steps ...
Create a new recipe	1. Click <b>New</b> in the top button bar. A new recipe is initialized with the checks selected in the rule file.
Copy an existing recipe	1. Choose the recipe to copy in the Recipe dropdown list. 2. Click <b>Copy</b> in the top button bar.

5. Proceed to “[Basic Editing of Custom Check Selection Recipes](#)” on page 84 or “[Advanced Editing of Check Selection Recipes in Calibre Interactive](#)” on page 86.

## Basic Editing of Custom Check Selection Recipes

The basic editing controls allow you to construct check recipes based on a short list of common expressions, such as “Checks selected in the rule file” and “Checks with density”. You can also include and exclude individual checks and check groups.

Custom check recipes (also called user recipes) allow you to control which Calibre rule checks are executed during a Calibre nmDRC or Calibre nmLVS run. Check recipe definitions are saved in the Calibre Interactive runset, along with the currently active check recipe.

### Prerequisites

- Calibre Interactive nmDRC or Calibre Interactive nmLVS is running.
- A rule file is loaded on the Rules pane.
- “Disable check selection” is not enabled in the Disable section of the Preferences page ([Settings > Show Pages > Preferences](#)).

### Procedure

1. Click the **Rules** button on the left panel of Calibre Interactive to view the **Rules** page.
2. Specify a rule file. You cannot edit a check recipe without specifying a rule file.
3. In the Check Selection section, choose the recipe you want to edit in the Recipe dropdown list.

---

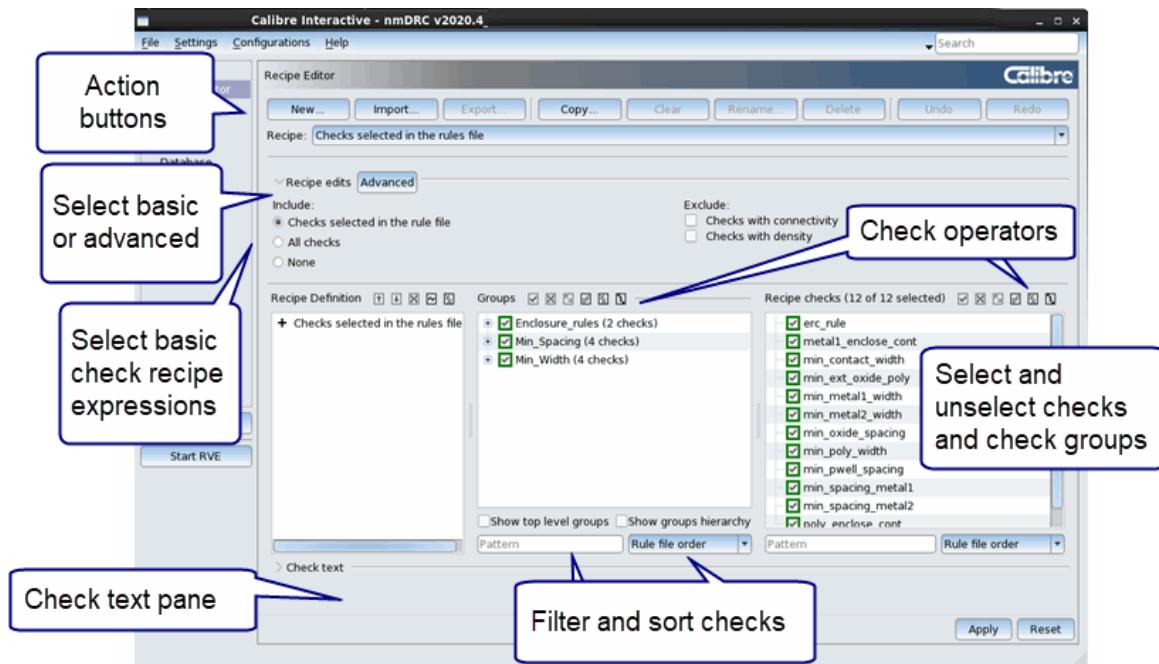
#### **Tip**

If you edit a built-in recipe, it is automatically saved as a user recipe with “(Modified)” appended to the recipe name. Only one modified version of each built-in recipe is saved at a time—new modifications overwrite any existing modified recipe.

---

4. Click the **Edit** button at the far right of the Recipe field to open the **Recipe Editor** page.

- For basic recipe editing, make sure the **Recipe Editor** page appears as shown in the following figure. If it doesn't, click the **Advanced** button to collapse the advanced controls.



- Use the Include and Exclude selections to select the types of checks in the check recipe.
- Enable and disable individual checks and check groups in the Recipe Checks area. The Groups section is only displayed if check groups are defined with the **Group** statement.
- (Optional) Click **Export** to save the check recipe to a file; recipe files are given the **.rcp** file extension. The recipe file can be imported by other users with the **Import** button. Saved DRC check recipes can be imported by **Calibre RealTime Custom** and **Calibre RealTime Digital**.
- Click **OK** to save the recipe and exit the dialog box.

#### **Tip**

If you edited a built-in recipe, you can click the **Rename** button to give the modified recipe a new name.

- (Recommended) Choose **File > Save Runset** to save the recipe definition to a runset.

#### **Tip**

You must either save a runset or export the recipe to a file if you want the new check recipe to be available in later sessions of Calibre Interactive.

# Advanced Editing of Check Selection Recipes in Calibre Interactive

The advanced editing controls in the Recipe Editor allow you to select checks by layer and group, apply a wildcard filter, and select and unselect individual rule checks.

Custom check recipes (also called user recipes) allow you to control which Calibre rule checks are executed during a Calibre nmDRC or Calibre nmLVS run. Check recipe definitions are saved in the Calibre Interactive runset, along with the currently active check recipe.

## Prerequisites

- Calibre Interactive nmDRC or Calibre Interactive nmLVS is running.
- A rule file is loaded on the Rules pane.

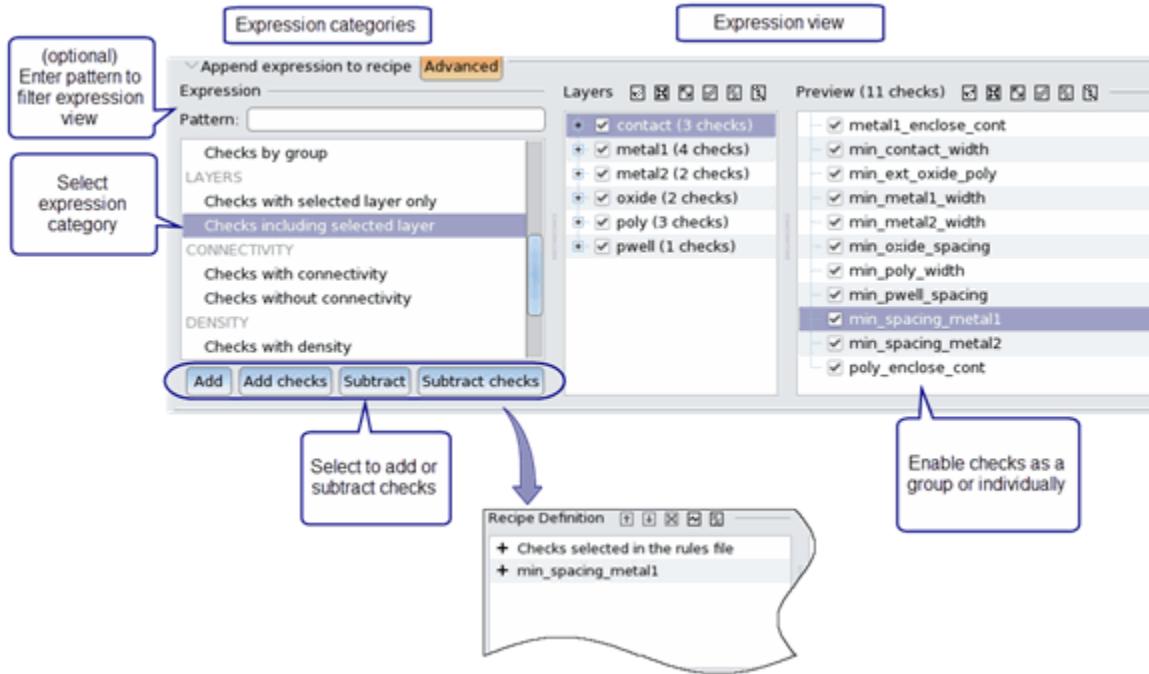
## Procedure

1. Open a recipe for editing and make selections using the basic editing controls, as described in “[Basic Editing of Custom Check Selection Recipes](#)” on page 84.
2. Click the **Advanced** button to expand the advanced editing controls if they are not visible. The page should look similar to that shown in [Figure 3-4](#) in “[Recipe Editor Page](#)” on page 90.
3. Do the following to add an expression to the recipe definition:
  - a. Choose an expression category in the left pane in the Expression area.
  - b. (Optional) You can add and subtract individual checks from the expression category selected in Step a. In the Preview pane on the right, select and unselect checks. The view in this area depends on the expression category and on the Pattern field; wildcard expressions are supported. You can use the following methods to select checks:
    - Click in the checkbox next to the rule check (or layer name if selecting by layer).
    - Right-click for a selection menu.
  - c. Click one of the following buttons to update the check recipe definition:
    - **Add** — Adds the expression and any modifications from the Preview pane.
    - **Add checks** — Adds the checks selected in the Preview pane as a list of rule checks.
    - **Subtract** — Subtracts the expression and any modifications from the Preview pane.
    - **Subtract checks** — Subtracts the checks selected in the Preview pane as a list of rule checks.

### Note

 Recipes built with expressions can often be used with a different rule file, unlike recipes that list individual rule checks.

**Figure 3-2. Adding a Recipe Expression**



4. Repeat Step 3 as necessary to add more expressions to the check recipe definition.
5. View the check recipe in the Recipe definition area. You can use the      buttons to move an expression up or down in the list, remove an expression, resolve an expression into the equivalent list of rule checks, or remove all recipe expressions.

### Tip

 If you want to only run checks that are selected in the rule file when using a custom check recipe, go to the Expression area, select “Checks not selected in the rules file,” select “Subtract,” and click the **Apply** button. Make sure the expression “- Checks not selected in the rules file” is the last expression in the recipe definition.

6. View the list of included and excluded checks in the Recipe checks area. A **Groups** tab is included if check groups are created in the rule file (the **Group** statement).

A green check mark indicates that the rule check or group is included in the recipe. A red empty space indicates the item is not included. Click in the checkbox next to an item to include or exclude it from the recipe.

In the Recipe Checks area you can right-click for a menu to add and subtract checks. You can also enter a filter pattern to filter the rule checks.

7. (Optional) Click **Export** to save the check recipe to a file. The recipe file can be imported by other users with the **Import** button. Saved DRC check recipes can be imported by [Calibre RealTime Custom](#) and [Calibre RealTime Digital](#).
8. Click **Apply**.

---

**Tip**

If you edited a built-in recipe, you can click the **Rename** button to give the modified recipe a new name.

---

9. (Recommended) Choose **File > Save Runset** to save the recipe definition to a runset.

---

**Tip**

You must either save a runset or export the recipe to a file if you want the new check recipe to be available in later sessions of Calibre Interactive.

---

## Running Only Checks Selected in the Rule File with a Custom Check Selection Recipe

You can subtract the check expression “Checks not selected in the rules file” from a custom check recipe in order to run only checks that are selected in the rule file. This step is necessary because most check expressions do not obey DRC [Un]Select Check statements in the rule file.

### Prerequisites

- Calibre Interactive nmDRC is running.
- A rule file is loaded on the **Rules** page.

### Procedure

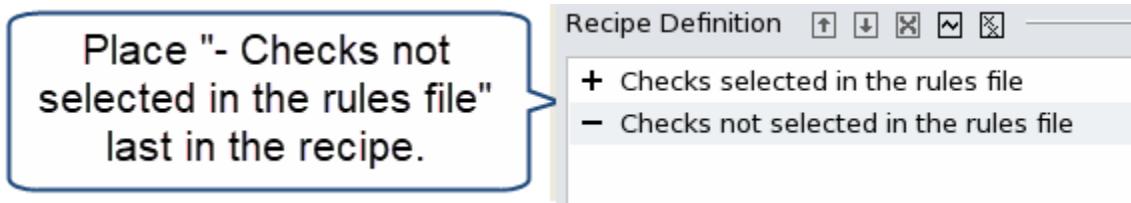
1. Do one of the following to open a recipe for editing:
  - Create a custom check recipe. See “[Creating a Custom Check Selection Recipe](#)” on page 83.
  - Open an existing check recipe for editing:
    - i. On the **Rules** page, choose the recipe to edit in the Recipe dropdown list in the Check Selection section.
    - ii. Click **Edit**.

Leave the recipe open in the Check Selection Recipe editor dialog box.

2. Click **Advanced** to view advanced options.
3. Select “Checks not selected in the rules file” in the Expression area.
4. Click the **Subtract** button.

5. View the “Recipe definition” area and confirm that “- Checks not selected in the rules file” is the last expression in the recipe definition.

## Examples



## Recipe Editor Page

To access: In the **Rules** page, click the **Edit** button to the right of the Recipe dropdown field in the Check Selection section.

The Recipe Editor page includes controls to create, copy, save, and edit a check selection recipe. Basic and advanced editing controls are available.

### Description

The Edit Recipe Dialog box has two modes: basic and advanced.

Use the basic controls (Figure 3-3) for the following tasks:

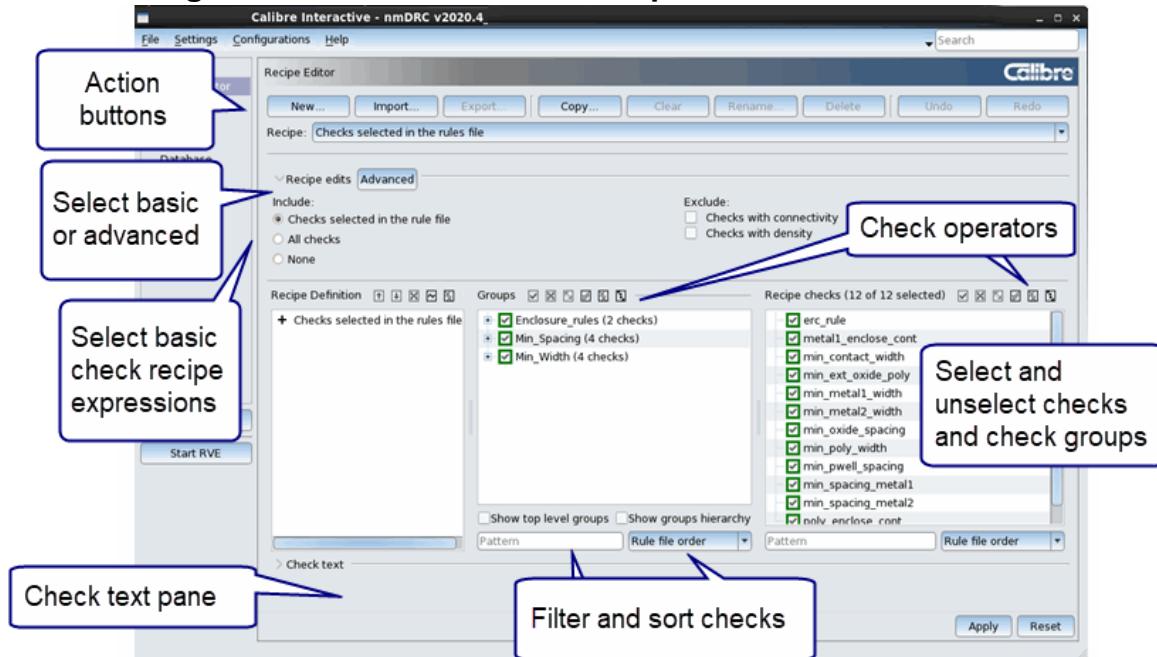
- Include and exclude basic recipe expressions.
- Include and exclude individual checks or check groups.

Use the advanced controls (Figure 3-4) for the following tasks:

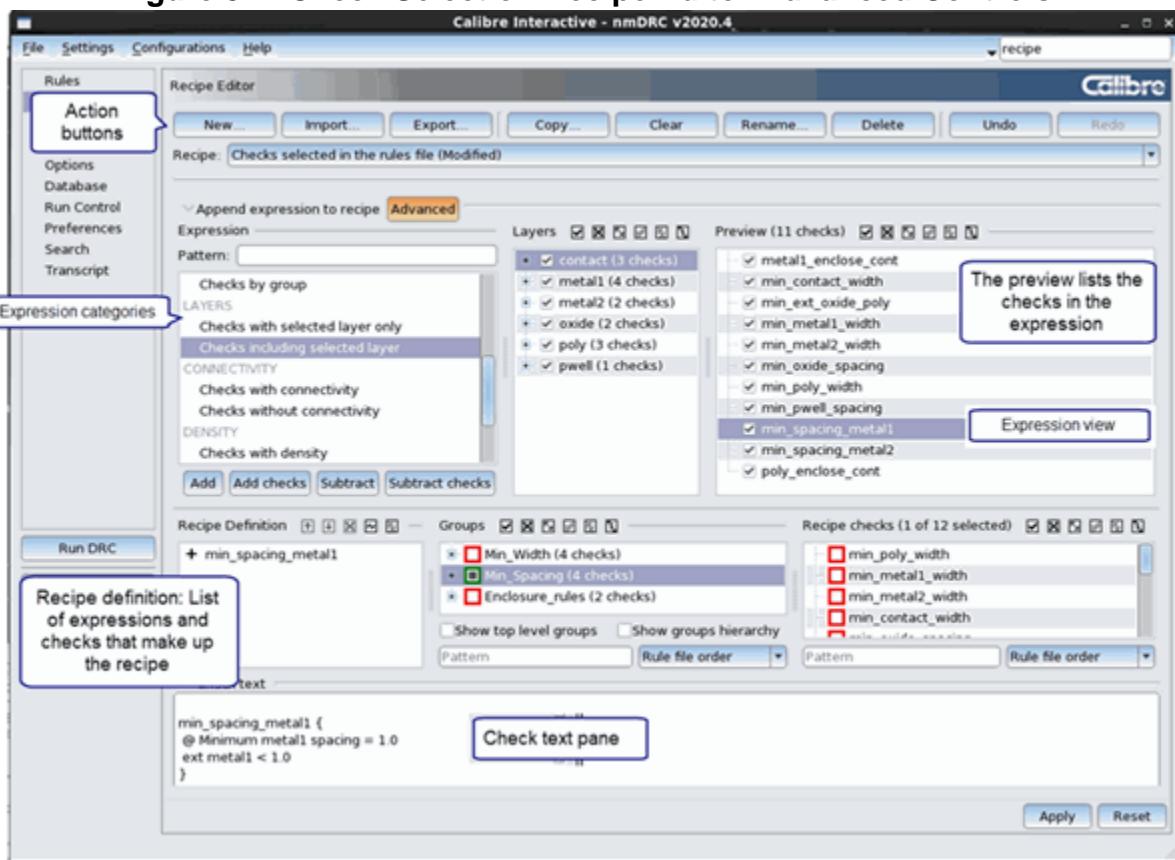
- Develop advanced recipe expressions involving selected layers, visible or recently edited layers, rule check comments, and check groups.
- View and edit the recipe definition.

Descriptions of the fields and selections in the dialog box follow the figures.

**Figure 3-3. Check Selection Recipe Editor Basic Controls**



**Figure 3-4. Check Selection Recipe Editor Advanced Controls**



## Objects

**Table 3-2. Check Selection Recipe Editor — Action Buttons**

Item	Type	Description
New	Button	Open a new recipe.
Import	Button	Open a recipe that was saved to file with the <b>Export</b> button. You can import a check recipe file exported from <b>Calibre RealTime Custom</b> and <b>Calibre RealTime Digital</b> ; however, the recipe expressions involving visible and recently edited layers have no meaning in Calibre Interactive.
Export	Button	Save a recipe to file. A recipe saved in this manner can be read using the <b>Import</b> button. <b>Calibre RealTime Custom</b> and <b>Calibre RealTime Digital</b> can import check recipes saved to file from Calibre Interactive.
Copy	Button	Create a copy of the selected recipe.
Clear	Button	Clear all expressions from the Recipe definition area and return to the default of “Checks selected in the rules file.”

**Table 3-2. Check Selection Recipe Editor — Action Buttons (cont.)**

Item	Type	Description
Rename	Button	Rename the selected check recipe.
Delete	Button	Delete the selected check recipe.
Undo	Button	Undo the last modification to the recipe definition.
Redo	Button	Redo the last modification to the recipe definition.

**Table 3-3. Check Selection Recipe Editor — Basic Controls**

Item	Type	Description
Recipe	Dropdown list	Specifies the active recipe.
Advanced	Button	Toggle between basic and advanced editing controls.
Include	Part of Recipe edits pane	<p>Specify checks to include:</p> <ul style="list-style-type: none"> <li>• <b>Checks selected in the rules file</b> — Include checks selected in the rule file with DRC [Un]Select Check statements.</li> <li>• <b>All checks</b> — Include all checks in the rule file.</li> <li>• <b>None</b> — Do not include any checks.</li> </ul>
Exclude	Part of Recipe edits pane	<p>Specify checks to exclude:</p> <ul style="list-style-type: none"> <li>• Checks with connectivity.</li> <li>• Checks with density.</li> </ul>
Recipe checks Groups	Pane	<p>Displays tabs with the rule checks and check groups in the rule file; the Groups section is only displayed if check groups are defined in the rule file.</p> <p>A green check mark indicates that the rule check or group is included in the recipe. A red space indicates the item is not included. Click in the checkbox next to an item to include or exclude it from the recipe. Right-click for a menu to add and subtract checks.</p> <p>You can also enter a filter pattern to filter the rule checks that are shown in the Recipe checks and Groups sections.</p>

**Table 3-3. Check Selection Recipe Editor — Basic Controls (cont.)**

Item	Type	Description
Recipe definition	Pane	<p>Displays the expressions that make up the recipe definition. Use the following buttons to operate on the selected expression:</p> <ul style="list-style-type: none"> <li> — Move expression up in the list</li> <li> — Move expression down in the list</li> <li> — Delete the expression</li> <li> — Resolve recipe into equivalent list of rule checks</li> <li> — Remove all recipe expressions</li> </ul>
Check text	Pane	Displays the check text for the selected rule check.

**Table 3-4. Check Selection Recipe Editor — Advanced Controls**

Item	Type	Description
Recipe	Dropdown list	Specifies the active recipe.
Advanced	Button	Toggle between basic and advanced editing controls.
Expression	Pane	<p>The list of available expression categories for the Check Selection Recipe. See the expression categories table for the definition of each expression.</p> <p>The Preview pane to the right displays the checks that are selected by the expression. Click in the checkbox next to a check name to change whether the check is included. You can also right-click for a menu.</p>
Add	Button	<p>Update the check recipe by adding the selected expression; for example, “+ Checks by group: width_rules”. Recipes built with expressions can often be used with a different rule file.</p> <p>The recipe definition is not modified until you click either <b>Add</b>, <b>Add checks</b>, <b>Subtract</b>, or <b>Subtract checks</b>.</p>
Add checks	Button	<p>Add the checks selected in the Preview pane to the recipe definition as a list of checks. Recipes built with a list of rule checks may not be able to be used with a different rule file.</p> <p>The recipe definition is not modified until you click either <b>Add</b>, <b>Add checks</b>, <b>Subtract</b>, or <b>Subtract checks</b>.</p>
Subtract	Button	<p>Update the check recipe by subtracting the selected expression; for example, “- Checks by group: width_rules”. Recipes built with expressions can often be used with a different rule file.</p> <p>The recipe definition is not modified until you click either <b>Add</b>, <b>Add checks</b>, <b>Subtract</b>, or <b>Subtract checks</b>.</p>

**Table 3-4. Check Selection Recipe Editor — Advanced Controls (cont.)**

Item	Type	Description
Subtract checks	Button	<p>Subtract the checks selected in the Preview pane to the recipe definition as a list of checks. Recipes built with a list of rule checks may not be able to be used with a different rule file.</p> <p>The recipe definition is not modified until you click either <b>Add</b>, <b>Add checks</b>, <b>Subtract</b>, or <b>Subtract checks</b>.</p>
Recipe definition	Pane	<p>Displays the expressions that make up the recipe definition. Use the following buttons to operate on the selected expression:</p> <ul style="list-style-type: none"> <li> — Move expression up in the list</li> <li> — Move expression down in the list</li> <li> — Delete the expression</li> <li> — Resolve recipe into equivalent list of rule checks</li> <li> — Remove all recipe expressions</li> </ul>
Recipe checks Groups	Pane	<p>Displays the rule checks and check groups in the rule file; the Groups pane is only displayed if check groups are defined in the rule file.</p> <p>A green check mark indicates that the rule check or group is included in the recipe. A red X indicates the item is not included. Click in the checkbox next to an item to include or exclude it from the recipe.</p> <p>In the Recipe Checks pane, you can right-click for a menu to add and subtract checks. You can also enter a filter pattern to filter the rule checks.</p>
Check text	Pane	Displays the check text for the selected rule check.

**Table 3-5. Expression Categories for Check Selection Recipes**

Expression Category	Definition and Usage
The expression categories appear in the Expression pane in the dialog box, as shown in <a href="#">Figure 3-4</a>	
All checks	<p>All checks in the rule file, except those excluded by preprocessor directives. <a href="#">DRC [Un]Select Check</a> and <a href="#">ERC [Un]Select Check</a> statements are <i>not</i> considered.</p> <p>All checks are listed and enabled in the expression view. Click in the checkbox next to a check name to include or exclude the check from the recipe or right-click for a menu to add and subtract checks.</p>

**Table 3-5. Expression Categories for Check Selection Recipes (cont.)**

Expression Category	Definition and Usage
Checks selected in the rules file	<p>Checks selected in the rule file. This expression and “Checks not selected in the rules file” are the only expressions that obey the DRC [Un]Select Check and ERC [Un]Select Check statements in the rule file.</p> <p>Note that if the rule file does not include any DRC Select Check statements, then all checks are executed; this is the same behavior as for batch Calibre nmDRC.</p> <p>Checks selected in the rule file are listed and enabled in the expression view. Click in the checkbox next to a check name to include or exclude the check from the recipe or right-click for a menu to add and subtract checks.</p>
Checks not selected in the rules file	<p>Checks <i>not</i> selected in the rule file. This expression and “Checks selected in the rules file” are the only expressions that obey the DRC [Un]Select Check and ERC [Un]Select Check statements in the rule file.</p> <p>Checks not selected in the rule file are listed and enabled in the expression view. Click in the checkbox next to a check name to include or exclude the check from the recipe or right-click for a menu to add and subtract checks.</p> <p><b>i Tip:</b> You can subtract this expression at the end of a recipe definition to ensure that only checks selected in the rule file are included in the recipe.</p>
Checks by rule check comments	<p>Choose checks by filtering on the rule check comment. Enter a wildcard pattern in the Pattern field. For example, enter *metal1* to have only checks that include “metal1” in the check text comment listed in the Preview pane. Click in the checkbox next to a check name to include or exclude the check from the recipe or right-click for a menu to add and subtract checks.</p> <p>See “<a href="#">Rule Comments</a>” in the Calibre Verification User’s Manual.</p>
Checks selected in recon mode	<p>Choose checks automatically selected in Calibre nmDRC Reconnaissance mode. Requires a license for Calibre RealTime Digital.</p> <p>Calibre nmDRC Recon selects checks that identify gross errors early in the design and assembly process. For more information, see “<a href="#">Chip-Level Verification</a>” in the <i>Calibre Solutions for Physical Verification</i> manual.</p>

**Table 3-5. Expression Categories for Check Selection Recipes (cont.)**

Expression Category	Definition and Usage
Checks selected in recon inverse mode	Choose checks <i>not</i> automatically selected in Calibre nmDRC Reconnaissance mode. Requires a license for Calibre RealTime Digital.
Checks by group	Choose checks by group name. Rule check groups are listed in the expression view. You can expand each group to view the rule checks. Click in the checkbox next to the group name or individual check to enable or disable it. Click the <b>Preview</b> tab to view the list of enabled and disabled checks.
Checks with selected layer only	Choose checks by layer, where the rule check includes <i>only</i> the indicated layer.  Original layers that are the only layer in a rule check are displayed in the expression view. Layers can be expanded to view the rule checks.  Click in the checkbox next to a layer to enable the rule checks that include only the indicated layer or right-click for a menu to select checks. Click the <b>Preview</b> tab to view the list of enabled and disabled checks.
Checks including selected layer	Choose checks by layer, where the rule check includes the indicated layer and possibly other layers.  Original layers that are included in a rule check are displayed in the expression view. Layers that are included in a rule check can be expanded to view the rule checks.  Click in the checkbox next to a layer to enable the rule checks that include the indicated layer; rule checks that include other layers in addition to the indicated layer are also enabled. Click the <b>Preview</b> tab to view the list of enabled checks.
Checks with connectivity	Choose checks that include connectivity.  All checks with connectivity are listed and enabled in the expression view. Click in the checkbox next to a check name to include or exclude the check from the recipe or right-click for a menu to add and subtract checks.
Checks without connectivity	Choose checks that do not include connectivity.  All checks without connectivity are listed and enabled in the expression view. Click in the checkbox next to a check name to include or exclude the check from the recipe or right-click for a menu to add and subtract checks.

**Table 3-5. Expression Categories for Check Selection Recipes (cont.)**

Expression Category	Definition and Usage
Checks with density	Choose Density checks. All checks with the Density operation are listed and enabled in the expression view; this includes checks in which the Density operation is used to derive an input layer for the check. Click in the checkbox next to a check name to include or exclude the check from the recipe or right-click for a menu to add and subtract checks.
Checks without density	Choose checks that do not include the Density operation. All checks without the Density operation are listed and enabled in the expression view. Click in the checkbox next to a check name to include or exclude the check from the recipe or right-click for a menu to add and subtract checks.

## Viewing and Editing Environment Variables

Calibre Interactive allows you to quickly view and edit environment variables relevant to your design. The environment variables referenced in your rule file are automatically listed in the **Environment** page. In addition to viewing rule file environment variables, you can view the environment variables defined in the shell session and create and edit runset environment variables.

Calibre Interactive maintains a distinction between rule file and runset environment variables, and between shell and runset values, as summarized here:

- **Rule file environment variable** — One referenced in the rule file.
- **Shell environment value** — The value defined in your shell environment for the environment variable.
- **Runset environment variable** — One created as part of your Calibre Interactive session or saved in a runset.
- **Runset environment value** — The value defined in your Calibre Interactive session or saved in the runset for the environment variable.

For information on using environment variables in GUI fields and runsets, see “[Environment Variables in Calibre Interactive GUI Fields](#)” on page 52:

See “[Environment Page in Calibre Interactive](#)” on page 116 for more information on how to access and use this page.

## Configuration Files for Startup Preferences

Calibre Interactive automatically saves your startup preferences to special setup files, stored in your \$HOME directory by default. Separate setup files allow you to define different startup preferences for the different Calibre Interactive applications.

The filenames used for classic Calibre Interactive are supported:

- *.cgidrcdb* for DRC
- *.cgilvsdb* for LVS
- *.cgipercdb* for PERC
- *.cgipexdb* for PEX

These setup files contain the last 10 runsets opened in Calibre Interactive and are used to populate the dropdown list of runsets in the Open Runset dialog box. The setup files also

contain the settings from the Startup Runset and Calibre Interactive Window Position sections of the Startup section of the **Preferences** page.

When you save over these classic Calibre Interactive setup files, the filenames are changed to avoid overwriting them.

---

**Note**

---

 Calibre Interactive also creates a *.cgi<tool>db* file in the current run directory. This file is for internal tool use only, and is not the same as the startup preference file in your home directory.

---

---

**Tip**

---

 You can populate the dropdown list of runsets in the Open Runset dialog box using the environment variable MGC\_CALIBRE\_DRC\_RUNSET\_LIST; replace DRC with LVS, PERC, or PEX for other applications. See “[Methods to Load a Calibre Interactive Runset](#)” on page 35.

---

## Templates for File Naming

You can use file naming templates to form the names that Calibre Interactive generates. The template preference settings are applied when you invoke Calibre Interactive from a layout or schematic editor. Template settings make use of the cell and view open in the design tool to name input and output files. You define file naming templates in the **Templates** page (**Settings > Show Pages > Templates**).

If “Apply Templates” is enabled in the **Templates** page the template settings are used when you invoke Calibre Interactive from a design tool. The settings formed by templates take precedence over other sunset settings for the same fields. The template settings are saved with the sunset.

The default behavior for extracted layout files, extracted SPICE netlist files, and output files is to generate the names for these files based upon layout and source top cell names given by the replaceable parameters %l and %s. You can tailor the file naming templates to fit your needs. Also see “[Export from Viewer Preferences](#)” on page 103 for important information about the “Export from layout viewer” and “Export from schematic viewer” template settings.

You can use absolute or relative pathnames on **Templates** tab; relative paths are relative to the run directory. The replaceable parameters given in [Table 3-6](#) are supported in template names.

**Table 3-6. Replaceable Parameters in Template Names**

Parameter	Definition
%l	The primary cell in the layout database.
%s	The primary cell in the source netlist (used only for LVS, PERC, and PEX runs).
%L	The name of the layout library for OpenAccess databases. Click the <b>Templates &gt; Database</b> sub-tab to set the template.
%V	The name of the layout view for OpenAccess databases. Click the <b>Templates &gt; Database</b> sub-tab to set the template.
%S	The name of the source library for OpenAccess databases. Click the <b>Templates &gt; Database</b> sub-tab to set the template.
%W	The name of the source view for OpenAccess databases. Click the <b>Templates &gt; Database</b> sub-tab to set the template.
Replaceable parameters for the output parasitic netlist filename. Click the <b>Templates &gt; Outputs</b> sub-tab to set the “PEX Netlist” template.	
%X	The extraction type: <ul style="list-style-type: none"><li>• R+C+CC — rcc</li><li>• R+C — rc</li><li>• R — r</li><li>• C+CC — cc</li><li>• No R/C — norc</li></ul>

**Table 3-6. Replaceable Parameters in Template Names (cont.)**

Parameter	Definition
%x	The output netlist format: <ul style="list-style-type: none"><li>• Calibre View — cv</li><li>• DSPF — dspf</li><li>• Eldo — spi</li><li>• HSPICE — sp</li><li>• Spectre — scs</li><li>• SPEF — spef</li></ul>
%o	The netlist format option: <ul style="list-style-type: none"><li>• PRIMETIME — pt</li><li>• HSIM — hsim</li><li>• ADITRAIL — ar</li><li>• Net Geometry — ng</li></ul>

See the following topics for more information.

- “[Example for File Naming Template](#)” on page 101
- “[Disabling Templates So That Saved Runset Values Are Used](#)” on page 102
- “[Export from Viewer Preferences](#)” on page 103
- “[Naming the Extracted Parasitic Netlist Based on the Extraction Type and Format](#)” on page 211

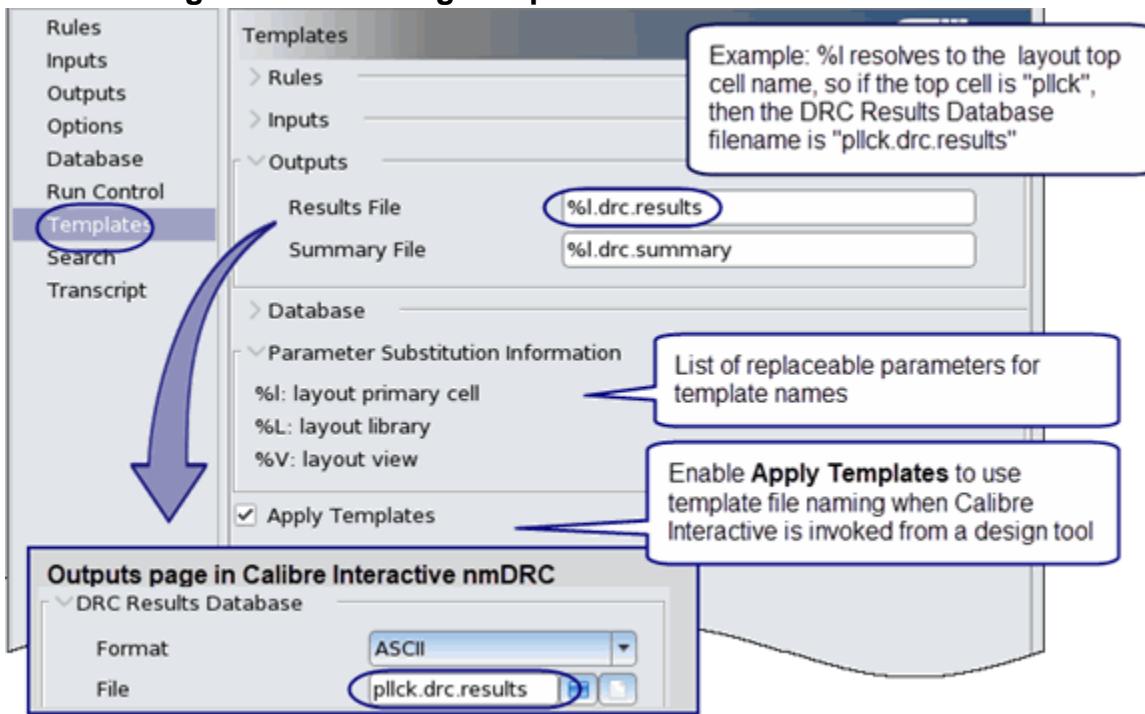
## Example for File Naming Template

You can use a template definition to create an output file name based on the top cell in the design. The method for doing this is shown in an example and with a short video.

For example, suppose you open a design with a top cell of “pllclk” in your design tool, then start Calibre Interactive nmDRC. File naming templates apply because Calibre Interactive was opened from a design tool. The default file naming template for the output database is “%l.drc.results”, where %l is replaced with the name of the topcell. When you view the DRC Results Database filename on the Outputs pane, it is set to “pllclk.drc.results”.

[Figure 3-5](#) shows the setup for this example. The figure shows the **Templates** page and a section of the **Outputs** page for Calibre Interactive nmDRC.

**Figure 3-5. Defining Templates for Generated Filenames**



Template definitions are saved to a runset parameter with “Template” in the runset option name, as in the following two lines from a runset file:

```
*drcTemplate_SF: %l.summary  
*cmnTemplate_RL: %l.rules
```

where drcTemplate\_SF and cmnTemplate\_RL define templates for the DRC summary file and the rule file, respectively.

## Disabling Templates So That Saved Runset Values Are Used

You may sometimes see that saved runset settings are not used when you open Calibre Interactive from a design tool. This occurs because templates are used to fill in certain GUI fields based on the open design when you launch Calibre Interactive from a design tool. You can disable templates if you want to use only saved runset options. You can disable all templates or delete the template for a specific GUI field.

### Prerequisites

- Calibre Interactive is open.

### Procedure

1. Choose **Settings > Show Pages > Templates** and open the Templates page.

2. Do one of the following to disable templates:

To ...	Do this ...
Disable all templates	Uncheck “Apply Templates.”
Disable a template for a specific GUI field	<ol style="list-style-type: none"><li>1. View the different sections and locate the setting for the GUI field.</li><li>2. Clear the text in the entry field for the template.</li></ol>

3. Make sure all GUI fields have the correct values. You may need to correct GUI fields that were filled in automatically using template settings.
4. Save the runset.

---

**Note**

 To insure that runset settings are always used, you may want to check the setting of “Read Inputs/Outputs fields while loading rules file” in the Rules section of the **Preferences** page. When this setting is enabled, input and output statements from the rule file populate the Inputs and Outputs pane when the rule file is loaded (**Load** button on the Rules pane).

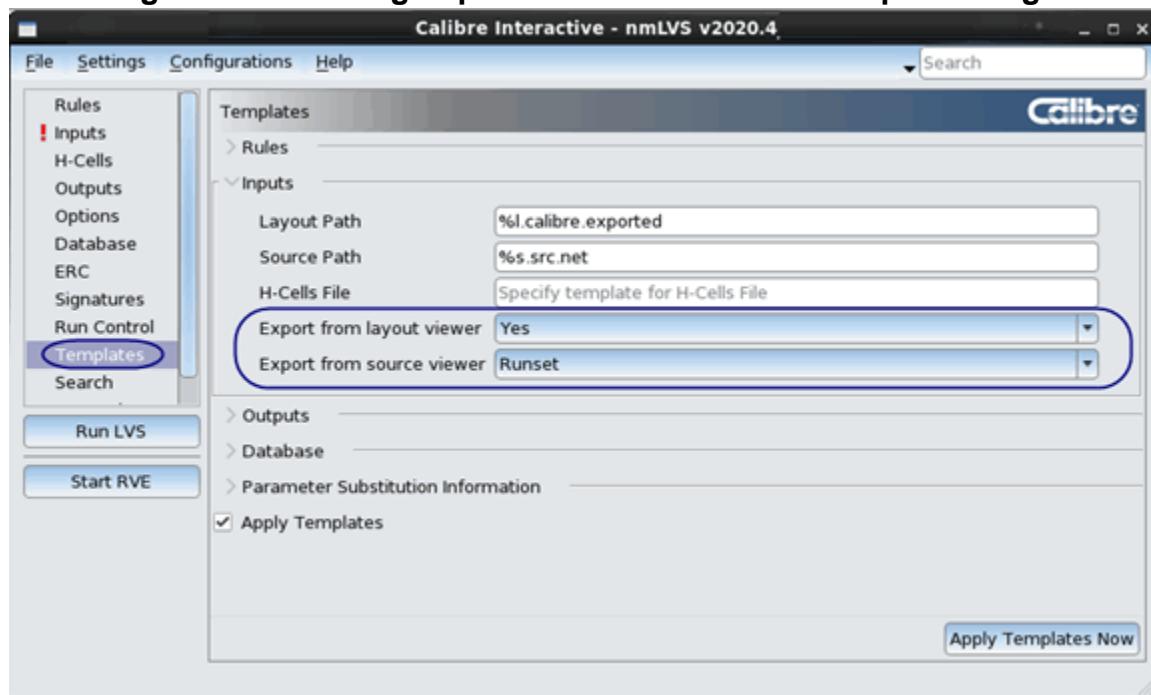
---

## Export from Viewer Preferences

If Calibre Interactive is connected to a layout or schematic viewer, you have the option of specifying the open layout or netlist as the input file for the run; this is called layout or schematic export. You can set a preference for the export setting that is used when Calibre Interactive is invoked from a layout or schematic viewer. The preference setting is on the **Templates** page, under the Inputs section. This preference setting is not used in batch mode or command line execution.

Figure 3-6 shows the **Inputs** page for Calibre Interactive nmLVS. The options for input file templates are different for different run types, and the “Export from source viewer” setting is not present for Calibre nmDRC runs. Table 3-7 explains the possible settings of Yes, No, and Runset for the export preference.

**Figure 3-6. Defining Export Preferences in the Templates Page**



**Table 3-7. Export Preference Settings in the Setup Preferences Dialog**

Setting	Description
Export from layout viewer  Default = Yes	If Calibre Interactive is invoked from a layout viewer: <ul style="list-style-type: none"> <li>• <b>Yes</b> — Export from layout viewer, regardless of the sunset setting for “Export from layout viewer” on the <b>Layout</b> tab of the Inputs pane.</li> <li>• <b>No</b> — Do not export from layout viewer, regardless of the sunset setting for “Export from layout viewer” on the <b>Layout</b> tab of the Inputs pane.</li> <li>• <b>Runset</b> — Use the setting from “Export from layout viewer” on the <b>Layout</b> tab of the Inputs pane.</li> </ul>
Export from source viewer  (for LVS, PERC, and PEX only)  Default = Runset	If Calibre Interactive is invoked from a schematic viewer: <ul style="list-style-type: none"> <li>• <b>Yes</b> — Export from schematic viewer, regardless of the sunset setting for “Export from schematic viewer” on the <b>Netlist</b> tab of the Inputs pane.</li> <li>• <b>No</b> — Do not export from schematic viewer, regardless of the sunset setting for “Export from schematic viewer” on the <b>Netlist</b> tab of the Inputs pane.</li> <li>• <b>Runset</b> — Use the setting from “Export from schematic viewer” on the <b>Netlist</b> tab of the Inputs pane.</li> </ul>

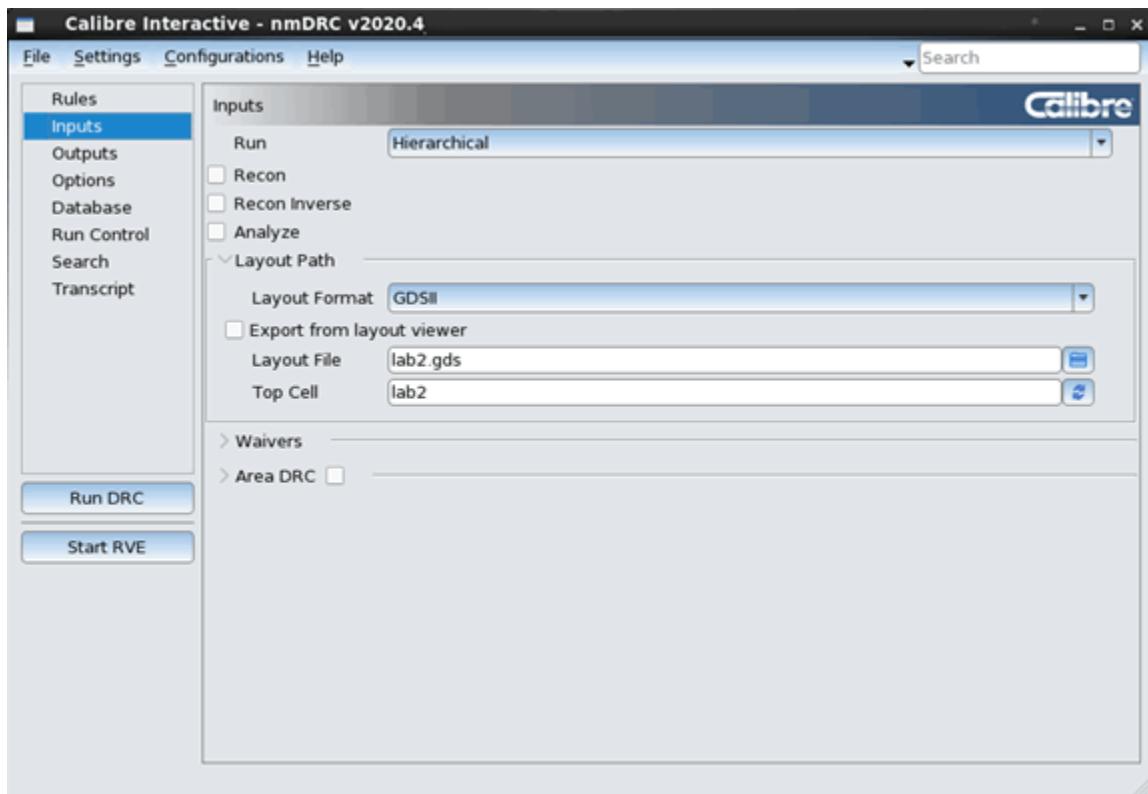
The template settings in [Table 3-7](#) affect the “Export from layout viewer” and “Export from source viewer” settings on the **Inputs** page under the Layout Path and Source Path sections.

# Chapter 4

## Calibre Interactive GUI Reference

This chapter includes reference information for GUI pages. The settings for a Calibre run are made in several pages selected from the left panel of the Calibre Interactive GUI. Some pages, such as the Rules page, are similar in all applications.

In most cases the name of the GUI control corresponds to an SVRF command or command option. Tooltips provide additional information. Choose **Settings > Show Pages** to hide and show pages. Some pages have a shortcut, or hotkey, for going to the page; this is indicated in parentheses.



**Table 4-1. Calibre Interactive GUI Items**

GUI Item	Description
Rules Page (Alt+r / Alt+R)	<p>Specify the rule file and the run directory on the Rules page.</p> <p>Click the  button to load rule file settings into the GUI. The rule file settings overwrite current GUI settings, including the path to the input design and the top cell name.</p> <p>To specify additional rule files, click Options on the left panel, select “Include Rules Files,” and expand the control area to specify the additional rule files.</p> <p>The Rules page includes the <a href="#">Layer Derivations Tree</a>.</p>
Inputs Page (Alt+i / Alt+I)	<p>Specify the input layout, schematic source (if used), and other input files.</p> <p>To specify additional layout or schematic input files, click Database on the left panel and expand the Library area.</p> <p>The Inputs page also specifies the run type for most applications.</p>
Outputs Page (Alt+o / Alt+O)	Specify parameters related to the output of the run.
Options Page (Alt+p / Alt+P)	<p>Specify options related to the application. Some options are common to all applications.</p> <p><a href="#">Options Page in Calibre Interactive: Common Options</a></p>
Application-specific pages	Specify additional application-specific options. The visibility of these additional pages is often dependent on other settings in the GUI.
Waivers page	Specify options for a waiver run or waiver creation. See “ <a href="#">Calibre Auto-Waivers in Calibre Interactive</a> ” on page 161.
<a href="#">Database Page</a>	Specify the Resolution and Precision statements, and additional layout and schematic input files.
<a href="#">OA/LEFDEF Page</a>	<p>Specify database conversion options for third party databases.</p> <p>The page is only displayed if “Layout Format” on the Inputs page is OPENACCESS or LEFDEF.</p> <p>See “<a href="#">Providing OpenAccess Input with Calibre Interactive</a>” on page 73 and “<a href="#">Providing LEF/DEF Input with Calibre Interactive</a>” on page 75.</p>
<a href="#">Environment Page</a>	View and specify environment variables for the run.

**Table 4-1. Calibre Interactive GUI Items (cont.)**

<b>GUI Item</b>	<b>Description</b>
V2LVS Page	Specify options for the v2lvs translator, which translates VERILOG format source input to SPICE format. The V2LVS page is available when VERILOG or MIXED is selected as the “Source Format” on the Inputs page. This page is similar to the Setup Verilog Translator dialog box in classic Calibre Interactive nmLVS—see “Setting Up the Verilog Translator (v2lvs)” in the <i>Calibre Interactive User’s Manual</i> .
<a href="#">Run Control Page</a>	Specify the run mode, processor information, options for invoking Calibre RVE, other run-related options. The page includes settings for connecting to a design tool.
<a href="#">Triggers Page</a>	Specify trigger functions that execute before and after the run.
<a href="#">Templates Page</a>	Specify settings for file-naming templates.
<a href="#">Preferences Page</a>	Specify settings for loading and saving the runset, saving the transcript, and loading a custom configuration file. The miscellaneous (Misc) preferences section controls warnings, prompts, automatic behaviors, and other options.
<a href="#">Search Toolbar (Ctrl+f / Ctrl+F)</a>	Search the GUI for a specified string.
<a href="#">Transcript Page (Alt+t / Alt+T)</a>	View the transcript of the run. Errors and warnings are listed in the bottom portion of the page. Enter Ctrl-f to open a search bar.
<a href="#">Run &lt;app&gt; button (Alt+d / Alt+D)</a>	Starts the Calibre run. To view the Calibre Interactive control file, click the <b>Run &lt;app&gt;</b> button while pressing the Ctrl key.
<a href="#">Start RVE button</a>	Open Calibre RVE and load the results database specified on the Outputs page.

Errors and warnings are indicated as follows:



A missing or incorrect setting. Hover over the icon for a message about the error. A red exclamation point (!) is also displayed by the page name if any settings have an error.



A warning, such as for a setting that conflicts with another option setting. The run can proceed with warnings. Hover over the icon for a message about the warning. The warning icon is not displayed next to the page name.

See these topics for information on application-specific pages:

- “[LVS-Specific Pages in Calibre Interactive](#)” on page 197

- “[DRC-Specific Pages in Calibre Interactive](#)” on page 156
- “[PEX-Specific Pages in Calibre Interactive](#)” on page 214

# Options Page in Calibre Interactive: Common Options

To access: Choose **Settings > Show Pages > Options** (shortcut: Alt-p)

The **Options** page for all Calibre Interactive applications has settings for including rule files, rule file statements, and layout text files.

## Objects

Object	Description
Virtual Connect	Specifies how nets with identical names are connected and reported. See <a href="#">Virtual Connect ... Commands</a> in the <i>SVRF Manual</i>
Include Rules Files	Specifies additional rule files to include in the run. Rule file statement: <b>Include</b> The additional rule files are included before the main rule file by default. To change this, open the Preferences page, expand the Rules area, and enable “Include additional rules files and statements after main rules file.”
Include Rule Statements	Specifies additional rule file statements to include in the run. The statements are included in the Calibre Interactive control file. If the main rule file specified on the Rules page is a TVF file, you can specify SVRF or TVF statements. If SVRF statements are included in a TVF rule file, the SVRF statements are enclosed within a TVF::VERBATIM block. The additional rule file statements are included before the main rule file by default. To change this, open the Preferences page, expand the Rules area, and enable “Include additional rules files and statements after main rules file.”
Include Layout Text File	Specifies a file with Layout Text statements. Layout Text statements in the Layout Text File are added to any Layout Text statements present in the rule file. Rule file statement: <b>Layout Text File</b>

## Related Topics

[DRC-Specific Pages in Calibre Interactive](#)

[LVS-Specific Pages in Calibre Interactive](#)

[PEX-Specific Pages in Calibre Interactive](#)

# Database Page in Calibre Interactive

To access: Choose **Settings > Show Pages > Database**

The **Database** page has settings for precision, resolution, and additional layout and source input files.

## Objects

Object	Description
<b>Units</b>	
Precision	<p>Specifies the ratio of database units to user units.</p> <p>Enter one floating-point value specifying the precision, or two positive integers, where <i>int_1/int_2</i> is the precision.</p> <p>Rule file statement: Precision</p>
Override layout value	<p>Specifies whether to override the layout database precision.</p> <p>This option is only visible if the Precision option is checked.</p> <p>Rule file statement: Layout Input Exception Severity PRECISION_RULE_FILE 1</p>
Resolution	<p>Specifies the layout grid step-size (the number of database units on which original geometry is required to be aligned).</p> <p>Enter a positive integer to specify an equal grid step size in the x and y directions.</p> <p>Enter two positive integers to specify the x and y grid step size, respectively.</p> <p>Rule file statement: Resolution</p>
<b>Library</b>	
Additional Layout Files	<p>Specify layout input files in addition to what is specified on the Inputs page.</p> <p>The additional layout files are added to the Layout Path statement.</p>
Additional Golden Layout Files	<p>Specify additional layout files that are to be specified as golden.</p> <p>See the GOLDEN keyword with the Layout Path statement.</p>
Additional SPICE Files	<p>Specify SPICE input files in addition to what is specified on the Inputs page.</p> <p>The additional source files are added to the Source Path statement.</p>

Object	Description
Additional VERILOG Files	<p>Specify VERILOG input files in addition to what is specified on the Inputs page.</p> <p>The additional source files are added to the Source Path statement. This option is only available if the source format is VERILOG.</p>
DFM RDB Defaults (DRC only)	Specify options for the DFM Defaults RDB statement.

# OA/LEFDEF and OPENACCESS Pages in Calibre Interactive

To access: Click OA/LEFDEF on the left panel of the GUI when the Layout Format on the Inputs page is set to OPENACCESS or LEFDEF. The page name is OPENACCESS for Calibre Interactive PEX.

You can control database conversion options for third party databases using Calibre Interactive.

## Note

 If the environment variable MGC\_CALIBRE\_DB\_READ\_OPTIONS is defined, the page displays the options set with the environment variable and the GUI controls are not active.

---

## Caution

 Runs started from Calibre Interactive do not use the statement [SVRF FDI Options](#) to set third-party database read options in the rule file. When using Calibre Interactive, the FDI options must be set in the GUI or with MGC\_CALIBRE\_DB\_READ\_OPTIONS.

Make sure database conversion options are set correctly in the GUI or the database may not be converted correctly.

---

## Description

The options on the page are described in the following tables. See “[Database Translation Utilities](#)” in the *Calibre Layout Comparison and Translation Guide* for a more detailed description.

## Fields

**Table 4-2. Read Options**

Option (fdi2gds/fdi2oasis option)	Description
Read	Select objects to read in the dropdown list. The corresponding fdi2gds or fdi2oasis option is given in parentheses. <ul style="list-style-type: none"><li>• Texts (-noText) — Read text objects. The default is to read text objects.</li><li>• Pins (-noPinShape) — Read pin shapes. The default is to read pin shapes.</li><li>• Blockages (-outputBlockages) — Read blockages for LEF/DEF or OpenAccess databases. The default is to not read blockages.</li><li>• Instance Pins (-instPinAsShape) — Read instance pin shapes. The default is to not read instance pin shapes.</li></ul>

**Table 4-2. Read Options (cont.)**

<b>Option (fdi2gds/fdi2oasis option)</b>	<b>Description</b>
Read Net Names as Text (-annotateNets TEXT [ <u>ALL</u>   TOP])	Specifies that annotations of net names are added as text objects placed on a shape object.  The default is to add net name annotations in all cells. Specify TOP to add annotations in the top-level cell only.  This option is off by default.
Read Net Names as Property (-annotateNets PROPERTY { <i>pname</i>   <i>pnum</i> })	Specifies that annotations of net names are added as properties <sup>1</sup> on the shape object.
Read Pin Names As Text (-annotatePins TEXT)	Specifies that annotations of pin names are added as text objects placed on a pin object.  This option is off by default.
Read Pin Names As Property (-annotatePins PROPERTY { <i>pname</i>   <i>pnum</i> })	Specifies that annotations of pin names are added as properties <sup>1</sup> on the pin shape.
Read Net Type As Property (-annotateNetType)	Specifies to read net types and annotate them as a property.  This option is off by default.
Read Instance Names As Property (-annotateInsts { <i>pname</i>   <i>pnum</i> })	Specifies to read instance names and annotate them as properties <sup>1</sup> .  This option is off by default.
Abort on Reading Empty PCells (-abortOnEmptyPCell)	Causes the Calibre application to exit when an empty PCell appears in an OpenAccess database.  This option is on by default in the GUI.

<sup>1</sup>The *pname* and *pnum* parameters specify the property name. For fdi2gds the *pnum* is an integer. Property names are limited to integers in GDS. For fdi2oasis the *pname* can be any string.

**Table 4-3. Mapping Files Options**

<b>Option (fdi2gds/fdi2oasis option)</b>	<b>Description</b>
Use Layer Map File (-layerMap)	Specifies the pathname of the layer mapping file to use when the database is read. The layer mapping file maps specific input layers to a user-specified list of output layers and datatypes. By default, the application maps objects with their corresponding layer number and a datatype of 0.

**Table 4-3. Mapping Files Options (cont.)**

Option (fdi2gds/fdi2oasis option)	Description
Use Object Map File (-objectMap)	Specifies the pathname of the object mapping file to use when the database is read. The object mapping file determines what blockage objects to convert, and how they are mapped.
Use Net Properties File (-netProp)	Specifies the pathname of the net properties file to use when the database is read.
Use Cell Map File (-cellMap)	Specifies the pathname of the cell mapping file to use when the database is read. The cell mapping file maps input cell names to user-specified cell names. The contents of the cell mapping file are layout system dependent. By default, input cell names are used throughout the run.
Use Input Exception File (-inputException)	Specifies the pathname to a file that gives exception levels for certain tool behaviors.

**Table 4-4. Additional Files Options**

Option (fdi2gds/fdi2oasis option)	Description
Use Cell GDS Files (-cellGDS)	Specifies that lower-level cells and any instantiated hierarchy should be read from the specified files and directories. Only used for GDS output.

**Table 4-5. Other Options**

Option (fdi2gds/fdi2oasis option)	Description
OpenAccess View Lists (-viewList)	Specifies the order in which to open views for lower-level cells. By default the view list is empty.
Write Log File (-logFile)	Specifies to write a runtime log file for the translation.
Write Map Info File (-mapInfo)	Specifies to write a output file with mapping information. This file shows what mappings the translator applied, including any user-specified mappings.
Write Layer Names File (-layerNames)	Specifies to write an output file with layer mapping information.

**Table 4-6. OpenAccess Environment Options**

<b>Option</b>	<b>Description</b>
Set OA_HOME when CDS_INST_ROOT or CDS_INST_DIR is set	Specifies that OA_HOME is set to <code>\$CDS_INST_DIR/share/oa</code> or <code>\$CDS_INST_ROOT/share/oa</code> if CDS_INST_ROOT or CDS_INST_DIR is defined. CDS_INST_ROOT is used if both variables are set.
Set CALIBRE_READDB_LD_LIBRARY_PATH when CDS_INST_ROOT or CDS_INST_DIR is set	Specifies that <code>\$CDS_INST_ROOT/tools/lib</code> and <code>\$CDS_INST_ROOT/tools/lib/64bit</code> are added to CALIBRE_READDB_LD_LIBRARY_PATH if CDS_INST_ROOT is defined. The corresponding action is taken if CDS_INST_DIR is defined. CDS_INST_ROOT is used if both variables are set.
Set PATH when CDS_INST_ROOT or CDS_INST_DIR is set	Specifies that <code>\$CDS_INST_ROOT/share/oa/bin</code> , <code>\$CDS_INST_ROOT/tools/dfII/bin</code> , and <code>\$CDS_INST_ROOT/tools/bin</code> are added to the PATH variable if CDS_INST_ROOT is defined. The corresponding action is taken if CDS_INST_DIR is defined. CDS_INST_ROOT is used if both variables are set.
Set CDS_EXP_PCELL_DIR when CDS_INST_ROOT or CDS_INST_DIR is set	Specifies that CDS_EXP_PCELL_DIR is set to <code>./expressPcells</code> if CDS_INST_ROOT or CDS_INST_DIR is defined.
Set CDS_EXP_ALL_PARAMS when CDS_INST_ROOT or CDS_INST_DIR is set	Specifies that CDS_EXP_ALL_PARAMS is set to true if CDS_INST_ROOT or CDS_INST_DIR is defined.
Set CDS_ENABLE_EXP_PCELL when CDS_INST_ROOT or CDS_INST_DIR is set	Specifies that CDS_ENABLE_EXP_PCELL is set to TRUE if CDS_INST_ROOT or CDS_INST_DIR is defined.

## Related Topics

[Providing OpenAccess Input with Calibre Interactive](#)

[Providing LEF/DEF Input with Calibre Interactive](#)

# Environment Page in Calibre Interactive

To access: Choose **Settings > Show Pages > Environment**

The **Environment** page has controls to view and define environment variables for the Calibre Interactive run.

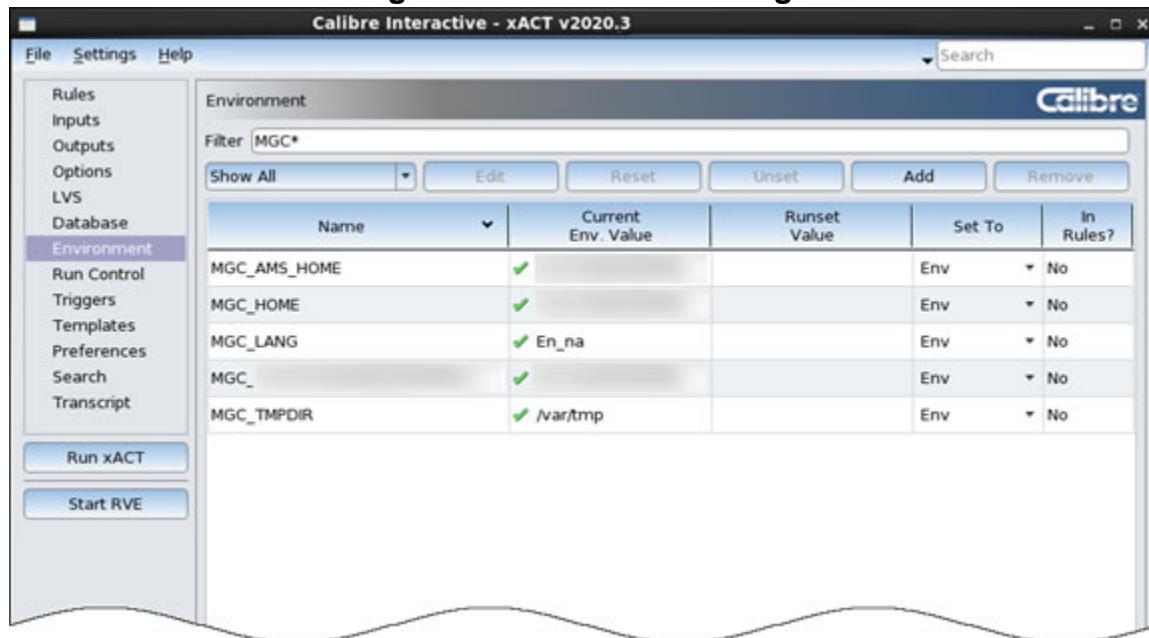
## Description

You can view, edit, and add environment variables using the Environment page. Calibre Interactive maintains a distinction between rule file and runset environment variables, and between shell and runset values, as summarized here:

- **Rule file environment variable** — One referenced in the rule file, such as with a “#IFDEF \$MY\_VAR” statement.
- **Runset environment variable** — One created as part of your Calibre Interactive session.
- **Runset environment value** — The value defined in your Calibre Interactive session.
- **Shell environment value** — The value defined in your shell environment for the environment variable.

You can add an environment variable to have it used for the Calibre run. Such runset environment variables are saved when you save the runset. You can also override the value of a shell environment variable for the run—the new value is saved as a runset environment value and the variable is treated as a runset environment variable.

**Figure 4-1. Environment Page**



## Objects

GUI Object	Description
Filter (text entry)	Specify a wildcard filter to limit the environment variables displayed in the table. Only one filter string may be entered.
Show All Show Runset Only Show Rules File Only (dropdown menu)	<ul style="list-style-type: none"> <li>• <b>Show All</b> — Show shell, runset, and rule file environment variables.</li> <li>• <b>Show Runset Only</b> — Show only runset environment variables. Runset environment variables are displayed in bold font.</li> <li>• <b>Show Rules File Only</b> — Show only environment variables referenced in the rule file.</li> </ul>
Edit (button)	Click to change the selected variable name or value. If you change a shell environment variable it is stored as a runset variable for the run. You can also click in the table cell to edit the entry.
Reset (button)	Click to set the selected shell environment variable back to the value defined in the shell.
Unset (button)	Click to unset the selected environment variable for the run.
Add (button)	Click to add a runset environment variable. A new row is added to the table with the variable name ENV_VAR_<i>.
	<b>Tip:</b> Make sure the Filter entry is * so that the new variable name is not filtered out of the table.
Remove (button)	Click to remove the selected runset environment variable. You cannot remove rule file environment variables.
Table	The table displays the environment variables for the run, subject to the Filter entry and the <b>Show ...</b> dropdown selection.

# Run Control Page in Calibre Interactive

To access: Click **Run Control** in the left panel of Calibre Interactive (shortcut: Alt-c)

You set the host processor, run, and licensing options on the **Run Control** page. You can also set options to connect to a design tool and specify whether to open Calibre RVE after the run.

## Objects

Category	Description
Run Calibre On	<p>Settings for the host processor or remote cluster.</p> <ul style="list-style-type: none"><li>• <b>Local Host</b> — Calibre runs on the host currently running Calibre Interactive.</li><li>• <b>Remote Host</b> — Calibre runs on the specified remote host.</li><li>• <b>Other Cluster</b> — Calibre runs on a custom computer cluster. Enter the appropriate command in the Queue Command field. You can use the replaceable parameters listed in the “Parameter Substitution Information” area.</li></ul> <p>Synchronous checkbox:</p> <ul style="list-style-type: none"><li>• Checked — Calibre Interactive waits for the Calibre run to complete. Calibre RVE and reports are displayed when the run is finished if appropriate options are set.</li><li>• Not checked — Calibre Interactive executes the queue command as a background process. The run transcript, Calibre RVE, and reports are not displayed when the run finishes.</li></ul> <ul style="list-style-type: none"><li>• <b>Platform LSF</b> — Calibre runs on the IBM® Spectrum™ LSF® (Load Sharing Facility) distributed computing solution. Specify the command in the “LSF Command” field.</li></ul> <p>Also see the Calibre Licensing and Remote Calibre Environment options when Remote Host or Other Cluster is selected.</p>

Category	Description
Run Calibre	<p>Settings for run mode (single-threaded, multithreaded, or Calibre MTflex) and behavior if licenses are not available for all requested CPUs.</p> <ul style="list-style-type: none"> <li>• <b>Single Thread</b> — Single-threaded execution.</li> <li>• <b>Multiple Threads</b> — Multithreaded execution on a single host computer. Specify the number of CPUs to use in the entry field. (-turbo num_processors)</li> <li>• <b>All Threads</b> — Multithreaded execution on a single host computer with all available processors. (-turbo)</li> <li>• <b>Distributed (MTflex)</b> — Use your existing computer network as a distributed computing platform with Calibre® MTflex™. Specify the remote hosts file, which is used as the parameter for the -remotefile command line argument.</li> </ul> <p>If desired, specify Remote Data Server (RDS) and the number of servers. See “<a href="#">Remote Data Server</a>” in the <i>Calibre Administrator’s Guide</i>.</p>
Run Calibre LVS Compare On Remote Host (LVS only)	<p>Settings for running LVS compare on a remote host. This option is available when the run mode is not set to Single Thread. Requires a Calibre nmLVS Advanced license.</p> <ul style="list-style-type: none"> <li>• <b>Specify Host</b> — Specify the host with the host name</li> <li>• <b>Specify File</b> — Specify the host with the remote host file</li> </ul>

Category	Description
RVE Options	<ul style="list-style-type: none"><li>• <b>Show Results in RVE</b> — Automatically open the results database in Calibre RVE when the run finishes.</li><li>• <b>Start RVE as soon as results are available</b> — As described. (DRC only)</li><li>• <b>Do not open empty main results database in RVE</b> — (DRC only) When enabled, an empty main results database (RDB) is not opened if at least one side (auxiliary) RDB will open automatically. If no side RDBs are specified to open automatically, or if all side RDBs are empty, then an empty main RDB is opened. Side RDBs can be specified to open automatically with the Calibre Interactive setting “Side RDBs” on the <b>Outputs</b> page; see “<a href="#">Outputs Page in Calibre Interactive nmDRC</a>” on page 158. This option is not available if “Start RVE as soon as results are available” is checked.</li><li>• <b>Start RVE after batch run</b> — Automatically open Calibre RVE after a batch run finishes.</li><li>• <b>Run RVE on local host</b> — When checked, Calibre RVE runs on the local host. When unchecked, Calibre RVE runs on the host specified for “Run Calibre On”.</li><li>• <b>Close RVE before running Calibre &lt;app&gt;</b> — As described.</li><li>• <b>Close RVE when Calibre Interactive closes</b> — As described.</li></ul>
Design Tool Settings	Settings for connecting to a layout or schematic design tool, including the socket number. See “ <a href="#">Connecting to a Design Tool in Calibre Interactive</a> ” on page 64.
PEX Steps (PEX only)	<ul style="list-style-type: none"><li>• <b>Create the PHDB</b> — Creates the PHDB with Calibre xRC if PEX Netlist specifies the use of layout names (LAYOUTNAMES). Creates the PHDB with Calibre nmLVS if PEX Netlist specifies the use of source names (SOURCENAMES, SOURCEBASED, or SCHEMATICONLY).</li><li>• <b>Create the PDB</b> — Runs the PDB step.</li><li>• <b>Create the parasitic netlist</b> — Runs the formatter step.</li></ul>
Calibre Licensing	Settings for the licensing mode. See “ <a href="#">Setting Licensing Options in Calibre Interactive</a> ” on page 445. The “Remote Calibre” section is available to set remote licensing options when “Run Calibre on” is set to Remote Host, Platform LSF, or Other Cluster.

Category	Description
Calibre Interactive License Timeout	<p>Settings the for license timeout.</p> <p>The license timeout can also be set with the environment variable MGC_CALGUI_RELEASE_LICENSE_TIME; the default unit is hours. See “<a href="#">License Timeout in Calibre Interactive</a>” on page 60 for instructions on using other units.</p>
Remote Calibre Environment	<p>Specify the value of MGC_HOME, MGC_LIB_PATH, the remote shell, and the remote username. See “<a href="#">Configuring the Remote Environment</a>” on page 443.</p> <p>This section is available when “Run Calibre on” is set to Remote Host, Platform LSF, or Other Cluster.</p>

# Triggers Page in Calibre Interactive

To access: Choose **Settings > Show Pages > Triggers**

The **Triggers** page defines trigger functions that execute before and after the batch Calibre run. Trigger functions can run in the shell environment or in the design tool environment. Trigger definitions are saved in the runset. You can also specify trigger functions in the custom configuration file.

## Description

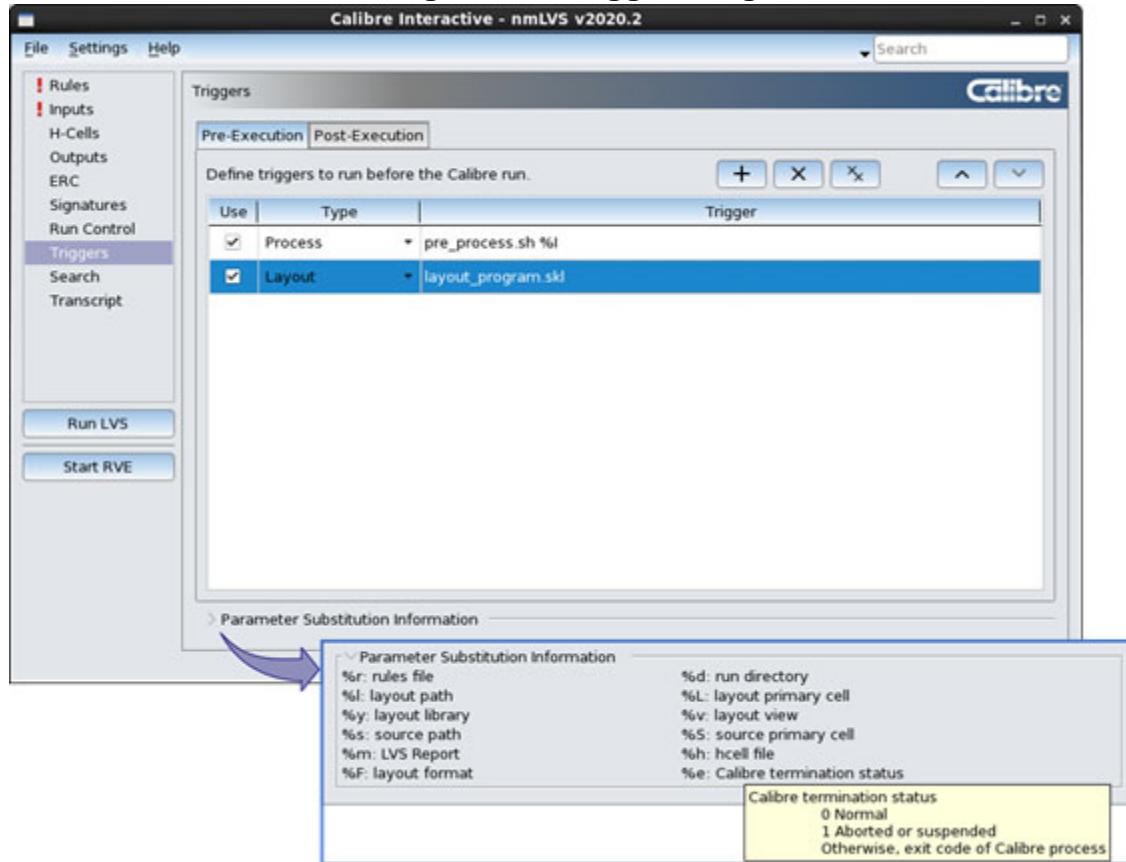
The transcript prints the following lines when a pre- or post-execution trigger is executed:

```
INFO: Running ... pre-trigger: <trigger_function>
```

```
INFO: Running ... post-trigger: <trigger function>
```

These lines are displayed in the warnings and errors section of the Transcript page as type “Info.”

**Figure 4-2. Triggers Page**



## Objects

Option	Control Type	Description
Pre-Execution Post-Execution	Tabs	<p>Pre-execution triggers run before the Calibre application executes; post-execution triggers run after the Calibre application finishes. Each tab contains the list of trigger functions for that internal trigger.</p> <p> <b>Note:</b> Calibre Interactive PEX and xACT have additional pre- and post-trigger functions. See “<a href="#">Internal Trigger Execution in Calibre Interactive</a>” on page 415.</p>
	Buttons	<p>The buttons take the following actions:</p> <ul style="list-style-type: none"> <li>• Add trigger at end of list</li> <li>• Remove selected trigger</li> <li>• Remove all triggers</li> <li>• Move selected trigger up in list</li> <li>• Move selected trigger down in list</li> </ul>
Use	Table column	Execute or not execute the trigger function.
Type	Table column	<ul style="list-style-type: none"> <li>• <b>Process</b> — Execute in the shell.</li> <li>• <b>Layout</b> — Execute in the layout design tool.</li> <li>• <b>Schematic</b> — Execute in the schematic design tool.</li> </ul>
Trigger	Table column	The trigger function, including arguments. The arguments listed in the Parameter Substitution area can be used.
Parameter Substitution	GUI area	The listed parameters are used to pass various filenames and other values specified in the GUI to the trigger program. For example, %L is replaced with the name of the layout primary cell and passed to the trigger program. See “ <a href="#">Trigger Parameters in Calibre Interactive</a> ” on page 390.

## Related Topics

[Configuration File Commands for Trigger Functions](#)

[Configuration File Format for Calibre Interactive GUI Customization](#)

# Templates Page in Calibre Interactive

To access: Choose **Settings > Show Pages > Templates**

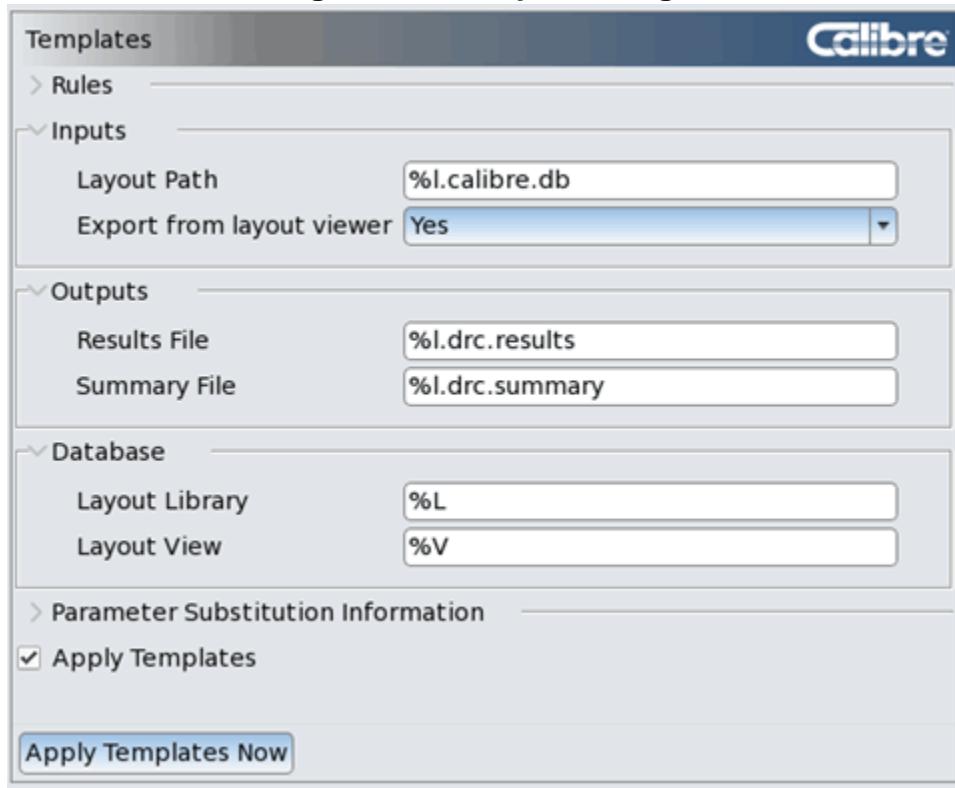
The **Templates** page has settings for file-naming templates, which form the names that Calibre Interactive generates. For example, you can specify input and output filenames based on the open cell in a connected design tool.

---

## Note

-  Templates are automatically applied if you open Calibre Interactive from a design tool.  
Conceptually, templates are the same as in classic Calibre Interactive; see “[Templates for File Naming](#)” on page 100.
- 

**Figure 4-3. Templates Page**



## Objects

Object	Object Type	Description
Rules	Expandable area	Templates for the rule file, run directory, and transcript file.

Object	Object Type	Description
Inputs	Expandable area	Templates for input files. See “ <a href="#">Export from Viewer Preferences</a> ” on page 103 for an explanation of the “Export from layout viewer” and “Export from source viewer” template settings.
Outputs	Expandable area	Templates for output files.
Database	Expandable area	Templates related to the source and layout databases, such as the library and view names.
Parameter Substitution Information	Expandable area	Lists the replaceable parameters that can be used in templates. These replaceable parameters are the same as those used in classic Calibre Interactive.
Apply Templates	Checkbox	Specifies whether to apply templates: <ul style="list-style-type: none"> <li>• <b>On</b> — Apply templates if a design tool is connected to the GUI session. Templates are not applied if there is no design tool connection.</li> <li>• <b>Off</b> — Do not apply templates.</li> </ul> When “On” is selected and a sunset is loaded, templates are applied at the end of the sunset loading process.
Apply Templates Now	Button	Applies the templates to the current session. The templates are applied regardless of whether a design tool is connected.

# Preferences Page in Calibre Interactive

To access: Choose **Settings > Show Pages > Preferences**

The **Preferences** page has settings for loading and saving the runset, saving the transcript, loading a custom configuration file, and other settings.

## Objects

Object	Description
Startup	Specify the startup behavior for loading a runset.
Recipes (DRC and LVS/ERC)	Specify check recipe files to load at startup. The check recipes are listed on the Recipe Editor page, in the USER RECIPES section of the Recipe dropdown list.   <b>Tip:</b> Check recipes are saved to file with the <b>Export</b> button in the Recipe Editor.
Runset	See “ <a href="#">Runset Preferences</a> ” on page 127.
Rules	See “ <a href="#">Rules Preferences</a> ” on page 128.
Run (PEX)	The following option is included for Calibre Interactive PEX: <ul style="list-style-type: none"><li>• Stop parasitic extraction if LVS indicates mismatch</li></ul>
Transcript	Specify whether to save the transcript to file.
Errors and Warnings	Specify whether to save errors and warnings to file, the file name, and the save mode (replace or append).
Configuration	<ul style="list-style-type: none"><li>• <b>Configuration</b> — (checkbox) Specify whether to load configuration files on invocation.</li><li>• <b>Configuration files</b> — Specify one or more configuration files.</li><li>• <b>Load Configuration Files</b> — (button) Click to load the specified configuration file(s) in the current GUI session.</li></ul> <p>The configuration files are saved to the runset. If the “Configuration” checkbox is checked, the configuration files are loaded the next time you load the runset.</p>  <b>Note:</b> Configuration files specified on the command line or with an environment variable have higher precedence. If either is specified, configuration files specified in the runset are not loaded and the <b>Load Configuration Files</b> is not present.
Disable	Disables certain GUI options. See “ <a href="#">Disable Preferences</a> ” on page 130.
Misc	See “ <a href="#">Miscellaneous Preferences</a> ” on page 131.

## Runset Preferences

To access: Click **Preferences** on the left panel of Calibre Interactive , then expand the Runset area. If **Preferences** is not visible in the left panel, choose **Settings > Show Pages > Preferences**.

The runset preferences in Calibre Interactive control the runset open, close, and save behavior, and runset tagging information.

### Objects

- **Type**

Specify a runset type.

- **Info**

Specify descriptive information.

- **On Close Runset**

- Prompt to save changes before closing runset — Display a prompt dialog when a modified runset is about to be closed.
- Save changed settings before closing runset — Automatically save a modified runset before closing it.
- Do not save changed settings — Do not save a modified runset and do not display a prompt dialog to save a modified runset.

- **Save runset each time Calibre is run**

Save the runset automatically each time Calibre is run.

- **Save runset to run directory each time Calibre is run**

When enabled, each time Calibre is run a copy of the runset is saved to the run directory specified on the Rules page.

If you loaded multiple runsets and this setting is enabled, the runset is saved as *runset.autosaved*. If you also enabled “Save changed settings before closing runset,” the runset saved on closing a runset is automatically named *runset.autosaved*. If one of the multiple runsets loaded is named *runset.autosaved*, that runset is overwritten.

- **Run Calibre when runset opened**

Start a run immediately after a runset is opened.

## Rules Preferences

To access: Click **Preferences** on the left panel of Calibre Interactive , then expand the Rules area. If **Preferences** is not visible in the left panel, choose **Settings > Show Pages > Preferences**.

The Rules preferences in Calibre Interactive control behavior for reading, saving, and loading the rule file.

### Objects

- **Save TVF Rules to SVRF before running Calibre**

Enables saving of TVF rules to an SVRF file.

- **Specify control file name**

Specifies a control filename. This value is stored in the runset.

- **Write control file suitable for command line run**

If enabled, the #IFDEF \$MGC\_CALIBRE\_INTERACTIVE line is omitted before the DRC ICSTATION YES line in the control file, along with the corresponding #ENDIF. A control file saved with this option enabled can be used in a command line Calibre run. (Default: true)

See “[Running Batch Calibre with a Calibre Interactive Control File](#)” on page 46.

If the layout database format is OpenAccess, you also need to set database translation options with an environment variable; this is described in the task reference in the preceding paragraph.

- **Only compile the rules file when Run button is pressed**

Prevents Calibre invocation when you click the **Run** button. Instead, only the rule file compiler is invoked and any compilation errors are displayed in a dialog box. If compilation succeeds, a message to that effect is printed in the Transcript page. This setting is saved to the runset.

If “Save TVF Rules to SVRF before running Calibre” is checked, the SVRF rule file is generated.

- **Always compile the rules file when Run button is pressed**

If enabled, the rule file is compiled in a separate step before starting the Calibre run. If compilation fails, the error is displayed in a pop-up message and the Calibre run is not started.

By default, rule file compilation occurs as part of the Calibre run and compilation errors are displayed in the Transcript page. When this option is enabled, compilation occurs twice for a successful run.

- **Read Inputs/Outputs fields when loading rules file**

If enabled, input and output statements from the rule file populate the Inputs and Outputs pages when a rule file is loaded with the  button.

- **Display text of included rules files in transcript**

Displays contents of included rule files in the transcript when Calibre is run.

This setting takes precedence over the environment variable  
CALIBRE\_ECHO\_RULE\_FILE.

- **Automatically load rules file**

When “Automatically load rules file” is enabled, Calibre Interactive automatically loads the rule file at these times:

- When you load a runset with a rule file definition
- When you start a Calibre run and the rule file has been modified since it was last loaded

**Show prompt** — Enable or disable displaying a confirmation dialog before automatically loading the rule file. If “Show prompt” is disabled, the setting for “Automatically load rules file” applies during Calibre Interactive batch operation.

- **Include additional rules files and statements after main rules file**

If enabled, additional rules files and rule file statements are placed after the rule file specified on the Rules page. This is useful if the order of rule file statements matters, which can happen if the main rule file includes conditional compilation directives, such as #DEFINE, which can impact the compilation of statements found in an included file. This option is off by default.

Additional rule files and rule statements are specified on the Options page with the options “Include Rules Files” and “Include Rules Statements.”

## Disable Preferences

To access: Choose **Settings > Show Pages > Preferences** and expand the Disable area.

You can disable several Calibre Interactive options on the **Preferences** page. The available disable options depend on the run type.

### Objects

- **DRC Runs**

- Disable check selection
    - Disables changing the DRC Check Selection Recipe.
  - Disable check selection recipe editing
    - Disables editing of custom check selection recipes.

- **Fast XOR Runs**

- Disable Fast XOR max diff count control
    - Disables the “Max Diff” control on **Fast XOR** page; this option is on by default, meaning the control is not displayed. Caution should be used when using this option. The “Max Diff” control adds the -maxdiff argument for DBdiff execution with FastXOR; see “[FastXOR Database Comparison](#)” in the *Calibre Layout Comparison and Translation Guide* for more information.
  - Disable Fast XOR options
    - Disables the Options area on the **Fast XOR** page. This disable preference is on by default, meaning the options are not displayed. See “[Options Area](#)” on page 154 with the documentation of the **Fast XOR** page.
  - Disable Fast XOR generated rules file custom name
    - Disables the option to specify a custom name for the generated rule file in a FastXOR run. The “Rules File” option is in the Rule File Generation area on the **Fast XOR** page. This disable preference is on by default, meaning the option is not displayed. See “[Rule File Generation Area](#)” on page 153.
  - Disable Fast XOR layer map and base layers combined usage
    - Disables simultaneous specification of the Use Layer Map File and Use Base Layers options in the Mapping Files area on the **Fast XOR** page. These options are available when “Create XOR rules file” is checked. This disable preference is on by default, meaning these options are mutually exclusive.

Clear this checkbox if you want to specify both Use Layer Map File and Use Base Layers.

## Miscellaneous Preferences

To access: Choose **Settings > Show Pages > Preferences** and expand the Misc area.

The miscellaneous (Misc) preferences in Calibre Interactive control warnings and prompts, automatic behaviors, and other options.

### Objects

- **Warn before overwriting layout database**

Display a warning before overwriting the system-generated layout database. This occurs when the layout GDSII or OASIS®<sup>1</sup> database is extracted from the layout viewer.  
(Default: true)

- **Warn before overwriting source database**

Display a warning before overwriting the source SPICE netlist. This occurs when the source SPICE netlist is extracted from the schematic viewer. (Default: true)

- **Verify checks are selected before running DRC**

Verify that at least one check is selected before starting a DRC run.

- **Check source netlist is complete before running <app>**

Verify that the source netlist is complete before starting the run; the source netlist check is case insensitive.

Only available when the run uses a source netlist.

- **Warn when library layout file format differs from primary layout format**

When enabled, Calibre Interactive displays a warning at runtime if additional layout files on the **Database** page in the “Library” options are not the same format as the layout file on the **Inputs** page.

- **Ignore layout and source export in batch mode**

This setting is enabled by default, causing batch runs of Calibre Interactive to ignore the settings for “Export from layout viewer” and “Export from schematic viewer” on the Inputs page.

- **SPICE instance prefix**

Specifies the instance prefix that the netlister adds to each instance in the schematic. Calibre Interactive adds this prefix to instance names obtained by clicking in the schematic.

- **Specify waiver setup file name**

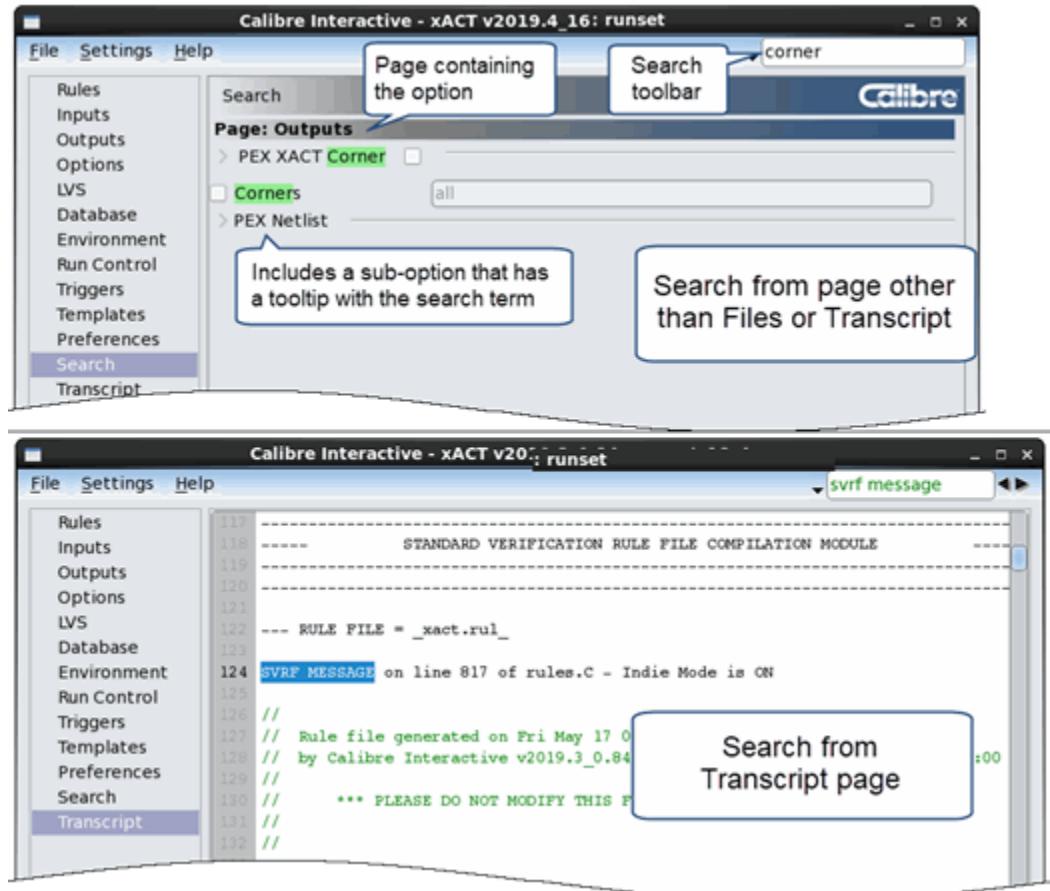
When enabled, the waiver setup file generated by Calibre Interactive during waiver creation or during a waiver run is saved to the specified filename. By default, the generated waiver setup file is named `_waiver_setup_`. (DRC and LVS only)

---

1. OASIS® is a registered trademark of Thomas Grebinski and licensed for use to SEMI®, San Jose. SEMI® is a registered trademark of Semiconductor Equipment and Materials International.

## Search Toolbar in Calibre Interactive

To access: Enter a search term in the entry field in top right corner of the GUI. (shortcut: Ctrl-f)  
The Search toolbar finds matches to a search string. The behavior depends on the GUI page that is active. You can also enter runset option names to find the corresponding control.



## Description

Active Page	Behavior
Page other than Files or Transcript	<p>Finds matches to a search string and places the GUI control in the Search page. Matches are also made to text in tooltips. You can make changes to the settings, if desired. Settings made in the Search page are reflected when you view the control in its original page.</p> <p>If a match is found in a sub-option, the top-level option and all its sub-options are displayed.</p> <p>The search examines all pages except Transcript and Files. Only visible pages and options are searched. For example, if the Inductance page is not visible, matches to “inductance” on the Inductance page are not found.</p>
Transcript or Files	Searches in only the active page and highlights matches.

## Objects

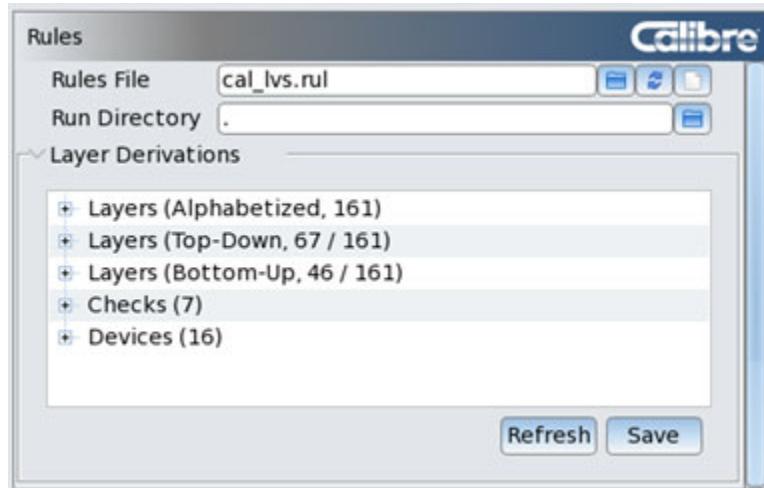
Object	Description
 Search	<p>Enter the search term. Click the dropdown button to select case-sensitive search.</p> <p>You can enter runset option names to find the corresponding control, including classic Calibre Interactive runset options.</p>
Search page	Contains search results for pages other than the Files or Transcript page.

## Layer Derivations Tree

To access: Click **Rules** on the left panel to display the Rules page, specify a rule file, and expand the Layer Derivations area.

The layer derivation tree displays the layers found in the rule file and the sequence of derivations that lead up to a particular derived layer. The layer derivation tree also displays the checks and device definitions found in the rule file and the corresponding layer derivations. The layer derivation tree is available for all Calibre Interactive applications.

**Figure 4-4. Layer Derivations Tree**



## Objects

**Table 4-7. Derivation Tree Listings**

Derivation	Description
Layers (Alphabetized, $n_{layers}$ )	Lists all layers alphabetically, where $n_{layers}$ is the total number of layers. Each derived layer can be expanded to see the full derivation.
Layers (Top-Down, $n_{top} / n_{layers}$ )	Lists all original and derived layers at the top of the derivation tree; layers at the top of the derivation tree are not used in further layer derivations.
Layers (Bottom-Up, $n_{bottom} / n_{layers}$ )	Lists all layers at the bottom of the derivation tree; such layers are not the result of any layer derivation.
Checks ( $n_{checks}$ )	Lists all rule checks and the derived layer(s) that result from the check. Only present for run types that include checks.
Devices ( $n_{devices}$ )	Lists the devices that are defined in the rule file, along with the layers that are part of the device definition.

**Table 4-8. Layer Derivation Buttons**

Button	Description
Refresh	Collapse the layer derivation tree.
Save	Save the layer derivation to a text file.

## Usage Notes

The layer derivation tree uses the following conventions:

- Original Layers — The layer derivation is displayed as follows:  
 $\text{“orig\_layer”} = \text{OR “orig\_layer”} = \text{layer\_number}$
- Inside Cell derivations — Some layer derivations, such as Inside Cell, are processed by the Calibre database constructor, which is not accessed by Calibre Interactive. As a result, such layers are displayed with a convention similar to that for original layers:  
 $\text{“layer\_name”} = \text{“orig\_layer” INSIDE CELL “cell”} = \text{layer\_number}$
-  — The layer has connectivity.
- *rule\_check::<num>* — Indicates the derived layer output from the check *rule\_check*. Some rules have more than one output layer, and this is indicated by *num*.
- *rule\_check::layer* — Indicates that *layer* is derived within the check *rule\_check*.
- DEVICE ... (*rule\_file:line\_num*) — Gives the device definition, ending with the file and line number for the definition in the rule file.



# Chapter 5

## Calibre Interactive nmDRC

---

You can specify inputs, outputs, and options for a Calibre nmDRC run using Calibre Interactive nmDRC.

<b>Setting Up a Calibre Interactive nmDRC Run . . . . .</b>	<b>137</b>
<b>Specifying the Checks to Execute in a Calibre Interactive nmDRC Run . . . . .</b>	<b>140</b>
<b>Running with Area DRC . . . . .</b>	<b>141</b>
<b>Creating a Check Text Override (CTO) File With Calibre Interactive nmDRC . . . . .</b>	<b>143</b>
<b>Creating a DRC HTML Report from Calibre Interactive nmDRC . . . . .</b>	<b>145</b>
<b>Reporting DRC Results in Local Cell Space or Top Cell Space . . . . .</b>	<b>148</b>
<b>Limiting the DRC Results Count . . . . .</b>	<b>148</b>
<b>FastXOR Database Comparison in Calibre Interactive . . . . .</b>	<b>149</b>
Running FastXOR Database Comparison in Calibre Interactive . . . . .	149
Fast XOR Page . . . . .	152
<b>DRC-Specific Pages in Calibre Interactive . . . . .</b>	<b>156</b>
Options Page in Calibre Interactive nmDRC . . . . .	157
Outputs Page in Calibre Interactive nmDRC . . . . .	158

## Setting Up a Calibre Interactive nmDRC Run

You can control inputs, outputs, and options for Calibre nmDRC run using Calibre Interactive.

### Prerequisites

- A SVRF rule file.
- A layout database.
- Calibre Interactive nmDRC is open. See “[Invoking the Calibre Interactive GUI](#)” on page 58
- See “[Calibre Interactive GUI Reference](#)” on page 105 for information on setting the options on the Environment, Run Control, Triggers, Preferences, and Templates pages.

### Procedure

1. (Optional) Click **File > Load Runset** to load an existing runset.

You can load a classic Calibre Interactive runset; see “[Converting Classic Calibre Interactive Runsets](#)” on page 37.

2. Click **Rules** on the left panel.

- Specify the rule file and run directory.

If desired, click the **Load** button () to load rule file settings into the GUI.

To specify additional rule files, enable “Include Rules Files” on the Options page.

---

**Tip**

 The setting “Read Inputs/Outputs fields while loading rules file” controls whether or not rule file settings populate the GUI options on the Inputs and Outputs pages when the **Load** button is clicked. This setting is enabled by default; go to the Rules section on the **Preferences** page (**Settings > Show Pages > Preferences**) to change the setting.

---

- Specify the check selection recipe.

To create or edit check a selection recipe, click the **Edit** button to open the **Recipe Editor** page. The Recipe Editor is similar to the Check Selection Recipe Editor in Calibre Interactive nmDRC.

3. Click **Inputs** on the left panel and specify the following:

- Specify the Run option:

- **Hierarchical** — Run Calibre nmDRC in hierarchical mode, with the -hier command line option.
- **Flat** — Run Calibre nmDRC in flat mode, without the -hier command line option.
- **Calibre DB** — Run Calibre nmDRC using the Calibre Cell/Block license package. This license package is only used for flat runs.
- **Autowaiver Creation** — Generate a waiver shape database for the Calibre Auto-Waivers flow. This selection adds the Waivers page to the left panel. Select the page and provide the required input.
- **Fast XOR** — Run Calibre Fast XOR. See “[Running FastXOR Database Comparison in Calibre Interactive](#)” on page 149.

- (Optional) For hierarchical runs, you can run in other modes:

- **Recon** — Runs with the -recon command line option. Calibre nmDRC Reconnaissance mode automatically selects a subset of DRC rule checks that identify gross errors early in the design and assembly process.
- **Recon Inverse** — Runs with the “-recon inverse” command line option. This mode runs the subset of checks that Calibre nmDRC Recon mode does not run.

Check recipe selection is disabled with Calibre nmDRC Recon and Calibre nmDRC Recon Inverse mode. See “[Chip-Level Verification](#)” in the *Calibre Solutions for Physical Verification* manual.

- **Analyze** — Runs with the -analyze command line option. Creates a DFM database containing error distribution results data for gross analysis of areas in the design that require attention.

When “Analyze” is enabled, all options in the **Outputs** page are disabled and the page is hidden. See “[Reviewing DRC Analyze Results](#)” in the *Calibre Solutions for Physical Verification* manual.

- c. Enter the “Layout Path” information.

Select “Export from layout viewer” to stream the layout from a connected design tool.

- d. (Optional) To run with Calibre Auto-Waivers, check the “Use Waivers when running DRC” option in the Waivers section.

The **Waivers** page is added to the left panel. Select the page and fill in the required information; see “[Calibre Auto-Waivers in Calibre Interactive](#)” on page 161.

The “Use Waivers when running DRC” option is not displayed if the run type is “Autowaiver Creation”.

- e. (Optional) To run DRC on a selected area of the layout, check “Area DRC” and provide the required input. See “[Running with Area DRC](#)” on page 141.

4. (Optional) To specify additional layout files, click **Database** on the left panel, expand the “Library” area, and provide the information.

5. Click **Outputs** on the left panel and select options as needed.

Use the tooltips to view the SVRF statements and keywords corresponding to a particular option.

Also see “[Outputs Page in Calibre Interactive nmDRC](#)” on page 158.

6. Click **Options** on the left panel and select desired options.

7. Click **Run Control** on the left panel.

- a. Expand the “RVE Options” area to select options related to opening Calibre RVE.
- b. Set run control and licensing options; see “[Run Control Page in Calibre Interactive](#)” on page 118.

8. Click **Run DRC** to start the run. The button label is **Run Autowaiver Creation** for the “Autowaiver creation” run type.

## Results

The run information is displayed in the Transcript page. The DRC Summary Report is displayed in the Files page if “View DRC Summary Report after run finishes” is checked on the Outputs page.

Errors and warnings are listed in the bottom portion of the Transcript page. Right-click and select **Save As** to save the transcript.

## Related Topics

[DRC-Specific Pages in Calibre Interactive](#)

[Database Page in Calibre Interactive](#)

[Options Page in Calibre Interactive nmDRC](#)

# Specifying the Checks to Execute in a Calibre Interactive nmDRC Run

The Check Selection Recipe specifies the checks that are executed in a Calibre Interactive nmDRC run. You can use a built-in check recipe or define your own check recipe.

## Prerequisites

- The prerequisites listed in “[Setting Up a Calibre Interactive nmDRC Run](#)” on page 137 are met.
- “Disable check selection” is not enabled in the Disable section of the **Preferences** page ([Settings > Show Pages > Preferences](#)).

## Procedure

1. Click **Rules** to display the Rules pane.
2. Specify a rule file; see “[Setting Up a Calibre Interactive nmDRC Run](#)” on page 137 for additional details and a view of the Rules pane.
3. Choose a check recipe using the dropdown list for **Check Selection Recipe**.

### BUILT-IN RECIPES

- **Checks selected in the rules file** — (Default) Run only checks selected in the rule file. This is the only built-in recipe that obeys [DRC Select Check](#) and DRC Unselect Check statements in the rule file.

Note that if the rule file does not include any DRC Select Check statements, then all checks are executed; this is the same behavior as for batch Calibre nmDRC.

- **All checks** — Run *all* checks in the rule file, except those excluded by pre-processor directives. DRC [Un]Select Check statements are *not* considered.

## USER RECIPES

- Select a previously defined custom check recipe.

See “[Basic Editing of Custom Check Selection Recipes](#)” on page 84 for information on creating a Check Selection Recipe.

## Results

The checks specified in the check recipe are executed during the Calibre nmDRC run.

# Running with Area DRC

You can run DRC checks on only a specified rectangular area of the layout. Area DRC checks the specified area plus a *halo* region around the area. You can specify the halo size and whether results in the halo region are filtered out or kept.

## Prerequisites

- Calibre Interactive nmDRC is open and set up for a run. See “[Setting Up a Calibre Interactive nmDRC Run](#)” on page 137.

## Procedure

1. Click **Inputs** on the left panel.
2. Enable the Area DRC option.
3. Select the Layout Window with one of the following methods:
  - Enter space-separated coordinates for the rectangular window you want to check. Coordinates are the lower-left and upper-right corners of the rectangle, in that order. You must use this method if not connected to a layout viewer.
  - If connected to a layout viewer, click the layout button () to select the area you want to check in the layout viewer. Click and drag out an area selection rectangle in the layout viewer. The coordinates of the area selection rectangle appear in the Layout Window field.

4. (Optional) Set area DRC options, such as halo size and coordinate scaling, as described in the following table:

**Table 5-1. Area DRC Options**

Option	Description
Automatic Halo	<p>Specifies how to determine the size of the halo which expands the area DRC window.</p> <ul style="list-style-type: none"><li>• <b>Yes</b> — Width of halo is the greater of the width or height of the specified Area DRC region.</li><li>• <b>No</b> — Width of halo is an explicit user-specified width in user-units.</li></ul> <p>Note: Depending on the geometry of your area selection, the automatic setting can result in a halo region that is much larger than needed, resulting in slow performance.</p>
Remove results from halo region after area DRC  (Only applies to the ASCII format results database.)	<p>Enabled — (default). As described.</p> <p>Disabled — Keep results in the halo region.</p>
Apply coordinate scaling	<p>Enabled — (default) Scale window coordinates by: (DRC Results Database Precision/Precision) * (Layout Magnify)</p> <p>Disabled — Do not apply scaling.</p>

5. Click **Run DRC** to start the run.

## Results

The following changes in output occur when Area DRC is enabled:

- **Results database** — Results in the halo region are filtered out based on the setting “Remove results from halo region after area DRC.” Result filtering only applies to ASCII format results databases.
- **Auxiliary databases** — If an auxiliary results database is specified in the rule file, the results are filtered as specified for the main results database.
- **Transcript and summary report output** — If the setting “Remove results from halo region after area DRC” is enabled, then both the transcript and the DRC Summary Report report the number of results that were filtered out of the halo region.

For example, the following line appears in both the transcript and the summary report:

```
TOTAL DRC Results Generated:      422 (783), FILTERED 397,  
FINAL Results Generated = 25
```

# Creating a Check Text Override (CTO) File With Calibre Interactive nmDRC

Calibre Interactive nmDRC can create a Check Text Override file to control layer visibility for highlight actions and to define categories for filtering by category. The CTO file is used by Calibre RVE for DRC and DRC HTML reporting. The automatically generated CTO file causes only the layers used in the rule check to be displayed when highlighting a result. Category definitions are optional.

The CTO file includes RVE Show Layers statements based on the rule checks in your DRC rule file. If your design tool uses a layer map file to control the layout export to GDS, the layer map file can be specified so that the layout viewer layer names are used in the CTO file.

[Layer](#) and [Layer Map](#) statements in the rule file are considered when creating the RVE Show Layers command in the CTO file. Restrictions apply if the Layer Map statement has an open-ended range for the layer or datatype.

## Restrictions and Limitations

- **With a Layer Map File**

If a layer map file is used, the following must be true in order for the layer to be included in the RVE Show Layers statement:

- Each layer must be listed in the layer map file.
- Layer Map statements in the rule file must have closed ranges for the layer and datatype.

If any of the above conditions is not met when using a layer map file, the corresponding layer is not used in the RVE Show Layers statement.

When using a layer map file, if there are Layer statements in your rule file without a corresponding Layer Map statement, all layer:purpose pairs mapped to the corresponding GDS layer in the layer map file are used in the RVE Show Layers statement for that layer.

- **Without a Layer Map File**

If a layer map file is *not* used, Layer Map statements in the rule file that have open ended ranges for the layer or datatype are ignored and a warning is issued. The corresponding Layer statement alone is used to determine the layer number for the RVE Show Layers statement.

## Prerequisites

- A Calibre nmDRC rule file.

- (Optional) A layer map file in one of the following formats:
  - Calibre DESIGNrev layerprops file — See “[layerprops File Format](#)” in the *Calibre DESIGNrev Layout Viewer User’s Manual*
  - Third-party layer map file format:
    - One entry per line with the following format:

```
viewer_layer_name viewer_purpose_name GDS_layer_num GDS_datatype
```

- ASCII text file.
- Comment lines start with #.

The Cadence Innovus layer map file (*streamOut.map*) is not supported.

Use the layer map file if you want the RVE Show Layers statements in the CTO file to use layer and purpose pairs or layer names rather than layer numbers. The layer map file is also used when generating the Layers category definition; see Step 5.

## Procedure

1. Open Calibre Interactive—nmDRC.
  2. Specify the rule file and other settings for the run.
  3. Click **Outputs** on the left panel.
  4. For DRC Results Database, select ASCII as the Format.
  5. Enable “Create CTO file” and specify options:
    - Viewer Layer Map File — Specify a layer map file if you want design tool layer and purpose pairs to be used in the RVE Show Layers statements rather than the GDS or OASIS layer numbers.
 

See the “Prerequisites” section for the format of the layer map file and the “Restrictions and Limitations” section for additional information.
    - Generate Categories from — Select Groups, Layers, and/or Recipes to include the selected categories in the CTO file. The category definitions are used to filter by category in Calibre RVE for DRC. See “Filtering DRC Results by Category” in the *Calibre RVE User’s Manual*.
  6. Click **Run DRC**.
  7. Examine the resulting CTO file. If you received any warnings, examine the layers and rules checks affected by the warnings and correct the CTO file if necessary.
- If you are using a layer map file and there are Layer statements in your rule file without a corresponding Layer Map statement, all layer:purpose pairs mapped to the corresponding GDS layer in the layer map file are used in the RVE Show Layers statement for that layer.

## Results

A CTO file with RVE Show Layers statements for each rule check is created. The format of the layer parameter in the RVE Show Layers statement is as follows:

- **Layer map file is used** — The layer is given as *layer:purpose*.
- **Layer map file not used** — The layer is given as *layer*, where *layer* is the layer number, or *layer.datatype* if Layer Map statements in the rule file provide a datatype.

Results with and without a layer map file are given in the “Examples” section.

If a results database with the name *results\_rdb* is open in Calibre RVE and the created CTO file is named *results\_rdb.cto* and in the same directory, then the results database is reloaded and the CTO file is applied.

Also see the examples included with “Creating a Check Text Override (CTO) File Using Calibre RVE” in the *Calibre RVE User’s Manual*.

# Creating a DRC HTML Report from Calibre Interactive nmDRC

You can have Calibre Interactive nmDRC automatically create a DRC HTML Report after the Calibre run is finished. You can use a pre-defined configuration file or specify your own configuration file for the report format. Highlight images are automatically included. A shell script is created so that you can run the report at a later time from the command line.

DRC HTML report creation is only possible if the input layout format is GDS or OASIS.

## Prerequisites

- Calibre Interactive nmDRC is open. See “[Setting Up a Calibre Interactive nmDRC Run](#)” on page 137.
- The input layout format is GDS or OASIS.
- (Optional) A Calibre DESIGNrev layer properties file named *<layout>.layerprops* where *<layout>* is the name of the layout specified on the Inputs page. The file *<layout>.layerprops* is used automatically if it exists.
- (Optional) A DRC RVE Check Text Override (CTO) file named *<drc\_db>.cto*, where *<drc\_db>* is the filename of the DRC results database. The file *<drc\_db>.cto* should exist in the same directory as the results database. The file is read automatically during the creation of the DRC HTML report.

The CTO file is used to set layer visibility, highlight color, and other options for the rule checks in the results database.

**Tip**

 If you do not have an existing CTO file, the following procedure describes how to create one automatically with Calibre Interactive.

- (Optional) Set the environment variable MGC\_RVE\_HTML\_BROWSER to the path for a browser executable to specify the HTML browser that is opened when viewing the DRC HTML Report. A recent browser version is recommended, as unexpected results may occur in older browsers. Your default browser is used if this environment variable is not set.

**Note**

 See the *Calibre RVE User's Manual* for information on DRC HTML Reporting and the CTO file.

**Procedure**

1. Click the **Outputs** button on the left panel of Calibre Interactive nmDRC.
2. Enable “Create HTML Report.”
3. (Optional) Choose options for the DRC HTML report. The default options may be satisfactory.
  - a. Choose the report configuration with the Template dropdown list:

**Table 5-2. Template Options for DRC HTML Report Creation**

Template	Description
Cell-Check	Group the results tree by Cell/Check.
Check-Cell-SC	Group the results tree by Check/Cell/#SC, where #SC is the Shape Class calculated property.  Only one result per shape class is shown for each check/cell combination; this is to reduce the size of the report.
Check-Cell	Group the results tree by Check/Cell.
Check-SC	Group the results tree by Check/#SC, where #SC is the Shape Class calculated property.  Only one result per shape class is shown for each check/cell combination; this is to reduce the size of the report.
Custom Template	Provide a custom DRC HTML Report configuration file; the file is specified in the Custom Template field, which appears when “Custom Template” is selected in the dropdown list.

- b. Choose the output format with the Format dropdown list.

The Format option is not available if “Custom Template” is selected.

**Table 5-3. Format Options for DRC HTML Report Creation**

Format	Description
HTML	Standard HTML format.
MHTML	Single file HTML format

- c. Specify the output directory in “Output Directory” field. The directory is created if it does not exist.
- d. (Optional) Enable “View Report” to have the report opened automatically after the Calibre run finishes.
4. (Optional) Do the following if you do not have an existing CTO file and want highlight images to show only the layers used in each rule check.
  - a. Enable “Create CTO file,” which is also on the Outputs page.
  - b. (Optional) To use layer names rather than layer numbers in the CTO file, enable “Viewer Layer Map File” and specify a file.

See “[Creating a Check Text Override \(CTO\) File With Calibre Interactive nmDRC](#)” on page 143 for information on the format of the layer map file.

When “Create CTO file” is enabled, Calibre Interactive automatically creates a CTO file with RVE Show Layers statements based on the rule file. The CTO file is used during HTML report creation. (Category definitions in the CTO file are not used.)
5. Make other Calibre Interactive nmDRC settings for the run.
6. Click **Run DRC**.

## Results

The report is created in the output directory after the Calibre run finishes. The report is named *index.htm* for HTML formatted reports and *report.mht* for MHTML formatted reports. The report is opened automatically if “View Report” was enabled in Step 3.d.

The settings on the Run Control page are used when creating the DRC HTML report.

The tool also performs the following actions:

- Creates a *report.ini* configuration file in the output directory.
- Creates *rdbSpec.txt* (DRC HTML Report Database Map file) in the output directory if layout snapshots were included in the report.
- Creates a shell script in the run directory. You can execute the shell script to recreate the HTML report at any time. The shell script is named *createReport.sh* if the leaf name of output directory is *report* (the default) or *createReport\_rptDir.sh* otherwise, where

*rptDir* is the leaf name of the output directory. If the “View Report” option was enabled in Step 3.d, you can include the “-open” switch when running the script to automatically open the report in a web browser after it has been created.

## Reporting DRC Results in Local Cell Space or Top Cell Space

You can choose to report DRC results in local cell coordinates or top cell space. By default Calibre Interactive nmDRC reports errors in local cell space. This option is for Calibre nmDRC-H only.

### Procedure

1. Click the **Outputs** button on the left panel.
2. In the DRC Results Database section, enable or disable “Output cell errors in cell space”:
  - **Enabled** — Use local cell space coordinates when reporting errors; this is the default for Calibre Interactive nmDRC. This option is required for [Calibre Auto-Waivers](#). The following statement is included in the Calibre Interactive control file:

DRC Cell Name YES CELL SPACE XFORM

- **Disabled** — Use top cell space coordinates when reporting errors. The following statement is included in the Calibre Interactive control file:

DRC Cell Name NO

---

#### Note

 The default in command line Calibre runs is [DRC Cell Name NO](#).

---

## Limiting the DRC Results Count

Use this option to control the number of results reported for each DRC rule check.

### Procedure

1. Click DRC Options.
2. Enable the “Maximum Result Count” section, and specify the “Upper Limit” option:
  - **All** — Report all results. Choose this option if you are using Calibre nmDRC to output a modified version of your layout as mask data
  - **Number** — Enter the maximum allowed result count.

Corresponding rule file statement: [DRC Maximum Results](#)

# FastXOR Database Comparison in Calibre Interactive

---

You can perform a Calibre FastXOR layout comparison using Calibre Interactive.

Calibre FastXOR uses internal DBdiff processing to decrease Layout Versus Layout (LVL) XOR comparison run times. FastXOR often runs five times faster than a normal dual-database XOR run. See “[FastXOR Database Comparison](#)” in the *Calibre Layout Comparison and Translation Guide* for general information.

Running FastXOR Database Comparison in Calibre Interactive.....	149
Fast XOR Page.....	152

## Running FastXOR Database Comparison in Calibre Interactive

You can start a Calibre FastXOR run from Calibre Interactive nmDRC. Calibre FastXOR is targeted for comparison between design iterations and can complete XOR comparison many times faster than traditional XOR flows.

Calibre FastXOR uses pre-processing by DBdiff to analyze two layout databases to determine the objects that differ between the two databases. Then, Calibre nmDRC-H is used to perform XOR operations on just the objects and layers that differ as determined by DBdiff.

### Prerequisites

- Two complete GDSII, OASIS, LEF/DEF, or OpenAccess layout databases to be compared. The input layouts should not use 64-bit coordinates—the layouts should use 32-bit coordinate space.

If the layout format is OpenAccess or you are comparing two different LEF files, then a layer mapping file is needed. See “[FastXOR Layer Mapping and Comparison](#)” in the *Calibre Layout Comparison and Translation Guide* for instructions on providing a layer mapping file.

- (Optional) An XOR comparison rule file; this is not required if you instruct Calibre Interactive to generate the required XOR rule file.

The XOR comparison rule file should meet the requirements given in the “[Restrictions](#)” section of the *Calibre Layout Comparison and Translation Guide*.

In addition, the rule file should handle layer mapping correctly so the appropriate layers are compared. For instructions on creating the rule file using DBdiff, see “[Generating the LVL Comparison Rule File](#)” in the *Calibre Layout Comparison and Translation Guide*.

- (Optional) Set DBdiff options for FastXOR using the environment variables CALIBRE\_FX\_DB\_DIFF\_OPTIONS and CALIBRE\_FX\_PASS\_THROUGH\_LAYER\_ENABLE, or the FastXOR template file. See the section “[DBdiff Command Execution in Fast XOR](#)” in the *Calibre Layout Comparison and Translation Guide* for details.

In particular, the -transform option for DBdiff may be useful in some cases. The -transform option specifies transformations to apply to the layout specified on the Inputs page.

## Procedure

1. Click **Inputs** on the left panel to display the Inputs page.
2. Choose a run type of “Fast XOR” in the Run dropdown list.

The Fast XOR page is added to the left panel of the GUI.

3. Specify the rule file with one of the following methods. You can supply a FastXOR rule file or instruct Calibre Interactive to generate the required XOR rule file before the run.

- Have Calibre Interactive generate the required XOR rule file:
  - i. Click **Fast XOR** on the left panel.
  - ii. Check “Create XOR rules file.”

When this setting is enabled, DBdiff creates the rule file *rules.fxor* using the -write\_xor\_rules option. The filename *rules.fxor* is entered as the rule file on the Rules page and the Rules File field is inactive.

  - iii. (Optional) Set output and DBdiff options for rule file generation in the Rule File Generation area of the Fast XOR page. See “[Rule File Generation Area](#)” on page 153.

Recommended: “Use Base Layers” in the **Mapping Files** category. This option adds the Layout Base Layer statement with the specified layers to the generated rule file.

- Specify an existing FastXOR rule file:
  - i. Click **Rules** on the left panel to display the Rules page.
  - ii. Enter the FastXOR comparison rule file in the Rules File entry field. Make sure the rule file meets the requirements given in the “Prerequisites” section.
  - iii. (Optional) Click  to populate the GUI with the settings in the rule file.

*After you load a rule file, any information you specify in the GUI supersedes the information in your loaded rule file.*

4. Click **Rules** on the left panel to display the Rules page, then specify the run directory in the “DRC Run Directory” field.

5. Specify the current (modified) layout as follows:
  - a. Click **Inputs** on the left panel and specify the current layout in the first Layout Path area. Do one of the following, depending on the source of the layout:
    - o Layout is read from file — Disable “Export from layout viewer” and specify the layout format, filename, and top cell name.
    - o Layout is exported from attached viewer — Enable “Export from layout viewer” and verify the filename for the extracted layout database (Layout File field) and the top cell name.

When exporting a layout database from a connected viewer, an intermediate file is extracted and saved to the file named in the Layout File field. This filename is generated automatically based on the Layout Path setting on the **Templates** page.

If the filename is red, the file does not exist and Calibre Interactive creates the file. If the filename is green, the file exists and will be overwritten. Enter another filename if you want to keep the existing file.

---

### **Caution**

 When “Export from layout viewer” is enabled, the file specified in the File field is overwritten if it exists already. Do not enter a tapeout database in the File field, or use the extracted layout named in the File field for tapeout or any other use.

---

- b. Select the database format in the Format dropdown list. Only GDSII, OASIS, LEF/DEF, and OpenAccess formats are allowed.

If the layout format is OpenAccess or you are comparing two different LEF files, then a layer mapping file is needed, as mentioned in the “Prerequisites.”

6. Specify the reference (previous) layout in the second Layout Path area.
7. (Optional) If the layout is in a third-party database format, set database options on the OA/LEFDEF page.
8. (Optional) Check Calibre Interactive file template settings.

Click **Templates** on the left panel and check the file naming templates in each area. In particular, you may want to change the default templates for the output files. (Choose **Settings > Show Pages > Templates** if the Templates page is not visible.)

9. Click **Outputs** on the left panel and check the settings on the Outputs page.
10. Do the following to set additional options for the run, such as the DBdiff report file, template file, and compare text option.
  - a. Click **Preferences** on the left panel. Choose **Settings > Show Pages > Preferences** if the page is not visible.

- b. In the Disable area, uncheck “Disable Fast XOR options.”
  - c. Click **Fast XOR** on the left panel and set options in the Options area. See “[Options Area](#)” on page 154.
11. Set other options as needed.
  12. Click **Run DRC** on the left panel.

## Results

The transcript output from the Calibre run appears in the Transcript window as the run progresses. Calibre RVE for DRC opens when the run finishes if you enabled “Show Results in RVE” on the Run Control page.

If your FastXOR comparison rule file was created using DBdiff, then the following output files are specified in the rule file, where *rules.fxor* is the name of the FastXOR comparison rule file:

- DRC Summary Report file: *rules.fxor.summary*
- ASCII DRC Results Database: *rules.fxor.asc*
- Geometric output database: *rules.fxor.gds*

If you clicked **Load** on the Rules page to load the rule file settings into the GUI and did not later change the output filenames in the GUI, then these are the files created by a FastXOR run when using the default XOR comparison rule file created by DBdiff.

However, if you did not load the rule file settings into the GUI, then the filenames in the GUI for the DRC Summary Report and DRC Results database take precedence over the filenames found in the rule file. If you started Calibre Interactive from a layout editor and are using file naming templates, then the output filenames are formed automatically using the templates.

For more detail about the output from a FastXOR run, see the “Results” section in the topic “[Using FastXOR to Perform LVL](#)” in the *Calibre Layout Comparison and Translation Guide*.

## Related Topics

[Templates Page in Calibre Interactive](#)

[OA/LEFDEF and OPENACCESS Pages in Calibre Interactive](#)

## Fast XOR Page

To display: On the **Inputs** page, select a Run type of Fast XOR.

The Fast XOR page has options related to automatically generating the rule file for a Calibre FastXOR run. The page also has options for the report file, RDB file, and DBdiff template file.

---

### Note

 If an option is set with the environment variable CALIBRE\_FX\_DB\_DIFF\_OPTIONS or CALIBRE\_FX\_PASS\_THROUGH\_LAYER\_ENABLE, the GUI setting reflects the environment variable setting and the GUI control is not active.

If you use an existing FastXOR rule file for the run and include the rule file statement SVRF DBdiff Options to set FastXOR options, those options are not used in a run started from Calibre Interactive. When using Calibre Interactive, the DBdiff options must be set in the GUI or with CALIBRE\_FX\_DB\_DIFF\_OPTIONS.

---

The options present on the Fast XOR page depend on other settings in the GUI.

- [Create XOR rules file](#)
- [Rule File Generation Area](#)
- [Options Area](#)

### Create XOR rules file

**Create XOR rules file** — When enabled, the FastXOR rule file is created automatically by DBdiff using the -write\_xor\_rules option. and the “Rule File Generation” area is displayed on the Fast XOR page. When not enabled, you must specify a FastXOR rule file on the Rules page.

### Rule File Generation Area

To display: Check “Create XOR rules file” on the **Fast XOR** page.

**Table 5-4. Format Options for FastXOR Rule File Generation**

Option	Description
Result Format	Specifies the format of the output database(s). Select multiple formats to output multiple results databases in different formats.
Rules File	Specifies the name of the generated rule file. The default is <i>rules.fxor</i> .  This option is not displayed by default. To display the option, click <b>Preferences</b> on the left panel, expand the Disable area, and clear the checkbox for “Disable Fast XOR generated rules file custom name.” (Choose <b>Settings &gt; Show Pages</b> if the page is not visible)
Output Files Prefix	Specifies a prefix for the created output files. See the “prefix” option in the DBdiff -write_xor_rules syntax.  Note: The prefix specified here does not override output filenames specified on the Outputs page of Calibre Interactive, because GUI filenames take precedence over output filenames specified in the rule file.

**Table 5-4. Format Options for FastXOR Rule File Generation (cont.)**

Option	Description
Add New and Missing shapes rules	Adds checks in the FastXOR rules file which use the NOT operation to identify new and missing shapes.
Common Only	When enabled, only layers that are common to both designs are compared.

**Table 5-5. Mapping Files Options for FastXOR Rule File Generation**

Option	Description
Use Layer Map File	Specifies a layer mapping file. See the -layermap argument for -write_xor_rules option in DBdiff.
Use Base Layers	Specifies layers for the -base_layers argument for the -write_xor_rules option in DBdiff. This option adds the Layout Base Layer statement with the specified layers to the generated rule file. (Recommended)
Use Include Rules File	Specifies a file containing SVRF statements that are imported directly into the generated FastXOR rule file. See the -include_rules argument to DBdiff.
Use Include Layers File	Specifies a file that lists layers to compare. See the -include_layer argument to DBdiff.
Use Exclude Layers File	Specifies a file that lists layers to exclude from comparison. See the -exclude_layer argument to DBdiff.

**Note**

 The Use Layer Map File and Use Base Layers options are mutually exclusive by default. To use both options simultaneously, select **Settings > Show Pages > Preferences** to display the **Preferences** page, expand the Disable area, and clear the checkbox “Disable Fast XOR layer map and base layers combined usage.”

---

## Options Area

To display: Click **Preferences** on the left panel, expand the Disable area, and clear the checkbox for “Disable Fast XOR options.”

**Table 5-6. Options for FastXOR Runs**

Setting	Description (command-line option)
Report File	Specifies the name of the DBdiff report file. (-report)

**Table 5-6. Options for FastXOR Runs (cont.)**

<b>Setting</b>	<b>Description (command-line option)</b>
RDB File	Specifies the name of the RDB output file with optional filter settings. Use the following syntax: <i>filename</i> [FC   FP   FT] [-sortlayer] FC — exclude cell differences FP — exclude polygon differences FT — exclude text differences -sortlayer — generate output on a sorted layer basis (-rdb)
Template File	Specifies a template file that contains command line options for DBdiff. (-template)
Compare Text	Enables text comparison. (-comparetext)
Compare All Placed Cells	Specifies 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. (-compareallplacedcells)
Enable Pass-Through Layers	Sets the environment variable CALIBRE_FX_PASS_THROUGH_LAYER_ENABLE, which improves performance if one or more layers between the compared designs has been added, significantly modified, or removed before a FastXOR comparison. Setting this option instructs FastXOR to pass unique layers directly to the traditional DRC XOR engine, bypassing unnecessary DBdiff operations.

## Related Topics

[Layout Comparison Command Line Syntax \[Calibre Layout Comparison and Translation Guide\]](#)

[FastXOR Database Comparison \[Calibre Layout Comparison and Translation Guide\]](#)

## DRC-Specific Pages in Calibre Interactive

---

Some GUI pages are specific to Calibre Interactive nmDRC.

**Table 5-7. DRC-Specific Pages in Calibre Interactive**

Page	Description
Outputs	Specify settings for the DRC results database, opening side RDBs, and for creating a CTO file, DRC HTML report, and DRC summary report
Fast XOR	Specify options for a Calibre FastXOR layout comparison run. See “ <a href="#">Fast XOR Page</a> ” on page 152.
Recipe Editor	Create and edit check selection recipes. The Recipe Editor page is displayed when you click <b>Edit</b> next to the Recipe entry on the Rules page. See “ <a href="#">Recipe Editor Page</a> ” on page 90.
Waivers	Specify information for the Calibre Auto-Waivers Flow. This page is available if “Use Waivers when running DRC” is checked on the Inputs page or a run type of “Autowaiver creation” is specified. See “ <a href="#">Calibre Auto-Waivers in Calibre Interactive</a> ” on page 161.

## Options Page in Calibre Interactive nmDRC

To access: Click **Options** in the left panel

The Options page includes settings that control the Calibre nmDRC run.

---

### Note

 “[Options Page in Calibre Interactive: Common Options](#)” on page 109 discusses these options that are common to all applications: “Virtual Connect,” “Include Rules Files,” “Include Rule Statements,” and “Include Layout Text File.”

---

### Objects

Object	Description
DRC Maximum Results	Specifies the maximum results count for any given rule check. Rule file statement: <a href="#">DRC Maximum Results</a>
DRC Maximum Vertex	Specifies the maximum vertex count of any polygon written to the DRC results database. Rule file statement: <a href="#">DRC Maximum Vertex</a>
Preserve Cells	Specifies layout cells to protect from most hierarchy modifications, such as automatic expansion. Only used in hierarchical runs. To read cell names from an existing waiver file, click the <b>Import from RVE Waiver File</b> button and selected a <i>.waived</i> file. All the cells that contain waived results are entered into the cell list. You can then delete or add cell names as needed. Rule file statements: <a href="#">Layout Preserve Cell List</a> and <a href="#">Layout Cell List</a>

## Outputs Page in Calibre Interactive nmDRC

To access: Click **Outputs** in the left panel

The Outputs page includes settings for the DRC results database, opening side RDBs, and for creating a CTO file, DRC HTML report, and DRC summary report.

### Objects

Object	Description
DRC Results Database	<p>Specifies the name, format, and related options for the DRC results database.</p> <p>Filenames that end in <code>.gz</code> are compressed using gzip. Filenames that end in <code>.bz2</code> are compressed using pbzip2. If the compression utility (gzip or pbzip2) is not in your path, you are prompted to supply the path.</p> <p>Rule file statements: DRC Results Database, DRC Cell Name, and DRC Check Text</p>
Side RDBs	<p>Specify the side RDBs to open automatically when Calibre RVE is invoked after the run completes.</p> <p>To populate the side RDB table, click the  button on the Rules page next to the rule file entry. Select the checkbox in the Open column to have the RDB opened automatically.</p> <p>The following RDB types are recognized:</p> <ul style="list-style-type: none"><li>• <b>Check Map</b> — DRC Check Map</li><li>• <b>Density</b> — Density operation with the RDB keyword</li><li>• <b>DFM</b> — DFM RDB</li><li>• <b>LIER</b> — Layout Input Exception RDB</li><li>• <b>NAR</b> — Net Area Ratio</li></ul> <p><b>Tip:</b> You can also specify side RDBs to open using Calibre RVE. See “<a href="#">Viewing Auxiliary Databases</a>” in the <i>Calibre RVE User’s Manual</i>.</p>
Create CTO File  (only displayed if the DRC Results Database format is ASCII)	<p>Specifies whether to create a Check Text Override (CTO) file during the run. The CTO file includes RVE Show Layers statements based on the rule checks in the rule file. It can optionally include category information.</p> <p>See “<a href="#">Creating a Check Text Override (CTO) File With Calibre Interactive nmDRC</a>” on page 143.</p>

Object	Description
Create HTML Report	<p>Specifies whether to create a DRC HTML Report of the results.</p> <p><b>Template:</b></p> <ul style="list-style-type: none"> <li>• Cell-Check — Group the results tree by Cell/Check.</li> <li>• Check-Cell-SC — Group the results tree by Check/Cell/#SC, where #SC is the Shape Class calculated property.</li> <li>• Check-Cell — Group the results tree by Check/Cell.</li> <li>• Check-SC — Group the results tree by Check/#SC.</li> <li>• Custom Template — Provide a custom DRC HTML Report configuration file. Specify the file in the Custom Template field which appears when this option is selected.</li> </ul> <p>When #SC (Shape Class) is in the group hierarchy, only one result per shape class is shown for each check/cell combination. This is useful in order to reduce the size of the report.</p> <p><b>Format:</b></p> <ul style="list-style-type: none"> <li>• HTML — Standard HTML format.</li> <li>• MHTML — Single file HTML format.</li> </ul> <p><b>Output Directory:</b> Specifies the directory in which to create the report. The directory is created if it does not exist.</p> <p><b>View Report:</b> Open the report after it is created.</p> <p>See “DRC HTML Reporting” in the <i>Calibre RVE User’s Manual</i>.</p>
Reports	<p>Specifies the name and options for the DRC Summary Report.</p> <p>Rule file statement: DRC Summary Report</p>

Object	Description
Reusable Hierarchical Database	<p>Specifies whether to save or restore an RHDB (Reusable Hierarchical Database) file. An RHDB is useful in some flows to save time by avoiding the overhead of repeatedly constructing a hierarchical database. Specify the RHDB file name and the Mode:</p> <ul style="list-style-type: none"><li>• Save — Saves a proprietary image of the hierarchical database, generated from all input files in the rule file, as an RHDB file. The RHDB file is saved with an embedded checksum. This option disables CTO file generation, HTML report generation, Area DRC, and automatic RVE invocation.</li><li>• Restore — Restores an RHDB file from the specified file. A checksum is generated from all input files and compared with the checksum in the specified file. A mismatch between the checksums generates an error.</li></ul>

# Chapter 6

## Calibre Auto-Waivers in Calibre Interactive

---

When verifying a design, some rule check violations may be waived by the foundry. A waived result is a violation that is considered acceptable for the design and process. Designers often want to eliminate waived errors from their Calibre results so they can focus on the errors that need to be fixed. You can perform many of the tasks related to Calibre Auto-Waivers using Calibre Interactive.

Calibre Auto-Waivers manages waivers by generating waiver shapes that are merged with your design so that waivers can be applied automatically during a verification run. These waivers take into account hierarchical polygon interactions, as well as interactions of multiple waivers that may apply to a single result. See the *Calibre Auto-Waivers User's and Reference Manual* for detailed information.

<b>Introduction to Calibre Auto-Waivers in Calibre Interactive.....</b>	<b>161</b>
<b>Calibre Auto-Waivers Requirements for Calibre Interactive.....</b>	<b>162</b>
<b>Generating Waiver Shapes with Calibre Interactive .....</b>	<b>163</b>
<b>Starting a Waiver Run in Calibre Interactive.....</b>	<b>167</b>

## Introduction to Calibre Auto-Waivers in Calibre Interactive

Calibre Auto-Waivers automates the process of applying waivers during design verification. Calibre Auto-Waivers can be used to waive the following types of results:

- Calibre nmDRC results
- ERC results from a Calibre nmLVS run
- CFA (Critical Feature Analysis) results in Calibre DFM (Design for Manufacturability).

You can use Calibre Interactive to do the following:

- Read in an existing waiver setup file and save a waiver setup file
- Read waiver setup parameters from a runset and save waiver setup parameters to a runset
- Adjust waiver setup parameters
- Run the waiver\_flow tool to create waiver shapes

- Start a verification run with waivers

Since the Calibre Interactive controls for specifying Calibre Auto-Waivers are similar for each of the supported applications, the instructions in this chapter apply to all supported applications.

See the topics in the following table for additional information on Calibre Auto-Waivers.

**Table 6-1. Calibre Auto-Waivers Information Sources**

<b>Section (manual)</b>	<b>Description</b>
“Getting Started With Calibre Auto-Waivers” ( <i>Calibre Auto-Waivers User’s and Reference Manual</i> )	General information and getting started procedures.
“Calibre Auto-Waivers Overview” ( <i>Calibre Auto-Waivers User’s and Reference Manual</i> )	General overview.
“ERC Automatic Waivers” and “DFM Automatic Waivers” ( <i>Calibre Auto-Waivers User’s and Reference Manual</i> )	Summary information for ERC and DFM-CFA waivers.
“Calibre PERC Waiver Flows” ( <i>Calibre PERC User’s Manual</i> )	The PERC waiver process, which is separate from Calibre Auto-Waivers.

## Calibre Auto-Waivers Requirements for Calibre Interactive

In addition to the standard requirements for the Calibre application, there are additional licensing, rule file, and input file requirements when running Calibre Auto-Waivers with Calibre Interactive.

The requirements are as follows:

- A Calibre Auto-Waivers product license, plus other licenses as described in “Requirements for Running Calibre Auto-Waivers” in the *Calibre Auto-Waivers User’s and Reference Manual*.
- A rule file that meets the requirements described in “Requirements for Running Calibre Auto-Waivers” in the *Calibre Auto-Waivers User’s and Reference Manual*.
- A design in a format supported by Calibre Interactive: GDS, OASIS, or LEF/DEF.
- (Optional) A waiver setup file. See “Writing the Waiver Setup File for Waiver Verification” and “Writing the Waiver Setup File for Waiver Generation” in the *Calibre Auto-Waivers User’s and Reference Manual*. Calibre Interactive creates the waiver setup file at runtime based on the current GUI settings. You can read in an existing

waiver setup file to populate the GUI settings, or specify the waiver setup parameters using the GUI.

- The waiver descriptions.
  - A waiver criteria file and a waiver cells file. The waiver criteria file is required for both waiver generation and the waiver run; the waiver cells file is only required for waiver generation. See “[Waiver Description Files](#)” in the *Calibre Auto-Waivers User’s and Reference Manual*.

If you are doing a waiver run, you have an additional option for providing the waiver criteria:

- A waiver geometry file that has waiver criteria embedded in it. In this case “Use criteria from the waiver shape files” must be selected on the Waivers page in the DRC Run area.

The waiver\_flow tool always adds the waiver criteria text object to the waiver shapes. If you are using Calibre RVE to generate the waiver shape database, then “Annotate waiver cells with waiver criteria text” must be enabled. See “[Text Object Annotations](#)” in the *Calibre Auto-Waivers User’s and Reference Manual*.

- The waiver shapes if performing a Calibre verification run with waivers applied. The waiver shapes can be embedded in the design or exist in a separate database.

## Generating Waiver Shapes with Calibre Interactive

You can use Calibre Interactive to generate a waiver shape database for the Calibre Auto-Waivers flow. All the waiver setup parameters are controlled through Calibre Interactive, then the waiver\_flow tool is used to create the waiver database.

All settings on the **Waivers** page are saved to the runset. If you have a runset loaded with desired settings, you can skip to the end of the following procedure.

---

### Tip

 Calibre RVE can also be used to create a waiver database. Calibre RVE is often preferable to using the waiver\_flow tool because individual results can be waived while you examine the result database. Only the waived results are exported to the waiver database. See “[Creating Embedded Waiver Shapes Using Calibre RVE](#)” in the *Calibre Auto-Waivers User’s and Reference Manual* for more information.

---

### Prerequisites

- “[Calibre Auto-Waivers Requirements for Calibre Interactive](#)” on page 162
- You are running Calibre Interactive nmDRC or Calibre Interactive nmLVS with ERC.

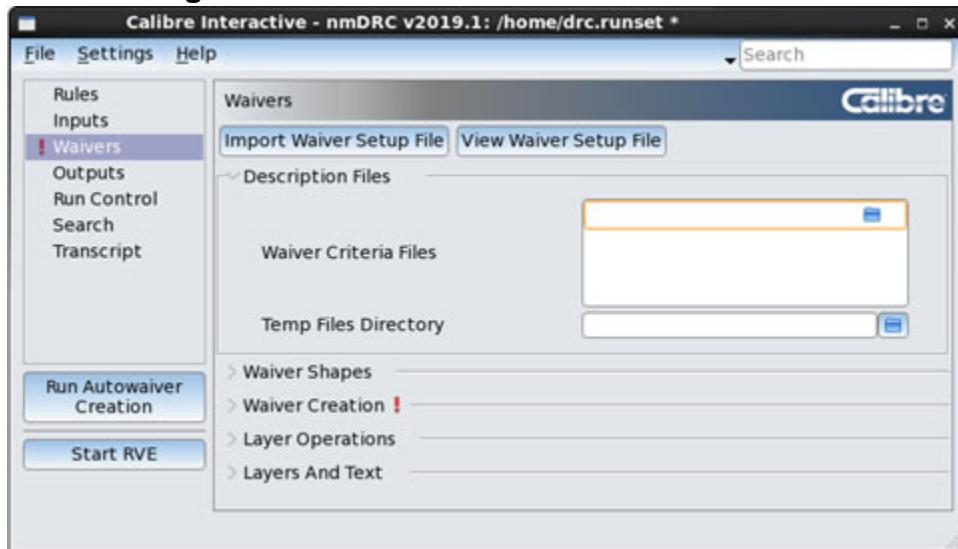
## Procedure

1. Click **Rules** on the left panel of Calibre Interactive and specify the rule file and run directory.
2. Click **Inputs** on the left panel and choose a run type of “Autowaiver Creation” in the Run dropdown list.  
The label of the run button on the left panel changes to **Run Autowaiver Creation** and the Waivers page is added to the left panel.
3. Specify information for the layout database on the Inputs page. The layout format must be GDSII or OASIS for Calibre Auto-Waivers.
4. (Optional) Specify the name of the generated waiver setup file.

A waiver setup file is generated by Calibre Interactive based on the GUI parameters and named `_waiver_setup_` by default. To specify a different file name, view the **Preferences** page, expand the Misc section, enable the option “Specify waiver setup file name,” and specify the file name.

5. Click **Waivers** on the left panel of the GUI. A view of the **Waivers** tab is shown in the following figure for Calibre Interactive nmDRC.

**Figure 6-1. Waivers Tab for Waiver Creation**



6. (Optional) If you have a waiver setup file already created and want to use it for waiver shape generation, click the **Import Waiver Setup File** button and select the file. This populates the GUI fields with the values in the selected waiver setup file; only GUI settings specified in the imported file are overwritten.

Use the following steps to confirm that the imported waiver setup parameters are correct. The waiver setup parameter that corresponds to the GUI control is given in parentheses.

7. (Optional) Provide information in the Description Files area:
  - a. If you have non-default waiver criteria, specify the Waiver Criteria File(s). Default waiver criteria is used if you do not specify a waiver criteria file. (WAIVER\_CRITERIA)

See “[Waiver Criteria File Format](#)” in the *Calibre Auto-Waivers User’s and Reference Manual* for complete information.

7. (Optional) Specify the directory for temporary files in the Temp Files Directory field. The default is the working directory. (TMP\_DIR)

8. Expand the Waiver Shapes area and specify the waiver shape file.

8. a. Specify a merged or standalone waiver database:
  - o **Merge waivers with existing design ON** — Embed waiver shapes in the layout specified on the Inputs page. (MERGE YES)
  - o **Merge waivers with existing design OFF** — Create a waiver database that contains only waiver geometry. (MERGE NO)
8. b. Specify the name of the output waiver database file in the Waiver Shapes File field. This entry is optional—the default output waiver database filename is *waived.gds*. (WAIVER\_DATABASE)
8. c. (Optional) Select the setting for “Use rule file precision”:
  - o **Enabled** — (default) Specifies PRECISION\_CONVERSION YES in the waiver setup file.
  - o **Disabled** — Specifies PRECISION\_CONVERSION NO in the waiver setup file.

This setting controls how waiver shape precision and magnification is handled if the rule file precision differs from the layout database precision; it has no effect if the precisions are the same. See “[Waiver Cell Precision and Magnification Considerations](#)” in the *Calibre Auto-Waivers User’s and Reference Manual* for details. (PRECISION\_CONVERSION)

9. Expand the Waiver Creation area and specify settings for the following options:

- **Waiver Cell File(s)** — Specify one or more Waiver Cell files.

See “[Waiver Cells File Format](#)” in the *Calibre Auto-Waivers User’s and Reference Manual* for complete information. (WAIVER\_CELLS)

To create a new waiver cells file using the GUI, enter a filename, and supply entries in the table at the end of the Waiver Creation area. If + and x icons are not visible for adding a removing table rows, click Show Expanded View.

- **Summary File** — Specify a name for the waiver summary report file. The summary file is always created and named *waiver.summary* by default if you do not specify a filename. (WAIVER\_SUMMARY)
  - **Text Magnification** — Specify a magnification factor for texts written during waiver creation (*waiver\_flow*). Text magnification only applies when the output format is GDS. The default is 0.005. (TEXT\_MAG)
  - **Magnify Text Spacing** — Magnify the spacing between text written during waiver creation; this only applies when the output format is GDS. The magnification level specified for the “Text Magnification” option is used; the default of 0.005 is used if “Text Magnification” is not specified. (MAGNIFY\_TEXT\_SPACING)
  - **Place waiver cells in intermediate cells** — Place waiver cells in intermediate cells instead of the parent cell. See the section “[Intermediate Container Cells](#)” in the *Calibre Auto-Waivers User’s and Reference Manual*.  
(ADD\_WAIVER\_HIERARCHY)
10. Expand the Layer Operations area, and specify active and inactive layer operations. (NOT\_IGNORE and IGNORE)
- Also see the section “[SVRF Command Support](#)” in the *Calibre Auto-Waivers User’s and Reference Manual*.
11. Expand the Layers and Text area and specify layer numbers and datatypes for the waiver shapes and text annotations, if desired.
- See “[Waiver Setup File Format for Waiver Generation](#)” in the *Calibre Auto-Waivers User’s and Reference Manual* for a description of the options and their defaults; also see “[Text Object Annotations](#)” in the same manual.
12. (Optional) Click the **View Waiver Setup File** button to view the waiver setup parameters. You can choose **File > Save Text File As** to save the file.
13. (Optional) Choose **File > Save Runset** to save the runset.
14. Click the **Create Autowaivers** button on the left panel to create new waivers.
- This step runs the [waiver\\_flow](#) tool with the specified parameters.

## Results

The following files are created:

- **Waiver setup file** — A waiver setup file is generated by Calibre Interactive based on the GUI parameters and named *\_waiver\_setup\_* unless you specified a different name in Step 4.

The RULE\_FILE, WORKING\_DIRECTORY, INPUT\_LIBRARY, and LAYOUT\_SYSTEM parameters are taken from the GUI settings on the Rules and Inputs pages.

- **Waiver shape file** — The filename is given in the Waiver Shapes area of the Waivers page, or *waived.gds* by default.
- **Waiver summary file** — The filename is given in the Waiver Creation area, or *waiver.summary* by default.

See “[Generating Waivers with the waiver\\_flow Tool](#)” and “[Waiver Cell Description](#)” in the *Calibre Auto-Waivers User’s and Reference Manual* for complete information.

## Starting a Waiver Run in Calibre Interactive

You can use Calibre Interactive to start a Calibre verification run with Calibre Auto-Waivers. The waiver setup parameters are specified on the **Waivers** page. The waiver run adds the “-waiver” option to the Calibre command line.

All settings on the **Waivers** page are saved to the runset. If you have a runset loaded with desired settings, you can skip to the end of the following procedure.

### Prerequisites

- “[Calibre Auto-Waivers Requirements for Calibre Interactive](#)” on page 162.
- You are running Calibre Interactive nmDRC or Calibre Interactive nmLVS with ERC.
- If you are using waivers from multiple sources, then all waiver shapes must be on the same layer/datatype. This may not be the case if waivers are provided by different IP vendors and a foundry, for example. For additional requirements and instructions for handling this situation, see “[Performing a DRC Waiver Run Using Multiple Waiver Criteria Files](#)” in the *Calibre Auto-Waivers User’s and Reference Manual*.

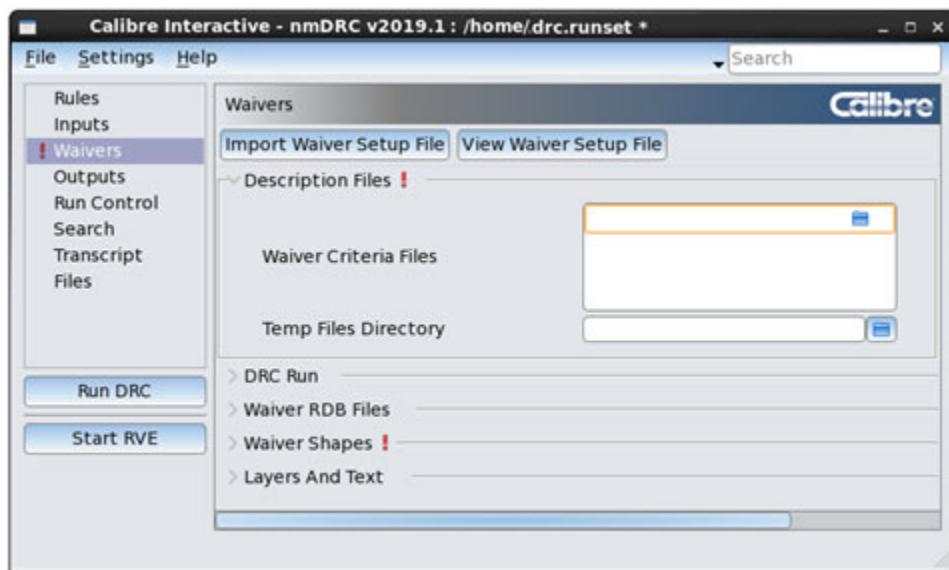
### Procedure

1. Click **Rules** on the left panel of Calibre Interactive and specify the rule file and run directory.
2. Click **Inputs** on the left panel of Calibre Interactive. Specify information for the layout database. The layout format must be GDSII or OASIS for Calibre Auto-Waivers.
3. Expand the Waivers area on the Inputs page and enable “Use Waivers when running <app>.”  
This adds the Waivers page to the left panel of the GUI.
4. (Optional) Specify the name of the generated waiver setup file.  
A waiver setup file is generated by Calibre Interactive based on the GUI parameters and named *\_waiver\_setup\_* by default. To specify a different file name, view the **Preferences** page, expand the Misc section, enable the option “Specify waiver setup file name,” and specify the file name.
5. Click **Waivers** on the left panel of the GUI.

6. (Optional) If you have a waiver setup file already created and want to use it for the waiver run, click the **Import Waiver Setup File** button. This populates the GUI fields with the values in the selected waiver setup file. Only GUI settings specified in the imported file are overwritten.

Use the following steps to confirm that the imported setup parameters are correct.

7. Enter settings in the Description Files area.



- a. Specify the “Waiver Criteria File(s).”

It is not necessary to specify the Waiver Criteria file if you specify “Use criteria from the waiver shape files” in the DRC Run area in Step 8.

See “[Waiver Criteria File Format](#)” in the *Calibre Auto-Waivers User’s and Reference Manual* for complete information. (WAIVER\_CRITERIA)

- b. (Optional) Specify the directory for temporary files in the Temp Files Directory field. The default is the working directory. (TMP\_DIR)

8. Expand the DRC Run area and specify run options as desired:

The corresponding parameter in the waiver setup file is given in parentheses.

- **Use criteria from the waiver criteria files** — As described. (WAIVER\_CRITERIA *file*)
- **Use criteria from the waiver shape files** — Extract waiver criteria from the waiver shape file. The specified waiver criteria file is overwritten or created if it does not exist. (WAIVER\_CRITERIA EXTRACT *file*)

**Note**

 If both “Use criteria from the waiver criteria files” and “Use criteria from the waiver shape files” are enabled, then APPEND is used instead of EXTRACT in the WAIVER\_CRITERIA specification.

---

- **Verify waiver cell checksums** — Verifies that the contents of the waiver cells match the checksum. (RUN\_LAYOUT\_CHECKSUM)
- **Verify Calibre version** — Verifies the Calibre version in waiver cells. (RUN\_CALIBRE\_VERSION\_CHECKSUM)
- **Retain edge and error result types** — Retain unwaived error and edge results from a waiver run instead of converting them to polygons. (RETAIN\_RESULT\_TYPE)
- **Verify IP\_MATCH checksums** — As described. (RUN\_IP\_MATCH)
- **Verify rule checksums** — Use rule checksums to verify that the rule that produced the waiver shape has not changed. (RUN\_RULE\_CHECKSUM)
- **Ignore waiving density operation** — Do not apply waivers to rules with density checks. (IGNORE\_DENSITY)
- **Enable dynamic results reporting** — Enables viewing of waived results in Calibre RVE as soon as they are available. This option only applies to Calibre nmDRC runs. (ENABLE\_DYNAMIC\_RESULTS\_REPORTING)

This option is available only when “Show Results in RVE” and “Start RVE as soon as results are available” are enabled.

9. Expand the Waiver RDB Files area and specify settings as desired to control the output of RDB files. (WAIVER\_RDB, UNUSED\_WAIVER\_RDB, USED\_WAIVER\_RDB, DENSITY\_WAIVER\_RDB)

Select “Enable waiver cell reporting” to add the cell names as an extra property in generated waiver RDB files. This option corresponds to ENABLE\_WAIVER\_CELL\_REPORTING, and is disabled by default.

The option “Ignore waiver statistics in summary file” corresponds to OPTIMIZE\_NONE\_WAIVER\_STATISTICS. The option is available if any of the waived, unused, or waived RDB options are set to NONE.

10. (Optional) Expand the Waiver Shapes area and specify the waiver database information.

A waiver shape file is not needed if *all* waiver shapes are embedded in the main layout. A waiver shape file is needed if at least some waiver shapes exist in separate waiver shape files and if “Use criteria from the waiver shape files” was specified in the DRC Run area in Step 8.

- Waiver Database — Specifies the source of waiver shapes:
  - BOTH** — Use waivers both from the waiver database and from the input layout. (MERGE NO)
  - EMBEDDED** — Use waivers only from the input layout. (MERGE YES)
  - EXTERNAL** — Use waivers only from the waiver database. (MERGE IGNORE\_DESIGN\_WAIVERS)
- Waiver Shape Files — Specify the waiver shape file and check the checkbox to use the file in the run. (WAIVER\_DATABASE)

11. (Optional) Select the setting for “Use rule file precision” (in the Waiver Shapes area):

- **Enabled** — (default) Specifies PRECISION\_CONVERSION YES in the waiver setup file.
- **Disabled** — Specifies PRECISION\_CONVERSION NO in the waiver setup file.

This setting controls how waiver shape precision and magnification is handled if the rule file precision differs from the layout database precision; it has no effect if the precisions are the same. See “[Waiver Cell Precision and Magnification Considerations](#)” in the *Calibre Auto-Waivers User’s and Reference Manual* for details. (PRECISION\_CONVERSION).

12. Expand the Layers and Text area and specify settings as desired.

These settings control the layer and datatype for the waiver shapes and text annotations. During a waiver run these settings specify the layer and datatype pairs that contain the waiver shapes and text annotations.

See “[Waiver Setup File Format for Waiver Verification](#)” in the *Calibre Auto-Waivers User’s and Reference Manual* for a description of the options and their defaults; also see “[Text Object Annotations](#)” in the same manual.

13. (Optional) Click the **View Waiver Setup File** button to view the waiver setup parameters. You can choose **File > Save Text File As** to save the file.

14. Make other settings as desired.

---

**Note**

 In Calibre Interactive nmDRC, the option “Start RVE as soon as results are available” has no effect if Calibre Auto-Waivers is enabled.

---

15. (Optional) Choose **File > Save Runset** to save the runset.

16. Click **Run DRC** when you are ready to start your Calibre run.

**Note**

 When waiver shapes are saved in a separate file rather than embedded in the design, it is possible for the precision of the waiver database to be different than the precision of the layout database. Calibre adjusts the magnification automatically if this happens, as described in the section “[Waiver Cell Precision and Magnification Considerations](#)” in the *Calibre Auto-Waivers User's Manual*.

---

## Results

In addition to the output files specified on the Outputs page, the following files are created or modified during a waiver run:

- **Waiver setup file** — A waiver setup file is generated by Calibre Interactive based on the GUI parameters and named `_waiver_setup_` unless you specified a different name in Step 4.  
The parameter `WORKING_DIRECTORY` is set to the run directory specified on the Rules page of Calibre Interactive.
- **Waiver criteria file** — If “Use criteria from the waiver shape files” is specified on the Waivers page, then the waiver criteria file is created or modified based on the waiver criteria included in text objects in the waiver shape file(s).
- *calibre-waiver.summary* — Contains runtime messages and results statistics.

The presence of the following RDB files depends on the settings on in the Waiver RDB Files area on the Waivers page:

- *waived.rdb*
- *unused\_waiver.rdb*
- *used\_waiver.rdb*
- *density\_waived.rdb*

See these sections in the *Calibre Auto-Waivers User's Manual* for a complete description of the output from a waiver verification run:

- “Results” section in “[Performing a Waiver Run](#)”
- “[Calibre Auto-Waivers File Usage](#)”



# Chapter 7

## Calibre Interactive nmLVS

---

You can specify inputs, outputs, and options for a Calibre nmLVS run using Calibre Interactive nmLVS.

<b>Setting Up a Calibre Interactive nmLVS Run .....</b>	<b>173</b>
<b>Supplying Hcell Data.....</b>	<b>176</b>
<b>Executing Electrical Rule Check (ERC) Operations in Calibre Interactive nmLVS ..</b>	<b>177</b>
<b>Running Short Isolation in Calibre Interactive nmLVS .....</b>	<b>178</b>
<b>Changing Trace Property Statements .....</b>	<b>179</b>
<b>Adding and Changing LVS Box Statements in Calibre Interactive .....</b>	<b>181</b>
<b>Performing Hcell Analysis in Calibre Interactive .....</b>	<b>184</b>
Creating and Analyzing an H-Cell List With Calibre Interactive .....	184
H-Cells Analysis Results .....	186
<b>Device Signatures: Using and Creating in Calibre Interactive nmLVS.....</b>	<b>187</b>
Generating Device Signatures Using Calibre Interactive .....	187
Using a Device Signature File in a Calibre Interactive nmLVS Run .....	190
Edit Parameters for Device Dialog Box in Calibre Interactive.....	191
Layers File for Device Signature Creation With Calibre Interactive .....	194
<b>LVS-Specific Pages in Calibre Interactive .....</b>	<b>197</b>

## Setting Up a Calibre Interactive nmLVS Run

You can control inputs, outputs, and options for a Calibre nmLVS run using Calibre Interactive.

### Prerequisites

- Calibre Interactive nmLVS is open. See “[Invoking the Calibre Interactive GUI](#)” on page 58.
- See “[Calibre Interactive GUI Reference](#)” on page 105 for information on setting the options on the Environment, Run Control, Triggers, Preferences, and Templates pages.

---

#### Note

 If a page name is not visible on the left panel, choose **Settings > Show Pages** and select the page name. Also make sure any options that enable the page are set.

---

### Procedure

1. (Optional) Click **File > Load Runset** to load an existing runset.

You can load a classic Calibre Interactive runset; see “[Converting Classic Calibre Interactive Runsets](#)” on page 37.

2. Click **Rules** on the left panel and specify the rule file and run directory.

If desired, click the **Load** button () to load rule file settings into the GUI.

To specify additional rule files, enable “Include Rules Files” on the Options page. To specify additional rule file statements, enable “Include Rule Statements” on the Options page.

3. Click **Inputs** on the left panel and specify the following:

- a. Specify the “Run” option:

Run Type	Description
Hierarchical	Run in hierarchical mode.
Flat	Run in flat mode. The -flatten command line option is used.
Recon	Run Calibre nmLVS Reconnaissance Short Isolation. The -si command line option is used. Several options not used when running Calibre nmLVS Recon SI are hidden, and the “Step” option changes to “Mode”: <ul style="list-style-type: none"> <li>• <b>All Shorts</b> — Perform both IO and Power Ground short isolation</li> <li>• <b>Power Ground Shorts</b> — Perform only Power Ground short isolation</li> <li>• <b>IO Shorts</b> — Perform only IO short isolation</li> </ul>
Calibre CB	Run with the Calibre CB (Cell/Block) flat license package. LVS comparison and circuit extraction are performed in flat mode. The -cb and -flatten command line options are used.
H-cells Analysis	Perform hcell analysis. See “ <a href="#">Performing Hcell Analysis in Calibre Interactive</a> ” on page 184.
Signature Generation	Generate device signatures. Device signature generation is similar to that in classic Calibre Interactive nmLVS. See “ <a href="#">Generating Device Signatures Using Calibre Interactive</a> ” on page 187 for more information.

For “H-cells Analysis” and “Signature Generation” runs, skip to Step 4.

- b. When not running Calibre nmLVS Recon, specify the “Step” (analysis) option:

Step	Description
Layout vs Netlist	Perform layout netlist extraction and LVS comparison in one step. You supply a layout database and the source netlist. Calibre extracts the layout netlist from the layout database. If the Run type is “Hierarchical”, the extracted layout netlist is saved.
Netlist vs Netlist	Perform LVS comparison in a separate step from layout netlist extraction. You supply a layout netlist file in SPICE format and a source netlist.
Netlist Extraction	Perform layout netlist extraction only. You supply a layout database. The extracted layout netlist is saved.

- c. Enter the “Layout Path” and “Source Path” information.

Select “Export from layout viewer” or “Export from source viewer” to stream the layout or source from a connected viewer.

“Source Path” information is not required for a Netlist Extraction run.

If your source input is in VERILOG or MIXED format, enter v2lvs translator options on the V2LVS page. This page is similar to the Setup Verilog Translator dialog box (**Setup > Verilog Translator**) in classic Calibre Interactive nmLVS.

4. (Optional) To specify hcell input, click **H-Cells** on the left panel.

This page is only available for hierarchical runs. See “[Supplying Hcell Data](#)” on page 176.

5. (Optional) To specify additional layout or source files, open the **Database** page, expand the “Library” area, and provide the information.

6. (Optional) To specify a device signature file, open the **Signatures** page and fill out the Signatures File field. See “[Using a Device Signature File in a Calibre Interactive nmLVS Run](#)” on page 190 for more information.

7. Click **Outputs** on the left panel and do the following:

- a. Enable “Mask SVDB” to create the Standard Verification Database. This database is required in order to view and debug results with Calibre RVE.

Specify the SVDB Directory name and other options. Hover over an option to view the corresponding keyword in the Mask SVDB Directory statement.

- b. (Optional) Enter information regarding the LVS Report in the “Reports” area.

Other reports and report-related options are specified on the **Options** page.

8. (Optional) Specify LVS Short Isolation. See “[Running Short Isolation in Calibre Interactive nmLVS](#)” on page 178.
9. (Optional) Specify ERC checks. See “[Executing Electrical Rule Check \(ERC\) Operations in Calibre Interactive nmLVS](#)” on page 177.
10. Open the **Options** page and select desired options. In most cases you can view the tooltip for a description. Consult the *SVRF Manual* for additional details.
11. Click **Run Control** on the left panel.
  - a. Expand the “RVE Options” area to select options related to opening Calibre RVE.
  - b. Set run control and licensing options; see “[Run Control Page in Calibre Interactive](#)” on page 118.
12. Click **Run LVS** to start the run.

Calibre Interactive first verifies that the source netlist is complete; the source netlist check is case insensitive. If an internal pre-execution trigger is defined, the check for a complete source netlist is done after the trigger executes. You can turn off the source netlist check by disabling the setting “Check source netlist is complete before running LVS” in the Misc section of the **Preferences** page (**Settings > Show Pages > Preferences**). See “[Miscellaneous Preferences](#)” on page 131 for information on the source netlist check.

## Results

The run information is displayed in the Transcript page. The LVS Report is displayed in the Files page if “View LVS report after run finishes” is checked on the Outputs page.

Errors and warnings are listed in the bottom portion of the Transcript page. Right-click and select **Save As** to save the transcript.

# Supplying Hcell Data

Hcell files are used in hierarchical Calibre nmLVS to define the correspondence between layout and source cell names.

## Prerequisites

- Calibre Interactive nmLVS is open.

## Procedure

1. Click **Inputs** on the left panel.
2. Select Hierarchical for the Run type.

The **H-Cells** page is only available for hierarchical runs.

3. Click **H-Cells** on the left panel.
4. Select the desired options:
  - **H-Cells File** — Specifies to use an hcell file. The hcell file specifies cell correspondence for hierarchical LVS comparison. This option preserves hcells as subcircuits during both netlist extraction and LVS comparison.
  - **Match cells by name** — Specifies automatic correspondence by name for cells in Calibre nmLVS-H comparison and should only be used if the layout cells actually contain the same devices as the source subcircuits of the same name.

This option applies to LVS comparison (the -automatch command line option); this option does not apply to netlist extraction.

If “H-Cells File” is also specified, automatched cells are added to the hcell list.

  - **Automatically generate an H-Cell list** — Specifies to automatically generate an hcell list during circuit extraction using the Query Server Tcl shell. This option adds the -genhcells option to the command line. This option cannot be specified with “Match cells by name.”

Check “Use default thresholds as hcell selection criteria” to use the default thresholds when selecting hcells. This option adds the -select keyword to the command line.

Check “Query Server Tcl script” to execute a user-defined Tcl script to generate the hcell list. If not specified, a default Tcl script is used.

If “H-Cells File” is also specified, the hcell lists are concatenated.

For more details, see the -genhcells option in the section “[Calibre nmLVS and Calibre nmLVS-H Command Line](#)” of the *Calibre Verification User’s Manual*.

  - **LVS Expand On Error** —Specifies to examine LVS discrepancies and, in certain cases, to automatically perform additional comparison runs with fewer hcells.

## Executing Electrical Rule Check (ERC) Operations in Calibre Interactive nmLVS

Corresponding SVRF statements: [LVS Execute ERC](#), [ERC Results Database](#), [ERC Maximum Results](#), [ERC Maximum Vertex](#), [ERC Summary Report](#), [ERC Select Check](#), [ERC Unselect Check](#)

You can specify ERC checks in Calibre Interactive nmLVS.

### Prerequisites

- Specify the rule file and inputs for the run. See “[Setting Up a Calibre Interactive nmLVS Run](#)” on page 173.

## Procedure

1. Click **ERC** on the left panel to view the ERC page.
2. Enable the checkbox for “LVS Execute ERC” and set the value in the dropdown list to Yes.
3. Select settings for the other options on the ERC page. Use the tooltips as needed, or refer to the *SVRF Manual* for information about the corresponding rule file statement.
4. Expand the area “ERC Check Selection Recipe Editor” to select the ERC checks to run.

The check selection recipe “Checks selected in the rules file” executes the ERC checks that are selected for execution in the rule file.

You can create a new check selection recipe in the “ERC Check Selection Recipe Editor” area. See “[Check Selection in Calibre Interactive](#)” on page 82 for more information.

5. (Optional) Specify Calibre Auto-Waivers for the run.

On the Inputs page, expand the Waivers area and check “Use Waivers when running LVS.” See “[Calibre Auto-Waivers in Calibre Interactive](#)” on page 161 for details.

## Related Topics

[Electrical Rule Checks \[Calibre Verification User's Manual\]](#)

# Running Short Isolation in Calibre Interactive nmLVS

Corresponding SVRF statement: LVS Isolate Shorts

You can specify short isolation options in Calibre Interactive nmLVS. Short isolation helps pinpoint the location of layout shorts between texted nets.

## Prerequisites

- You have specified the rule file and layout for a Calibre Interactive nmLVS run.  
See “[Setting Up a Calibre Interactive nmLVS Run](#)” on page 173.

## Procedure

1. Click **Options** on the left panel of the GUI.
2. Enable “Short Isolation.”
3. For the “LVS Isolate Shorts” option, specify a type, if desired:
  - **Yes** — Isolate all shorts. This setting is enabled by default. Adds “LVS ISOLATE SHORTS YES”.

- **Power Shorts Only** — Isolate shorts between power net labels. Adds “`&& NAME LVS_POWER_NAME`”.
  - **Ground Shorts Only** — Isolate shorts between ground net labels. Adds “`&& NAME LVS_GROUND_NAME`”.
  - **Power Ground Shorts** — Isolate shorts between power net labels, shorts between ground net labels, and shorts between power-ground nets. Adds “`&& NAME LVS_POWER_NAME LVS_GROUND_NAME`”.
  - **IO Shorts** — Isolate shorts between signal nets and shorts between signal and power-ground nets. Adds “`&& !NAME LVS_POWER_NAME LVS_GROUND_NAME`”.
4. Specify additional options as desired.

Hover over an option to view a description and the corresponding keyword for the rule file statement.

In order to perform interactive short isolation with Calibre RVE for LVS, the following options are recommended. The corresponding keyword for the LVS Isolate Shorts statement is given in parentheses.

- Report shorts by layer (BY LAYER ALSO)
  - Report shorts by cell (BY CELL ALSO)
  - Isolate shorts in “All Cells” (CELL ALL)
5. Enable “LVS Maximum Short Results,” if desired. This option appears in the **Options** page if “Short Isolation” is enabled.

If set to “Count,” this option limits the maximum number of isolated short paths that are identified and reported to the provided value. When not enabled or set to “All,” all short paths are reported.

## Changing Trace Property Statements

Corresponding SVRF statement: [Trace Property](#)

You can use the Trace Properties option in Calibre Interactive to override the Trace Property statements in your rule file. You can add new statements, edit existing statements, and disable statements. This feature is available in Calibre Interactive nmLVS and Calibre Interactive PERC on the Options page.

### Prerequisites

- Calibre Interactive nmLVS or Calibre Interactive PERC is open. See “[Setting Up a Calibre Interactive nmLVS Run](#)” on page 173 and “[Calibre Interactive PERC](#)” on page 219.

- A rule file is specified on the Rules page.

## Procedure

1. Click **Options** on the left panel of the GUI.
2. Check the “Trace Property” option. This also expands the “Trace Property” area.  
When “Trace Property” is checked, only the Trace Property statements enabled in the Trace Property table are used in the Calibre run.
3. Use the following table to determine the steps to take depending on the action you want:

---

### Note

---

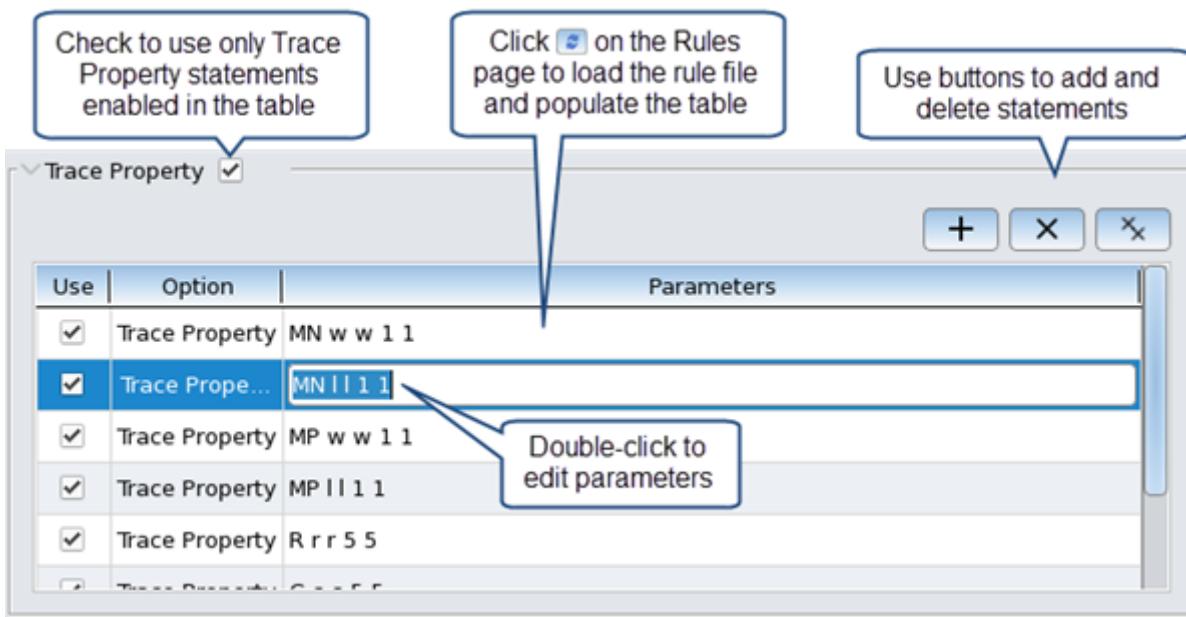
 Trace Property statements that include trace property computation procedures are not loaded into the Trace Property table.

---

**Table 7-1. Changing Trace Property Statements in Calibre Interactive nmLVS**

Action	Steps
Disable all Trace Property statements in the rule file.	<p>Do one of the following:</p> <ul style="list-style-type: none"><li>• Do <i>not</i> load the rule file to populate the Trace Property table.</li><li>• If the Trace Property table is already populated, disable all checkboxes in the Use column or click  to remove all statements.</li></ul>
Disable or enable specific Trace Property statements.	<ol style="list-style-type: none"><li>1. Click  (Load) on the Rules page to populate the table.</li><li>2. Use the checkboxes in the Use column to select the statements that are included in the run.</li></ol>
Edit the parameters for a Trace Property statement.	<ol style="list-style-type: none"><li>1. Click  (Load) on the Rules page to populate the table.</li><li>2. Click in the Parameters column and edit the parameters.</li></ol>
Add a new Trace Property statement.	<ol style="list-style-type: none"><li>1. (Optional) Click  (Load) on the Rules page to populate the table with rule file statements. Do this if you want to run the rule file Trace Property statements in addition to the new statement.</li><li>2. Click  to add a new statement, then fill in the parameters in the Parameters column.</li></ol>

**Figure 7-1. Changing Trace Property Statements**



4. (Optional) Save the runset with **File > Save Runset**. The statements in the Trace Property table and the Use column settings are saved to the runset.

## Results

When the “Trace Property” checkbox is checked, only the statements enabled in the Trace Property table and the Use column settings are used in the Calibre run.

### Note

 When you load a rule file, all corresponding GUI settings are set to the settings in the rule file. Therefore, any custom settings in the Trace Property table are reset to rule file settings when you load the rule file.

# Adding and Changing LVS Box Statements in Calibre Interactive

You can use Calibre Interactive to override the LVS Box related statements in your rule file; this is done on the LVS page with the LVS Box, LVS Black Box Port, and LVS Black Box Port Depth options. You can add new statements, edit existing statements, and disable statements.

Corresponding rule file statements: [LVS Box](#), [LVS Black Box Port](#), [LVS Black Box Port Depth](#)

## Prerequisites

- Calibre Interactive is open for the LVS, PERC, PEX, or xACT applications.
- A rule file is specified on the Rules page.

## Procedure

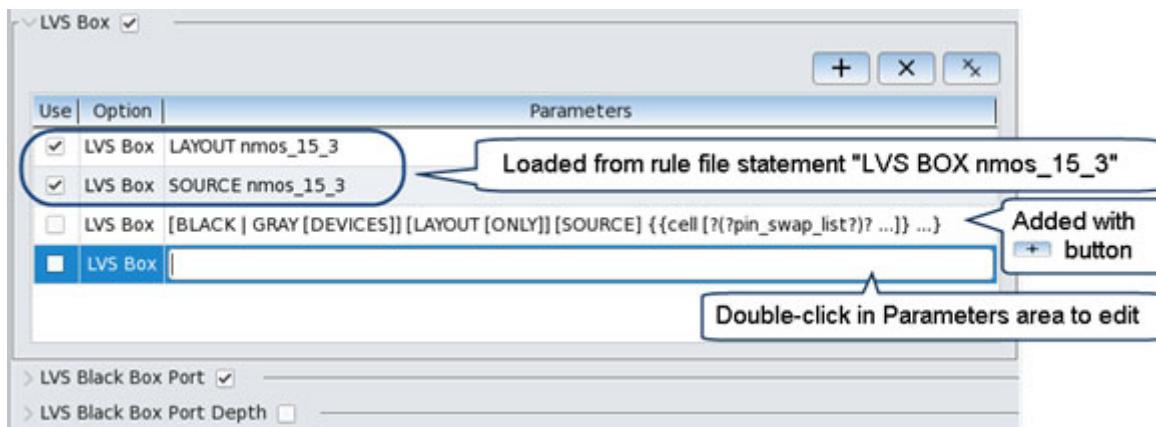
1. (Optional) Click **Rules** on the left panel and click the  button to load the rules file.

This loads the LVS Box statements in the rule file into the GUI's table of LVS Box statements. This step is required if you want to change existing LVS Box statements.

A single LVS Box statement in the rule file may specify multiple cells and pin swap lists, and the combination LAYOUT SOURCE is used if neither of those keywords is specified; when loaded into the LVS Box table such statements are broken up into multiple LVS Box statements.

2. Do one of the following depending on your Calibre Interactive application:
  - LVS and PERC — Select the Options page.
  - PEX and xACT — Select the LVS page.
3. Check the “LVS Box” checkbox and expand the option area. If you loaded a rule file with LVS Box statements, this is already done.

When “LVS Box” is checked, the run uses only LVS Box statements from the GUI.



“LVS Black Box Port” is displayed if “LVS Box” is checked, and “LVS Black Box Port Depth” is displayed if “LVS Black box Port” is checked.

4. Select, unselect, and edit statements in the “LVS Box” table as described in the following table:

Action	Steps
Disable or enable specific statements.	Use the checkboxes in the “Use” column to select statements to include in the run.
Add a statement.	Click the  button to add a statement, then click in the Parameters area to enter the statement parameters.

Action	Steps
Edit the parameters for a statement.	Click in the Parameters area.
Remove statements.	Click the  button to remove all statements, or the  button to remove a single statement.
Disable all LVS Box statements.	Uncheck all statements with the “Use” column.

The GUI does not check the syntax of the parameter entries.

5. If needed, expand the “LVS Black Box Port” option area and supply (or edit) the LVS Black Box Port statement.

---

**Note**

 The LVS Black Box Port statement is required if “LVS Box BLACK ...” is specified. If you add a LVS Box BLACK statement, you must also add a LVS Black Box Port statement—the GUI does not enforce this.

---

6. If needed, check “LVS Black Box Port Depth” and select the hierarchy level.

# Performing Hcell Analysis in Calibre Interactive

---

You can use Calibre Interactive nmLVS to calculate the effectiveness of hcells in reducing the total hierarchical instance count of your design. You can also extract and view the hierarchical structure of the design.

Creating and Analyzing an H-Cell List With Calibre Interactive .....	184
H-Cells Analysis Results .....	186

## Creating and Analyzing an H-Cell List With Calibre Interactive

A number of options affect the output of your hcell analysis.

### Prerequisites

- Calibre Interactive nmLVS is open with the layout netlist and source netlist specified on the **Inputs** page.  
If the layout netlist file does not exist, specify a geometric layout and the layout netlist will be extracted from the layout. If the source netlist file does not exist and “Export from schematic viewer” is enabled, the source netlist will be exported from a connected schematic viewer.

### Procedure

- Click **Inputs** on the left panel and select a run type of “H-cells Analysis.”
- Click **H-Cells** on the left panel.
- Select hcell options. You must select at least one of the following in order to perform an analysis.
  - H-Cells File** — Select this option if you have an existing hcell file that you want to analyze.  
This option applies to hcell analysis, LVS comparison, and netlist extraction.
  - Match cells by name** — Select this option if you want hcell candidates to be selected and analyzed based upon cell names that match between layout and source. (Do this only if you are certain the subcircuit definitions match between such cells.)  
This option applies to both hcell analysis and LVS comparison; it does not apply to netlist extraction.
  - Match cells by number of placements** — Select this option if you want to compare cells with the same number of placements and equivalent pin counts as hierarchical entities. Use caution when specifying this option, since it can identify invalid hcell

pairings, as described in [hcells::placementmatch](#) and [NETLIST PLACEMENTMATCH](#) in the *Calibre Query Server Manual*.

This option only applies to hcell analysis in Calibre Interactive; it does not apply during LVS comparison or circuit extraction.

The option “Automatically generate an H-Cell list” does not apply for hcell analysis runs.

4. Select analysis options in the H-Cells Analysis Options dialog box:

- **Evaluate Current H-Cells during analysis** — Calibre Interactive evaluates cells in the specified hcells file against a specific threshold during the analysis.
- **Effectiveness Evaluation Threshold (%)** — This setting defaults to 30%, and specifies the effectiveness threshold for saving an hcell candidate. The effectiveness is a measurement of computational savings based on the reduction of Total Hierarchical Instance Count (THIC) for additional hcells. Only hcell candidates that collectively are at or above this threshold are saved after analysis. This setting specifies the first parameter (*THIC\_threshold*) for the [NETLIST EVALUATION THRESHOLD](#) and “[hcells::select -threshold](#)” Query Server commands.
- **Total Hier Inst Count Reduction Threshold(%)** — This setting specifies the cutoff threshold for instance count evaluation. If a cell's instance count is greater than (total flat instance count \* *inst\_count\_threshold* / 100), it appears in the hcell list even if it does not improve the computational effectiveness. The default is 1%. This setting specifies the second parameter (*flat\_threshold*) for the [NETLIST EVALUATION THRESHOLD](#) and “[hcells::select -threshold](#)” Query Server commands.

If both “Evaluate Current H-Cells during analysis” and “Match cells by name (automatch)” are selected, the hcell analysis considers cells from the specified hcell list and cells determined by automatch.

5. Click the **Run H-Cells Analysis** button in the left panel.

Calibre Interactive nmLVS performs the hcell analysis and displays the results in four tabs. See “[H-Cells Analysis Results](#)” on page 186.

Evaluate the hcell list for correctness and effectiveness. The hcell entries are on the H-Cells Evaluation tab in the last two columns (Layout Cell Name and Source Cell Name). If needed, you can adjust options and run again.

6. (Optional) Click the **Save H-Cells** button below the result tabs to save the results to an hcell file.

After saving the hcells file, a pop-up dialog prompts you to choose whether to use this file as the hcells file for the run. To use the file, you must also check the checkbox for H-Cells File.

Before using the hcells file, check the list for correctness.

## H-Cells Analysis Results

Hcell analysis results are shown in four sub-tabs within the H-Cells page of Calibre Interactive nmLVS.

See the task “[Creating and Analyzing an H-Cell List With Calibre Interactive](#)” on page 184 for a description of how to perform Hcell analysis.

When hcell analysis is complete, the results are displayed as shown in [Figure 7-2](#) in the following four sub-tabs:

- **H-Cells Evaluation** — This tab displays the results of your hcells evaluation.  
Use the **Save H-cells** button to save the selected hcell names to an hcell file.
- **Source Analysis** — This tab displays information about the contribution of devices present given certain conditions and the resulting memory savings. Hover your mouse over the column heading for a description of the data provided.
- **Layout Analysis** — This tab displays information similar to the **Source Analysis** tab, but for the layout netlist.
- **Hierarchy** — This tab displays the hierarchical structure of both the layout and source.

**Figure 7-2. H-Cells Analysis Result Tabs**

H-Cells Evaluation								
Total Hier Inst. Count ^ Layout	Total Hier Inst. Count Source	Inst. Count In this Cell Layout	Inst. Count In this Cell Source	Saved By this Cell	Total Savings So Far	Potential Remaining Saving	Layout Cell Name	Source Cell Name
407	251		42		45	3.6	vco	vco
406	250		40	11	45	3.4	phasedetpath	phasedetpath
386	250		6		46	0.3	pll	pll
386	250	16		0	46	0.3	pliclik	pliclik
385	249		8	4.8	47	0	buf4	buf4

**Save H-Cells**

Also see “[Hcell and Hierarchical Analysis](#)” in the *Calibre Query Server Manual*.

# Device Signatures: Using and Creating in Calibre Interactive nmLVS

Calibre Interactive nmLVS enables you to easily generate and use unique signature strings for devices. Device signatures enable the matching of devices in the layout to a specific geometric pattern, rather than matching to just the layers that are present.

For general information on device signatures, signature generation, and signature comparisons, refer to “[Device Signatures](#)” in the *Calibre Verifications User’s Manual* and the [Device](#) command in the *SVRF Manual*. It will help if you are familiar with device signature concepts when using the GUI to create a device signature file.

<b>Generating Device Signatures Using Calibre Interactive .....</b>	<b>187</b>
<b>Using a Device Signature File in a Calibre Interactive nmLVS Run .....</b>	<b>190</b>
<b>Edit Parameters for Device Dialog Box in Calibre Interactive .....</b>	<b>191</b>
<b>Layers File for Device Signature Creation With Calibre Interactive.....</b>	<b>194</b>

## Generating Device Signatures Using Calibre Interactive

You can use Calibre Interactive nmLVS to create a device signature file. Device signatures enable the matching of devices in the layout to a specific geometric pattern, rather than matching to just the layers that are present. The signature creation process from the GUI is a standalone step, separate from your LVS comparison or netlist extraction run. You can save the device commands with the generated device signatures to a new file or append to an existing file.

### Prerequisites

- A Calibre Advanced Device Properties (ADP) product license in addition to a Calibre nmLVS license.
- A layout file in a geometric format, specified on the **Inputs** page.
- A rule file, specified on the **Rules** page.
- (Optional) A Layers File. The Layers File specifies the device layers and the layers used to generate the device signature. The file is used to populate fields in the Edit Parameters for Device dialog. See “[Layers File for Device Signature Creation With Calibre Interactive](#)” on page 194 for the syntax of this file.
- (Optional) A signatures file from a previous signature generation run. An existing signature file can be used to populate the GUI with device information.

### Procedure

1. Click **Inputs** on the left panel to display the Inputs page.

- a. For the Run setting, choose **Signature Generation** in the dropdown menu.
  - b. Enter layout information in the Layout Path area. The layout must be in a geometric format.
2. Click **Rules** on the left panel and enter a rule file suitable for device generation.
  3. Click **Signatures** on the left panel.
  4. (Optional) Populate the GUI with a Layers File or signature file.

---

**Note**

 If you do not populate the GUI with a Layers File or signature file, all necessary information can be entered manually.

---

Do one of the following:

- Specify a Layers File in the text entry field. See “[Layers File for Device Signature Creation With Calibre Interactive](#)” on page 194 for the syntax of this file.

Click the  button to load the Layers File data. The Layers File populates the Edit Parameters for Device dialog box with layer information. A pop-up note displays if any layer parameters are missing in the Layers File. However, because all layers may be entered manually, this note is for information only and does not interfere with the signature creation process.

- Specify a previously generated signature file.

Click the  button to load the signature data. The Device Statements table and the Device Signatures field are populated based on information in the signatures file.

5. Do one or both of the following, depending on what you did in the previous step.

- **Add a new device** — Click the  (**Add Device**) button to add a new device. The Edit Parameters for Device dialog box opens. This step may not be needed if you loaded a signature file.

Enter all required layers and options for signature creation in the Edit Parameters for Device dialog box.

- **Edit a device** — Select a device row in the Device Statements table and click the  (**Edit Device**) button to open the Edit Parameters for Device dialog box.

Use the action buttons and right-click menus in the Edit Parameters for Device dialog box to add table rows for pins and layers. You can also type directly in the Device Name, Seed Layer, and Reference Layer entry boxes. See “[Edit Parameters for Device Dialog Box in Calibre Interactive](#)” on page 191 for a definition of the fields in the dialog box.

6. Click **Save Device** or **Save As New Device** to update the entries in the Device Statements table.

If you enabled “Use only seed layers with attached text label” in the Edit Parameters for Device dialog box, the Label column is filled in with the With Text statement that defines the labeled seed layer.

Note: The new statement is *not* added if any of the following errors occur:

- The device statement already exists in the Device Statements table.
- There are one or more unspecified parameters for the device statement.

7. (Optional) Add additional device statements if needed.
8. Click **Close** to close the Edit Parameters for Device dialog box.
9. (Optional) If your rule file includes an [LVS Summary Report](#) statement with a Tcl script for custom report generation, it is required to generate the SVDB. Click the **Outputs** button and enable “Create SVDB Database.”
10. Click **Run Signature Generation** to generate signatures for all of the device statements selected in the Device Statements table.



**Tip**

You can hold the Ctrl key and click the **Run Signature Generation** button to view the rule file that will be used in the signature creation run.

---

If a duplicate device definition is detected between the created device statements from the GUI and the device statements in the original rule file, an error message displays and the run is aborted. Remove the duplicate statement from either the rule file or the Device Statements table, and click **Run Signature Generation** again.

After a successful run, the Device Signature area is displayed with the generated signature(s).

11. Click **Save Device Signatures** to save the generated signature statements in a file. A dialog prompts you for the filename; you can choose to overwrite the file or append to it.

If you enabled “Use only seed layers with attached text label” in the Edit Parameters for Device dialog box, the With Text statement that defines the labeled seed layer is also written to the saved file. The With Text statement can also be viewed in the Label column of the Device Statements table.

## Results

At the end of the signature generation process, the Device statements with the generated signatures are saved in a signature file. To use the created signatures in your Calibre nmLVS run, see “[Using a Device Signature File in a Calibre Interactive nmLVS Run](#)” on page 190.

A SVDB is created if the rule file includes an [LVS Summary Report](#) statement and “Create SVDB Database” is enabled.

## Using a Device Signature File in a Calibre Interactive nmLVS Run

Calibre Interactive nmLVS can include a device signature file in the Calibre nmLVS run. The controls are on the Signatures page in the GUI.

See “[Device Signatures](#)” in the *Calibre Verifications User’s Manual* for details.

### Prerequisites

- A device signature file. To create a signature file, see one of the following:
  - **GUI method:** “[Generating Device Signatures Using Calibre Interactive](#)” on page 187
  - **Command line method:** “[Device Signatures](#)” in the *Calibre Verifications User’s Manual*
- Calibre Interactive nmLVS is open and set up for a Calibre nmLVS run.

### Procedure

1. Click **Signatures** on the left panel of the GUI to view the Signatures page.

---

#### Note

 If you just finished a Signature Generation run using the GUI, the generated signatures are displayed in the Device Signatures area. If you want to use these, skip to Step 4.

---

2. Enter your device signature file in the Signatures File entry.

3. Click the  button to load the signatures file.

The Device Signatures area is displayed with the signature information.

4. Enable the checkbox “Include Device Signatures in LVS Run.”

### Results

When the LVS run is started, the Device command with the signature string is added to the Calibre Interactive control file, and used in the run.

---

#### Tip

 To view the control file without starting a run, press and hold the Ctrl key while clicking the **Run LVS** button.

---

# Edit Parameters for Device Dialog Box in Calibre Interactive

To access: On the **Signatures** page with a run mode of Signature Creation, expand the Device Statements area and do one of the following:

- Click the button to add a device to the Device Statements table.
- Select an existing device in the Device Statements table, and click the (Edit Device) button.

Use the Edit Parameters for Device dialog box to specify the layers, pin names, and options used for device signature creation in Calibre Interactive.

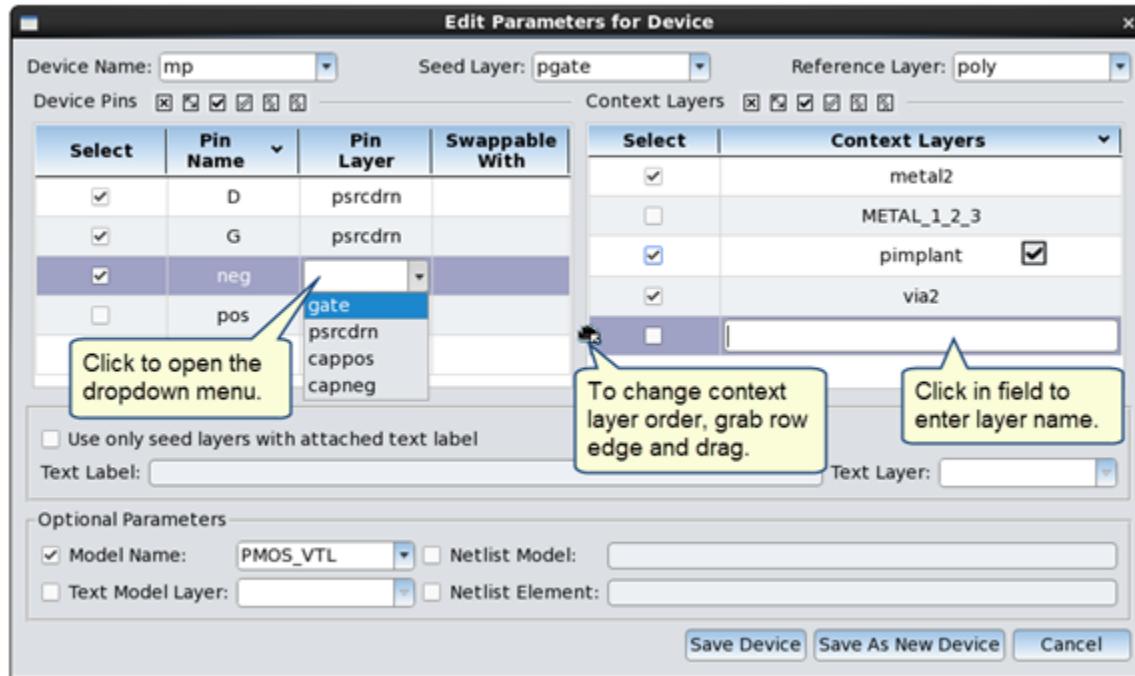
## Description

### Note

You can specify and load a Layers File to pre-populate the dropdown lists and tables in the dialog box; see “[Layers File for Device Signature Creation With Calibre Interactive](#)” on page 194.

However, you can add table rows, and type in any entry field or table row to enter an arbitrary parameter, so loading a Layers File is not necessary.

**Figure 7-3. Edit Parameters for Device Dialog Box**



## Objects

**Table 7-2. Device Parameter Fields for Create Device Signature**

Device Parameter Field	Description
Device Name	The component name.
Seed Layer	The layer the device body is on; also known as the device layer.
Reference Layer	<p>The reference layer defines the area around the seed shape used for deriving a signature.</p> <p>Once a reference layer is used with a specific set of context layers, it may only be used with that set of context layers in the rule file.</p>
Device Pins	<p>The Device Pins table has the pin name and pin layer pairs necessary for the Device statement.</p> <p>Use the right-click menu or button bar to add, delete, and select pins.</p> <p>Swappable With — Enter the pin names that are interchangeable with the pin in the row. If this entry is left blank, the default pin swappability rule applies: pins on the same layer are interchangeable.</p>
Context Layers	<p>Shapes on the context layer that intersect the reference layer are considered in the signature.</p> <p>You can specify multiple context layers. The order of the context layers matters, since the generated signature reflects the given order. You can change the order of context layers by dragging the layer entry—grab the row at either end and drag.</p> <p>Use the right-click menu or button bar to select or unselect all layers and to toggle the layer selection.</p> <p>You can use the layers file to map a list of context layers to a list name; see the *CONTEXT_LAYER entry in “<a href="#">Layers File for Device Signature Creation With Calibre Interactive</a>” on page 194.</p> <p>Note: The seed layer and pin layers are not part of the signature calculation unless the layers are included in the context layer list.</p>
Use only seed layers with attached text label Text Label Text Layer	<p>When enabled, only device seed layers with the specified text label on the given text layer are used in the Device statement. The text label string is appended to the seed layer name and a new layer is created which is used in the Device statement. These statements are used:</p> <pre>&lt;seed_layer&gt;_&lt;label&gt; = &lt;seed_layer&gt; WITH TEXT     "label" &lt;text_layer&gt; DEV &lt;device&gt; &lt;seed_layer&gt;_&lt;label&gt; ...</pre>
Model Name	Use the specified model name (or component subtype).

**Table 7-2. Device Parameter Fields for Create Device Signature (cont.)**

Device Parameter Field	Description
Text Model Layer	Use the TEXT MODEL LAYER keyword and parameter,
Netlist Model	Use the NETLIST MODEL keyword and parameter
Netlist Element	Use the NETLIST ELEMENT keyword and parameter.

# Layers File for Device Signature Creation With Calibre Interactive

Input for: The **Signatures** page in Calibre Interactive nmLVS, in Signature Generation mode

The Layers File is used to define the device types, device layers, pin names and pin layers for signature creation. The entries in the Layers File populate the fields of the Edit Parameters for Device dialog box in Calibre Interactive. You create the Layers File with a text editor.

The Parameters section defines the entries in the Layers File for device signature creation. An example is provided in the “Examples” section.

## Format

- ASCII text file.
- One entry per line.
- Comment lines start with a “//”.
- Layer entries may have an optional *abbreviation* entry. If present, the layer *abbreviation* is used in the Edit Parameters for Device dialog box.
- For the CONTEXT\_LAYER entry only, a list of layers enclosed by double quotes may be mapped to an abbreviation.

## Parameters

**Table 7-3. Layers File for Device Signature Creation Contents**

Contents	Definition
*DEVICE_TYPES <device_type> ...	List the device types you plan to create signatures for. The device types should be those permitted in the SVRF Device command.
*DEVICE_MODELS <device_model> ...	List the device models you plan to create signatures for. The device models should be those permitted in the SVRF Device command.
*REFERENCE_LAYERS <reference_layer> [<abbreviation>] ...	List the layers that serve as the reference_layer parameter for signature creation in the Device statement.

**Table 7-3. Layers File for Device Signature Creation Contents (cont.)**

Contents	Definition
<pre>*CONTEXT_LAYERS &lt;context_layer&gt; [&lt;abbreviation&gt;] ... or: “&lt;layer&gt; &lt;layer&gt; ...” [&lt;list_name&gt;]</pre>	<p>List the layers that serve as the context_layer parameter for signature creation in the Device statement.</p> <p>A list of context layers enclosed in double quotes may also be mapped to a layer list name. If the context layer list is selected in the Edit Parameters for Device dialog box, the list contents are enclosed in braces ({ }) when shown in the device statement; the braces are not included in the device statement written to the control file.</p>
<pre>*PIN_LAYERS &lt;pin_layer&gt; [&lt;abbreviation&gt;] ...</pre>	<p>List the layers that serve as the pin_layer parameter in the Device statement.</p>
<pre>*PIN_NAMES &lt;pin_name&gt; [&lt;abbreviation&gt;] ...</pre>	<p>List the layers that serve as the pin_name parameter in the Device statement.</p>
<pre>*SEED_LAYERS &lt;seed_layer&gt; [&lt;abbreviation&gt;] ...</pre>	<p>List the layers that serve as the seed layer (or device_layer parameter in the Device statement).</p>
<pre>*TEXT_MODEL_LAYERS &lt;model_text_layer&gt; [&lt;abbreviation&gt;] ...</pre>	<p>List the layers that serve as the text_layer parameter for the TEXT MODEL LAYER keyword in the Device statement.</p>
<pre>*SEED_LABEL_LAYERS &lt;seed_text_layer&gt; [&lt;abbreviation&gt;] ...</pre>	<p>If you are using seed shapes that are texted for the purpose of signature generation, list the layers that have text labels for the seed layer. See “<a href="#">Edit Parameters for Device Dialog Box in Calibre Interactive</a>” on page 191 for an explanation of how this parameter is used with the option “Use only seed layers with attached text label”.</p>

## Examples

```
// layers file for device signature creation
*DEVICE_TYPES
mp
C

*DEVICE_MODELS
PMOS_VTL
PMOS_VTH
```

```
*PIN_NAMES
G
S
D
pos
neg

*PIN_LAYERS
gate
psrcdrn
cappos
capneg

*SEED_LAYERS
pgate
hvtpgate
capbody

*REFERENCE_LAYERS
poly
fieldpoly fpoly
capbody

*CONTEXT_LAYERS
pimplant
metal2
via2
// create context layer list
"metal1 metal2 metal3" METAL_1_2_3
// the seed label layer entry is optional
// *SEED_LABEL_LAYERS
```

# LVS-Specific Pages in Calibre Interactive

Several pages are specific to Calibre Interactive nmLVS.

**Table 7-4. LVS-Specific Pages in Calibre Interactive**

Page	Description
H-Cells	Enter Hcell information. This page is available for “Hierarchical” and “H-cells Analysis” runs. See “ <a href="#">Supplying Hcell Data</a> ” on page 176 and “ <a href="#">Performing Hcell Analysis in Calibre Interactive</a> ” on page 184.
ERC	See “ <a href="#">Executing Electrical Rule Check (ERC) Operations in Calibre Interactive nmLVS</a> ” on page 177.
Signatures	Specify device information for a device signature generation run, or specify a device signature file for a Calibre nmLVS run. See “ <a href="#">Device Signatures: Using and Creating in Calibre Interactive nmLVS</a> ” on page 187.
V2LVS	Enter options for the v2lvs translator, which translates VERILOG format source input to SPICE format. The V2LVS page is available when VERILOG or MIXED is selected as the “Source Format” on the Inputs page.
Waivers	Specify information for the Calibre Auto-Waivers Flow. Waivers are supported for ERC checks in Layout vs Netlist and Netlist Extraction runs. This page is available if “Use Waivers when running LVS” is checked on the Inputs page or a run type of “Autowaiver creation” is specified. See “ <a href="#">Calibre Auto-Waivers in Calibre Interactive</a> ” on page 161.



# Chapter 8

## Calibre Interactive PEX

---

You can specify inputs, outputs, and options for a Calibre xRC run using Calibre Interactive PEX.

<b>Introduction to Calibre Interactive PEX</b> .....	<b>199</b>
<b>Inputs for a PEX Run</b> .....	<b>202</b>
<b>Setting Up a Calibre Interactive PEX Run</b> .....	<b>202</b>
<b>Using Iterative Extraction to Create a Mixed Parasitic Network</b> .....	<b>205</b>
<b>Setting Cell Extraction Options for Blocks</b> .....	<b>206</b>
<b>Setting Probe Points</b> .....	<b>208</b>
<b>Specifying Extracted Netlist Annotation of Net Geometries to Calibre View</b> .....	<b>210</b>
<b>Naming the Extracted Parasitic Netlist Based on the Extraction Type and Format</b> ..	<b>211</b>
<b>Controlling PEX Step Execution</b> .....	<b>213</b>
<b>PEX-Specific Pages in Calibre Interactive</b> .....	<b>214</b>
LVS Page in Calibre Interactive PEX.....	215

## Introduction to Calibre Interactive PEX

The Calibre Interactive PEX Graphical User Interface (GUI) is an interface to the Calibre xRC, Calibre xACT, and Calibre xACT 3D batch parasitic extraction tools. The Calibre Interactive PEX GUI allows you to specify command line options and selected rule file statements that execute the desired Calibre PEX application.

Calibre Interactive facilitates performing the following types of extraction:

- **Transistor-level extraction** — Generates a flat netlist which includes the original intentional devices in your circuit along with the specified parasitic elements.
- **Gate-level extraction** — Generates a netlist which includes the top-level cells (logic gates) in your circuit, along with the specified parasitic elements associated with the interconnection of those cells.
- **Hierarchical extraction** — Generates a netlist which includes connectivity, intentional device and parasitic element information for all cells in the hierarchy of your design.

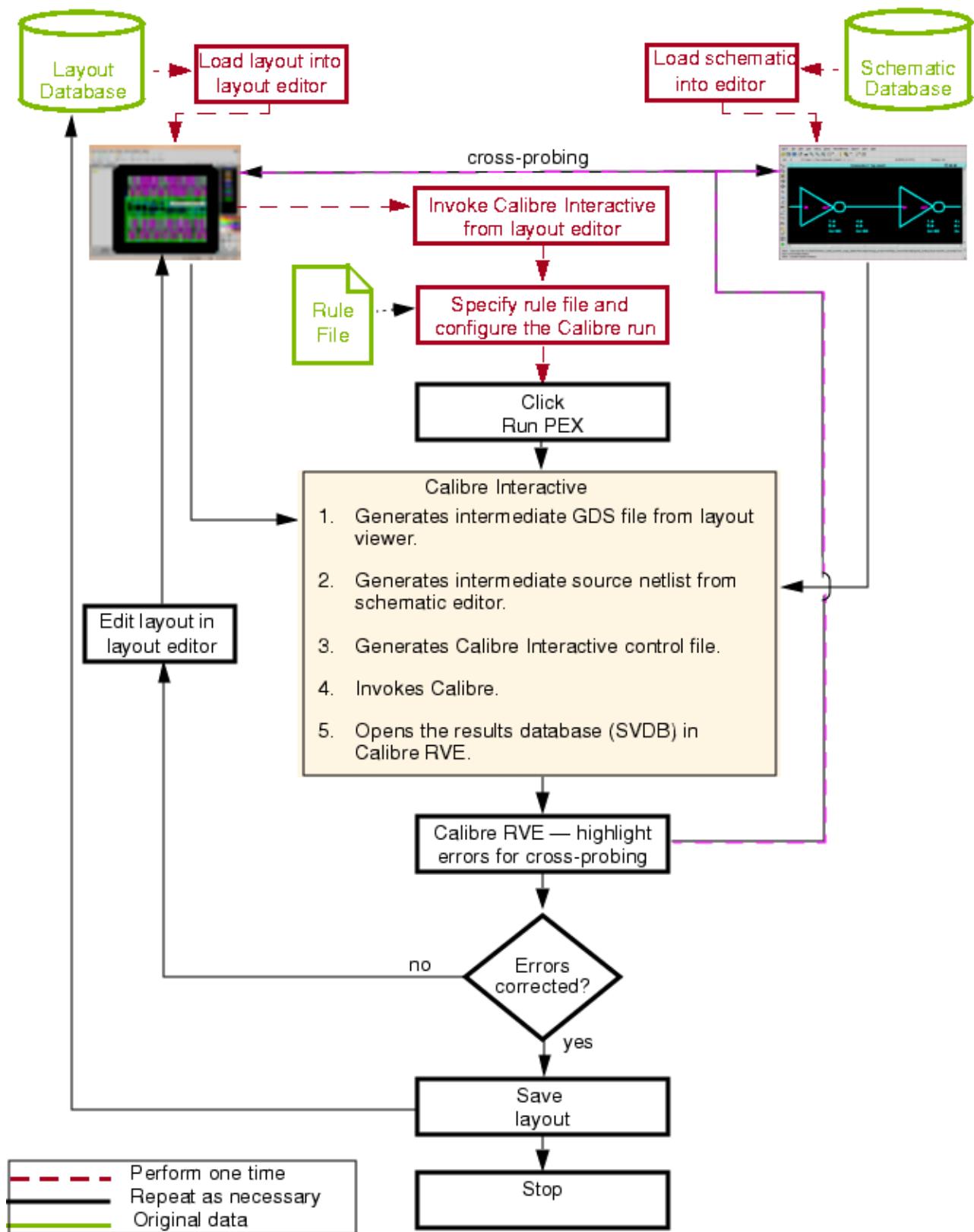
## Input Files You Use With Calibre Interactive PEX

The following input files are required when using the Calibre Interactive PEX GUI:

**Table 8-1. Files Required by the Calibre Interactive PEX GUI**

Name	Description
Layout database	The layout database, or design, is a physical representation of your IC.
SVRF rule file	A text input file you must create using Standard Verification Rule Format (SVRF) statements and operations. It controls the parasitic extraction tool's execution.
Source netlist (optional)	The source netlist, or schematic, is an electrical representation of your IC. Use it for input if you want source names rather than layout names in your output.

**Figure 8-1. Generic PEX Flow**



## Inputs for a PEX Run

The Inputs pane includes the tabs for each type of input. Use these tabs to specify the inputs needed for your PEX run.

**Table 8-2. Inputs Pane for PEX**

Tab	Purpose	Required
Layout	Specify the layout design database and format.	Yes
Netlist	Specify the schematic netlist. This is used when you want the extracted netlist to use source schematic names. Netlist data is also needed when controlling cell extraction options for an ADMS run.	No
H-Cells	Specify cell lists for performing non-flat extraction.	No
Blocks	Specify blocks for ADMS extraction. See “ <a href="#">Setting Cell Extraction Options for Blocks</a> ” on page 206 and the <i>Calibre xRC User’s Manual</i> .	No
Probes	Specify probe points. See “ <a href="#">Setting Probe Points</a> ” on page 208 and <a href="#">PEX Probe File</a> in the <i>SVRF Manual</i> .	No

## Setting Up a Calibre Interactive PEX Run

You can control inputs, outputs, and options for a Calibre xRC run using Calibre Interactive PEX.

### Prerequisites

- Calibre Interactive PEX is open. See “[Invoking the Calibre Interactive GUI](#)” on page 58.
- See “[Calibre Interactive GUI Reference](#)” on page 105 for information on setting the options on the Environment, Run Control, Triggers, Preferences, and Templates pages.

**Tip** If a page name is not visible on the left panel, choose **Settings > Show Pages** and select the page name. Also make sure any options that enable the page are set.

---

### Procedure

1. (Optional) Click **File > Load Runset** to load an existing runset.  
You can load a classic Calibre Interactive runset; see “[Converting Classic Calibre Interactive Runsets](#)” on page 37.
2. Click **Rules** on the left panel and specify the rule file and run directory.

If desired, click the **Load** button () to load rule file settings into the GUI.

To specify additional rule files, enable “Include Rules Files” on the Options page. To specify additional rule file statements, enable “Include Rule Statements” on the Options page.

3. Click **Outputs** on the left panel and select the xRC Mode and Extraction Type.

---

**Note**

 The run mode, extraction type, and PEX Netlist options affect the options that are available on other pages, so it is useful to set these options first.

---

- a. Select the “xRC Mode”:
  - o **xRC** — Use model-based extraction. See the *Calibre xRC User’s Manual*.
  - o **xACT 3D** — Use the Calibre xACT 3D fieldsolver. See the *Calibre xACT User’s Manual*.
  - o **xACT 3D Hybrid** — Use hybrid xACT 3D/rule-based extraction (the -hybrid command line option).

The two xACT 3D options add the FieldSolver page for selecting **PEX Fieldsolver Mode** options.

- b. If the extraction mode is “xACT 3D” or “xACT 3D Hybrid”, select the “Accuracy Mode”.
- c. Make selections in the “Extraction Type” area for the extraction level and parasitics to extract. The Level selection affects the available options for “Extraction Type”.
- d. (Optional) To specify corner extraction, enable the Corners checkbox and enter the corner names.
- e. Select options for the PEX Netlist statement.
- f. Select options for the Mask SVDB statement.
- g. Make selections for the report-related settings.
- h. (Optional) Expand the Formatter area and make selections for the parasitics to output to the RC netlist and other formatter options.

4. Click **Inputs** on the left panel and specify the following:

- a. Enter the “Layout Path” information.

Select “Export from layout viewer” to stream the layout from a connected viewer.

- b. If required, enter the “Source Path” information.

If “Source Path” information is not required for the run, the option is not displayed.

Select “Export from source viewer” to stream the source from a connected viewer.

If your source input is in VERILOG or MIXED format, enter v2lvs translator options on the V2LVS page. This page is similar to the Setup Verilog Translator dialog box (**Setup > Verilog Translator**) in classic Calibre Interactive PEX.

- c. (Optional) Specify hcell input.
- d. Specify xcell input if required for your run.
5. (Optional) To specify additional layout or source files, open the **Database** page, expand the “Library” area, and provide the information.

If you specify multiple source netlist files, Calibre Interactive creates a file named `_source.net_` with .INCLUDE statements for each netlist file; `_source.net_` is used as the source netlist for the Source Path statement in the control file. Files entered in the **Database** page are listed before files in the **Inputs** page.
6. Open the **Options** page and select desired options. In most cases you can view the tooltip for a description. Consult the *SVRF Manual* for additional details.
7. Click **LVS** on the left panel and select LVS-related options.
8. Select options on the Inductance and FieldSolver pages if these pages are available for your run.
9. Click **Run Control** on the left panel.
  - a. Expand the “RVE Options” area to select options related to opening Calibre RVE.
  - b. Set run control and licensing options; see “[Run Control Page in Calibre Interactive](#)” on page 118.
10. Click **Run PEX** to start the run.

## Results

The run information is displayed in the Transcript page. Calibre RVE opens if “Show Results in RVE” is enabled on the Run Control page.

Errors and warnings are listed in the bottom portion of the Transcript page. Right-click and select **Save As** to save the transcript.

## Related Topics

[Basic Calibre Interactive Usage](#)

[Using Iterative Extraction to Create a Mixed Parasitic Network](#)

# Using Iterative Extraction to Create a Mixed Parasitic Network

Iterative extraction runs the PDB stage multiple times with different parasitic elements extracted for different nets. This can improve simulation time because the netlist has fewer elements.

In this flow, the PHDB is created once, the PDB is generated in several steps for selected nets and specified parasitics, and the extracted netlist is created as the last step.

## Prerequisites

- A Calibre Interactive PEX run is set up. See “[Setting Up a Calibre Interactive PEX Run](#)” on page 202.

The initial extraction setup is typically “background” parasitics for the complete design.

---

### Note

 The extraction level must stay the same throughout the procedure.

---

## Procedure

1. Disable netlist creation:
  - a. Click **Run Control** on the left panel.
  - b. Expand the Run Steps area and uncheck “Create the parasitic netlist.” Leave “Create the PHDB” and “Create the PDB” checked.
2. Enable cumulative extraction for the PDB:
  - a. Click **Outputs** on the left panel.
  - b. Expand the Mask SVDB area and check “Generate PDB incrementally.” This is often done when extracting a netlist with mixed parasitic networks. See the sections “Extracting a Netlist with Mixed Parasitic Networks” and “Mixing Parasitics from Calibre Interactive” in the *Calibre xRC User’s Manual*.
3. Click **Run PEX** to create the PHDB and extract the initial parasitics to the PDB.
4. Disable PHDB creation:
  - a. Click **Run Control** on the left panel.
  - b. Uncheck “Create the PHDB,” so that only “Create the PDB” is checked.
5. If the Options page is not visible on the left panel, choose **Settings > Show Pages > Options**.

6. Add parasitics for specified nets to the PDB:
  - a. On the **Options** page, expand the PEX Extract Include Nets area and specify the nets to extract parasitics for.
  - b. On the **Outputs** page, specify the extracted parasitics with the Resistance/ Capacitance and Inductance dropdown menus.
  - c. Click **Run PEX**.
7. Repeat the previous step as needed.
8. Generate the parasitic netlist:
  - a. Click **Run Control** on the left panel.
  - b. Uncheck “Create the PDB.”
  - c. Check “Create the parasitic netlist.”
  - d. Click **Run PEX**.

## Results

The run information is displayed in the Transcript page. Calibre RVE opens if “Show Results in RVE” is enabled on the Run Control page.

Errors and warnings are listed in the bottom portion of the Transcript page. Right-click and select **Save As** to save the transcript.

---

### Caution

 Do not extract more parasitics to the same SVDB database after generating a netlist. Netlist generation can eliminate internal nodes, so adding new parasitic elements can introduce subtle errors.

---

## Related Topics

[Basic Calibre Interactive Usage](#)

# Setting Cell Extraction Options for Blocks

The **Blocks** page displays an alphabetical list of cells found in the source netlist files. Each cell entry contains an expansion control and the number of placements of each subcell found within its parent cell. You can select the extraction format, extraction mode, cross-annotate flag, and output netlist file for each extracted cell.

A  next to a cell name in the hierarchy tree indicates that the cell is listed in the xcell file as a primitive cell (the -P option), and the cell contents will not be extracted.

**Tip**

**i** You can discard changes in the **Blocks** page and refresh the view at any time by clicking the **Cell Options** button in the top right corner of the page and choosing **Restore from Xcells File and Re-read Netlist Files**.

---

## Prerequisites

- Calibre Interactive PEX is open.  
See “[Setting Up a Calibre Interactive PEX Run](#)” on page 202.
- The Extraction Type is Hierarchical or ADMS on the Outputs page.

## Procedure

1. (Optional) Load the rules file: On the Rules page, click  next to the rule file name.  
If the rule file includes PEX Netlist ADMS statements, the information is displayed on the **Blocks** page.
2. Choose **Settings > Show Pages > Blocks** to display the **Blocks** page.
3. Locate the cell you want to specify extraction options for by expanding the tree view as needed. If needed, click **Cell Options** in the top right corner of the page and choose **Show All Cells**.
4. Click the checkbox next to a cell name to mark the cell for extraction.
5. Double-click the cell name to open the Block Extraction Property dialog box.  
A light green box with extraction information is displayed to the right of the cell. Additional extraction options are available once the format is specified.
  - a. Select the Format (DSPF, ELD0, HSPICE, SPEF, or SDF).
  - b. Select the extraction mode (RCC, RC, R, or C). C-only extraction is not available for all formats.
  - c. Select additional options.
    - **Cross-annotate** — Flag the cell for cross-annotation.
    - **Filter X** — Only available for SDF format. Enable this checkbox to specify the FILTERX option for the PEX Netlist ADMS statement. The setting is disabled by default.
    - **Netlist File** — The output netlist filename for the cell.
    - **Top Verilog Module** — Only available for SDF format. Specify the top Verilog module for the cell. .

**Note**

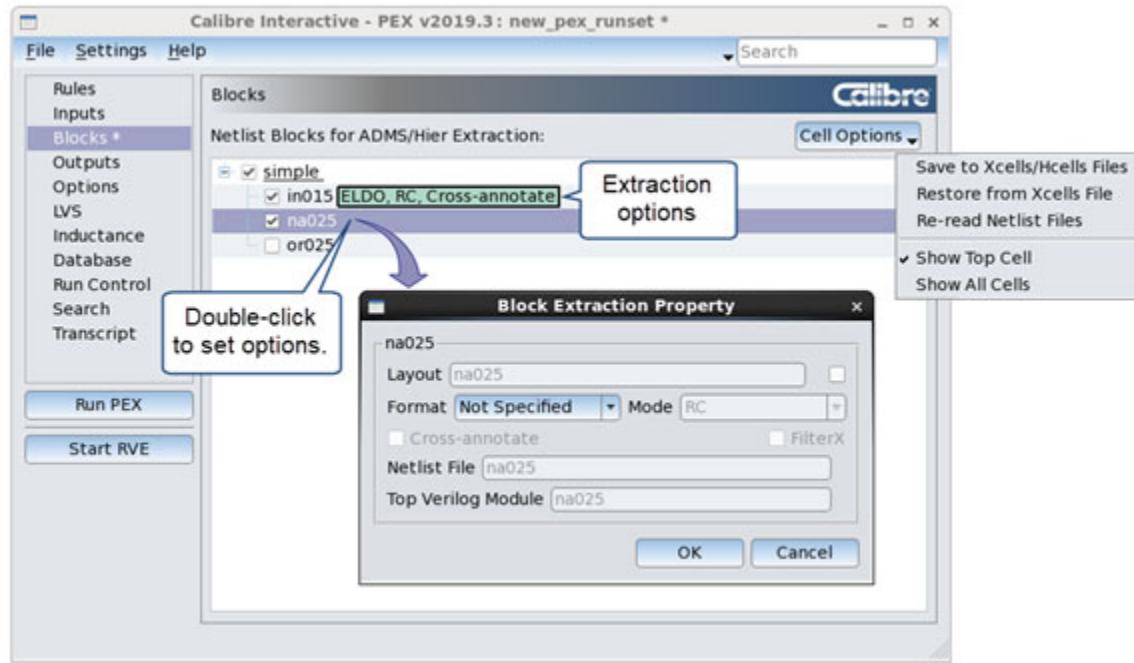
 Any changes on the **Blocks** page cause an asterisk to appear at the top of the GUI next to the runset name. The asterisk indicates that cell information has been modified and may be different from what appears in the xcell file.

6. Click **Cell Options** in the top right corner of the page and choose **Save to Xcells/Hcells Files** when done making changes. This writes your changes to the hcell and xcell files specified on the Inputs page.

The asterisk to the right of the runset name disappears when your changes have been saved to the xcell and hcell files.

A view of the **Blocks** page is shown in the following figure.

**Figure 8-2. Blocks Page in Calibre Interactive PEX**



## Setting Probe Points

You can specify probe points in your parasitic model using Calibre Interactive PEX. Probe points are used to verify timing from specific points on each net.

### Prerequisites

- You are running Calibre Interactive PEX, and have specified the rule file, input layout, and input netlist. See “[Setting Up a Calibre Interactive PEX Run](#)” on page 202.

### Procedure

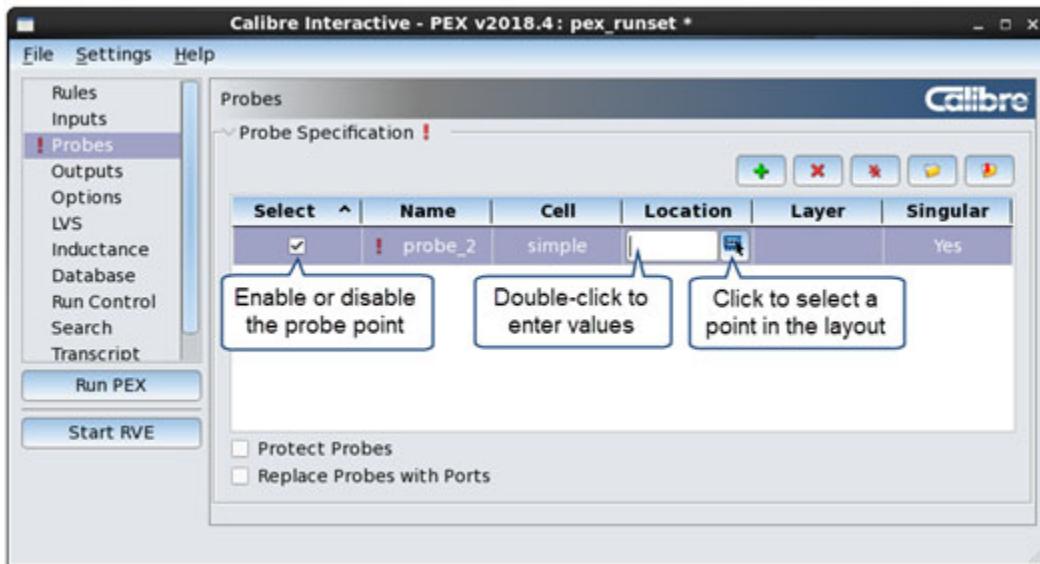
1. Choose **Settings > Show Pages > Probes** to display the Probes page.

2. Do one of the following, depending on whether you have an existing probe file:

- Click the  button to select and load an existing probe file.
- Click the  button to manually add a probe point.

The following figure shows the GUI after clicking the  button. If you loaded an existing probe file, the probe parameters are filled in.

**Figure 8-3. Calibre Interactive PEX Probes Page**



3. If any probe parameters are missing, double-click in each column to specify the required parameter. You can hover your mouse over each column header to view a tooltip for the parameter.

To provide the probe location, click the  button to choose a location in a connected layout design tool. To enter coordinates, click next to the  button and enter a space-separated coordinate pair.

4. (Optional) Specify the options “Protect Probes” and “Replace Probes with Ports” below the Probe Points table.

See [PEX Probe File](#) in the *SVRF Manual* for details.

5. When done, click the  button to save the probe file.

# Specifying Extracted Netlist Annotation of Net Geometries to Calibre View

You can choose to simplify the backannotation of net geometries into Calibre View by selecting the “Net Geometry” option for Calibre View and the “Show Parasitic Polygons” option in the Calibre View setup.

Backannotation of net geometries allows you to visualize the parasitics and their corresponding layout shapes, and also helps in identifying plot points for simulation from within the Cadence Analog Design Environment (ADE).

## Prerequisites

- A cellmap file with layer mapping statements. See the following sections:
  - [“Creating the Cellmap File” on page 498](#)
  - [“Layer Mapping Syntax” on page 506](#)
  - [“Identifying Cadence Virtuoso Layers for Mapping” on page 515](#)
- The following licenses:
  - A Calibre Connectivity Interface (CCI)
  - Calibre Interactive
  - Calibre RVE
  - Calibre nmLVS-H

## Procedure

1. In the **Outputs** page, expand the PEX Netlist section.
2. Enable the following options:
  - a. In “Netlist Format” dropdown button list, select the “CALIBREVIEW” option.
  - b. Click the “Net Geometry” checkbox.
  - c. In the “Use Names From” dropdown button list, select the “Source Names” option. Backannotation with the net geometries requires using names from the schematic netlist.
  - d. Enable these options: “LOCATION,” “RWIDTH,” “RLENGTH,” and “RLAYER.”
3. Enable the Mask SVDB option and expand the section.
4. Enable the “Generate Calibre Connectivity Interface data” option.
5. In Cadence Virtuoso, choose **Calibre > Setup > Calibre View** to open the Calibre View dialog box.

- a. Enable “Always Show Dialog” at the bottom of the form.
  - b. Click **Save** to close the dialog box. Do not click **OK**, as this creates the Calibre View.
6. Click **Run PEX** on the left panel of Calibre Interactive.
  7. When the Calibre View Setup dialog box opens, make the following selections:
    - a. Set the “View” field to “masklayout.”
    - b. Set the “Parasitic Placement” field to “Layout Location.”  
The “Show Parasitic Polygons” option should become active.
    - c. Enable the “Show Parasitic Polygons” option.
  - d. Click **Save** to close the dialog box. Do not click **OK**, as this starts the creation of the Calibre View.

## Results

Calibre places the intended devices and parasitics in the Cadence Virtuoso layout and annotates the original resistance polygons. Each polygon is split into two parts; one polygon connects to the resistor's PLUS pin, the other polygon connects to the resistor's MINUS pin, and the resistor is located in the middle of the two polygons. The polygons' layer is the same as the original resistance layer.

---

### Note

 The message “Calibre Error: Cannot create net objects in views of type schematic” is issued if creating a schematic view with Net Geometry enabled. This message is expected and the Calibre View is still created, although net objects are not created in the schematic view.

---

## Naming the Extracted Parasitic Netlist Based on the Extraction Type and Format

You can use a file naming template to name the extracted parasitic netlist based on the cell, extraction type, netlist format, and format options. For example, if the netlist format is DSPF, the file extension can be set to .dspf.

File naming templates are used to define file names based on properties of the open design. File naming templates are applied when Calibre Interactive is invoked from a design tool and a runset is initially loaded. The file naming template is not re-evaluated when options in the Calibre Interactive GUI change. Replaceable parameters can be used in the template file names. See “[Templates for File Naming](#)” on page 100 for complete information, including a table of replaceable parameters ([Table 3-6](#)).

All replaceable parameters may be used in the template for the extracted parasitic netlist filename. The following replaceable parameters are replaced with the GUI settings for the

extraction type, netlist format, and format options, and may only be used in extracted parasitic netlist filename:

**Table 8-3. Replaceable Parameters for Output PEX Netlist**

Parameter	Definition
%X	The extraction type: <ul style="list-style-type: none"> <li>• R+C+CC — rcc</li> <li>• R+C — rc</li> <li>• R — r</li> <li>• C+CC — cc</li> <li>• No R/C — norc</li> </ul>
%x	The output netlist format: <ul style="list-style-type: none"> <li>• Calibre View — cv</li> <li>• DSPF — dspf</li> <li>• Eldo — spi</li> <li>• HSPICE — sp</li> <li>• Spectre — scs</li> <li>• SPEF — spef</li> </ul>
%o	The netlist format option: <ul style="list-style-type: none"> <li>• PRIMETIME — pt</li> <li>• HSIM — hsim</li> <li>• ADITRAIL — ar</li> <li>• Net Geometry — ng</li> </ul>

## Prerequisites

- Calibre Interactive PEX is open.

## Procedure

1. Select **Settings > Show Pages > Templates** and open the **Templates** page.
2. Select the **Outputs** section.
3. Enter the desired filename template in the “PEX Netlist” field. The default is `%l.pex.netlist`, where `%l` is replaced by the name of the open layout cell.

For example, to include the extraction type (%X), format options (%o), and netlist format (%x) in the file name, enter the following for the “PEX Netlist”:

`%l.pex.%X_%o.%x`

4. Select **File > Save Runset** to save the runset.

## Results

When you invoke Calibre Interactive from a design tool and load the runset, the extracted parasitic netlist (Outputs pane, Netlist tab) is named according to the template.

For example, with the template given in Step 4, a cell name of opamp, extraction type of R+C+CC, and an output netlist format of HSPICE, the extracted netlist is named *opamp.pex.rcc\_sp*. If the netlist format is DSPF with the PRIMETIME option, the filename is *opamp.pex.rcc\_pt.dspf*.

The extracted netlist filename is not updated if the GUI options change. For example, if you load a runset then change the netlist format in the GUI, the filename does not change. The filename is generated from the template when the runset is loaded.

# Controlling PEX Step Execution

By default, the GUI executes all three PEX steps (Persistent Hierarchical Database creation, Parasitic Database creation and formatted pex netlist creation) each time you click Run PEX. You can select each step individually, if desired.

## Prerequisites

- Calibre Interactive PEX is open.

## Procedure

1. Click **Run Control** on the left panel.
2. Click the PEX Steps section.
3. Choose the desired PEX steps to execute.
4. (Optional) Set options to control PDB and formatter message output:
  - a. Click on the **Outputs** button on the left panel.,
  - b. Click the Formatter section on the **Outputs** page and expand the Options dropdown.
  - c. Enable “Display formatter warnings” to add the -fmt\_warnings option to the command line for the formatting stage.
  - d. Enable “Display formatter information” to add the -fmt\_info option to the command line for the formatting stage.

## PEX-Specific Pages in Calibre Interactive

---

Several pages are specific to Calibre Interactive PEX.

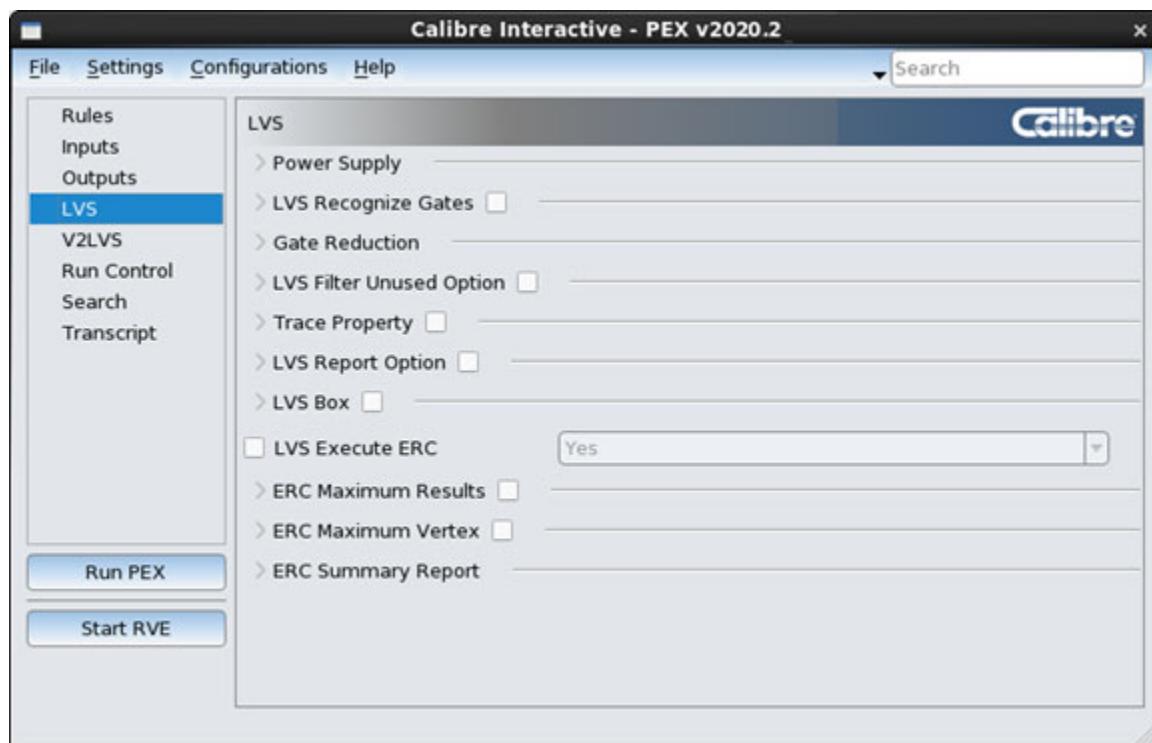
**Table 8-4. PEX-Specific Pages in Calibre Interactive**

Page	Description
Blocks	See “ <a href="#">Setting Cell Extraction Options for Blocks</a> ” on page 206.
Probes	Specify probe points in the parasitic model. See “ <a href="#">Setting Probe Points</a> ” on page 208.
LVS	Specify LVS related options for the run. These include power and ground net names, gate recognition and reduction, device filtering, LVS Report options, and LVS Box settings.
Inductance	Specify inductance extraction options. This page is only available if inductance extraction is specified on the Outputs page.
Fieldsolver	Specify fieldsolver options. This page is only available if the “xRC Mode” on the Outputs page is xACT 3D or xACT 3D Hybrid.
V2LVS	Enter options for the v2lvs translator, which translates VERILOG format source input to SPICE format.  The V2LVS page is available when VERILOG or MIXED is selected as the “Source Format” on the Inputs page.

## LVS Page in Calibre Interactive PEX

To access: Click **LVS** in the left panel of the Calibre Interactive PEX GUI

You can adjust LVS-related settings on the LVS page. These include power and ground net names, gate recognition and reduction, device filtering, LVS Report options, and LVS Box settings.



### Tip

**i** Settings in the GUI override those in the rule file. To populate the GUI with the rule file settings, click **Rules** on the left panel and click the  button (to the right of the text entry field for the rule file).

## Objects

- Power Supply  
Specify the power and ground nets. (LVS Power Name and LVS Ground Name statements)
- LVS Recognize Gates  
Enable the checkbox next to “LVS Recognize Gates” to specify gate recognition options using the GUI. See [LVS Recognize Gates](#) in the *SVRF Manual* for details.
- Gate Reduction  
Select options related to gate reduction. Use the tooltips for more information.

- LVS Filter Unused Option

Enable the checkbox next to “LVS Filter Unused Option” to control the filtering of unused devices using the GUI. Do the following for each option:

- a. Enable or disable the checkbox in the “Use” column. Hover over the option keyword to see a description.
- b. (Optional) Click in the “Design Type” column to select whether the option applies to the layout, source, or both.
- c. (Optional) Click in the “Devices” column to specify which devices the option applies to. Specify devices as “type” or “type (sub-type)”, without the quotes. For example: MP or MN(NZ)

See [LVS Filter Unused Option](#) for details.

- LVS Report Option

Enable the checkbox next to LVS Report Option and select the desired keywords in the dropdown list. Type in the filter field to search for specific options.

See [LVS Report Option](#) for details. For most Calibre runs, you should not specify any of these options, unless you actually need the information they provide. The default report is usually all that is needed for most runs.

- LVS Box

Enable the checkbox next to “LVS Box” to control LVS Box related statements using the GUI. See “[Adding and Changing LVS Box Statements in Calibre Interactive](#)” on page 181.

The option “LVS Black Box Port” is displayed if “LVS Box” is checked, and “LVS Black Box Port Depth” is displayed if “LVS Black Box Port” is checked.

- LVS Execute ERC

Enable the checkbox next to “LVS Execute ERC” to specify to execute ERC operations during the circuit extraction phase. See [LVS Execute ERC](#) for details. The option “ERC Results Database” is displayed if “LVS Execute ERC” is checked. Under that option, specify the path to the ERC results database, how to handle pseudocell results, and whether to output hierarchical ERC errors in their context cells. See [ERC Results Database](#) and [ERC Cell Name](#) for details.

- ERC Maximum Results

Enable the checkbox next to “ERC Maximum Results” to specify a maximum result count for any given check. See [ERC Maximum Results](#) for details.

- ERC Maximum Vertex

Enable the checkbox next to “ERC Maximum Vertex” to specify a maximum vertex count for any ERC result polygon written to the ERC results database. See [ERC Maximum Vertex](#) for details.

- ERC Summary Report

Enable the checkbox next to “ERC Summary Report” to specify a report file name, the writing method, and whether to open the report after the run finishes. See [ERC Summary Report](#) for details.



# Chapter 9

## Calibre Interactive PERC

---

You can specify inputs, outputs, and options for a Calibre PERC run using Calibre Interactive PERC.

Setting Up a Calibre PERC Topological Run with Calibre Interactive.....	219
Setting Up a Calibre PERC LDL Run with Calibre Interactive.....	221
Specifying Waivers for a Calibre PERC Run.....	224
Generating a Calibre PERC Rule File with High Level Checks.....	224
Using the Calibre PERC Cell-Based Flow.....	226

## Setting Up a Calibre PERC Topological Run with Calibre Interactive

You can start a Calibre PERC topological run from Calibre Interactive. You can specify inputs, outputs, and run options using the GUI.

### Prerequisites

- Calibre Interactive PERC is open. See “[Invoking the Calibre Interactive GUI](#)” on page 58.
- See “[Calibre Interactive GUI Reference](#)” on page 105 for information on setting the options on the Environment, Run Control, Triggers, and Templates pages.

---

#### Tip

 If a page name is not visible on the left panel, choose **Settings > Show Pages** and select the page name. Also make sure any options that enable the page are set.

---

### Procedure

1. (Optional) Click **File > Load Runset** to load an existing runset.

You can load a classic Calibre Interactive runset; see “[Converting Classic Calibre Interactive Runsets](#)” on page 37.

2. Click **Rules** on the left panel and specify the rule file and run directory.

If desired, click the **Load** button () to load rule file settings into the GUI.

To specify additional rule files, enable “Include Rules Files” on the Options page.

3. Click **Inputs** on the left panel and specify the following:
  - a. Specify the “Run” option as Hierarchical or Flat.
  - b. Specify the “Step” option, which specifies the analysis type:
    - **Layout** — Run Calibre PERC on a geometric layout database. Calibre PERC extracts a layout SPICE netlist and runs on that netlist.  
Layout analysis inserts the PERC Netlist LAYOUT command in the rule file and uses the -spice option on the command line.
    - **Layout Netlist** — Run Calibre PERC on an existing layout netlist. The netlist must be in SPICE format, and can be a netlist extracted during a previous Calibre PERC Layout analysis.  
Layout netlist analysis inserts the PERC Netlist SOURCE command in the rule file; the -spice option is not used on the command line.
    - **Source Netlist** — Run Calibre PERC on a source netlist.  
Source netlist analysis inserts the PERC Netlist SOURCE command in the rule file; the -spice option is not used on the command line.
  - c. Depending on the option selected for “Step,” provide information for “Layout Path” or “Source Path”.  
Enable “Export from layout viewer” or “Export from source viewer” to export the input from a connected viewer.  
If the source format is VERILOG or MIXED, enter v2lvs translator options on the V2LVS page. This page is similar to the Setup Verilog Translator dialog box (**Setup > Verilog Translator**) in classic Calibre Interactive nmLVS.
  - d. (Optional) Provide Hcell input. This option is not available in Flat runs.  
Specifying an hcell file adds the -hcell command-line option. The “Match cells by name” option adds the -automatch command-line option. See the “calibre -perc” section in the *Calibre PERC User’s Manual* for the meaning of these options in a Calibre PERC run.
  - e. (Optional) Specify Calibre PERC waiver information in the “PERC Waiver Path” area.
4. (Optional) To specify additional layout or source files, open the **Database** page, expand the “Library” area, and provide the information.

---

**Note**

 Calibre Interactive verifies that the source netlist is complete before starting the run. You can turn off this check by disabling “Check source netlist is complete before running PERC” in the Misc section of the **Preferences** page (**Settings > Show Pages > Preferences**). See “[Miscellaneous Preferences](#)” on page 131 for information on the source netlist check.

---

5. Click **Outputs** on the left panel and do the following:
  - a. Enable “Mask SVDB” to create the Standard Verification Database. This database is required in order to view and debug results with Calibre RVE.  
Specify the SVDB Directory name and other options. Hover over an option to view the corresponding keyword in the Mask SVDB Directory statement.
  - b. (Optional) Enter information regarding the PERC Report and PERC Summary Report in the “Reports” area.
6. Open the **Options** page and select desired options.
7. Click **Run Control** on the left panel.
  - a. Expand the “RVE Options” area to select options related to opening Calibre RVE.
  - b. Set run control and licensing options; see “[Run Control Page in Calibre Interactive](#)” on page 118.
8. Click **Run PERC** to start the run.

## Results

The run information is displayed in the Transcript page.

Errors and warnings are listed in the bottom portion of the Transcript page. Right-click and select **Save As** to save the transcript.

## Related Topics

[Adding and Changing LVS Box Statements in Calibre Interactive](#)

[Changing Trace Property Statements](#)

# Setting Up a Calibre PERC LDL Run with Calibre Interactive

You can start a Calibre PERC logic driven layout (LDL) run from Calibre Interactive. You can specify inputs, outputs, and run options using the GUI.

## Prerequisites

- Calibre Interactive PERC is open. See “[Invoking the Calibre Interactive GUI](#)” on page 58.
- See “[Calibre Interactive GUI Reference](#)” on page 105 for information on setting the options on the Environment, Run Control, Triggers, and Templates pages.
- See these topics in the *Calibre PERC User’s Manual* for special requirements for the Calibre PERC-LDL applications:
  - “[Current Density Checks](#)” (PERC LDL CD)
  - “[Point-to-Point Resistance Checks](#)” (PERC LDL P2P)
  - “[High-Level Checks and Topological DRC](#)” (PERC LDL DRC)

---

### Tip

 If a page name is not visible on the left panel, choose **Settings > Show Pages** and select the page name. Also make sure any options that enable the page are set.

---

## Procedure

1. (Optional) Click **File > Load Runset** to load an existing runset.

You can load a classic Calibre Interactive runset; see “[Converting Classic Calibre Interactive Runsets](#)” on page 37.

2. Click **Rules** on the left panel and specify the rule file and run directory.

If desired, click the **Load** button () to load rule file settings into the GUI.

To specify additional rule files, enable “Include Rules Files” on the Options page.

3. Click **Inputs** on the left panel and specify the following:

- a. Specify the “Run” option as LDL.

This option uses the -ldl and -hier command-line options.

- b. Specify the “Step” option, which specifies the analysis type:

- **Layout** — Run Calibre PERC LDL on a geometric layout database.

Layout analysis inserts the PERC Netlist LAYOUT command in the rule file and uses the -spice option on the command line.

- **Source Based Flow** — Run Calibre PERC LDL DRC on a source netlist. The rule file must include the “ldl::export\_perc -source” command.

- c. Provide information for the “Layout Path”. If “Source-Based Flow” is selected, also provide information for the “Source Path”.

Enable “Export from layout viewer” or “Export from source viewer” to export the input from a connected viewer.

If the source format is VERILOG or MIXED, enter v2lvs translator options on the V2LVS page. This page is similar to the Setup Verilog Translator dialog box (**Setup > Verilog Translator**) in classic Calibre Interactive nmLVS.

d. (Optional) Provide Hcell input.

Specifying an hcell file adds the -hcell command-line option. The “Match cells by name” option adds the -automatch command-line option. See the “calibre -perc” section in the *Calibre PERC User’s Manual* for the meaning of these options in a Calibre PERC run.

e. (Optional) Specify waiver information in the “PERC LDL Waiver Path” area. This option is only available for the LDL layout flow.

f. (Optional) Specify information for the PERC LDL Probe Path statement in the option area. This option is only available for the LDL layout flow.

4. (Optional) To specify additional layout or source files, open the **Database** page, expand the Library area, and provide the information.

5. Click **Outputs** on the left panel and do the following:

a. Enable “DFM Database” to create the DFM Database. This database is required in order to view and debug results with Calibre RVE.

Specify the DFM Database Directory name and Keywords. Type in the filter field to search for specific options, and hover over a keyword to view a short description.

b. (Optional) Enter information regarding the PERC Report and PERC Summary Report in the “Reports” area.

6. Click **Options** on the left panel and select desired options.

7. Click **Run Control** on the left panel.

a. Expand the “RVE Options” area to select options related to opening Calibre RVE.

b. Set run control and licensing options; see “[Run Control Page in Calibre Interactive](#)” on page 118.

8. Click **Run PERC** to start the run.

## Results

The run information is displayed in the Transcript page.

Errors and warnings are listed in the bottom portion of the Transcript page. Right-click and select **Save As** to save the transcript.

## Related Topics

- [Adding and Changing LVS Box Statements in Calibre Interactive](#)
- [Changing Trace Property Statements](#)

# Specifying Waivers for a Calibre PERC Run

You can specify a Calibre PERC Waiver description file in Calibre Interactive PERC. This causes the Calibre PERC Waiver flow to be used during the Calibre PERC run.

## Prerequisites

- A Calibre Auto-Waivers license in addition to the Calibre PERC license.
- A Calibre PERC waiver description file. See “[Calibre PERC Waiver Flows](#)” in the *Calibre PERC User’s Manual*.

## Procedure

1. Click the **Inputs** button on the left pane of Calibre Interactive.
2. Enable “PERC Waiver Path” and enter the path to the waiver description file.
3. (Optional) If the run type on the Inputs page is set to PERC-LDL, enable “PERC LDL Waiver Path” and enter the path to the file. The waivers in this file apply to geometric (RDB) results.

## Results

The [PERC Waiver Path](#) statement is added to the control file for the run, and optionally the [PERC LDL Waiver Path](#) statement.

# Generating a Calibre PERC Rule File with High Level Checks

You can generate a rule file for a Calibre PERC run with the Calibre PERC Rule Generator GUI. The rule checks that can be included are part of the high-level checks flow in Calibre PERC. You specify the various parameters and input files for each rule check using the rule generator GUI.

These sections in the *Calibre PERC User’s Manual* provide important information about the flow:

- “[High-Level Checks](#)”
- “[High-Level Check Types](#)”
- “[Design Flow Selection](#)”

## Prerequisites

- Calibre PERC and Calibre Interactive licenses.
- An LVS rule file.
  - (Optional) A parasitic resistance extraction rule file if you are running a check that requires it. These rules should be included in the LVS rules.
- Know the checks and design flow you want to run.
- Various setup files such as net voltages, layer lists, constraint tables, as required by the checks you plan to run.
- Calibre Interactive PERC is running.

## Procedure

1. Click **Rules** on the left panel of Calibre Interactive and enter the desired name for the generated Calibre PERC rule file. This file should not exist—if it does exist, Calibre Interactive automatically assigns a new name so that your previous rule file is not overwritten.
2. Click **Inputs** on the left panel of Calibre Interactive and do the following:
  - a. Specify the Run and Step options.
  - b. Enter the corresponding input information. The format for the input design must be GDS, OASIS, LEFDEF, or SPICE.

These values are automatically entered into the Calibre PERC Rule Generator GUI and cannot be changed. The design flow is determined by the Run and Step selections.
3. Click **Outputs** on the left panel of Calibre Interactive and do the following:
  - a. Specify the SVDB Directory name.
  - b. Expand the Reports area and specify the PERC Report filename.
4. (Optional) Specify the filename for the Calibre PERC Rule Generator setup file. The setup file saves the settings in the Calibre PERC Rule Generator GUI.
  - a. Click **Options** on the left panel.
  - b. Expand the Rule Generator area.
  - c. Specify the binary “PERC Rule Generator” setup file (.prg file). The default setup file is *percrulegen.prg*.

If the file specified in the “PERC Rule Generator File” field exists, the settings are automatically loaded when you start the Calibre PERC Rule Generator GUI.

The updated settings in the Calibre PERC Rule Generator GUI are automatically saved to the specified setup file when the rule file is generated.

5. Choose **Settings > Generate PERC rules** to open the Calibre PERC Rule Generator GUI.

---

**Note**

 When invoking the Calibre PERC Rule Generator GUI from Calibre Interactive, settings on the Rules, Inputs, and Outputs pages of Calibre Interactive take precedence over the corresponding settings in the *.prg* file.

---

6. Choose **Inputs** in the left panel of the Calibre PERC Rule Generator GUI and enter your LVS rule file in the Rules File field.

The Flow, Format, Layout File, and Top Cell entries are taken from the Calibre Interactive settings and cannot be changed.

7. Choose **Checks** in the left panel and add the desired checks.

See Steps 6 through 8 in the procedure “[Running the Rule File Generator GUI](#)” in the *Calibre PERC User’s Manual* for more information.

8. Choose **Outputs** in the left panel and verify the name of the output DFM Database.

The Generated Rule File, SVDB Database, and PERC Report entries are read from the Calibre Interactive GUI and cannot be changed.

9. Click **Generate** to generate the rule file.

The Calibre PERC Rule Generator GUI closes automatically. The Calibre Interactive “PERC Rules File” field on the Rules page is automatically populated with the generated rule file name.

10. Click **Run PERC**.

## Results

A rule file of the name provided in the “Generated Rule File” field is written to the working directory.

The Calibre PERC Rule Generator GUI settings are automatically saved to the setup file specified on the **Options** page (see Step 4).

After the run completes, you can view the results in Calibre RVE for PERC.

## Using the Calibre PERC Cell-Based Flow

You can specify the Calibre PERC cell-based flow in Calibre Interactive. This flow is part of the Calibre PERC LDL (Logic-Driven Layout) functionality. The flow assumes LEF/DEF design input instead of SPICE and performs topological checking without considering devices.

## Prerequisites

- A rule file set up for the Calibre PERC cell-based flow, as described in “[Cell-Based Checks](#)” in the *Calibre PERC User’s Manual*.
- Layout is in LEF/DEF format.
- (Optional) Define environment variables for database read. See “[FDI Environment Variables](#)” in the *Calibre Layout Comparison and Translation Guide*. These options can be set on OA/LEFDEF page in Calibre Interactive if not set with environment variables.
- (Optional) A cell info file. This is needed when you are using GDS input for connectivity of certain cells. See “[Cell Information File Format](#)” in the *Calibre PERC User’s Manual*.

## Procedure

1. Specify the rule file and run directory on the Rules page.
2. Click **Inputs** on the left panel.
3. Choose a run type of LDL.
4. In the Layout Path area, select LEFDEF for the Layout Format.
5. Fill in the DEF and LEF input information.

See “[Providing LEF/DEF Input with Calibre Interactive](#)” on page 75. Database read options are set on the OA/LEFDEF page.

6. Enable the “Enable Cell-Based flow” checkbox. This adds the -soc option to the command line.
7. (Optional) Enable “Cell Info File” and specify a cell info file. This adds the option -cell\_info\_file and the specified file to the command line.

## Related Topics

[Setting Up a Calibre PERC LDL Run with Calibre Interactive](#)

[OA/LEFDEF and OPENACCESS Pages in Calibre Interactive](#)



# Chapter 10

## GUI Customization for Calibre Interactive

---

You can customize the GUI by loading a configuration file. You can specify controls as hidden or not editable and define custom GUI panes. You can also specify #DEFINE and Variable statements that are added to the Calibre Interactive control file.

<b>Loading a Configuration File in Calibre Interactive .....</b>	<b>230</b>
<b>Loading a Classic Tcl Customization File in Calibre Interactive .....</b>	<b>232</b>
<b>Customizing the GUI With the Configuration Editor .....</b>	<b>233</b>
<b>Saving GUI Settings from Custom Controls.....</b>	<b>237</b>
<b>Configuration File Reference .....</b>	<b>239</b>
Configuration File Format for Calibre Interactive GUI Customization .....	240
Global Object Definitions.....	243
Arrays and Maps.....	244
Tool Properties .....	247
Option Properties .....	250
GUI Signals.....	252
Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive.....	252
Finding Option Names to Use in the Configuration File .....	254
<b>Function Reference for GUI Customization .....</b>	<b>256</b>
General Configuration File Functions .....	257
Configuration Commands to Execute Functions and Commands .....	293
Configuration File Commands to Add #DEFINE and #UNDEFINE Statements .....	303
Configuration File Functions to Add Variable Statements.....	342
Configuration File Commands to Add Buttons .....	382
Configuration File Commands for Trigger Functions .....	390
<b>Guide and Examples for Converting Tcl Customization Files to New Configuration Files</b>	<b>397</b>
Correspondence of Tcl Customization Commands to New Configuration Commands ..	397
Adaptation Example: #DEFINE and Variable Statements .....	401
Adaptation Example: Move and Set GUI Options .....	403
Adaptation Example: Callback Function .....	407
<b>Troubleshooting Configuration Files for the Calibre Interactive GUI .....</b>	<b>411</b>

# Loading a Configuration File in Calibre Interactive

You can load a configuration file with a command line option, by specifying the configuration file with an environment variable, loading it from the GUI, or loading a runset with a configuration file specified.

Multiple configuration files can be loaded with both methods. Files are loaded in order (left to right as specified) and settings in the most recently loaded (rightmost) file take precedence if there is a conflict.

Configuration files specified on the command line have the highest precedence, while configuration files specified in a runset have the lowest precedence.

If you invoke Calibre Interactive from a design tool, you must specify the configuration file with an environment variable, specify it in the runset, or load it manually from the GUI.

---

## Note

 Configuration files are loaded first and in the order specified, then runsets are loaded in the order specified. If configuration files are specified in a runset, the configuration files are loaded before the runset settings. If any settings conflict, the setting in the most recently loaded file is used. For example, if a configuration file and a runset both specify a value for the same GUI setting, the value in the runset is used.

---

## Prerequisites

- A configuration file. See “[GUI Customization for Calibre Interactive](#)” on page 229.

## Procedure

Do one of the following. Assume you have configuration files named *drc.cfg*, *lvs1.cfg*, and *lvs2.cfg*.

**Table 10-1. Loading a Configuration File**

Method	Steps
Command line specification (highest precedence)	Specify the configuration file with the -config command line option. You may enter more than one configuration file on the command line.  For example: <code>calibre -gui -drc -runset runset -config drc.cfg</code> <code>calibre -gui -lvs -config lvs1.cfg lvs2.cfg</code>

**Table 10-1. Loading a Configuration File (cont.)**

Method	Steps
GUI and runset specification	<p>You can specify a configuration file on the <b>Preferences</b> page in the GUI.</p> <ol style="list-style-type: none"> <li>1. Click <b>Preferences</b> on the left panel. Select <b>Settings &gt; Show Pages &gt; Preferences</b> if this page is not visible.</li> <li>2. Enable the “Configuration” checkbox.</li> <li>3. Specify one or more configuration files. The configuration files are loaded in the order specified.</li> <li>4. Click <b>Load Configuration Files</b>.</li> </ol> <p>The configuration files are saved to the runset. When the “Configuration” checkbox is checked, the configuration files are loaded the next time you load the runset.</p> <p> <b>Note:</b> Configuration files specified on the command line or with an environment variable have higher precedence. If either is specified, configuration files specified in the runset are not loaded and the <b>Load Configuration Files</b> button is not present.</p>
Environment variable specification	<ol style="list-style-type: none"> <li>1. Specify the configuration file(s) with one the following environment variables before invoking the GUI:           <ul style="list-style-type: none"> <li>MGC_CALIBRE_DRC_CONFIG_FILE</li> <li>MGC_CALIBRE_LVS_CONFIG_FILE</li> <li>MGC_CALIBRE_PERC_CONFIG_FILE</li> <li>MGC_CALIBRE_PEX_CONFIG_FILE</li> </ul>           For example:  <pre>setenv MGC_CALIBRE_DRC_CONFIG_FILE configDRC.cfg</pre>           Specify multiple configuration files as a space separated list enclosed in quotes.         </li> <li>2. Invoke Calibre Interactive.            See “<a href="#">Invoking the Calibre Interactive GUI</a>” on page 58</li> </ol>
Toolbar invocation (lowest precedence)	<p>From the toolbar in the GUI, select <b>Configurations &gt; Load</b> and specify a configuration file. You may select more than one configuration file.</p> <p>If a configuration is currently loaded when this option is selected, it is unloaded and all unsaved changes made to the current runset are discarded.</p> <p>To reload both the current runset and the current configuration, select <b>File &gt; Reset</b>.</p>

## Results

The configuration file(s) are loaded. If a runset is specified on the command line, it is loaded after the configuration file is initialized.

# Loading a Classic Tcl Customization File in Calibre Interactive

Calibre Interactive can load a Tcl customization file that was developed for classic Calibre Interactive. A new Custom page is created within the GUI and the customization controls are placed on the Custom page.

There are three methods to load the customization file: 1) with a command-line switch, 2) using a classic runset that defines the customization file, or 3) with an environment variable. A customization file specified on the command line has the highest precedence, while a customization file specified with an environment variable has lowest precedence.

The CUSTOM::CHECK command is not supported.

If you invoke Calibre Interactive from a design tool, you must specify the customization file with an environment variable or specify it in the runset.

## Prerequisites

- A Tcl customization file developed for classic Calibre Interactive.

## Procedure

Do one of the following.

**Table 10-2. Loading a Classic Tcl Customization File**

Method	Steps
Command line specification (highest precedence)	Specify the Tcl customization file with the -custom command line option. For batch operation, use the -customgui command line option.  For example, with interactive operation:  <code>calibre -gui -drc -runset runset -custom custom_drc.tcl</code>  Batch operation:  <code>calibre -gui -lvs -runset lvs.runset -customgui custom_lvs.tcl -batch</code>
Runset specification	You can load a classic runset that has a Tcl customization file defined in the runset. (In classic Calibre Interactive, customization files that are specified on the <b>Startup</b> tab of the Setup Preferences dialog box are saved to the runset.)  <code>calibre -gui -drc -runset runset_with_customfile</code>

**Table 10-2. Loading a Classic Tcl Customization File (cont.)**

<b>Method</b>	<b>Steps</b>
Environment variable specification (lowest precedence)	<p>1. Specify the Tcl customization file with the MGC_CALIBRE_CUSTOMIZATION_FILE environment variables before invoking the GUI  For example:  <pre>setenv MGC_CALIBRE_CUSTOMIZATION_FILE custom_drc.tcl</pre></p> <p>2. Invoke Calibre Interactive.  See “<a href="#">Invoking the Calibre Interactive GUI</a>” on page 58.</p>

## Results

The customization controls are displayed on the new Custom page. If a runset is specified on the command line, it is loaded after the Custom page is created.

In batch operation (with the -customgui option), a GUI with only the Custom page and the Run button is displayed. When the Run button is clicked the run continues in batch mode using the settings on the Custom page and in the runset.

Problems in converting the Tcl customization file are noted on the Transcript page.

The CUSTOM::SETWINGEOM command sets the size of the Calibre Interactive GUI, because there is not a separate customization window, as in classic Calibre Interactive. However, the minimum dimensions of the GUI are different than for the customization window, which may cause differences in the expected GUI size.

# Customizing the GUI With the Configuration Editor

You can easily customize the Calibre Interactive GUI using the Calibre Interactive Configuration Editor. You can add and hide GUI pages, move options to different pages, hide options, and make other GUI changes. You can also add controls for #DEFINE and Variable statements.

## Prerequisites

- “[Requirements for Calibre Interactive](#)” on page 24

## Procedure

- Open the configuration editor using the following command line syntax:

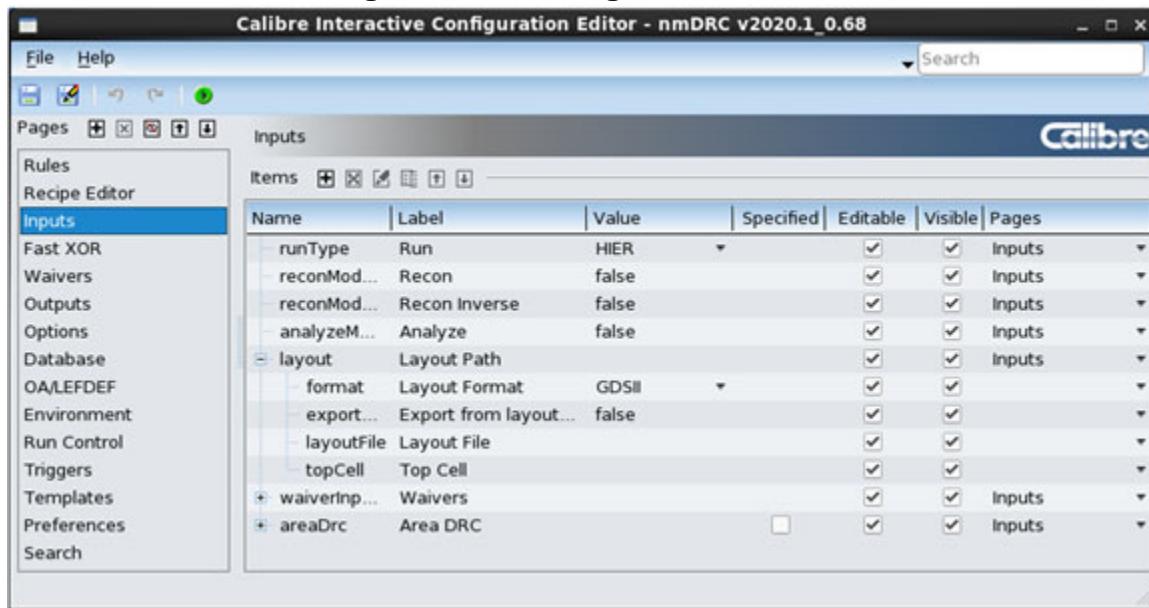
```
calibre -gui {-drc | -lvs | -perc | -pex } -config_editor
[-config config.cfg ...] [-custom custom.tcl ...]
```

The optional -config switch specifies one or more existing configuration files to load.

The optional `-custom` switch specifies one or more existing classic Tcl customization files to load. The controls from the customization file are placed on a new page named **Custom**. Some editor actions, such as delete, are not available for controls created by a Tcl customization file.

See the “Results” section for important information on how the updated configuration is saved, especially for the case when a classic Tcl customization file is loaded.

**Figure 10-1. Configuration Editor**



**Note** As with the actual GUI, the visibility of some options depends on the state of another option. For example, although the Waivers page is listed in the left panel, there are no options displayed on the page if “Use Waivers” is set to false.

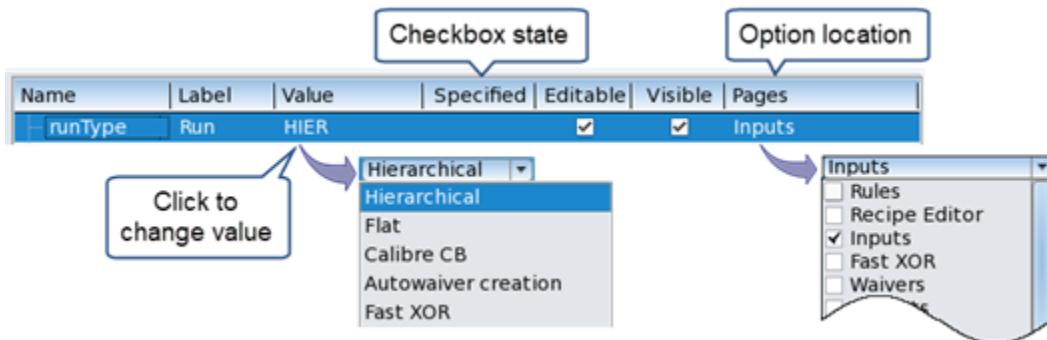
---

You can load configuration files under **File > Load Configuration** in the Configuration Editor toolbar. If you have a configuration currently open when you load configurations, you have the option of replacing or appending it. Appending configurations does not reset GUI changes made in the original configuration that are not affected by the new configurations.

You can do any of the actions in the following steps. Use the buttons for Undo and Redo.

2. Set values, options, and the location for controls:

Click in any cell to change the initial settings for the control.



You can choose multiple pages for an item to be displayed on.

3. Move controls with one of these methods:

To change the page:

- Drag the control to a page name in the left panel. You can drag and drop multiple items.
- In the Pages column, select the new page or multiple pages in the dropdown list.
- Right-click and choose **Change Item Page**.

To change the item order within the page:

- Drag the item to the new location.
- Use the up/down buttons in the Items toolbar.
- Right-click and choose **Move Item Up** or **Move Item Down**.

4. Add, delete, show/hide, and move pages with the Pages toolbar ( ). Built-in pages cannot be deleted.

5. Add custom GUI controls, such as for #DEFINE and Variable statements, labels, headers, and separators.

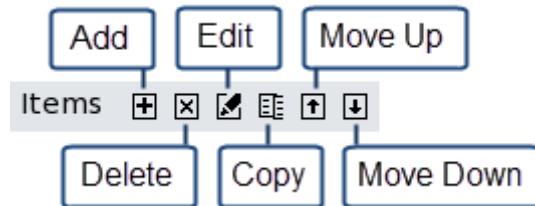
- a. Click the icon in the Items toolbar to add a custom control.
- b. Select the item Type:
  - **Define** — Adds a control for a #DEFINE statement. See “[Configuration File Commands to Add #DEFINE and #UNDEFINE Statements](#)” on page 303.
  - **Variable** — Adds a control for a Variable statement. See “[Configuration File Functions to Add Variable Statements](#)” on page 342.
  - **Label** — Adds a text label. See [addLabel](#).

- **Header** — Adds a text label with a shaded background. See [addHeader](#).
  - **Separator** — Adds a separator line. See [addSeparator](#).
  - **Table** — Adds a table. See [addTable](#).
- c. For Variable and Define controls, choose the sub-type.
  - d. Choose other options, which may depend on the control type.

For common options, see “[Option Properties](#)” on page 250. For information on “Add Master,” see [addMaster](#) and [addMasterSelect](#).

Items are added at the end of the page—the order can be changed with the buttons in the toolbar, or by drag and drop.

6. Use the other toolbar icons for more actions. Delete, Edit, and Copy are only available for custom items.



7. Click (or choose **File > Preview Configuration**) to view the GUI with the customization applied.
8. Click (**Save**) or (**Save As**) to save the custom settings to a configuration file.  
If you loaded multiple configuration files or loaded a classic Tcl customization file, only **Save As** is available.

## Results

The saved configuration file can be edited in any text editor, if needed. See “[Configuration File Reference](#)” on page 239.

The following behavior occurs when you load a configuration file or classic Tcl customization file in the configuration editor and save the updated configuration:

- If you load a single configuration file and click **Save**, the saved file includes both original settings and updates made during the session.
- If you load multiple configuration files, the final *combined* configuration settings are saved in the file specified with the **Save As** action. The **Save** selection is not available.
- If you load one or more classic Tcl customization files, only the configuration *changes* from the configuration editor session are saved in the file specified with the **Save As** action. The **Save** selection is not available. To use the updated configuration in a GUI

session, you must load *both* the original Tcl customization file and the configuration file saved from the editor session.

For example, to open the Calibre Interactive nmDRC GUI with a configuration file named *myConfig.config*, do the following:

```
calibre -gui -drc -config myConfig.config
```

If you loaded a classic Tcl customization file named *custom.tcl* in the editor and saved configuration updates to the file *myConfigUpdates.config*, you must load both files to use the updated configuration in a GUI session. For example:

```
calibre -gui -drc -custom custom.tcl -config myConfigUpdates.config
```

---

**Note**

 On IXL platforms, the Prompt String setting for TextBox Variable and Define controls does not work. The IXL platform is deprecated.

---

## Saving GUI Settings from Custom Controls

The configuration file in Calibre Interactive is used to customize the GUI. Changes made to custom controls are saved to the Calibre Interactive runset. By default, only changed settings are saved to the runset.

For example, suppose a configuration file includes this command to add an integer #DEFINE statement to the Calibre Interactive control file:

```
var test_a_def = tool.addIntegerDefine("test_a")
test_a_def.value = 1
```

“test\_a” is the variable name for the #DEFINE statement.

If the assigned value is changed to 2 in the GUI, this line is included in the Calibre Interactive control file:

```
#define "test_a" 2
```

When the runset is saved, this line is included in the runset, where the variable name test\_a is used as the object name:

```
test_a.value = 2
```

The behavior for commands that add Variable statements is similar.

If you make changes to controls added by a configuration file, these settings are saved in the runset. You can return to a saved GUI setup by loading both the saved runset and the configuration file.

## Allowed Characters in Variable and Define Names

In the preceding addIntegerDefine statement, the variable name for the #DEFINE statement is “test\_a”. The recommended best practice is for variable names to include only alphanumeric and underscore (\_) characters.

However, for backward compatibility with existing classic Calibre Interactive customization files, spaces and parentheses are also allowed in variable names. For example, the variable name in the following command is “test(a)”:

```
var test_a_def2 = tool.addIntegerDefine("test (a) ")
```

This line is written to the control file when the value is 2:

```
#define "test (a)" 2
```

However, spaces and parentheses are not permitted in the object names used in the runset, due to JavaScript parsing rules. Spaces are removed and parentheses are changed to underscores when saved to the runset. The corresponding line in the runset for this example is the following:

```
test_a_.value = 2
```

The use of spaces and parentheses in variable names is not recommended. If they are used, make sure that the corresponding object names are unique. For example, the variable names “test(a)” and “test\_a\_” have the same object names and therefore should not be used in the same configuration file.

# Configuration File Reference

---

The configuration file in Calibre Interactive contains commands and specifications to customize the appearance of the GUI.

<b>Configuration File Format for Calibre Interactive GUI Customization .....</b>	<b>240</b>
<b>Global Object Definitions .....</b>	<b>243</b>
<b>Arrays and Maps .....</b>	<b>244</b>
<b>Tool Properties.....</b>	<b>247</b>
<b>Option Properties.....</b>	<b>250</b>
<b>GUI Signals.....</b>	<b>252</b>
<b>Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive .....</b>	<b>252</b>
<b>Finding Option Names to Use in the Configuration File .....</b>	<b>254</b>

# Configuration File Format for Calibre Interactive GUI Customization

Input file for the Calibre Interactive GUI

The configuration file specifies a custom GUI configuration. You can specify new pages, hide pages, control the visibility of options, and add controls for #DEFINE and Variable statements. You can also connect a function to a GUI action such as changing a value.

The topics are also relevant:

- “[Loading a Configuration File in Calibre Interactive](#)” on page 230
- “[Finding Option Names to Use in the Configuration File](#)” on page 254

## Format

### File Syntax

The configuration file has the following structure, where detailed information about the different elements is provided in the “Parameters” section.

```
Configuration {  
    // Global object definitions  
    // Configuration object definitions  
    // Function definitions  
  
    onInitialized: {  
        // JavaScript commands  
  
        // Configuration API calls  
  
        // Function calls  
  
        // Property assignments  
    }  
}
```

### Formatting Rules

- Single line comments start with two slashes (//) and extend to the end of the line.
- Multiple line comments start with /\* and end with \*/. Comments cannot be nested.
- String values must be enclosed in quotes. This is generally shown in the command syntax.
- A newline within a string value is specified with “\n”. For example: “Test\nNewline”
- Whitespace, including newlines, is ignored. This means there is no need for a line continuation character.

- The semicolon (;) separates executable statements in JavaScript<sup>TM</sup><sup>1</sup>. While the semicolon is optional in some circumstances, it is good practice to include it at the end of executable statements.

## Parameters

- **Configuration { }**

Required. It must contain the onInitialized section.

The Configuration section can contain the following optional elements:

- Global object definitions

Objects defined in the Configuration section have a global scope. See “[Global Object Definitions](#)” on page 243 for the supported syntax.

- Configuration object definitions

The configuration objects FileIO and Command are defined in the Configuration section. See [FileIO](#), [launchCommand](#), and [launchCommandReturnString](#).

- Function definitions

Function definitions are written in JavaScript with the following syntax:

```
function functionName(arglist) {  
    // JavaScript statements  
}
```

The contents of the function definition follow the same rules as the onInitialized section.

- **onInitialized: { }**

Required. The contents of the onInitialized section are executed when the tool is initialized. If a runset is specified, it is loaded after the onInitialized section is executed. The onInitialized section can contain the following elements:

- JavaScript commands

Standard JavaScript constructs are allowed. See <https://www.w3schools.com/js/> or one of the many other online references.

- Configuration API calls

The configuration API includes commands to add #DEFINE and Variable statements, add buttons, add GUI pages, print to the transcript, and more. See “[Function Reference for GUI Customization](#)” on page 256.

- Function calls

---

1. JavaScript is a registered trademark of the Oracle Corporation.

Functions defined in the Configuration section can be called from the onInitialized section or called from other functions.

Functions can also be attached to a GUI signal with the `connect()` function. See “[GUI Signals](#)” on page 252.

You can attach a callback function to the Run button with the tool property `tool.runCallback`. See “[Example: Attach Callback Function to Run Button With tool.runCallback](#)” on page 249.

- o Property assignments

Property assignments can be made for the tool in general or for a particular GUI object using this general syntax:

```
tool.property = value
tool.option.property = value
tool.option.sub_option.property = value
```

Tool properties specify overall properties for the GUI, such as the visible GUI pages. Option properties specify properties for a GUI control, such as the value, visibility, or label. See “[Tool Properties](#)” on page 247 and “[Option Properties](#)” on page 250.

## Examples

```
Configuration {
    onInitialized: {
        // Create new combined "Rules and Input" page
        // Create new Netlist page and move Netlist options to it
        // Hide standard Rules, Inputs, Environment, and Run Control pages
        tool.pages = ["Rules and Inputs", "Outputs", "Netlist", "Options"]

        // Move rule file, run dir, and input settings to Rules and Inputs page
        tool.rulesFile.pages = ["Rules and Inputs"]
        tool.runDir.pages = ["Rules and Inputs"]
        tool.layout.pages = ["Rules and Inputs"]
        tool.source.pages = ["Rules and Inputs"]
        tool.hCells.pages = ["Rules and Inputs"]
        tool.xCells.pages = ["Rules and Inputs"]

        // Move Netlist options to Netlist page
        tool.pexNetlist.pages = ["Netlist"]

        //duplicate the Include and Exclude options on
        //Netlist and Options pages
        tool.pexExcludeNets.pages = ["Netlist", "Options"]
        tool.pexIncludeNets.pages = ["Netlist", "Options"]

        // Specify and lock out SVDB settings
        tool.svdb.specified = true
        tool.svdb.editable = false
        tool.svdb.dirPath.value = "svdb"
        tool.svdb.asciiXref.value = true
    }
}
```

Also see the examples provided with the command references.

## Global Object Definitions

Objects defined in the Configuration section of the Calibre Interactive configuration file have a global scope. A global object is declared with the “property variant” keyword.

### Uninitialized Object

```
property variant myVal;  
property variant myStr;
```

A value is assigned in the onInitialized section:

```
myVal = 25;  
myStr = "top";
```

### Initialized Object

An object can be declared and initialized with a numeric value, a string, a boolean, an array, or a map of key-value pairs. A JavaScript expression can be used to initialize the value. An example is shown for each.

A numeric value:

```
property variant dist: 4.5;  
property variant count: 1;  
property variant count: 100 + 1; // initialize with expression
```

A string value, which must be enclosed in quotes:

```
property variant companyName: "ACME";
```

A Boolean value, which must be true or false:

```
property variant isDRC: false;
```

An array, which can have numeric, string, and Boolean values, or the objects returned by a configuration API call that creates a GUI control:

```
property variant arrShapes: ["Square", "Circle", "Triangle"];  
property variant arrMixed: ["box", 5, 4]
```

A map:

```
property variant mapTest: {"name": "M1", "width": 100}
```

See “[Arrays and Maps](#)” on page 244 for important information on manipulating arrays and maps.

## Arrays and Maps

Arrays and key-value maps can be used in the Calibre Interactive configuration file with some limitations.

The examples make use of the tool.[transcript](#) command, which prints a message to the Transcript page of Calibre Interactive.

### Array Syntax

An array has the following syntax:

```
[value0, value1, ...]
```

where the elements can be a mixture of numeric, string, and Boolean values, and objects returned by one of the configuration file API calls that create a GUI control. String values must be enclosed in quotes. Boolean values must have the value true or false. Array indexing starts at zero.

### Map Syntax

A map is an array of key-value pairs, with the following syntax:

```
{key1: value1, key2: value2, ...}
```

where the keys can be numeric or string values and the values may be numeric, string, or Boolean values, or objects returned by one of the configuration file API calls that create a GUI control. Key values must be unique within the map. String values must be enclosed in quotes. Boolean values must have the value true or false.

### Declaring Arrays and Maps

The method for declaring an array or map depends on whether the declaration is in the Configuration or onInitialized section of the configuration file.

Object names must start with a lower-case letter [a-z].

- In the Configuration section

An array or map is declared with the “property variant” keyword in the Configuration section and has a global scope. The following example declares and initializes the array arrNums and the map mapTest.

```
property variant arrNums: [1, 2, 3, "four", "five"]
property variant mapTest: { "shape": "square", "width": 20}
```

- In the onInitialized section

An array or map is declared with the “var” keyword in the onInitialized section and has a local scope. The following example declares and initializes the array arrTemp and the map mapTemp.

```
var arrTemp = [10, 9, 8, 7];
var mapTemp = {"test": 1, 42: "forty-two"};
```

## Accessing Arrays and Maps

Array elements are accessed using the index number enclosed in brackets ([ ]), where the index starts at zero. For example:

```
tool.transcript("The second element of arrNums is " + arrNums[1]);
```

Map values are accessed using the key enclosed in brackets. For example, using the mapTemp object declared in the previous section:

```
tool.transcript("mapTemp test = " + mapTemp["test"] +
    " mapTemp 42 = " + mapTemp[42]);
```

## Assigning Values to Arrays and Maps

The values in a globally declared array or map cannot be changed. To change a value, you can copy the global array or map object to a locally declared object, assign values, and copy the local array or map object back to the global object. This is demonstrated in the following example.

### Example

This example demonstrates global and local arrays and maps, value assignments, and the JavaScript for loop used to access the array and map values.

```
Configuration {
    property variant arrNums: [1, 2, 3, "four", "five"]
    property variant mapTest: { "shape": "square", "width": 20}

    onInitialized: {
        var i;
        var prop;
        var arrTemp = [10, 9, 8, 7];
        var mapTemp = {"test": 1, 42: "forty-two"};

        // uninitialized objects have the value undefined
        tool.transcript("prop = " + prop + " index i = " + i);

        tool.transcript("The second element of arrNums is " + arrNums[1]);
        tool.transcript("mapTemp test = " + mapTemp["test"] +
            " mapTemp 42 = " + mapTemp[42]);
```

```
tool.transcript("\ninitial arrTemp");
for (i=0; i<arrTemp.length; i++) {
    tool.transcript(arrTemp[i]);
}
tool.transcript("\ninitial mapTemp");
for (prop in mapTemp) {
    tool.transcript(prop + " is " + mapTemp[prop]);
}

tool.transcript("\ninitial arrNums");
for (i=0; i<arrNums.length; i++) {
    tool.transcript(arrNums[i]);
}
tool.transcript("\ninitial mapTest");
for (prop in mapTest) {
    tool.transcript(prop + " is " + mapTest[prop]);
}

// change an array value
arrTemp = arrNums;                      // copy values to temp array
arrTemp[2] = "three";                    // change an array value
arrNums = arrTemp;                      // copy temp array back to global array
tool.transcript("\nnnew arrNums");
for (i=0; i<arrNums.length; i++) {
    tool.transcript(arrNums[i]);
}

// add to the map and change a value
mapTemp = mapTest;
mapTemp["shape"] = "circle";   // change value for shape key
mapTemp["width"] = 30;        //change value for width key
mapTemp["metal"] = "copper"; // add a key and value
mapTest = mapTemp;
tool.transcript("\nnnew mapTest");
for (prop in mapTest) {
    tool.transcript(prop + " is " + mapTest[prop]);
}
}
```

The following output is printed to the Transcript page:

```
prop = undefined index i = undefined
The second element of arrNums is 2
mapTemp test = 1 mapTemp 42 = forty-two

initial arrTemp
10
9
8
7

initial mapTemp
42 is forty-two
test is 1
```

```
initial arrNums
1
2
3
four
five

initial mapTest
shape is square
width is 20

new arrNums
1
2
three
four
five

new mapTest
shape is circle
width is 30
metal is copper
```

## Tool Properties

Certain properties apply at the tool level to set overall behavior and appearance of the Calibre Interactive GUI. For example, you can specify the pages that are visible in the GUI.

Tool properties are specified as follows, where the available properties are given in the table.

```
tool.property = value
```

**Table 10-3. Tool Properties**

Property	Allowed Values	Description
blockRun	true   <u>false</u>	Specifies whether to block the Calibre run when the <b>Run &lt;app&gt;</b> button is clicked and display a pop up dialog box. The pop up dialog box displays the string property blockReason.  The default value is false.  See the example in <a href="#">getCalibreVersion</a> .  A run is automatically blocked if a required option is missing information.
blockReason	string	The text displayed in the pop up dialog box that is displayed when the <b>Run &lt;app&gt;</b> button is clicked and blockRun is set to true.

**Table 10-3. Tool Properties (cont.)**

Property	Allowed Values	Description
defaultPages	[“name”, “name”, ...]	Defines the default page or pages; if not defined, the default page is “Custom”.  If the pages property is not defined for a control defined in the configuration file, the control is placed on the default page(s).
pages	[“name”, “name”, ...]	Defines the visible pages, where <i>name</i> is the name that appears in the left panel of the GUI; <i>name</i> can be a standard page name or a custom page name. All visible pages must be listed if this line is included. All standard pages are visible if this line is not included.  Example: <code>tool.pages = [“Inputs”, “Custom Page”]</code>
name	drc   lvs   perc   xrc   pex	The name property is set to drc, lvs, perc, xrc, or pex, corresponding to the DRC, LVS, PERC, PEX, and xACT Calibre Interactive GUI applications.  This property should only be read in the configuration file.
runCallback	“callbackFunction”	Specifies the name of a callback function that is executed when the Run button is clicked. The function should return true or false. The callback function is automatically executed in a batch run.  The function must be defined in the configuration file, outside the onInitialized section.  If the return value is false, a pop-up warning is displayed and execution continues. Define <code>tool.runCallbackError</code> to include a user-defined message in the pop-up warning.  Calibre execution continues regardless of the return value. Use other means, such as <code>tool.blockRun</code> , to control execution.  See “ <a href="#">Example: Attach Callback Function to Run Button With tool.runCallback</a> ” on page 249.

**Table 10-3. Tool Properties (cont.)**

<b>Property</b>	<b>Allowed Values</b>	<b>Description</b>
runCallbackError	string	Specifies an error message to display when <i>callbackFunction</i> returns false. When tool.runCallbackError is defined, the pop-up warning message includes a button “Show Details.”
showSearchPage	<u>true</u>   false	Specifies the visibility of the Search widget at the top right corner of the GUI. The default is true. The Search widget is not controlled by the tool.pages property.  Example: tool.showSearchPage = true
userRunLabelSuffix	string	Specifies a different label for the <b>Run ...</b> button. The specified string is appended to the word “Run”.  For example: tool.userRunLabelSuffix = “PERC LDL” results in a button label of “Run PERC LDL”.

**Example: Attach Callback Function to Run Button With tool.runCallback**

This example uses tool.runCallback to attach a callback function to the Run button. If the callback function returns false, a warning is displayed and tool.blockRun is set to halt the run.

The callback function tests the value of an integer define control that is added to the GUI.

```
Configuration {
    property variant defvar;

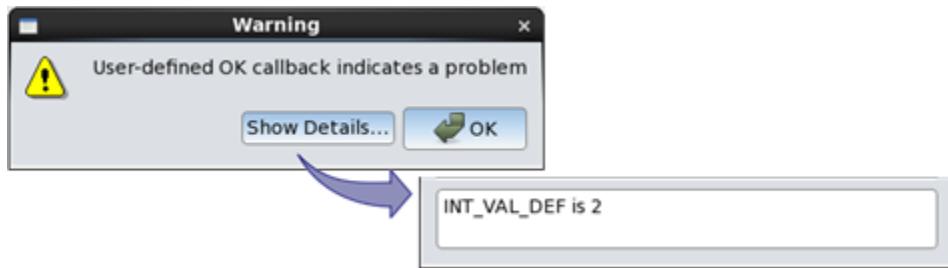
    onInitialized: {

        tool.addPage("Defines and Variables")

        defvar = tool.addIntegerDefine("INT_VAL_DEF")
        defvar.pages      = ["Defines and Variables"]
        defvar.label     = "Define INT_VAL_DEF"
        defvar.description = "SVRF Define for INT_VAL_DEF"
        defvar.value      = 2
        tool.runCallback = "defTest";

    // end onInitialized
}
```

```
function defTest() {
    tool.runCallbackError = "INT_VAL_DEF is 2";
    if (defvar.value == 2) {
        tool.blockRun = true;
        tool.blockReason = "INT_VAL_DEF cannot be 2, halting run";
        return false;
    } else {
        tool.blockRun = false;
        return true;
    }
}
//end Configuration
}
```



This simple example demonstrates the use of `tool.runCallback`. To display the warning message only, without halting run, remove the lines with `tool.blockRun`.

Alternatively, you can use the `minValue` and `maxValue` properties for `addIntegerDefine` to set limits on the possible values for the define control, rather than use `tool.runCallback` and `tool.blockRun`.

## Option Properties

Certain properties are common to the all GUI objects for option settings. For example, you can specify the tooltip, label, visibility, and the page on which an option is displayed. The objects created by the configuration API commands also have some of the same option properties.

See “[Finding Option Names to Use in the Configuration File](#)” on page 254 for instructions on finding the option name that corresponds to a standard GUI control (those not added with the configuration file).

With the exception of the “specified” property, the properties in the following table are available with all standard GUI control objects. The reference page for the configuration API commands that create a GUI object lists the properties available with the object; see “[Function Reference for GUI Customization](#)” on page 256.

**Table 10-4. Common Option Properties**

Property	Allowed Values	Description
description	string	The tooltip.

**Table 10-4. Common Option Properties (cont.)**

<b>Property</b>	<b>Allowed Values</b>	<b>Description</b>
editable	true   false	<p>Specifies whether the value can be changed in the GUI.</p> <p>This setting only affects whether the value can be changed in the GUI. The value can always be set with a runset option or the application of a template, even when editable=false.</p>
expanded	true   false	Specifies whether an option is expanded (true) or collapsed (false) when the GUI is opened.
label	string	The displayed label.
optional	true   <u>false</u> (Boolean)	<p>Places a checkbox next to the control. The default is false, resulting in a required control.</p> <p>The “optional” property is not available for all controls. For example, addDefine does not support the “optional” property.</p>
pages	list of strings	<p>A list of the pages that the control appears on. For example: [“Outputs”, “Options”]</p> <p>The pages property can only be set for a top-level option; it cannot be set for a sub-option. For example, you can set tool.layout.pages, but not tool.layout.format.pages.</p>
promptText	string	The prompt text for a text entry box. The prompt text is only used if the value property is not specified.
specified	true   false	Specifies the state of the checkbox for an optional control. The “specified” property is not used with Boolean controls (controls with <i>only</i> a checkbox).
value	See description column	The value for the control. For Boolean controls, which are displayed with only a checkbox, value is set to true or false.
visible	true   false	<p>Specifies whether to show or hide the control in the GUI.</p> <p>If the control has a missing or incorrect setting, it is made visible even if this property value is set to false.</p>

## GUI Signals

You can connect functions to signals generated by the GUI. Signals are generated when a property value changes and from user-defined buttons.

**Table 10-5. Signals from the Calibre Interactive GUI**

GUI Signal	To connect a function ...
Change in property value for a standard GUI control (those not added by the configuration file)	<b><i>tool.option.propertyChanged.connect(cmd)</i></b> See <a href="#">connect</a> .
Change in property value from an object added by the configuration file (for example, addDefine or addIntegerVariable)	<b><i>ctrl_var.propertyChanged.connect(cmd)</i></b> See <a href="#">connect</a> and “ <a href="#">New Configuration File to Attach a Callback Function to a DEFINE Control</a> ” on page 409.
Button click from user-defined button	<b><i>button_var.activated.connect(cmd)</i></b> See <a href="#">addAction</a> .
New choice from a button with a dropdown list	<b><i>button_var.valueChanged.connect(cmd)</i></b> See <a href="#">addChoicesAction</a> .

---

### Tip

 You can block a run from executing by setting the property `tool.blockRun` to false. See “[Tool Properties](#)” on page 247.

You can attach a callback function to the Run button with the tool property `tool.runCallback`. See “[Example: Attach Callback Function to Run Button With `tool.runCallback`](#)” on page 249.

---

## Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive

You can use the hyperlink option property to add a link to a custom control. The hyperlink property is supported for addHeader, addLabel, and the commands that add Variable, #DEFINE, and #UNDEFINE statements to the control file.

### Syntax

```
ctrl_var.hyperlink = "link_path[#anchor_tag]"
```

### Arguments

- *ctrl\_var* — The variable name for the custom control.

- ***link\_path*** — The path for the link. This can be a URL, an HTML file, a text file, or a PDF file. For files, some HTML browsers may require “file://” to precede the file path of the document.
- **#*anchor\_tag*** — An optional parameter that specifies a named destination within the document at ***link\_path***.

Some typical anchor tags supported in PDF documents are #page=<page>, #destination, and #tag=destination. However, when viewing PDF documents in an HTML browser, support for the #page anchor tag depends on the browser and the browser settings.

## Description

The label for the control is displayed in blue underline font. The specified ***link\_path*** is opened in your default web browser or default PDF viewer when you click on the label.

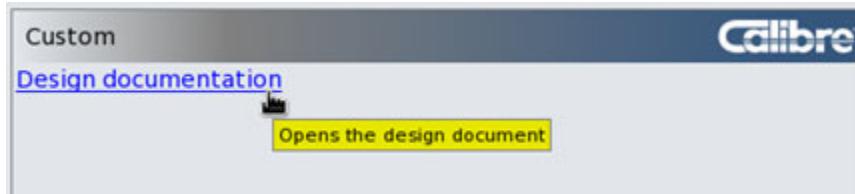
You can set a user-defined location for the web browser and PDF viewer executables with the environment variables MGC\_RVE\_HTML\_BROWSER and MGC\_RVE\_PDF\_VIEWER, respectively.

## Example 1

This is an example for the [addLabel](#) command:

```
var designDocLabel = tool.addLabel("Design documentation",
                                    "Opens the design document");
designDocLabel.pages = ["Custom"];
designDocLabel.hyperlink = "http://mySite.com/designDoc.html"
```

**Figure 10-2. hyperlink Property Example**



## Example 2

This example uses an anchor tag to open a PDF document at the tagged destination “Layout”. The PDF document is located in the working directory.

```
designDoc2Label.hyperlink = "designDoc.pdf#Layout"
```

## Finding Option Names to Use in the Configuration File

The configuration file uses the same option names as the runset file. You can determine option names by saving a runset in which only the options of interest have non-default settings.

The procedure gives a specific example that finds the option names corresponding to the DRC Cell Name statement and the option to start Calibre RVE after the run is finished.

---

### Note

---

 As an alternative to the following method, you can set the environment variable MGC\_CALIBRE\_SAVE\_ALL\_RUNSET\_VALUES to 1 before invoking the GUI and saving a runset. All runset options are saved, rather than just the non-default options. However, it is not always straightforward to determine the runset option that corresponds to a particular GUI option in the resulting runset file.

---

### Procedure

1. Open Calibre Interactive nmDRC with default settings—that is, do not load a runset or a configuration file.

```
calibre -gui -drc
```

2. Enter “DRC Cell Name” in the Search text entry box at the top right corner of the GUI.

The view automatically switches to the Search page, where matches are highlighted.

Expand the DRC Results Database area. The “Output cell errors in cell space” option is highlighted, and the tooltip shows that this is the option corresponding to DRC Cell Name.

Change the setting for “Output cell errors in cell space” so that the option no longer has its default setting.

3. Enter “RVE” in the Search text entry box. We are not sure of the name for the option to start Calibre RVE, so enter something general.

On the Search page, expand the area for “RVE Options” and change the setting for “Show results in RVE”.

4. Select **File > Save Runset** and save the runset as *drcoptions.runset*.

5. View the file *drcoptions.runset*:

```
drc.drcdb.resultsCellName.value = false
drc.rve.autoStart.value = true
```

## Results

Only the non-default runset options are saved. For this example the option names are drcdb.resultsCellName and rve.autoStart, where the initial “drc.” is dropped. When used in a configuration file the options are preceded with “tool”, as in tool.rve.autoStart.

## Function Reference for GUI Customization

---

The configuration file in Calibre Interactive supports many commands for GUI customization.

<b>General Configuration File Functions . . . . .</b>	<b>257</b>
<b>Configuration Commands to Execute Functions and Commands . . . . .</b>	<b>293</b>
<b>Configuration File Commands to Add #DEFINE and #UNDEFINE Statements . . . . .</b>	<b>303</b>
<b>Configuration File Functions to Add Variable Statements . . . . .</b>	<b>342</b>
<b>Configuration File Commands to Add Buttons . . . . .</b>	<b>382</b>
<b>Configuration File Commands for Trigger Functions . . . . .</b>	<b>390</b>

## General Configuration File Functions

The Calibre Interactive configuration file has functions to add and remove pages, write to the transcript, add a separator line, and add text to the GUI. With the exception of the transcript function, these functions do not affect the Calibre run.

See “[Configuration File Format for Calibre Interactive GUI Customization](#)” on page 240 for information on the file format and how to invoke Calibre Interactive with a custom configuration file.

Command	Description
<a href="#">addGroup</a>	Specifies various custom items to be grouped together.
<a href="#">addHeader</a>	Adds a header label with a shaded background to the Calibre Interactive GUI.
<a href="#">addLabel</a>	Adds a text label to the Calibre Interactive GUI.
<a href="#">addMaster</a>	Specifies that the visibility of a control depends on the state of another control (the primary control) defined in the Calibre Interactive configuration file.
<a href="#">addMasterSelect</a>	Specifies that the visibility of a control depends on the state of another control (the primary control) defined in the Calibre Interactive configuration file. The dependent control is displayed if the primary control is <i>not</i> selected.
<a href="#">addMastersWithExpr</a>	Specifies that the visibility of a control depends on one or more other controls (the primary controls) that are defined in the Calibre Interactive configuration file. The logic for displaying the dependent control is defined with an expression involving the primary controls.
<a href="#">addMastersWithFunction</a>	Specifies that the visibility of a control depends on one or more other controls (the primary controls) that are defined in the Calibre Interactive configuration file. The logic for displaying the dependent control is determined by a function defined in the configuration file.
<a href="#">addMutuallyExclusiveGroup</a>	Specifies a group of exclusive define controls, in which only one can be enabled at a time.
<a href="#">addPage</a>	Adds a custom page to the Calibre Interactive GUI or changes the page order of an existing page.
<a href="#">addSeparator</a>	Adds a separator line with an optional text label to the Calibre Interactive GUI.
<a href="#">addTable</a>	Adds a table to the Calibre Interactive GUI.

Command	Description
<a href="#">envVars</a>	The tool.envVars object includes function calls to read and set environment variables from the Calibre Interactive custom configuration file.
<a href="#">FileIO</a>	The FileIO component and associated functions provide the means to perform file I/O with the Calibre Interactive custom configuration file.
<a href="#">getCalibreVersion</a>	Returns a string containing the Calibre version, for example v2017.3_0.74.
<a href="#">getProcessID</a>	Returns the process ID for the Calibre Interactive GUI.
<a href="#">hasOwnProperty</a>	Returns a Boolean value indicating whether the specified option name exists.
<a href="#">removePage</a>	Removes a page from the Calibre Interactive GUI. Pages with invalid objects are visible even when removed.
<a href="#">setItemOrder</a>	Defines the order of controls on a page in Calibre Interactive.
<a href="#">transcript</a>	Outputs text to the transcript.

## addGroup

Function call for the custom configuration file for Calibre Interactive  
Specifies various custom items to be grouped together.

### Usage

`ctrl_var = tool.addGroup("name", item_list)`

### Object Properties

<code>ctrl_var.pages</code>	<code>= page_list</code>
<code>ctrl_var.label</code>	<code>= "label"</code>
<code>ctrl_var.description</code>	<code>= "tooltip"</code>
<code>ctrl_var.editable</code>	<code>= <u>true</u>   false</code>
<code>ctrl_var.visible</code>	<code>= <u>true</u>   false</code>

### Arguments

- **`ctrl_var`**  
A required argument specifying the variable name for the control.
- **`name`**  
A required argument specifying the variable name for the group. The **`name`** can include alphanumeric and underscore (\_) characters.  
Some special characters can be used in the **`name`**, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.
- **`item_list`**  
A required argument specifying the list of items to be grouped together. Only custom items, including groups, are allowed.
- Object Properties  
See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, and visible.  
See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified `link_path`.

### Description

`addGroup` designates a set of custom items to be grouped together. These items are physically grouped together in the GUI, even if they are not defined next to each other.

If there is a conflict between an individual item’s object property and the group’s object property, then the group’s object property value takes precedence. For example, if an item is set to appear on the Rules page, and a group it belongs to is set to appear on the Outputs page, the group and that item still appears on the Outputs page. However, setting a group’s object property does not affect each item in that group. For example, if you set an item to be not

editable and a group it belongs to as editable, the group is editable but that individual item remains not editable.

## Examples

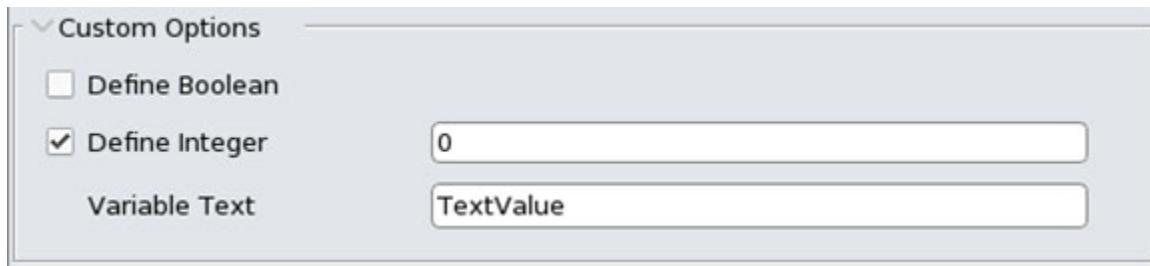
Create three different items, them join them in a group in the **Outputs** page:

```
Configuration {
    onInitialized: {
        var d1 = tool.addDefine("Boolean")

        var d2 = tool.addIntegerDefine("Integer")
        d2.optional = true

        var d3 = tool.addTextVariable("Text")
        d3.value = "TextValue"

        var group = tool.addGroup("Test Group", [d1, d2, d3]);
        group.pages = ["Outputs"]
        group.label = "Custom Options"
    }
}
```



## addHeader

Function call for the custom configuration file for Calibre Interactive  
Adds a header label with a shaded background to the Calibre Interactive GUI.

### Usage

#### Function Syntax

```
ctrl_var = tool.addHeader("label"[, "tooltip"])
```

#### Object Properties

```
ctrl_var.label      = "label"  
ctrl_var.description = "tooltip"  
ctrl_var.pages     = page_list  
ctrl_var.visible   = true | false  
ctrl_var.hyperlink = link_path
```

### Arguments

- ***ctrl\_var***  
A required argument specifying the variable name for the control.
- ***label***  
A required argument specifying the header text.
- ***tooltip***  
An optional argument specifying the hover text (tooltip) for the header. If “tooltip” is not included in the function call it can be defined with the description property.
- Object Properties  
See “[Option Properties](#)” on page 250 for a description of these common properties: pages, description, and visible.  
You can specify the label and description (tooltip) properties in the command call or as property definitions.  
See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

### Description

The function call `tool.addHeader` adds a header with label text to the GUI. The header has a shaded background.

The function [addLabel](#) adds a label without a shaded background.

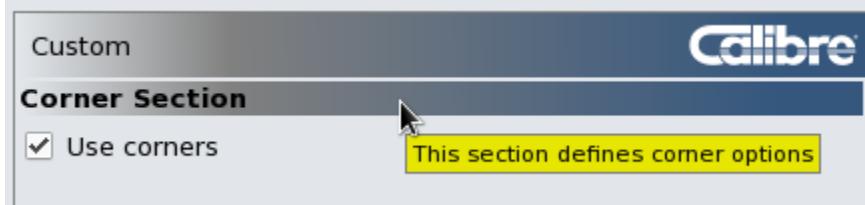
### Examples

The following code adds a header and a define control to the default Custom page.

## GUI Customization for Calibre Interactive General Configuration File Functions

---

```
var head = tool.addHeader("Corner Section",
                         "This section defines corner options");
var dCorner = tool.addDefine("USE_CORNER");
dCorner.value = true;
dCorner.label = "Use corners";
```



## addLabel

Function call for the custom configuration file for Calibre Interactive  
Adds a text label to the Calibre Interactive GUI.

### Usage

#### Function Syntax

```
ctrl_var = tool.addLabel("label"[, "tooltip"])
```

#### Object Properties

```
ctrl_var.label      = "label"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.visible     = true | false  
ctrl_var.hyperlink   = link_path
```

### Arguments

- ***ctrl\_var***  
A required argument specifying the variable name for the control.
- ***label***  
A required argument specifying the text added to the GUI page.
- ***tooltip***  
An optional argument specifying the hover text (tooltip) for the label. If “tooltip” is not included in the function call it can be defined with the description property.
- Object Properties  
See “[Option Properties](#)” on page 250 for a description of these common properties: pages, description, and visible.  
You can specify the label and description (tooltip) properties in the command call or as property definitions.  
See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

### Description

The function call `tool.addLabel` adds a text line to the GUI.

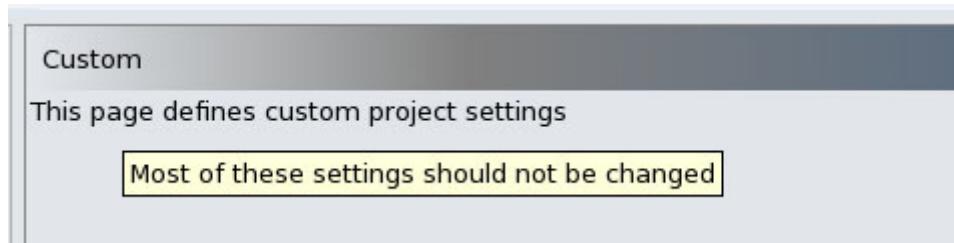
### Examples

The following code adds a text line and corresponding tooltip (hover text) to the default custom page.

## GUI Customization for Calibre Interactive General Configuration File Functions

---

```
var lb = tool.addLabel("This page defines custom project settings",  
                      "Most of these settings should not be changed");
```



## addMaster

Function call for the custom configuration file for Calibre Interactive

Specifies that the visibility of a control depends on the state of another control (the primary control) defined in the Calibre Interactive configuration file.

### Usage

`ctrl_var_dep.addMaster(ctrl_var_master[, value_list])`

### Arguments

- **`ctrl_var_dep`**  
A required argument specifying the variable name for the dependent control.
- **`ctrl_var_master`**  
A required argument specifying the variable name for the primary control.
- **`value_list`**  
An optional argument specifying a list of primary control values to match.

### Description

`addMaster` designates that the state of a primary control (**`ctrl_var_master`**) determines the visibility of the dependent control (**`ctrl_var_dep`**). By default (without `value_list`) the dependent control is displayed only if the checkbox for the primary control is checked. If `value_list` is provided, the dependent control is displayed if the value of the primary control matches an item in `value_list`.

### Examples

#### Example 1

A dependent control depends on the state of an `addDefine` control, which does not have a value.

```
// primary control
var dCornerMaster = tool.addDefine("USE_CORNER");
dCornerMaster.value = false;
dCornerMaster.label = "Use corners";

// #define for CORNERS_DEF, primary control is USE_CORNER
var dCornerDep = tool.addTextDefine("CORNERS_DEF", "typical")
dCornerDep.label      = "Define CORNERS_DEF"
dCornerDep.description = "Enter a space separated list of corners"
dCornerDep.allowSpaces = true;
dCornerDep.addMaster(dCornerMaster);
```

The image on the left shows the initial GUI display. The image on the right shows the display after checking “Use corners.”



### Example 2

A dependent control depends on the value of an addChoicesDefine primary control. The dependent control is only displayed if the primary control is set to “adv” and the checkbox for the primary control is checked.

```
// primary control for ADV_MODE define
var dModeMstr = tool.addChoicesDefine("MODE_DEFINE",
    ["adv", "basic"],
    ["advanced", "basic"],
    ["advanced mode", "basic mode"]);
dModeMstr.label = "MODE_DEFINE control";
dModeMstr.optional = true;

// dependent on MODE_DEFINE value
var dModeAdv = tool.addTextDefine("ADV_MODE",
    "p1",
    "enter value for advanced mode");
dModeAdv.addMaster(dModeMstr, ["adv"]);
```

The following images show the different possible states of the primary and dependent control.



### Example 3

Three controls depend on the state of an addIntegerVariable control. One of the dependent controls depends on matching to a single value for the primary control, one matches to a list of values, and one depends on the state of the primary checkbox only.

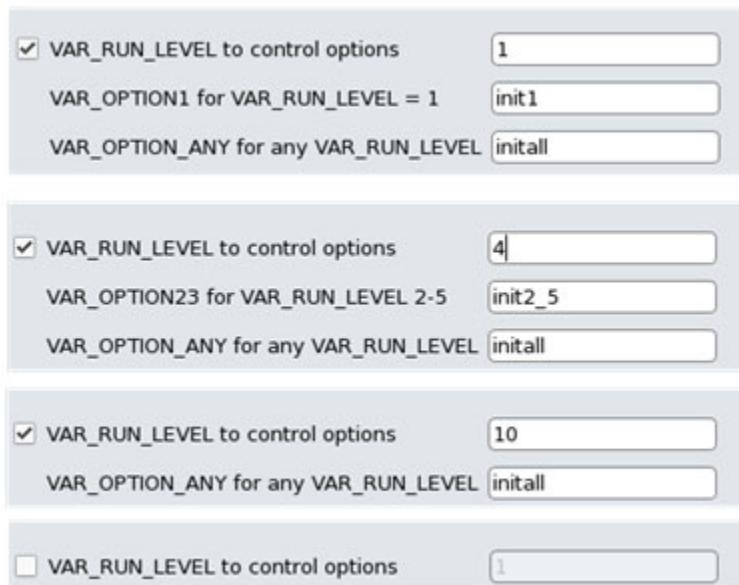
```
var vIntMaster = tool.addIntegerVariable("VAR_RUN_LEVEL",
    1, "controls run level");
vIntMaster.label = "VAR_RUN_LEVEL to control options";
vIntMaster.optional = true;
```

```
var vOptDep1 = tool.addTextVariable("VAR_OPTION1", "init1");
vOptDep1.label = "VAR_OPTION1 for VAR_RUN_LEVEL = 1";
vOptDep1.addMaster(vIntMaster, [1]);

var vOptDep23 = tool.addTextVariable("VAR_OPTION2_5", "init2_5");
vOptDep23.label = "VAR_OPTION23 for VAR_RUN_LEVEL 2-5";
vOptDep23.addMaster(vIntMaster, [2, 3, 4, 5]);

// this only shows up if primary control has optional =true, and checked
var vOptDepChk = tool.addTextVariable("VAR_OPTION_ANY", "initall");
vOptDepChk.label = "VAR_OPTION_ANY for any VAR_RUN_LEVEL";
vOptDepChk.addMaster(vIntMaster);
```

The following images show the different possible states of the primary and dependent controls.



## addMasterSelect

Function call for the custom configuration file for Calibre Interactive

Specifies that the visibility of a control depends on the state of another control (the primary control) defined in the Calibre Interactive configuration file. The dependent control is displayed if the primary control is *not* selected.

### Usage

`ctrl_var_dep.addMasterSelect(ctrl_var_master)`

### Arguments

- **`ctrl_var_dep`**  
A required argument specifying the variable name for the dependent control.
- **`ctrl_var_master`**  
A required argument specifying the variable name for the primary control.

### Description

`addMasterSelect` designates that the state of a primary control (`ctrl_var_master`) determines the visibility of the dependent control (`ctrl_var_dep`). The dependent control is displayed only if the checkbox for the primary control is *not* checked.

### Examples

#### Example 1

A dependent control depends on the state of an `addDefine` control, which does not have a value.

```
// primary control
var dProcessMaster = tool.addDefine("PROCESS");
dProcessMaster.value = false;
dProcessMaster.label = "Process";

// info label when primary control PROCESS is not selected
var noProcessLb = tool.addLabel("Process is not defined")
noProcessLb.addMasterSelect(dProcessMaster);
```

## addMastersWithExpr

Function call for the custom configuration file for Calibre Interactive

Specifies that the visibility of a control depends on one or more other controls (the primary controls) that are defined in the Calibre Interactive configuration file. The logic for displaying the dependent control is defined with an expression involving the primary controls.

### Usage

`ctrl_var_dep.addMastersWithExpr(['master_list'], "expression")`

### Arguments

- **`ctrl_var_dep`**  
A required argument specifying the variable name for the dependent control.
- **`master_list`**  
A required argument specifying the comma-separated list of one or more variable names for the primary controls that are used in the **`expression`**.
- **`expression`**  
A required argument specifying the expression that controls whether the dependent control is displayed. The expression should return true or false and be within quotes.

### Description

`addMastersWithExpr` designates that the visibility of a dependent control (**`ctrl_var_dep`**) is controlled by an expression that involves one or more primary controls.

### Examples

The primary controls must be defined outside the `onInitialized` section as global variables.

```
Configuration {
    property variant primary1;
    property variant primary2;
    property variant dpdntAND;

    onInitialized: {
        primary1 = tool.addDefine("Primary1");
        primary2 = tool.addDefine("Primary2");

        dpdntAND = tool.addDefine("AND_Dependent");

        // use expression to show dpdntAND when primary1 && primary2 are on
        // value for addDefine is true/false for the checkbox state
        dpdntAND.addMastersWithExpr([primary1, primary2],
                                     "primary1.value && primary2.value");
    }
}
```

## addMastersWithFunction

Function call for the custom configuration file for Calibre Interactive

Specifies that the visibility of a control depends on one or more other controls (the primary controls) that are defined in the Calibre Interactive configuration file. The logic for displaying the dependent control is determined by a function defined in the configuration file.

### Usage

`ctrl_var_dep.addMastersWithFunction(['master_list'],function)`

### Arguments

- ***ctrl\_var\_dep***  
A required argument specifying the variable name for the dependent control.
- ***master\_list***  
A required argument specifying a comma-separated list of one or more variable names for the primary controls that are used in the *function*.
- ***function***  
A required argument specifying the name of the function that controls whether the dependent control is displayed. The function should return true or false.

### Description

addMastersWithFunction designates that the visibility of a dependent control (*ctrl\_var\_dep*) is controlled by a function that returns true or false. addMastersWithFunction is useful when the logic for determining the dependent control visibility is more complicated than can be easily specified by an expression in [addMastersWithExpr](#).

### Examples

The primary controls must be defined outside the onInitialized section as global variables.

```
Configuration {
    property variant primary1;
    property variant primary2;
    property variant dpdntOR;

    onInitialized: {
        primary1 = tool.addDefine("Primary1");
        primary2 = tool.addDefine("Primary2");

        dpdntOR = tool.addDefine("OR_Dependent");

        // use function to show dpdntOR when primary1 || primary2 are on
        // value for addDefine is true/false for the checkbox state
        dpdntOR.addMastersWithFunction([primary1, primary2], control_or);
    }

    function control_or() {
        return (primary1.value || primary2.value);
    }
}
```

## addMutuallyExclusiveGroup

Function call for the custom configuration file for Calibre Interactive

Specifies a group of exclusive define controls, in which only one can be enabled at a time.

### Usage

```
ctrl_var = tool.addMutuallyExclusiveGroup("name", define_list)
```

### Object Properties

<i>ctrl_var.pages</i>	= <i>page_list</i>
<i>ctrl_var.label</i>	= "label"
<i>ctrl_var.description</i>	= "tooltip"
<i>ctrl_var.editable</i>	= <u>true</u>   false
<i>ctrl_var.visible</i>	= <u>true</u>   false
<i>ctrl_var.hyperlink</i>	= <i>link_path</i>

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- ***name***

A required argument specifying the variable name for the group. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- ***define\_list***

A required argument specifying the list of define control names to be grouped together. At least two names are required.

- Object Properties

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, and visible.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

### Description

addMutuallyExclusiveGroup designates a set of define controls to be exclusive, such that only one can be enabled (set to 1) at a time. These defines are physically grouped together in the GUI, even if they are not defined next to each other.

Only define controls with a checkbox can be added to a mutually exclusive group. [addDefine](#) controls always have a checkbox, with the checkbox state controlled by the value property. Other define controls must set the optional property to true to include a checkbox; the checkbox

state is controlled with the specified property. Only one control in the group can have the checkbox state set to true when the mutually exclusive group is created.

**Note**

 [addDefine](#) controls have their value properties set to false by default. Other defines, such as integers and reals, have their specified properties set to true by default; they must be explicitly set to false in the configuration file.

## Examples

Create three different define controls, them join them in a mutually exclusive group in the **Outputs** page:

```
Configuration {
    onInitialized: {
        var d1 = tool.addDefine("Boolean")

        var d2 = tool.addIntegerDefine("Integer")
        d2.optional = true

        var d3 = tool.addRealDefine("Real")
        d3.optional = true
        d3.specified = false

        var defines = tool.addMutuallyExclusiveGroup("Group", [d1, d2, d3]);
        defines.pages = ["Outputs"]
        defines.label = "New Defines"
    }
}
```



Set defines to optional, and set at most one define's specified or value property to true.

## addPage

Function call for the custom configuration file for Calibre Interactive

Adds a custom page to the Calibre Interactive GUI or changes the page order of an existing page.

### Usage

#### Function Syntax

```
tool.addPage("page_name"[, "before_page"])
```

### Arguments

- *page\_name*  
A required argument specifying the name of a new or existing page.
- *before\_page*  
An optional argument specifying the name of an existing page.

### Description

If *page\_name* does not exist, it is added. By default it is placed at the end of the page list on the left panel of the GUI, but before the Transcript and Files pages. If *before\_page* is specified, the new page is placed before the specified page.

If *page\_name* exists, it is moved with the logic described in the preceding paragraph.

A page that has no content is not created.

## addSeparator

Function call for the custom configuration file for Calibre Interactive

Adds a separator line with an optional text label to the Calibre Interactive GUI.

### Usage

#### Function Syntax

`ctrl_var = tool.addSeparator(["label"])`

#### Object Properties

`ctrl_var.label = "label"`  
`ctrl_var.description = "tooltip"`  
`ctrl_var.pages = page_list`  
`ctrl_var.visible = true | false`

### Arguments

- **`ctrl_var`**  
A required argument specifying the variable name for the control.
- **`label`**  
An optional argument specifying a text label for the separator line. The label text can be specified in the function call or it can be defined with the label property.
- Object Properties  
See “[Option Properties](#)” on page 250 for a description of these common properties: pages, description, and visible.  
You can specify the optional `label` in the command call or as a property definition.

### Description

The function call `tool.addSeparator` adds a separator line to the GUI, with an optional text label.

## addTable

Function call for the custom configuration file for Calibre Interactive  
Adds a table to the Calibre Interactive GUI.

### Usage

#### Function Syntax

```
ctrl_var = tool.addTable("name", rowCount, columnCount[, isRowCountFixed])
```

#### Object Properties

<i>ctrl_var.pages</i>	= <i>page_list</i>
<i>ctrl_var.label</i>	= "control_label"
<i>ctrl_var.description</i>	= "control_tooltip"
<i>ctrl_var.value</i>	= <i>value</i>
<i>ctrl_var.editable</i>	= <u>true</u>   false
<i>ctrl_var.visible</i>	= <u>true</u>   false
<i>ctrl_var.optional</i>	= true   <u>false</u>
<i>ctrl_var.specified</i>	= <u>true</u>   false
<i>ctrl_var.headers</i>	= <i>headers</i>
<i>ctrl_var.tooltips</i>	= <i>tooltips</i>
<i>ctrl_var.rowCount</i>	= <i>row_count</i>
<i>ctrl_var.columnCount</i>	
<i>ctrl_var.isRowCountFixed</i>	= <u>true</u>   false

#### Component Functions

<i>ctrl_var.getCell(row, column)</i>
<i>ctrl_var.setCell(row, column, value)</i>
<i>ctrl_var.makeTextColumn(column[, defaultValue])</i>
<i>ctrl_var.makeBooleanColumn(column[, defaultValue])</i>
<i>ctrl_var.makeChoicesColumn(column, value_list[, label_list])</i>
<i>ctrl_var.makeIntegerColumn(column[, defaultValue, minValue, maxValue, stepValue])</i>

### Arguments

- ***ctrl\_var***  
A required argument specifying the variable name for the control.
- ***name***  
A required argument specifying the variable name for the table.
- ***rowCount***  
A required argument specifying the number of rows in the table. You can change this value with *ctrl\_var.rowCount = row\_count* or in the GUI.

- ***columnCount***

A required argument specifying the number of columns in the table. This value cannot be changed after it is assigned.

- ***isRowCountFixed***

An optional argument specifying whether you can change the number of rows in the table.

- Object Properties

See [Option Properties](#) for a description of these common properties: pages, label, description, value, editable, visible, optional, and specified.

You can specify the label and description (tooltip) properties in the command call or as property definitions.

The properties specific to addTable are the following:

**Table 10-6. Object Properties for addTable**

Property	Allowed Values	Description
rowCount	<i>row_count</i> (integer)	The number of rows in the table. Does not work if the row count is fixed ( <i>isRowCountFixed</i> = true).
columnCount	None	The number of columns in the table. This property is read-only.
headers	<i>headers</i> (list of strings)	The text label of each column header.
tooltips	<i>tooltips</i> (list of strings)	The text of each column tooltip. Hover over the column to see the tooltip.
isRowCountFixed	<u>true</u>   false (Boolean)	Set to true to disable adding or deleting rows.

- Component Functions

**Table 10-7. Component Functions for addTable**

Function	Description
<code>getCell(<i>row</i>, <i>column</i>)</code>	Retrieves the value in the cell located at the specified row and column.
<code>setCell(<i>row</i>, <i>column</i>, <i>value</i>)</code>	Sets the value of the cell located at the specified row and column.
<code>makeTextColumn(<i>column</i>[, <i>defaultValue</i>])</code>	Sets the specified column as a text column. <ul style="list-style-type: none"> <li>• <b><i>column</i></b> — A required integer specifying a column. Column numbers are zero-indexed from left to right.</li> <li>• <b><i>defaultValue</i></b> — An optional string specifying the default text of every cell in the column. If not specified, the cells are empty.</li> </ul>

**Table 10-7. Component Functions for addTable (cont.)**

Function	Description
<code>makeBooleanColumn(     <i>column</i>[,     <i>defaultValue</i>])</code>	Sets the specified column as a Boolean column. Each cell is populated with a checkbox. <ul style="list-style-type: none"> <li>• <i>column</i> — A required integer specifying a column. Column numbers are zero-indexed from left to right.</li> <li>• <i>defaultValue</i> — An optional Boolean specifying the default value of every cell in the column. Default is true.</li> </ul>
<code>makeChoicesColumn(     <i>column</i>, <i>value_list</i>[,     <i>label_list</i>])</code>	Sets the specified column as a choices column. Each cell is populated with a dropdown menu of options. <ul style="list-style-type: none"> <li>• <i>column</i> — A required integer specifying a column. Column numbers are zero-indexed from left to right.</li> <li>• <i>value_list</i> — A required list of strings specifying the options in the dropdown menu.</li> <li>• <i>label_list</i> — An optional list of strings specifying the labels used in the dropdown menu. If this is not specified, <i>value_list</i> is used.</li> </ul>
<code>makeIntegerColumn(     <i>column</i>[, <i>defaultValue</i>,     <i>minValue</i>, <i>maxValue</i>,     <i>stepValue</i>])</code>	Sets the specified column as an integer column. Each cell is populated with an integer. <ul style="list-style-type: none"> <li>• <i>column</i> — A required integer specifying a column. Column numbers are zero-indexed from left to right.</li> <li>• <i>defaultValue</i> — An optional integer specifying the default value of every cell in the column. If unspecified, this value is set to 0. If this default value lies beyond the range specified by <i>minValue</i> and <i>maxValue</i>, it is clamped to the nearest valid number.</li> <li>• <i>minValue</i> — An optional integer specifying the minimum value of the cells in the column.</li> <li>• <i>maxValue</i> — An optional integer specifying the maximum value of the cells in the column.</li> <li>• <i>stepValue</i> — An optional integer specifying the step value of the spin box. Clicking the up and down controls of the spin box raise and lower the cell's value by <i>stepValue</i>. The default value is 1.</li> </ul>

## Description

The function call `tool.addTable` adds a table to the GUI. The table name and its cell values are not written to the control file during a run.

If the row count is not fixed (`isRowCountFixed = true`), the number of rows can be edited in the GUI using three buttons:

-  — Adds a row to the bottom of the table.
-  — Deletes the selected row.

-  — Deletes all rows.

## Examples

### Creating a Table

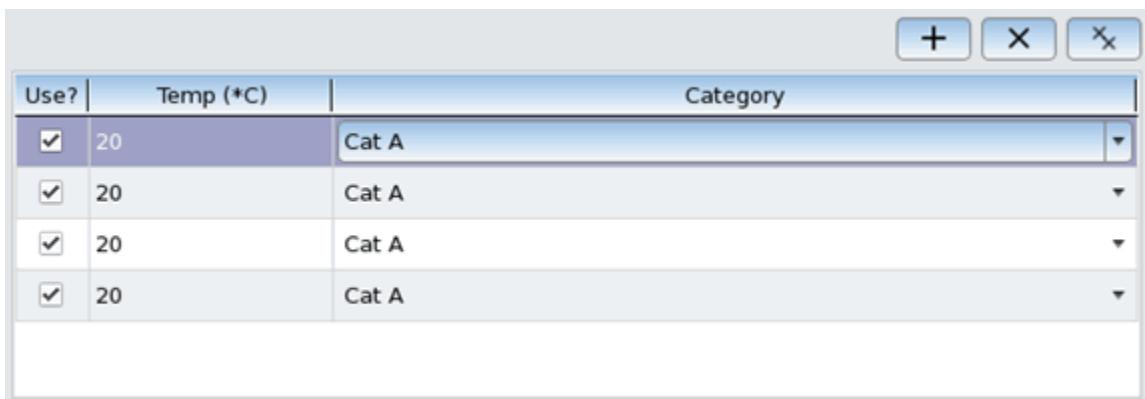
Add the following code to the configuration file:

```
var aTable = tool.addTable('Temperature Table', 4, 3, false)
aTable.headers = ["Use?", "Temp (*C)", "Category"]
```

aTable has four rows and three columns; its number of rows can be adjusted. Its three columns have those specified header labels, from left to right, and must have their types specified:

```
aTable.makeBooleanColumn(0, 1)
aTable.makeIntegerColumn(1, 20)
aTable.makeChoicesColumn(2, ["a", "b", "c"], ["Cat A", "Cat B", "Cat C"])
```

The cells in the leftmost column (column 0) each contain a single checkbox that is checked by default. The cells in the middle column are populated with the text “20”. The cells in the rightmost column each contain a dropdown menu displaying “Cat A”, “Cat B”, and “Cat C”; their default values are “Cat A”.



Use?	Temp (*C)	Category
<input checked="" type="checkbox"/>	20	Cat A
<input checked="" type="checkbox"/>	20	Cat A
<input checked="" type="checkbox"/>	20	Cat A
<input checked="" type="checkbox"/>	20	Cat A

### Validating a Table

You can create a function to verify that the data entries in your table follow a certain criteria. For example, to verify that every value in the Temp column is greater than 0, you could create the function:

```
function verifyTable(table) {
    for (var i = 0; i < table.rowCount; i++) {
        var value = table.getCell(i, 1)
        if (value <= 0) {
            return false
        }
    }
    return true
}
```

This function can be configured to run after a given action, such as upon clicking a custom button. The following button blocks a Calibre run if verifyTable returns false:

```
var validateTempButton = tool.addAction("Validate temperatures")
validateTempButton.activated.connect(function() {
    if (!verifyTable(aTable)) {
        tool.blockRun = true
        tool.blockReason = "One or more temperatures are below 0 C"
    } else {
        tool.blockRun = false
    }
})
```

See “[Configuration File Commands to Add Buttons](#)” on page 382 for more information on buttons.

### Exporting Table Data

You can create a function to export data in a table to a file. For example, you can use the following function to convert table entries to CSV format and write them to a file:

```
FileIO {
    id: tableFile
    fileName: ""
    accessMode: FileIO.Write
}

function convertCsv(table, fileLocation) {
    tableFile.filename = fileLocation
    tableFile.accessMode = FileIO.Write
    tableFile.writeLine("USE, TEMP, CATEGORY")
    for (var i = 0; i < table.rowCount; i++) {
        var row = ''
        for (var j = 0; j < table.columnCount - 1; j++) {
            row += table.getCell(i, j) + ','
        }
        row += table.getCell(i, table.columnCount - 1)
        tableFile.writeLine(row)
    }
}
```

Like with the previous example, you can configure this convertCsv function to run after a given action, such as pressing a button:

```
var exportTable = tool.addAction("Export table to file")
exportTable.activated.connect(function() {
    var defaultLocation = "./TEMP_CAT_DIR/TableFile.csv"
    convertCsv(aTable, defaultLocation)
})
```

See “[FileIO](#)” on page 283 for more information on performing file I/O.

## envVars

Object type for the custom configuration file for Calibre Interactive

The tool.envVars object includes function calls to read and set environment variables from the Calibre Interactive custom configuration file.

### Usage

```
tool.envVars.isEnvSet("name")
tool.envVars.envValue("name")
tool.envVars.setEnv("name", "value")
tool.envVars.unsetEnv("name")
```

### Arguments

#### Object Functions

- **isEnvSet("name")**  
Returns a Boolean values indicating whether the environment variable *name* is set.
- **envValue("name")**  
Returns a string containing the value of the environment variable *name*.
- **setEnv("name", "value")**  
Sets the environment variable *name* to *value*. The function can be used to change the value of an existing environment variable or to define a new environment variable. Specify *value* as an empty string to define the environment variable without an assigned value. The new settings are used within the current Calibre Interactive session. There is no return value.
- **unsetEnv("name")**  
Unsets the environment variable *name* within the current Calibre Interactive session. There is no return value.

### Examples

```
Configuration {
  onInitialized: {

    // set the inductance extraction mode depending on an env var
    if (tool.envVars.isEnvSet("PEX_RUN_INDUCTANCE_LM")) {
      tool.transcript("PEX_RUN_INDUCTANCE_LM is set, setting LM extraction");
      tool.pexExtractionType.inductanceMode.value = "lm";
    }
  }
}
```

```
// set the netlist format based on the value of an env var
// note, this code does not have any error checking for the value,
// it assumes the env var is set to valid values for the runset option
if (tool.envVars.isEnvSet("PEX_NETLIST_FORMAT")) {
    if (tool.envVars.envValue("PEX_NETLIST_FORMAT") != "") {
        tool.transcript("PEX_NETLIST_FORMAT is set to "
            + tool.envVars.envValue("PEX_NETLIST_FORMAT"));
        tool.pexNetlist.netlistFormat.value =
            tool.envVars.envValue("PEX_NETLIST_FORMAT");
    }
}

// end onInitialized
}
// end Configuration
}
```

## FileIO

Component definition for the custom configuration file for Calibre Interactive

The FileIO component and associated functions provide the means to perform file I/O with the Calibre Interactive custom configuration file.

### Usage

#### FileIO Component Definition

```
FileIO {  
    id: fileID;  
    fileName: "file_name";  
    [accessMode: FileIO.file_mode;]  
}
```

#### FileIO Component Properties

*fileID.fileName*

#### FileIO Component Functions

```
fileID.exists()  
fileID.open()  
fileID.close()  
fileID.readLine()  
fileID.writeLine("string")  
fileID.endOfFile()  
fileID.isDir()  
fileID.isFile()  
fileID.isReadable()  
fileID.isWritable()  
fileID.dirName()
```

### Arguments

- ***fileID***

A required argument specifying the string by which the file component is accessed. The *fileID* must be defined in the FileIO component definition. It must start with a lowercase letter ([a-z]).

- ***file\_name***

A required argument specifying the name of the file. The *file\_name* must be defined in the FileIO component definition and can be accessed with the *fileName* component property.

- **FileIO.*file\_mode***

An optional argument specifying the mode in which the file is opened.

FileIO.Read — Open the file in read-only mode. This is the default mode if *accessMode* is not defined in the FileIO component definition.

FileIO.Write — Open the file in write-only mode. The file is created if it does not exist. If the file exists, it is truncated to zero length on open.

FileIO.Append — Open the file in append mode, with no read access. The file is created if it does not exist. If the file exists, the file offset marking the current position in the file is placed at the end of the file.

- FileIO Component Functions

**Table 10-8. Functions for FileIO Component**

Function	Return Value
exists()	Boolean
open()	Boolean
close()	None
readLine()	String (the current line), or an empty string on failure
writeLine("string")	None
endOfFile()	Boolean
isDir()	Boolean
isFile()	Boolean
isReadable()	Boolean
isWritable()	Boolean
dirName()	String (the directory name in which the file exists), or an empty string on failure

## Description

The FileIO component definition is placed before the onInitialized section of the configuration file. The FileIO component functions are used inside the onInitialized section.

## Examples

This example opens two files, one for reading and one for writing. It prints two lines from the read-access file, and writes the process id to the write-access file. Some basic file property checks are performed before opening the read-access file.

```
Configuration {  
  
    FileIO {  
        id: fileRD1;  
        fileName: "test_read";  
    }  
  
    FileIO {  
        id: fileWr_log;  
        fileName: "log_file";  
        accessMode: FileIO.Write;  
    }  
  
    onInitialized: {  
  
        if (fileRD1.exists() && !fileRD1.isDir() && fileRD1.isReadable() ) {  
            fileRD1.open();  
            tool.transcript("Reading from " + fileRD1.fileName +  
                " in directory " + fileRD1.dirName() );  
            tool.transcript("    line 1: " + fileRD1.readLine());  
            tool.transcript("    line 2: " + fileRD1.readLine());  
            fileRD1.close();  
        } else {  
            if (!fileRD1.exists()) {  
                tool.transcript(fileRD1.fileName + " doesn't exist");}  
            if (fileRD1.isDir()) {  
                tool.transcript(fileRD1.fileName + " is a directory");}  
            if (!fileRD1.isReadable()) {  
                tool.transcript(fileRD1.fileName + " is not readable");}  
        }  
  
        fileWr_log.open();  
        fileWr_log.writeLine("Log file, process id " + tool.getProcessID());  
        fileWr_log.close();  
  
        // end onInitialized  
    }  
  
    //end Configuration  
}
```

## getCalibreVersion

Function call for the custom configuration file for Calibre Interactive

Returns a string containing the Calibre version, for example v2017.3\_0.74.

### Usage

`tool.getCalibreVersion()`

### Arguments

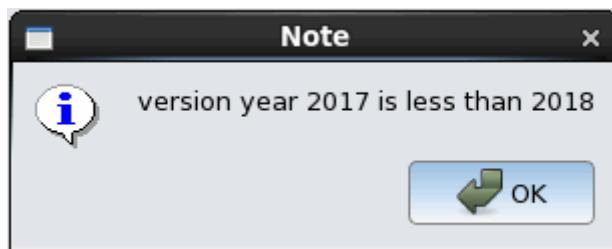
None.

### Examples

This example parses the year portion of the Calibre version string. If the year is less than 2018 the run is blocked by setting the `blockRun` property to true.

```
var str_ver = tool.getCalibreVersion();
var str_yr = str_ver.substr(1,4);
if (str_yr < 2018) {
    tool.blockRun = true;
    tool.blockReason = "version year " + str_yr + " is less than 2018";
}
```

For Calibre versions earlier than 2018, the run is not started when the **Run <app>** button is clicked and the following dialog box is displayed:



## getProcessID

Function call for the custom configuration file for Calibre Interactive

Returns the process ID for the Calibre Interactive GUI.

### Usage

`tool.getProcessID()`

### Arguments

None.

### Examples

This example prints the process id to the transcript.

```
var gui_pid = tool.getProcessID();
tool.transcript("process id " + gui_pid);
```

## hasOwnProperty

Function call for the custom configuration file for Calibre Interactive

Returns a Boolean value indicating whether the specified option name exists.

### Usage

#### Syntax 1

```
tool.hasOwnProperty('optionName')
```

#### Syntax 2

```
tool.optionName.hasOwnProperty('subOptionName')
```

### Arguments

- *optionName*

A required argument specifying an option name, such as rulesFile, runDir, layout, or source.

- *subOptionName*

A required argument specifying a sub-option to *optionName*. For example, layoutFile or format, where *optionName* is layout.

### Description

Returns a Boolean value indicating whether the specified option name exists. The function hasOwnProperty is useful in controlling the logic flow in the configuration file depending on the existence of an option.

Syntax 1 can be used with an option name that does not have sub-options (such as rulesFile and runDir) and with an option name that has sub-options (such as layout and source).

Syntax 2 is used to determine if a specific *subOptionName* exists for *optionName*. The syntax can be extended as needed for multiple levels of sub-options.

### Examples

```
if (tool.hasOwnProperty('rulesFile')) {
    tool.transcript("rule file option exists");
} else {
    tool.transcript("rule file option does not exist");
}

if (tool.layout.hasOwnProperty('topCell')) {
    tool.transcript("layout.topCell option exists");
} else {
    tool.transcript("layout.topCell option does not exist");
}
```

## removePage

Function call for the custom configuration file for Calibre Interactive

Removes a page from the Calibre Interactive GUI. Pages with invalid objects are visible even when removed.

### Usage

#### Function Syntax

```
tool.removePage("page_name")
```

#### Arguments

- *page\_name*

A required argument specifying the name of an existing page.

## setItemOrder

Defines the order of controls on a page in Calibre Interactive.

### Usage

```
tool.setItemOrder("page_name", item_list)
```

### Arguments

- *page\_name*

A required argument specifying the name of the containing the items to be ordered.

- *item\_list*

A required argument specifying a list of items, or controls, on *page\_name*. Only top-level items can be listed. For custom controls, the control variable name is used. For built-in controls, the runset option name is used.

Any items on *page\_name* that are not included in *item\_list* are placed after the items in *item\_list*, in the order they were created.

### Description

Defines the order of items, or controls, on the GUI page *page\_name*.

All controls in *item\_list* must be defined before the setItemOrder command is used.

### Examples

This example creates two checkboxes on the Rules page to add #DEFINE statements to the control file. The order of one checkbox is specified with setItemOrder, while the order of the other checkbox is not defined.

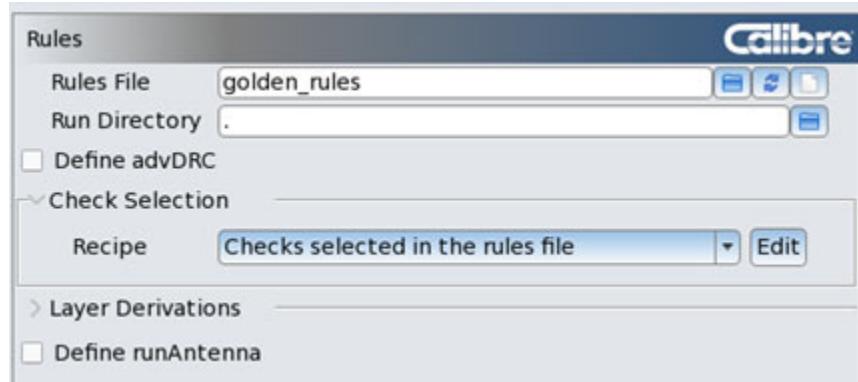
```
Configuration {
    onInitialized: {

        // add define controls
        var defAdvChecks = tool.addDefine("advDRC");
        defAdvChecks.description = "Use advanced DRC checks";
        defAdvChecks.pages = "Rules";

        var defAntennaChecks = tool.addDefine("runAntenna");
        defAntennaChecks.description = "Use antenna checks";
        defAntennaChecks.pages = "Rules";

        tool.setItemOrder("Rules", [tool.rulesFile, tool.runDir, defAdvChecks]);
    }
}
```

The defAdvChecks control (for advDRC) is placed after the run directory control. All remaining controls are placed in order, so the custom control defAntennaChecks is placed after the built-in controls for Check Selection and Layer Derivations.



## transcript

Function call for the custom configuration file for Calibre Interactive  
Outputs text to the transcript.

### Usage

#### Function Syntax

`tool.transcript("text")`

### Arguments

- *text*

A required argument specifying the text to send to the transcript.

### Description

The specified text is sent to the transcript.

When viewed on the Transcript page, text that begins with “ERROR:”, “WARNING:”, and “INFO:” is color coded within the transcript and also displayed in a sorted list at the bottom of the Transcript page.

When multiple configurations are loaded, messages from each configuration file are appended to the transcript window.

## Configuration Commands to Execute Functions and Commands

---

The custom configuration file in Calibre Interactive has commands available that call other commands or functions. The commands specify a callback function and provide a way to execute a shell command.

Command	Description
<a href="#">connect</a>	Specifies a function to execute on a signal from the Calibre Interactive GUI.
<a href="#">launchCommand</a>	Executes an external (shell) command.
<a href="#">launchCommandReturnString</a>	Executes an external (shell) command that returns a string.

## connect

Function call for the custom configuration file in Calibre Interactive

Specifies a function to execute on a signal from the Calibre Interactive GUI.

### Usage

**Syntax 1: Connect to Signal from an addAction Button**

***button\_var.activated.connect(cmd)***

**Syntax 2: Connect to Signal from an addChoicesAction Button**

***button\_var.valueChanged.connect(cmd)***

**Syntax 3: Connect to Change Signal from a Property**

***tool.option.propertyChanged.connect(cmd)***

***ctrl\_var.propertyChanged.connect(cmd)***

### Arguments

- ***button\_var***

A required argument specifying the variable name for an AddAction or AddChoicesAction button.

- ***cmd***

A required argument specifying the function name. The function must be defined in the Configuration section of the configuration file.

When used in Syntax 1 and 3, the function cannot have arguments. In Syntax 2 (for the addChoicesAction button), the function must accept one argument corresponding to the selection in the dropdown list.

- ***option.property***

- ***option*** — A required argument specifying an option name for a standard GUI controls (those not added by the configuration file). See “[Finding Option Names to Use in the Configuration File](#)” on page 254. If the option has sub-options, the syntax can be extended as needed, for example: ***option.subOption.property***.

- ***property*** — A required argument specifying an option property. See “[Option Properties](#)” on page 250.

- ***ctrl\_var.property***

- ***ctrl\_var*** — A required argument specifying the variable name for an object added by the configuration file; for example, and addDefine or addIntegerVariable control object.

- ***property*** — A required argument specifying a property associated with ***ctrl\_var*** object. See the relevant command reference for the list of available object properties.

## Description

The connect function specifies a function to call when a signal is received from the Calibre Interactive GUI.

### Syntax 1: Connect to Signal from an addAction Button

The activated.connect(cmd) function specifies the function that is executed when an [addAction](#) button is clicked. The [addAction](#) reference pages includes an example.

### Syntax 2: Connect to Signal from an addChoicesAction Button

The valueChanged.connect(cmd) function specifies the function that is executed when a new selection is made in the dropdown list of an [addChoicesAction](#) button. The new control value (*ctrl\_var.value*) is passed to the *cmd* function.

### Syntax 3: Connect to Change Signal from a Property

The connect functions specifies a function that is called when a property value changes. Any valid property may be used. The properties “specified” and “value” are typically used to call a function when the state of a checkbox or the option value changes.

---

#### Note

 For “value” properties that are provided by means of a text entry field in the GUI, the callback cmd is called at each character entry.

---

See the [addAction](#) or [addChoicesAction](#) functions for examples that connect a function to a user-defined button in the GUI. To add a callback function to the Run button, use the tool property `tool.runCallback`, as described in “[Example: Attach Callback Function to Run Button With tool.runCallback](#)” on page 249.

## Examples

The following examples connect a function to a change signal from a property.

### Example 1: Callback Function to Set Options when Checkbox Enabled

This example sets options associated with the SVDB database if the checkbox state is changed to true.

```
Configuration {
    onInitialized: {
        // Attach setSVDBOptions() as a callback function
        // if the svdb checkbox is changed
        tool.svdb.specifiedChanged.connect(setSVDBOptions)
    }

    // define a function to set SVDB options if the
    // Mask SVDB checkbox is changed to true
    function setSVDBOptions() {
        var checkbox = tool.svdb.specified

        if (checkbox) {
            tool.svdb.query.value = true
            tool.svdb.asciiXref.value = true
            tool.svdb.cci.value = true
            tool.svdb.genParasiticData.value = true
            tool.svdb.pinloc.value = "PINLOC"
            tool.svdb.pinloc.specified = true
        }
    }
}
```

#### Example 2: Callback Function to Set Parasitic Netlist File Name

This example defines a callback function to set the parasitic netlist file name based on the top cell name and the netlist format. The callback function is executed if the top cell name or netlist format changes.

```
Configuration {
    onInitialized: {
        // Attach updateNetlistFile() as a callback function
        // if netlist format or topcell name changes
        tool.pexNetlist.netlistFormat.valueChanged.connect(updateNetlistFile)
        tool.layout.topCell.valueChanged.connect(updateNetlistFile)
    } //onInit
```

```
// define the function to construct the parasitic netlist name
function updateNetlistFile() {
    var topcell = tool.layout.topCell.value
    var format = tool.pexNetlist.netlistFormat.value

    switch (format) {
        case "HSPICE":
            tool.pexNetlist.netlistFile.value = topcell.concat(".hsp")
            break;
        case "DSPF":
            tool.pexNetlist.netlistFile.value = topcell.concat(".dspf")
            break;
        case "SPEF":
            tool.pexNetlist.netlistFile.value = topcell.concat(".spef")
            break;
        case "SPECTRE":
            tool.pexNetlist.netlistFile.value = topcell.concat(".spectre")
            break;
        case "ELDO":
            tool.pexNetlist.netlistFile.value = topcell.concat(".eldo")
            break;
        case "CALIBREVIEW":
            tool.pexNetlist.netlistFile.value = topcell.concat(".calibre")
            break;
        case "SPICE":
            tool.pexNetlist.netlistFile.value = topcell.concat(".sp")
            break;
        default:
            break;
    }
}

}//Configuration
```

**Example 3: Callback Function Attached to Property Change Signal from an addDefine Control**

See “[New Configuration File to Attach a Callback Function to a DEFINE Control](#)” on page 409.

## launchCommand

Command definition and function call for the custom configuration file for Calibre Interactive  
Executes an external (shell) command.

### Usage

`cmdID.launchCommand`

#### Command Object Definition

```
Command { id: cmdID;  
          [commandString: "cmd";]  
        }
```

### Arguments

- ***cmdID***  
A required argument specifying the ID for the command. It must start with a lowercase letter ([a-z]).
- ***cmd***  
An optional argument specifying the external command to execute, including any arguments. The command must be in your path.  
The *cmd* can be defined in the command definition or with the commandString property.  
The *cmd* string must be enclosed in double quotes.
- Object Properties  
commandString — An optional argument specifying the command to execute. See the *cmd* argument description.

### Return Values

The return value from the *cmd*.

### Description

The launchCommand function provides a way to call an external (shell) command, such as echo. You can call launchCommand from a function, call it directly in the onInitialized section of the configuration file, or use it as the connect command.

For example, if bSVDB is a addAction control variable, the following launches myCmd when the button is clicked:

```
bSVDB.activated.connect (myCmd.launchCommand) ;
```

## Examples

### Example 1

This example defines a button that executes the setSVDB callback function. The user-defined function echoVar prints statements to the console. The commandString property is redefined within the echoVar function so that the new option values are printed.

```
Configuration {
    onInitialized: {
        // add button to turn on svdb options
        var bSVDB = tool.addAction("SVDB options"); //name is internal
        bSVDB.pages = ["Rules"];
        bSVDB.label = "SVDB options"; // the button label
        bSVDB.description = "turn on SVDB and some options"; // tooltip
        bSVDB.activated.connect(setSVDB);

    } // end onInitialized

    // user-defined command echoVar
    Command { id: echoVar;
        commandString: "echo var change: ";
    }

    // define function to set SVDB options if the Mask SVDB checkbox is set
    function setSVDB() {
        echoVar.launchCommand();

        tool.svdb.query.value = true
        echoVar.commandString = "echo --svdb.query " + tool.svdb.query.value;
        echoVar.launchCommand();

        tool.svdb.asciixref.value = true
        echoVar.commandString = "echo --ascii " + tool.svdb.asciixref.value;
        echoVar.launchCommand();
    }
} // end Configuration
```

### Example 2

This example calls the touch shell command. The return code is tested and a message printed to the transcript. The command string is defined in the command definition.

```
Configuration {  
    Command { id: cmdTouch;  
              commandString: "/bin/touch test.txt";  
    }  
  
    onInitialized: {  
  
        if (cmdTouch.launchCommand() == 0) {  
            tool.transcript("touch success\n");  
        } else {  
            tool.transcript("touch failed\n");  
        }  
    } // end onInitialized  
} // end Configuration
```

## launchCommandReturnString

Command definition and function call for the custom configuration file for Calibre Interactive  
Executes an external (shell) command that returns a string.

### Usage

`cmdID.launchCommandReturnString`

#### Command Object Definition

```
Command { id: cmdID;  
          [commandString: "cmd";]  
        }
```

#### Object Properties

`cmdID.returnCode`  
`cmdID.commandString`

#### Arguments

- ***cmdID***  
A required argument specifying the ID for the command. It must start with a lowercase letter ([a-z]).
- ***cmd***  
An optional argument specifying the external command to execute, including any arguments. The command must be in your path and should return a string value.

The *cmd* can be defined in the command definition or with the `commandString` property.  
The *cmd* string must be enclosed in double quotes.

- Object Properties

**Table 10-9. Object Properties for launchCommandReturnString**

Property	Description
returnCode	An optional argument specifying the return code from the command ( <i>cmd</i> ).
commandString	An optional argument specifying the command to execute. See the <i>cmd</i> argument description.

#### Return Values

The string value returned by the *cmd*.

#### Description

The `launchCommandReturnString` function provides a way to call an external (shell) command, such as `readlink`, that returns a string. You can call `launchCommandReturnString` from a

function, call it directly in the onInitialized section of the configuration file, or use it as the connect command.

## Examples

This example uses the readlink command to print the full path for the file *README*. When successful, the example actually prints the full path twice: once when readlink is executed and once when the transcript command prints the variable *readmePath* containing return string.

```
Configuration {

    Command { id: cmdPrint;
        commandString: "/usr/bin/readlink -f ./README";
    }

    onInitialized: {

        var readmePath = cmdPrint.launchCommandReturnString();

        if (cmdPrint.returnCode === 0) {
            tool.transcript("\nREADME path: " + readmePath);
        } else {
            tool.transcript("\ncmdPrint failed");
        }
    } // end onInitialized
} // end Configuration
```

## Configuration File Commands to Add #DEFINE and #UNDEFINE Statements

Several commands are available for the Calibre Interactive custom configuration file that add a #DEFINE or #UNDEFINE statement to the Calibre Interactive control file.

See “[Configuration File Format for Calibre Interactive GUI Customization](#)” on page 240 for information on the file format and how to invoke Calibre Interactive with a custom configuration file.

You can use the command [addMutuallyExclusiveGroup](#) to group multiple define controls.

Command	Description
<a href="#">addChoicesDefine</a>	Adds a control to specify a #DEFINE or #UNDEFINE statement with a value to the Calibre Interactive GUI; the value is taken from an enumerated list. The statement is added to the control file.
<a href="#">addDefine</a>	Adds a control to specify a #DEFINE or #UNDEFINE statement with no value to the Calibre Interactive GUI. The statement is added to the control file.
<a href="#">addInputDirTextDefine</a>	Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to a directory used for input. The statement is added to the control file. The control includes a text input field and a browser button for selecting the directory.
<a href="#">addInputFileTextDefine</a>	Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to an input file. The statement is added to the control file. The control includes a text entry field, a file browser button for selecting the file, and a view file button.
<a href="#">addIntegerDefine</a>	Adds a control to specify a #DEFINE or #UNDEFINE statement with an integer value to the Calibre Interactive GUI. The statement is added to the control file.
<a href="#">addOutputDirTextDefine</a>	Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to a directory used for output. The statement is added to the control file. The control includes a text input field and a browser button for selecting the directory.

Command	Description
<a href="#">addOutputFileTextDefine</a>	Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to an output file. The statement is added to the control file. The control includes a text input field, a file browser button for selecting the file, and a view file button.
<a href="#">addRealDefine</a>	Adds a control to specify a #DEFINE or #UNDEFINE statement with a floating-point value to the Calibre Interactive GUI. The statement is added to the control file.
<a href="#">addTextBoxDefine</a>	Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI. The GUI control is a multi-line text box. The statement is added to the control file.
<a href="#">addTextDefine</a>	Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI. The statement is added to the control file.

## addChoicesDefine

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a #DEFINE or #UNDEFINE statement with a value to the Calibre Interactive GUI; the value is taken from an enumerated list. The statement is added to the control file.

### Usage

#### Function Syntax

```
ctrl_var = tool.addChoicesDefine("name", value_list[, label_list[, tooltip_list]])
```

#### Object Properties

```
ctrl_var.pages      = page_list
ctrl_var.label      = "control_label"
ctrl_var.description = "control_tooltip"
ctrl_var.value       = value
ctrl_var.editable    = true | false
ctrl_var.visible     = true | false
ctrl_var.optional    = true | false
ctrl_var.specified   = true | false
ctrl_var.valueQuoted = true | false
ctrl_var.tvf         = true | false
ctrl_var.hyperlink   = link_path
ctrl_var.insertToControlFile = true | false
ctrl_var.choiceAt(i).label = "choice_label"
ctrl_var.choiceAt(i).value = "choice_value"
ctrl_var.choiceAt(i).description = "choice_tooltip"
ctrl_var.choiceAt(i).compatible = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- **“*name*”**

A required argument specifying the variable name for the define statement that is written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- ***value\_list***

A required argument specifying a list of possible values for the define statement.

For example: [1, 2, 3] or [on, off]

Values are quoted by default in the statement written to the control file—set the valueQuoted property to false for integer or real values.

- *label\_list*

An optional argument specifying the list of labels for the control's dropdown list.

For example: [“one”, “two”, “three”]

If *label\_list* is not included in the function call, the labels can be defined with the choiceAt(*i*).label property.

- *tooltip\_list*

An optional argument specifying the list of tooltips for the control's dropdown list.

For example: [“first choice”, “second choice”, “third choice”]

If *tooltip\_list* is not included in the function call, the tooltips can be defined with the choiceAt(*i*).description property.

- Object Properties

The description and label properties for the choices list can be defined in the function call or with a separate property definition using choiceAt(*i*).

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addChoicesDefine are the following:

**Table 10-10. Object Properties for addChoicesDefine**

Property	Allowed Values	Description
value	<i>value</i> (string or number)	The assigned value for the #DEFINE statement. Set this property to one of the choice values to provide an initial value for the control.
valueQuoted	true   false (Boolean)	The assigned value is enclosed in quotes by default. Set valueQuoted to false to not include the quotes, as for an integer or real value.
tvf	true   false (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.

**Table 10-10. Object Properties for addChoicesDefine (cont.)**

Property	Allowed Values	Description
label	“control_label” (string)	The label displayed for the control. If not specified, the default label is “Define <i>name</i> ”.
choiceAt( <i>i</i> ).label	“choice_label” (string)	The label for the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control. Numbering starts at zero.
choiceAt( <i>i</i> ).value	<i>choice_value</i> (string, integer, or real)	The assigned value for the <i>i</i> <sup>th</sup> choice.
choiceAt( <i>i</i> ).description	“choice_tooltip” (string)	The tooltip for the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control.
choiceAt( <i>i</i> ).compatible	<u>true</u>   false (Boolean)	Specifies whether the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control is available (the default) or unavailable (dimmed).
optional	true   <u>false</u> (Boolean)	Places a checkbox next to the control. The default is false, which creates a required control.
specified	<u>true</u>   false (Boolean)	Sets the state of the checkbox.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the #DEFINE or #UNDEFINE statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addChoicesDefine adds a define control with an assigned value to the GUI, where the assigned value is taken from a dropdown list of choices. The control causes a #DEFINE or #UNDEFINE statement to be written to the Calibre Interactive control file.

When combined with appropriate logic, the choiceAt(*i*).compatible property can be used to set a choice as available or unavailable depending on other settings in the GUI.

The following statement is written to the control file if the control is required or if the optional checkbox is checked, where *value* is the selected choice from the dropdown list:

```
#DEFINE name "value"
```

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" "value"
tvf::set_global_variable "name" "value"
```

The value is quoted by default; set valueQuoted to false to not include quotes for both the TVF and non-TVF case.

If the control is optional and the checkbox is unchecked the following statement is written to the control file:

```
#UNDEFINE name
```

For the corresponding TVF case, the following statements are written to the control file:

```
unset -nocomplain "name"
tvf::unset_global_variable "name"
```

## Examples

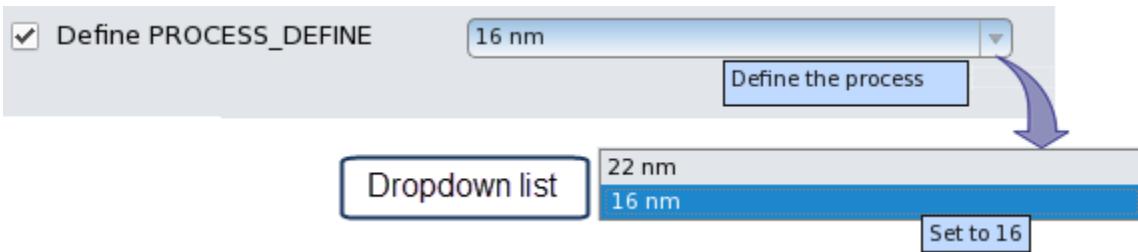
This example adds an optional define control with two possible choices. The assigned value is a number, so valueQuoted is set to false. The code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized.

The tool property “label” is not defined, so the default label of “Define **name**” is used, where **name** is the variable name for the define statement, PROCESS\_DEFINE in this case.

```
tool.addPage("Defines and Variables")
var d = tool.addChoicesDefine("PROCESS_DEFINE",
                               [22, 16],
                               ["22 nm", "16 nm"],
                               ["Set to 22", "Set to 16"])

d.pages      = ["Defines and Variables"]
d.description = "Define the process"
d.optional    = true;
d.specified   = true;

// set initial value to second choice, and do not include quotes
d.value = d.choiceAt(1).value
d.valueQuoted = false;
```



The following statement is added to the Calibre Interactive control file:

```
#DEFINE PROCESS_DEFINE 16
```

## Related Topics

[Configuration File Format for Calibre Interactive GUI Customization](#)

## addDefine

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a #DEFINE or #UNDEFINE statement with no value to the Calibre Interactive GUI. The statement is added to the control file.

### Usage

#### Function Syntax

`ctrl_var = tool.addDefine("name")`

#### Object Properties

<code>ctrl_var.pages</code>	<code>= page_list</code>
<code>ctrl_var.label</code>	<code>= "label"</code>
<code>ctrl_var.description</code>	<code>= "tooltip"</code>
<code>ctrl_var.value</code>	<code>= true   false</code>
<code>ctrl_var.editable</code>	<code>= true   false</code>
<code>ctrl_var.visible</code>	<code>= true   false</code>
<code>ctrl_var.tvf</code>	<code>= true   false</code>
<code>ctrl_var.hyperlink</code>	<code>= link_path</code>
<code>ctrl_var.insertToControlFile</code>	<code>= true   false</code>

### Arguments

- **`ctrl_var`**

A required argument specifying the variable name for the control.

- **`name`**

A required argument specifying the variable name for the define statement that is written to the Calibre Interactive control file. The **`name`** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the **`name`**, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- Object Properties

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, and visible.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified `link_path`.

The properties specific to addDefine are the following:

**Table 10-11. Object Properties for addDefine**

Property	Allowed Values	Description
value	(true   <u>false</u> )	Specifies the state of the checkbox for the addDefine control; the default is false.
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the #DEFINE or #UNDEFINE statement to the control file. The default is true; modify this property with caution.

**Note**

-  addDefine automatically adds a checkbox for the control. The “optional” property, which specifies a checkbox, is not valid for addDefine and is ignored if used.
- 

## Description

The function call tool.addDefine adds a define control with no value to the GUI, which causes a #DEFINE or #UNDEFINE statement to be written to the Calibre Interactive control file. The control has only a checkbox with the specified label.

The following statement is written to the control file if the control if the checkbox is checked:

```
#DEFINE name
```

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" 1
tvf::set_global_variable "name" 1
```

If the checkbox is unchecked the following statement is written to the control file:

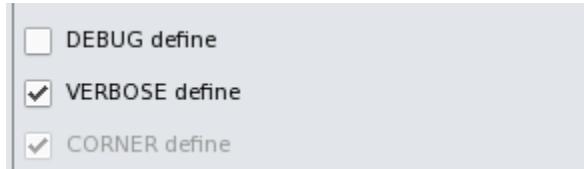
```
#UNDEFINE name
```

For the corresponding TVF case, the following statements are written to the control file:

```
set "name" 0
tvf::set_global_variable "name" 0
```

## Examples

The following code places these controls on the custom page named “Defines and Variables”. The code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized.



```
tool.pages = ["Rules", "Inputs", "Outputs", "Options", "Environment", "Run Control", "Defines and Variables"]

// add an optional define for the debug run, initial value off
var debugDef = tool.addDefine("DEBUG")
debugDef.pages = ["Defines and Variables"]
debugDef.label = "Define DEBUG"
debugDef.description = "include #DEFINE for DEBUG"

// add an optional define for the verbose run, initial value on
var verboseDef = tool.addDefine("VERBOSE")
verboseDef.value = true
verboseDef.pages = ["Defines and Variables"]
verboseDef.label = "Define VERBOSE"
verboseDef.description = "include #DEFINE for VERBOSE"

// add required define for CORNER
var d = tool.addDefine("CORNER")
d.pages = ["Defines and Variables"]
d.label = "CORNER define"
d.description = "required #DEFINE for CORNER"
d.value = true
d.editable = false
```

The following lines are added to the Calibre Interactive control file:

```
#UNDEFINE DEBUG
#define VERBOSE
#define CORNER
```

## Related Topics

[Configuration File Format for Calibre Interactive GUI Customization](#)

## addInputDirTextDefine

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to a directory used for input. The statement is added to the control file. The control includes a text input field and a browser button for selecting the directory.

### Usage

#### Function Syntax

```
ctrl_var = tool.addInputDirTextDefine("name"[, "dirName"[, "tooltip"]])
```

#### Object Properties

```
ctrl_var.value      = "dirName"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label       = "label"  
ctrl_var.valueQuoted = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.promptText = "prompt_string"  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional    = true | false  
ctrl_var.specified   = true | false  
ctrl_var.tvf         = true | false  
ctrl_var.hyperlink   = link_path  
ctrl_var.insertToControlFile = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- **“*name*”**

A required argument specifying the variable name for the define statement that is written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- **“*dirName*”**

An optional argument specifying the assigned text string for the define statement. If “***dirName***” is not included in the function call it can be defined with the value property.

- “*tooltip*”

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (*dirName*) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addInputDirTextDefine are the following:

**Table 10-12. Object Properties for addInputDirTextDefine**

Property	Allowed Values	Description
value	“ <i>dirName</i> ” (string)	The assigned value for the define statement.
valueQuoted	<u>true</u>   false (boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (boolean)	Allow spaces in strings; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the value property.
optional	<u>true</u>   <u>false</u> (boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (boolean)	Sets the state of the checkbox when optional = true.
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.

**Table 10-12. Object Properties for addInputDirTextDefine (cont.)**

Property	Allowed Values	Description
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the #DEFINE or #UNDEFINE statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addInputDirTextDefine adds a define control with a string value to the GUI, which causes a #DEFINE or #UNDEFINE statement to be written to the Calibre Interactive control file. The string value *dirName* is expected to correspond to a directory used for input. The control includes a text entry box and browser button. An error ! is displayed next to the control if the *dirName* is not readable or does not exist.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
#DEFINE name "dirName"
```

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" "dirName"
tvf::set_global_variable "name" "dirName"
```

The value is quoted by default; set valueQuoted to false to not include quotes for both the TVF and non-TVF case.

If the control is optional and the checkbox is unchecked (optional = true and specified = false) the following statement is written to the control file:

```
#UNDEFINE name
```

For the corresponding TVF case, the following statements are written to the control file:

```
unset -nocomplain "name"
tvf::unset_global_variable "name"
```

## Examples

See the example with [addInputFileDialog](#), which has similar syntax.

## addInputFileTextDefine

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to an input file. The statement is added to the control file. The control includes a text entry field, a file browser button for selecting the file, and a view file button.

### Usage

#### Function Syntax

```
ctrl_var = tool.addInputFileTextDefine("name"[, "fileName"[, "tooltip"]])
```

#### Object Properties

```
ctrl_var.value      = "fileName"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label       = "label"  
ctrl_var.valueQuoted = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.promptText = "prompt_string"  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional    = true | false  
ctrl_var.specified   = true | false  
ctrl_var.tvf         = true | false  
ctrl_var.hyperlink   = link_path  
ctrl_var.insertToControlFile = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- **“*name*”**

A required argument specifying the variable name for the define statement that is written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- **“*fileName*”**

An optional argument specifying the assigned text string for the define statement. If “*fileName*” is not included in the function call it can be defined with the value property.

- “*tooltip*”

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (*fileName*) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addInputFileTextDefine are the following:

**Table 10-13. Object Properties for addInputFileTextDefine**

Property	Allowed Values	Description
value	“ <i>fileName</i> ” (string)	The assigned value for the define statement.
valueQuoted	<u>true</u>   false (boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (boolean)	Allow spaces in strings; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the value property.
optional	<u>true</u>   <u>false</u> (boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (boolean)	Sets the state of the checkbox when optional = true.
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.

**Table 10-13. Object Properties for addInputFileTextDefine (cont.)**

Property	Allowed Values	Description
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the #DEFINE or #UNDEFINE statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addInputFileTextDefine adds a define control with a string value to the GUI, which causes a #DEFINE or #UNDEFINE statement to be written to the Calibre Interactive control file. The string value *fileName* is expected to correspond to an input file. The control includes a text entry box, a file browser button, and a view file button. An error  is displayed next to the control if the *fileName* is not readable or is not a file.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
#DEFINE name "fileName"
```

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" "fileName"  
tvf::set_global_variable "name" "fileName"
```

The value is quoted by default; set valueQuoted to false to not include quotes for both the TVF and non-TVF case.

If the control is optional and the checkbox is unchecked (optional = true and specified = false) the following statement is written to the control file:

```
#UNDEFINE name
```

For the corresponding TVF case, the following statements are written to the control file:

```
unset -nocomplain "name"  
tvf::unset_global_variable "name"
```

## Examples

The following code adds an optional control to provide a file name for the INPUT\_FILE define statement. The label property is not specified, so a default label is used. The code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized.

```
// #define for INPUT_FILE
var tdf = tool.addInputFileTextDefine("INPUT_FILE");
tdf.optional = true;
tdf.specified = false;
tdf.description = "optional input file";
```

The following control is added to the Custom page:



The default control adds the following statement to the Calibre Interactive control file:

```
#UNDEFINE INPUT_FILE
```

If the checkbox is clicked and a filename is provided, the following statement is added to the Calibre Interactive control file:

```
#DEFINE INPUT_FILE "my_input_file"
```

## addIntegerDefine

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a #DEFINE or #UNDEFINE statement with an integer value to the Calibre Interactive GUI. The statement is added to the control file.

### Usage

#### Function Syntax

```
ctrl_var = tool.addIntegerDefine("name"[, intValue[, "tooltip"]])
```

#### Object Properties

<i>ctrl_var.value</i>	= <i>intValue</i>
<i>ctrl_var.description</i>	= <i>"tooltip"</i>
<i>ctrl_var.pages</i>	= <i>page_list</i>
<i>ctrl_var.label</i>	= <i>"label"</i>
<i>ctrl_var.minValue</i>	= <i>min</i>
<i>ctrl_var.maxValue</i>	= <i>max</i>
<i>ctrl_var.singleStep</i>	= <i>step</i>
<i>ctrl_var.editable</i>	= <u>true</u>   false
<i>ctrl_var.visible</i>	= <u>true</u>   false
<i>ctrl_var.optional</i>	= true   <u>false</u>
<i>ctrl_var.specified</i>	= <u>true</u>   false
<i>ctrl_var.tvf</i>	= true   <u>false</u>
<i>ctrl_var.hyperlink</i>	= <i>link_path</i>
<i>ctrl_var.insertToControlFile</i>	= <u>true</u>   false

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- **“*name*”**

A required argument specifying the variable name for the define statement that is written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- ***intValue***

An optional argument specifying the assigned value for the define statement. If “***intValue***” is not included in the function call it can be defined with the value property.

- **“*tooltip*”**

An optional argument specifying the hover text (tooltip) for the control. If “***tooltip***” is not included in the function call it can be defined with the description property.

- Object Properties

The value (*intValue*) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addIntegerDefine are the following:

**Table 10-14. Object Properties for addIntegerDefine**

Property	Value (type)	Description
value	<i>intValue</i> (integer)	The assigned value for the define statement.
minValue	<i>min</i> (integer)	The minimum value allowed for the value property.  An error notification (⚠) is displayed next to the control if the condition is not satisfied.
maxValue	<i>max</i> (integer)	The maximum value allowed for the value property.  An error notification (⚠) is displayed next to the control if the condition is not satisfied.
singleStep	<i>step</i> (integer)	Specify a <i>step</i> value in order to use a spin control in the GUI to set the value property. The properties minValue and maxValue must also be specified.  When the spin control up or down arrow is pressed, the current value is incremented or decremented by <i>step</i> . It is still possible to enter a value directly into the entry field of a spin control, as long as <i>min</i> <= <i>intValue</i> <= <i>max</i> .
optional	true   false (Boolean)	Set to true to add a checkbox for the control. The default is false, which creates a required control.
specified	true   false (Boolean)	Sets the state of the checkbox when optional = true.

**Table 10-14. Object Properties for addIntegerDefine (cont.)**

Property	Value (type)	Description
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	true   false (Boolean)	Specifies whether to write the #DEFINE or #UNDEFINE statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addIntegerDefine adds a define control with an integer value to the GUI, which causes a #DEFINE or #UNDEFINE statement to be written to the Calibre Interactive control file. The control includes a text entry box.

The properties minValue and maxValue specify minimum and maximum allowed values for *intValue* and may be specified with or without the singleStep property. Without singleStep, an error notification (■) is displayed next to the control if the condition is not satisfied. A spin control is included if singleStep is defined. The minValue and maxValue properties are required if singleStep is specified, and place limits on the spin control.

The following statement is written to the control file if the control is required or if the optional checkbox is checked, where *intValue* is replaced with the actual value:

```
#DEFINE name intValue
```

For the corresponding TVF case, the following statements are written to the control file:

```
unset -nocomplain "name"
tvf::unset_global_variable "name"
```

If the control is optional and the checkbox is unchecked the following statement is written to the control file:

```
#UNDEFINE name
```

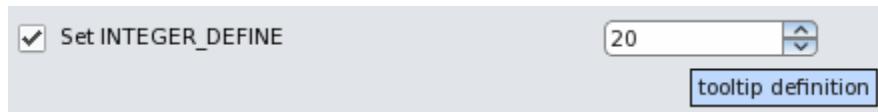
For the corresponding TVF case, the following statements are written to the control file:

```
unset -nocomplain "name"
tvf::unset_global_variable "name"
```

## Examples

This example code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized.

```
var d = tool.addIntegerDefine("INTEGER_DEFINE", 20)
d.pages      = ["Defines and Variables"]
d.label      = "Set INTEGER_DEFINE"
d.description = "tooltip definition"
d.minValue   = -100;
d maxValue    = 200;
d.singleStep = 10;
d.optional   = true
d.specified  = true
```



The following line is written to the control file:

```
#DEFINE INTEGER_DEFINE 20
```

## Related Topics

[Configuration File Format for Calibre Interactive GUI Customization](#)

## addOutputDirTextDefine

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to a directory used for output. The statement is added to the control file. The control includes a text input field and a browser button for selecting the directory.

### Usage

#### Function Syntax

```
ctrl_var = tool.addOutputDirTextDefine("name"[, "dirName"[, "tooltip"]])
```

#### Object Properties

```
ctrl_var.value      = "dirName"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label       = "label"  
ctrl_var.valueQuoted = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.promptText = "prompt_string"  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional    = true | false  
ctrl_var.specified   = true | false  
ctrl_var.tvf          = true | false  
ctrl_var.hyperlink   = link_path  
ctrl_var.insertToControlFile = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- **“*name*”**

A required argument specifying the variable name for the define statement that is written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- **“*dirName*”**

An optional argument specifying the assigned text string for the define statement. If “*dirName*” is not included in the function call it can be defined with the value property.

- “*tooltip*”

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (*dirName*) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addOutputDirTextDefine are the following:

**Table 10-15. Object Properties for addOutputDirTextDefine**

Property	Allowed Values	Description
value	“ <i>dirName</i> ” (string)	The assigned value for the define statement.
valueQuoted	<u>true</u>   false (boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (boolean)	Allow spaces in strings; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the value property.
optional	<u>true</u>   <u>false</u> (boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (boolean)	Sets the state of the checkbox when optional = true.
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.

**Table 10-15. Object Properties for addOutputDirTextDefine (cont.)**

Property	Allowed Values	Description
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the #DEFINE or #UNDEFINE statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addOutputDirTextDefine adds a define control with a string value to the GUI, which causes a #DEFINE or #UNDEFINE statement to be written to the Calibre Interactive control file. The string value *dirName* is expected to correspond to a directory used for output. The control includes a text entry box and browser button. An error ! is displayed next to the control if the *dirName* is not writable.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
#DEFINE name "dirName"
```

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" "dirName"  
tvf::set_global_variable "name" "dirName"
```

The value is quoted by default; set valueQuoted to false to not include quotes for both the TVF and non-TVF case.

If the control is optional and the checkbox is unchecked (optional = true and specified = false) the following statement is written to the control file:

```
#UNDEFINE name
```

For the corresponding TVF case, the following statements are written to the control file:

```
unset -nocomplain "name"  
tvf::unset_global_variable "name"
```

## Examples

See the example with [addInputFileTextDefine](#), which has similar syntax.

## addOutputFileTextDefine

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to an output file. The statement is added to the control file. The control includes a text input field, a file browser button for selecting the file, and a view file button.

### Usage

#### Function Syntax

```
ctrl_var = tool.addOutputFileTextDefine("name"[, "fileName"[, "tooltip"]])
```

#### Object Properties

```
ctrl_var.value      = "fileName"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label       = "label"  
ctrl_var.valueQuoted = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.promptText = "prompt_string"  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional    = true | false  
ctrl_var.specified   = true | false  
ctrl_var.tvf         = true | false  
ctrl_var.hyperlink   = link_path  
ctrl_var.insertToControlFile = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- **“*name*”**

A required argument specifying the variable name for the define statement that is written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- **“*fileName*”**

An optional argument specifying the assigned text string for the define statement. If “*fileName*” is not included in the function call it can be defined with the value property.

- “*tooltip*”

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (*fileName*) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addOutputFileTextDefine are the following:

**Table 10-16. Object Properties for addOutputFileTextDefine**

Property	Allowed Values	Description
value	“ <i>fileName</i> ” (string)	The assigned value for the define statement.
valueQuoted	<u>true</u>   false (boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (boolean)	Allow spaces in strings; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the value property.
optional	<u>true</u>   <u>false</u> (boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (boolean)	Sets the state of the checkbox when optional = true.
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.

**Table 10-16. Object Properties for addOutputFileTextDefine (cont.)**

Property	Allowed Values	Description
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the #DEFINE or #UNDEFINE statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addOutputFileTextDefine adds a define control with a string value to the GUI, which causes a #DEFINE or #UNDEFINE statement to be written to the Calibre Interactive control file. The string value *fileName* is expected to correspond to an output file. The control includes a text entry box, and file browser button, and a view file button. An error ! is displayed next to the control if the *fileName* is not writable.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
#DEFINE name "fileName"
```

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" "fileName"
tvf::set_global_variable "name" "fileName"
```

The value is quoted by default; set valueQuoted to false to not include quotes for both the TVF and non-TVF case.

If the control is optional and the checkbox is unchecked (optional = true and specified = false) the following statement is written to the control file:

```
#UNDEFINE name
```

For the corresponding TVF case, the following statements are written to the control file:

```
unset -nocomplain "name"
tvf::unset_global_variable "name"
```

## Examples

See the example with [addInputFileTextDefine](#), which has similar syntax.

## addRealDefine

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a #DEFINE or #UNDEFINE statement with a floating-point value to the Calibre Interactive GUI. The statement is added to the control file.

### Usage

#### Function Syntax

*ctrl\_var* = tool.addRealDefine("name"[, *realValue*[, "tooltip"]])

#### Object Properties

<i>ctrl_var.value</i>	= <i>realValue</i>
<i>ctrl_var.description</i>	= "tooltip"
<i>ctrl_var.pages</i>	= <i>page_list</i>
<i>ctrl_var.label</i>	= "label"
<i>ctrl_var.minValue</i>	= <i>min</i>
<i>ctrl_var.maxValue</i>	= <i>max</i>
<i>ctrl_var.singleStep</i>	= <i>step</i>
<i>ctrl_var.includeMinEndpoint</i>	= <u>true</u>   false
<i>ctrl_var.includeMaxEndpoint</i>	= <u>true</u>   false
<i>ctrl_var.editable</i>	= <u>true</u>   false
<i>ctrl_var.visible</i>	= <u>true</u>   false
<i>ctrl_var.optional</i>	= true   <u>false</u>
<i>ctrl_var.specified</i>	= <u>true</u>   false
<i>ctrl_var.tvf</i>	= true   <u>false</u>
<i>ctrl_var.hyperlink</i>	= <i>link_path</i>
<i>ctrl_var.insertToControlFile</i>	= <u>true</u>   false

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- ***name***

A required argument specifying the variable name for the define statement that is written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- ***realValue***

An optional argument specifying the floating-point assigned value for the define statement. If “***realValue***” is not included in the function call it can be defined with the value property.

- *tooltip*

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (*realValue*) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, value, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addRealDefine are the following:

**Table 10-17. Object Properties for addRealDefine**

Property	Value (type)	Description
value	<i>realValue</i> (floating-point)	The assigned value for the define statement.
minValue	<i>min</i> (floating-point)	The minimum value allowed for the value property.  An error notification (⚠) is displayed next to the control if the condition is not satisfied.
maxValue	<i>max</i> (floating-point)	The maximum value allowed for the value property.  An error notification (⚠) is displayed next to the control if the condition is not satisfied.
singleStep	<i>step</i> (floating-point)	Specify a <i>step</i> value in order to use a spin control in the GUI to set the value property. The properties minValue and maxValue must also be specified.  When the spin control up or down arrow is pressed, the current value is incremented or decremented by <i>step</i> . It is still possible to enter a value directly into the entry field of a spin control, as long as <i>realValue</i> satisfies the conditions dictated by minValue, maxValue, includeMinEndpoint, and includeMaxEndpoint.
includeMinEndpoint	<u>true</u>   false (Boolean)	Specifies whether to include minValue in the range.

**Table 10-17. Object Properties for addRealDefine (cont.)**

Property	Value (type)	Description
includeMaxEndpoint	true   false (Boolean)	Specifies whether to include maxValue in the range
optional	true   false (Boolean)	Set to true to add a checkbox for the control. The default is false, which creates a required control.
specified	true   false (Boolean)	Sets the state of the checkbox when optional = true.
tvf	true   false (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	true   false (Boolean)	Specifies whether to write the #DEFINE or #UNDEFINE statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addRealDefine adds a define control with a real value to the GUI, which causes a #DEFINE or #UNDEFINE statement to be written to the Calibre Interactive control file.

If the tvf property is set to true, then a Tcl variable is inserted in the Calibre Interactive control file, rather than a #DEFINE statement.

The properties minValue, maxValue, includeMinEndpoint, and includeMaxEndpoint specify minimum and maximum allowed values for *realValue* and may be specified with or without the singleStep property. Without singleStep, an error notification ( ! ) is displayed next to the control if the condition is not satisfied. A spin control is included if singleStep is defined. The minValue and maxValue properties are required if singleStep is specified, and place limits on the spin control.

The following statement is written to the control file if the control is required or if the optional checkbox is checked, where *realValue* is replaced with the actual value:

```
#DEFINE name realValue
```

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" realValue
tvf::set_global_variable "name" realValue
```

If the control is optional and the checkbox is unchecked the following statement is written to the control file:

```
#UNDEFINE name
```

For the corresponding TVF case, the following statements are written to the control file:

```
unset -nocomplain "name"
tvf::unset_global_variable "name"
```

## Related Topics

[Configuration File Format for Calibre Interactive GUI Customization](#)

[addIntegerDefine](#)

## addTextBoxDefine

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI. The GUI control is a multi-line text box. The statement is added to the control file.

### Usage

#### Function Syntax

```
ctrl_var = tool.addTextBoxDefine("name"[, "textValue"[, "tooltip"]])
```

#### Object Properties

```
ctrl_var.value      = "textValue"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label       = "label"  
ctrl_var.valueQuoted = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.emptyIsValid = true | false  
ctrl_var.promptText = "prompt_string"  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional    = true | false  
ctrl_var.specified   = true | false  
ctrl_var.tvf         = true | false  
ctrl_var.hyperlink   = link_path  
ctrl_var.insertToControlFile = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- **“name”**

A required argument specifying the variable name for the define statement that is written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- **“textValue”**

An optional argument specifying the assigned text string for the define statement. If “***textValue***” is not included in the function call it can be defined with the value property.

- “*tooltip*”

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (*textValue*) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, and visible.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addTextDefine are the following:

**Table 10-18. Object Properties for addTextDefine**

Property	Allowed Values	Description
value	“ <i>textValue</i> ” (string)	The assigned value for the define statement.
valueQuoted	<u>true</u>   false (Boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (Boolean)	Allow spaces in strings; the default is false.
emptyIsValid	true   <u>false</u> (Boolean)	Allow an empty <i>textValue</i> string; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the value property.
optional	true   <u>false</u> (Boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (Boolean)	Sets the state of the checkbox when optional = true.

**Table 10-18. Object Properties for addTextDefine (cont.)**

Property	Allowed Values	Description
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the #DEFINE or #UNDEFINE statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addTextBoxDefine adds a define control with a string value to the GUI, which causes a #DEFINE or #UNDEFINE statement to be written to the Calibre Interactive control file. The control includes a multi-line text entry box, in contrast to addTextDefine, which has a single line text entry box.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
#DEFINE name "textValue"
```

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" "textValue"
tvf::set_global_variable "name" "textValue"
```

The value is quoted by default; set valueQuoted to false to not include quotes for both the TVF and non-TVF case.

If the control is optional and the checkbox is unchecked (optional = true and specified = false) the following statement is written to the control file:

```
#UNDEFINE name
```

For the corresponding TVF case, the following statements are written to the control file:

```
unset -nocomplain "name"
tvf::unset_global_variable "name"
```

## Examples

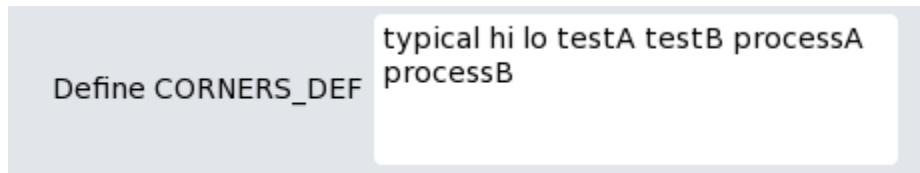
The following code adds a required control to provide a string value for the CORNERS\_DEF variable. The initial value is “typical”. The code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized.

```
// #DEFINE for CORNERS_DEF
var d = tool.addTextDefine("CORNERS_DEF",
                           "typical hi lo testA testB processA processB",
                           "space separated list of corners")
d.pages      = ["Defines and Variables"]
d.allowSpaces = true
```

The following statement is added to the Calibre Interactive control file:

```
#DEFINE CORNERS_DEF "typical hi lo testA testB processA processB"
```

The control looks like this in the GUI:



## addTextDefine

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a #DEFINE or #UNDEFINE statement with a string value to the Calibre Interactive GUI. The statement is added to the control file.

### Usage

#### Function Syntax

```
ctrl_var = tool.addTextDefine("name"[, "textValue"[, "tooltip"]])
```

#### Object Properties

```
ctrl_var.value      = "textValue"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label       = "label"  
ctrl_var.valueQuoted = true | false  
ctrl_var.emptyIsValid = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.promptText = "prompt_string"  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional     = true | false  
ctrl_var.specified    = true | false  
ctrl_var.tvf          = true | false  
ctrl_var.hyperlink    = link_path  
ctrl_var.insertToFile = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- “***name***”

A required argument specifying the variable name for the define statement that is written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- “***textValue***”

An optional argument specifying the assigned text string for the define statement. If “***textValue***” is not included in the function call it can be defined with the value property.

- “***tooltip***”

An optional argument specifying the hover text (tooltip) for the control. If “***tooltip***” is not included in the function call it can be defined with the description property.

- Object Properties

The value (*textValue*) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addTextDefine are the following:

**Table 10-19. Object Properties for addTextDefine**

Property	Allowed Values	Description
value	“ <i>textValue</i> ” (string)	The assigned value for the define statement.
valueQuoted	<u>true</u>   false (Boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (Boolean)	Allow spaces in strings; the default is false.
emptyIsValid	true   <u>false</u> (Boolean)	Allow an empty <i>textValue</i> string; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the “value” property.
optional	true   <u>false</u> (Boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (Boolean)	Sets the state of the checkbox when optional = true.
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.

**Table 10-19. Object Properties for addTextDefine (cont.)**

Property	Allowed Values	Description
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the #DEFINE or #UNDEFINE statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addTextDefine adds a define control with a string value to the GUI, which causes a #DEFINE or #UNDEFINE statement to be written to the Calibre Interactive control file. The control includes a text entry box.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
#DEFINE name "textValue"
```

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" "textValue"  
tvf::set_global_variable "name" "textValue"
```

The value is quoted by default; set valueQuoted to false to not include quotes for both the TVF and non-TVF case.

If the control is optional and the checkbox is unchecked (optional = true and specified = false) the following statement is written to the control file:

```
#UNDEFINE name
```

For the corresponding TVF case, the following statements are written to the control file:

```
unset -nocomplain "name"  
tvf::unset_global_variable "name"
```

## Examples

The following code adds a required control to provide a string value for the CORNERS\_DEF variable. The initial value is “typical”. The code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized.

```
// #define for CORNERS_DEF  
var d = tool.addTextDefine("CORNERS_DEF", "typical")  
d.pages      = ["Defines and Variables"]  
d.label      = "Define CORNERS_DEF for corner settings"  
d.description = "Enter a space separated list of corners"  
d.allowSpaces = true
```

The following statement is added to the Calibre Interactive control file:

```
#DEFINE CORNERS_DEF "typical"
```

## Related Topics

[Configuration File Format for Calibre Interactive GUI Customization](#)

## Configuration File Functions to Add Variable Statements

Several functions are available for the Calibre Interactive custom configuration file that add a Variable statement to the Calibre Interactive control file.

See “[Configuration File Format for Calibre Interactive GUI Customization](#)” on page 240 for information on the file format and how to invoke Calibre Interactive with a custom configuration file.

Command	Description
<a href="#">addChoicesVariable</a>	Adds a control to specify a Variable statement with a value to the Calibre Interactive GUI; the value is taken from an enumerated list. The statement is added to the control file.
<a href="#">addListVariable</a>	Adds a control to specify a Variable statement to the Calibre Interactive GUI, where the assigned value for the variable is a list. The statement is added to the control file.
<a href="#">addInputDirTextVariable</a>	Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to a directory used for input. The Variable statement is added to the control file. The control includes a text entry field and a browser button for selecting the directory.
<a href="#">addInputFileTextVariable</a>	Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to an input file. The Variable statement is added to the control file. The control includes a text entry field, a file browser button for selecting the file, and a view file button.
<a href="#">addIntegerVariable</a>	Adds a control to specify a Variable statement with an integer value to the Calibre Interactive GUI. The statement is added to the control file.
<a href="#">addMultiChoicesVariable</a>	Adds a control to specify a Variable statement with a value to the Calibre Interactive GUI. The value is taken from an enumerated list and multiple values can be selected. The statement is added to the control file.
<a href="#">addOutputDirTextVariable</a>	Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to a directory used for output. The Variable statement is added to the control file. The control includes a text entry field and a browser button for selecting the directory.

Command	Description
<a href="#">addOutputFileTextVariable</a>	Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to an output file. The Variable statement is added to the control file. The control includes a text entry field, a file browser button for selecting the file, and a view file button.
<a href="#">addRealVariable</a>	Adds a control to specify a Variable statement with a floating-point value to the Calibre Interactive GUI. The statement is added to the control file.
<a href="#">addTextBoxVariable</a>	Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI. The GUI control is a multi-line text box. The statement is added to the control file.
<a href="#">addTextVariable</a>	Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI. The statement is added to the control file.

## addChoicesVariable

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a Variable statement with a value to the Calibre Interactive GUI; the value is taken from an enumerated list. The statement is added to the control file.

### Usage

#### Function Syntax

```
ctrl_var = tool.addChoicesVariable("name", value_list[, label_list[, tooltip_list]])
```

#### Object Properties

<i>ctrl_var.pages</i>	= <i>page_list</i>
<i>ctrl_var.label</i>	= "control_label"
<i>ctrl_var.description</i>	= "control_tooltip"
<i>ctrl_var.value</i>	= <i>value</i>
<i>ctrl_var.editable</i>	= <u>true</u>   false
<i>ctrl_var.visible</i>	= <u>true</u>   false
<i>ctrl_var.optional</i>	= true   <u>false</u>
<i>ctrl_var.specified</i>	= <u>true</u>   false
<i>ctrl_var.valueQuoted</i>	= <u>true</u>   false
<i>ctrl_var.tvf</i>	= true   <u>false</u>
<i>ctrl_var.hyperlink</i>	= <i>link_path</i>
<i>ctrl_var.insertToControlFile</i>	= <u>true</u>   false
<i>ctrl_var.choiceAt(i).label</i>	= "choice_label"
<i>ctrl_var.choiceAt(i).value</i>	= "choice_value"
<i>ctrl_var.choiceAt(i).description</i>	= "choice_tooltip"
<i>ctrl_var.choiceAt(i).compatible</i>	= <u>true</u>   false

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- **“*name*”**

A required argument specifying the name of the variable for the [Variable](#) statement written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- ***value\_list***

A required argument specifying the list of possible values for the Variable statement. For example: [1, 2, 3]

Values are quoted by default in the statement written to the control file—set the *valueQuoted* property to false for integer or real values.

- *label\_list*

An optional argument specifying a list of labels for the control's dropdown list.

For example: [“one”, “two”, “three”]

If *label\_list* is not included in the function call, the labels can be defined with the choiceAt(*i*).label property.

- *tooltip\_list*

An optional argument specifying a list of tooltips (hover text) for the control's dropdown list.

For example: [“first choice”, “second choice”, “third choice”]

If *tooltip\_list* is not included in the function call, the tooltips can be defined with the choiceAt(*i*).description property.

- Object Properties

The label (*label\_list*) and description (*tooltip\_list*) properties for the choices list can be defined in the function call or with a separate property definition using choiceAt(*i*).

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addChoicesVariable are the following:

**Table 10-20. Object Properties for addChoicesVariable**

Property	Allowed Values	Description
value	<i>value</i> (string or number)	The assigned value for the Variable statement. Set this property to one of the choice values to provide an initial value for the control.
optional	true   <u>false</u> (Boolean)	Places a checkbox next to the control. The default is false, which creates a required control.
specified	<u>true</u>   false (Boolean)	Sets the state of the checkbox when optional=true.
valueQuoted	<u>true</u>   false (Boolean)	The assigned value is enclosed in quotes by default. Set valueQuoted to false to not include the quotes, as for an integer or real value.

**Table 10-20. Object Properties for addChoicesVariable (cont.)**

Property	Allowed Values	Description
tvf	true   false (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the Variable statement to the control file. The default is true; modify this property with caution.
choiceAt( <i>i</i> ).label	<i>choice_label</i> (string)	The label for the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control. Numbering starts at zero.
choiceAt( <i>i</i> ).value	<i>choice_value</i> (string, integer, or real)	The assigned value for the <i>i</i> <sup>th</sup> choice.
choiceAt( <i>i</i> ).description	<i>choice_tooltip</i> (string)	The tooltip for the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control.
choiceAt( <i>i</i> ).compatible	<u>true</u>   false (Boolean)	Specifies whether the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control is available (the default) or unavailable (dimmed).

## Description

The function call tool.addChoicesVariable adds a variable control with a dropdown list of choices to the GUI. The control causes a **Variable** statement to be written to the Calibre Interactive control file.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true), where “value” is one of the available choices:

```
VARIABLE name "value"
```

The value is quoted by default—set valueQuoted to false to not use quotes.

When combined with appropriate logic, the choiceAt(*i*).compatible property can be used to set a choice as available or unavailable depending on other settings in the GUI.

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" {value}
tvf::set_global_variable "name" {value}
```

The property valueQuoted is not used when the tvf property is set to true.

## Examples

This example adds an optional control to include a Variable statement to set VAR\_TEMP, where there are two possible choices for the assigned value. The property valueQuoted is set to false in order to define a numeric value. The label property is not defined so the default label of “Variable VAR\_TEMP” is used for the control. The code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized.

```
var cv = tool.addChoicesVariable("VAR_TEMP",
[25, 27],
["25", "27"],
["25 degrees", "27 degrees"])
cv.pages      = ["Defines and Variables"]
cv.description = "Defines SVRF Variable VAR_TEMP for the temperature"
cv.optional   = true;
cv.specified  = true;
// necessary to get numeric value rather than a string
cv.valueQuoted = false;
// set initial value to one of the choices
cv.value = cv.choiceAt(1).value
```

The following statement is added to the Calibre Interactive control file:

```
VARIABLE VAR_TEMP 27
```

## Related Topics

[Configuration File Format for Calibre Interactive GUI Customization](#)

[addMultiChoicesVariable](#)

## addListVariable

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a Variable statement to the Calibre Interactive GUI, where the assigned value for the variable is a list. The statement is added to the control file.

### Usage

#### Function Syntax

```
ctrl_var = tool.addListVariable("name"[, value_list[, "tooltip"]])
```

#### Object Properties

```
ctrl_var.value      = value_list
ctrl_var.description = "tooltip"
ctrl_var.pages      = page_list
ctrl_var.label       = "label"
ctrl_var.valueQuoted = true | false
ctrl_var.promptText = "prompt_string"
ctrl_var.editable    = true | false
ctrl_var.visible     = true | false
ctrl_var.optional    = true | false
ctrl_var.specified   = true | false
ctrl_var.tvf         = true | false
ctrl_var.hyperlink   = link_path
ctrl_var.insertToFile = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- ***“name”***

A required argument specifying the name of the variable for the [Variable](#) statement added to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- ***value\_list***

An optional argument specifying a list of items specifying the assigned value for the Variable statement. For example: [“buf”, “inv”, “or”]

If *value\_list* is not included in the function call it can be defined with the value property.

- ***“tooltip”***

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (*value\_list*) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addListVariable are the following:

**Table 10-21. Object Properties for addListVariable**

Property	Allowed Values	Description
value	<i>value_list</i> (list of strings or numbers)	The assigned value for the Variable statement.  For example: [“buf”, “inv”, “or”]
valueQuoted	<u>true</u>   false (Boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the value property.
optional	true   <u>false</u> (Boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (Boolean)	Sets the state of the checkbox when optional=true.
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the Variable statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addListVariable adds a variable control with a list value to the GUI, which causes a [Variable](#) statement to be written to the Calibre Interactive control file. The control includes a text entry box.

If the value property is set to [item1, item2], the following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
VARIABLE name "item1" "item2"
```

The list items are enclosed in quotes by default; set valueQuoted to false to not use the quotes. Spaces are not allowed within strings in the Variable statement, therefore a string with a space is considered as two separate strings; for example, “aa bb” is considered as “aa” “bb”.

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" {item1 item2}  
tvf::set_global_variable "name" {item1 item2}
```

The property valueQuoted is not used when the tvf property is set to true.

## Examples

This example defines a variable with a list of cell names. The code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized. The list value is included in the function call, while the tooltip is specified as a property definition. The label property is not specified and defaults to “Variable VAR\_CELL\_LIST”. The pages property is not specified, so the control appears on the default “Custom” page.

```
// VARIABLE VAR_CELL_LIST for list of strings for cell names  
var v = tool.addListVariable("VAR_CELL_LIST",  
                           ["inv", "buf", "cella cellb"]);  
v.description = "Adds SVRF Variable statement for VAR_CELL_LIST"
```

The following control is created:



Variable VAR\_CELL\_LIST inv buf cella cellb

This statement is added to the control file for Calibre Interactive:

```
VARIABLE VAR_CELL_LIST "inv" "buf" "cella" "cellb"
```

## Related Topics

[Configuration File Format for Calibre Interactive GUI Customization](#)

## addInputDirTextVariable

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to a directory used for input. The Variable statement is added to the control file. The control includes a text entry field and a browser button for selecting the directory.

---

### Note

 Variables defined as a directories may be used in the [tvf::svrf\\_var](#) command, however, such variables cannot be used to specify directories for SVRF operations and statements. See the [Variable](#) statement.

---

## Usage

### Function Syntax

```
ctrl_var = tool.addInputDirTextVariable("name"[, "dirName"[, "tooltip"]])
```

### Object Properties

```
ctrl_var.value      = "dirName"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label      = "label"  
ctrl_var.promptText = "prompt_string"  
ctrl_var.valueQuoted = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional    = true | false  
ctrl_var.specified   = true | false  
ctrl_var.tvf         = true | false  
ctrl_var.hyperlink   = link_path  
ctrl_var.insertToFile = true | false
```

## Arguments

- **ctrl\_var**

A required argument specifying the variable name for the control.

- **name**

A required argument specifying the name of the variable for the Variable statement written to the Calibre Interactive control file. The **name** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the **name**, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- “*dirName*”

An optional argument specifying the assigned text string for the Variable statement. If “*dirName*” is not included in the function call it can be defined with the value property.

- “*tooltip*”

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (“*dirName*”) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addInputDirTextVariable are the following:

**Table 10-22. Object Properties for addInputDirTextVariable**

Property	Allowed Values	Description
value	“ <i>dirName</i> ” (string)	The assigned value for the Variable statement.
valueQuoted	<u>true</u>   false (boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (boolean)	Allow spaces in strings; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the value property.
optional	<u>true</u>   false (boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (boolean)	Sets the state of the checkbox when optional=true.

**Table 10-22. Object Properties for addInputDirTextVariable (cont.)**

Property	Allowed Values	Description
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the Variable statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addInputDirTextVariable adds a variable control with a string value to the GUI, which causes a [Variable](#) statement to be written to the Calibre Interactive control file. The string value *dirName* is expected to correspond to a directory used for input. The control includes a text entry box and a browser button. An error  is displayed next to the control if the *dirName* is not readable or does not exist.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
VARIABLE name "dirName"
```

The value is quoted by default; set valueQuoted to false to not include quotes. Quotes are generally recommended.

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" {dirName}
tvf::set_global_variable "name" {dirName}
```

The property valueQuoted is not used when the tvf property is set to true.

## Examples

See the example with [addInputFileDialogVariable](#), which has similar syntax.

## addInputFileTextVariable

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to an input file. The Variable statement is added to the control file. The control includes a text entry field, a file browser button for selecting the file, and a view file button.

### Note

 Variables defined as filenames may be used in the `tvf::svrf_var` command, however, such variables cannot be used to specify filenames for SVRF operations and statements. See the [Variable](#) statement.

---

## Usage

### Function Syntax

```
ctrl_var = tool.addInputFileTextVariable("name"[, "fileName"[, "tooltip"]])
```

### Object Properties

```
ctrl_var.value      = "fileName"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label       = "label"  
ctrl_var.promptText = "prompt_string"  
ctrl_var.valueQuoted = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional    = true | false  
ctrl_var.specified   = true | false  
ctrl_var.tvf         = true | false  
ctrl_var.hyperlink   = link_path  
ctrl_var.insertToFile = true | false
```

## Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- ***name***

A required argument specifying the name of the variable for the Variable statement written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- “*fileName*”

An optional argument specifying the assigned text string for the Variable statement. If “*fileName*” is not included in the function call it can be defined with the value property.

- “*tooltip*”

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (“*fileName*”) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addInputFileTextVariable are the following:

**Table 10-23. Object Properties for addInputFileTextVariable**

Property	Allowed Values	Description
value	“ <i>fileName</i> ” (string)	The assigned value for the Variable statement.
valueQuoted	<u>true</u>   false (boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (boolean)	Allow spaces in strings; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the value property.
optional	<u>true</u>   false (boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (boolean)	Sets the state of the checkbox when optional=true.

**Table 10-23. Object Properties for addInputFileDialogVariable (cont.)**

Property	Allowed Values	Description
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the Variable statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addInputFileDialogVariable adds a variable control with a string value to the GUI, which causes a [Variable](#) statement to be written to the Calibre Interactive control file. The string value *fileName* is expected to correspond to an input file. The control includes a text entry box, a file browser button, and a view file button. An error  is displayed next to the control if the *fileName* is not readable or is not a file.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
VARIABLE name "fileName"
```

The value is quoted by default; set valueQuoted to false to not include quotes. Quotes are generally recommended.

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" {fileName}
tvf::set_global_variable "name" {fileName}
```

The property valueQuoted is not used when the tvf property is set to true.

## Examples

The following code adds a control to specify a Variable statement with a string value corresponding to a file name. The code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized. The variable name, initial setting for the file name, and tooltip are specified in the function call.

```
var var_f = tool.addInputFileTextVariable("VAR_INPUT_FILE",
                                         "test_input",
                                         "define Variable VAR_INPUT_FILE");
```

The following control is added to the Custom page:



The following statement is added to the control file for Calibre Interactive:

```
VARIABLE VAR_INPUT_FILE "test_input"
```

## addIntegerVariable

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a Variable statement with an integer value to the Calibre Interactive GUI. The statement is added to the control file.

### Usage

#### Function Syntax

```
ctrl_var = tool.addIntegerVariable("name"[, intValue[, "tooltip"]])
```

#### Object Properties

<i>ctrl_var.value</i>	= <i>intValue</i>
<i>ctrl_var.description</i>	= <i>"tooltip"</i>
<i>ctrl_var.pages</i>	= <i>page_list</i>
<i>ctrl_var.label</i>	= <i>"label"</i>
<i>ctrl_var.minValue</i>	= <i>min</i>
<i>ctrl_var.maxValue</i>	= <i>max</i>
<i>ctrl_var.singleStep</i>	= <i>step</i>
<i>ctrl_var.editable</i>	= <u>true</u>   false
<i>ctrl_var.visible</i>	= <u>true</u>   false
<i>ctrl_var.optional</i>	= true   <u>false</u>
<i>ctrl_var.specified</i>	= <u>true</u>   false
<i>ctrl_var.tvf</i>	= true   <u>false</u>
<i>ctrl_var.hyperlink</i>	= <i>link_path</i>
<i>ctrl_var.insertToControlFile</i>	= <u>true</u>   false

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- **“*name*”**

A required argument specifying the name of the variable for the **Variable** statement written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- ***intValue***

An optional argument specifying the assigned integer value for the Variable statement. If “***intValue***” is not included in the function call it can be defined with the value property.

- **“*tooltip*”**

An optional argument specifying the hover text (tooltip) for the control. If “***tooltip***” is not included in the function call it can be defined with the description property.

- Object Properties

The value (*intValue*) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, value, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addIntegerVariable are the following:

**Table 10-24. Object Properties for addIntegerVariable**

Property	Value (type)	Description
value	<i>intValue</i> (integer)	The assigned value for the Variable statement.
minValue	<i>min</i> (integer)	The minimum value allowed for the value property.  An error notification (⚠) is displayed next to the control if the condition is not satisfied.
maxValue	<i>max</i> (integer)	The maximum value allowed for the value property.  An error notification (⚠) is displayed next to the control if the condition is not satisfied.
singleStep	<i>step</i> (integer)	Specify a <i>step</i> value in order to use a spin control in the GUI to set the value property. The properties minValue and maxValue must also be specified.  When the spin control up or down arrow is pressed, the current value is incremented or decremented by <i>step</i> . It is still possible to enter a value directly into the entry field of a spin control, as long as <i>min</i> <= <i>intValue</i> <= <i>max</i> .
optional	true   false (Boolean)	Set to true to add a checkbox for the control. The default is false, which creates a required control.
specified	true   false (Boolean)	Sets the state of the checkbox when optional = true.

**Table 10-24. Object Properties for addIntegerVariable (cont.)**

Property	Value (type)	Description
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the Variable statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addIntegerVariable adds a variable control with an integer value to the GUI, which causes a [Variable](#) statement to be written to the Calibre Interactive control file. The control includes a text entry box.

The properties minValue and maxValue specify minimum and maximum allowed values for *intValue* and may be specified with or without the singleStep property. Without singleStep, an error notification () is displayed next to the control if the condition is not satisfied. A spin control is included if singleStep is defined. The minValue and maxValue properties are required if singleStep is specified, and place limits on the spin control.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
VARIABLE name intValue
```

where *intValue* is replaced with the actual value.

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" intValue
tvf::set_global_variable "name" intValue
```

## Related Topics

[Configuration File Format for Calibre Interactive GUI Customization](#)

[addTextVariable](#)

## addMultiChoicesVariable

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a Variable statement with a value to the Calibre Interactive GUI. The value is taken from an enumerated list and multiple values can be selected. The statement is added to the control file.

### Usage

#### Function Syntax

```
ctrl_var = tool.addMultiChoicesVariable("name", value_list[, label_list[, tooltip_list]])
```

#### Object Properties

```
ctrl_var.pages      = page_list
ctrl_var.label      = "control_label"
ctrl_var.description = "control_tooltip"
ctrl_var.value      = value
ctrl_var.editable    = true | false
ctrl_var.visible     = true | false
ctrl_var.optional    = true | false
ctrl_var.specified   = true | false
ctrl_var.valueQuoted = true | false
ctrl_var.tvf         = true | false
ctrl_var.hyperlink   = link_path
ctrl_var.insertToControlFile = true | false
ctrl_var.choiceAt(i).label = "choice_label"
ctrl_var.choiceAt(i).value = "choice_value"
ctrl_var.choiceAt(i).description = "choice_tooltip"
ctrl_var.choiceAt(i).compatible = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- ***name***

A required argument specifying the name of the variable for the Variable statement written to the Calibre Interactive control file.

- ***value\_list***

A required argument specifying a list of possible values for the Variable statement. For example: [1, 2, 3]

Values are quoted by default in the statement written to the control file—set the valueQuoted property to false for integer or real values.

- ***label\_list***

An optional argument specifying a list of labels for the control's dropdown list.

For example: [“one”, “two”, “three”]

If *label\_list* is not included in the function call, the labels can be defined with the choiceAt(*i*).label property.

- *tooltip\_list*

An optional argument specifying a list of tooltips (hover text) for the control’s dropdown list.

For example: [“first choice”, “second choice”, “third choice”]

If *tooltip\_list* is not included in the function call, the tooltips can be defined with the choiceAt(*i*).description property.

- Object Properties

The label (*label\_list*) and description (*tooltip\_list*) properties for the choices list can be defined in the function call or with a separate property definition using choiceAt(*i*).

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addMultiChoicesVariable are the following:

**Table 10-25. Object Properties for addMultiChoicesVariable**

Property	Allowed Values	Description
value	" <i>value</i> " (string or number)	The assigned value for the Variable statement. Set this property to one of the choice values to provide an initial value for the control.
optional	true   <u>false</u> (boolean)	Places a checkbox next to the control. The default is false, which creates a required control.
specified	<u>true</u>   false (boolean)	Sets the state of the checkbox when optional=true.
valueQuoted	<u>true</u>   false (boolean)	The assigned value is enclosed in quotes by default. Set valueQuoted to false to not include the quotes, as for an integer or real value.

**Table 10-25. Object Properties for addMultiChoicesVariable (cont.)**

Property	Allowed Values	Description
tvf	true   false (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the Variable statement to the control file. The default is true; modify this property with caution.
choiceAt( <i>i</i> ).label	<i>choice_label</i> (string)	The label for the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control. Numbering starts at zero.
choiceAt( <i>i</i> ).value	<i>choice_value</i> (string, integer, or real)	The assigned value for the <i>i</i> <sup>th</sup> choice.
choiceAt( <i>i</i> ).description	<i>choice_tooltip</i> (string)	The tooltip for the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control.
choiceAt( <i>i</i> ).compatible	<u>true</u>   false (Boolean)	Specifies whether the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control is available (the default) or unavailable (dimmed).

## Description

The function call tool.addMultiChoicesVariable adds a variable control to the GUI. The control includes a dialog box in which one or more items from the *value\_list* can be selected. The control causes a SVRF **Variable** statement to be written to the Calibre Interactive control file.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true) and two choice values are selected (value1 and “value2”):

```
VARIABLE "name" "value1" "value2"
```

The values are quoted by default—set valueQuoted to false to not use quotes.

When combined with appropriate logic, the choiceAt(*i*).compatible property can be used to set a choice as available or unavailable depending on other settings in the GUI.

For a TVF rule file with the tvf property set to true, the following statements are written to the control file when two choice values are selected:

```
set "name" {value1 value2}
tvf::set_global_variable "name" {value1 value2}
```

The property valueQuoted is not used when the tvf property is set to true.

## Examples

This example adds an optional control to include a Variable statement to set VAR\_NETS, where there are three possible choices for the assigned value. The label property is not defined so the default label of “Variable VAR\_NETS” is used for the control. The code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized.

```
var cvm = tool.addMultiChoicesVariable("VAR_NETS",
                                         ["a", "b", "rs1"],
                                         ["a", "b", "rs1"],
                                         ["net a", "net b", "net rs1"])
cvm.pages      = ["Defines and Variables"]
cvm.description = "Defines SVRF Variable VAR_NETS"
cvm.optional    = true;
cvm.specified   = true;
// set initial value to one of the choices
cvm.value = cvm.choiceAt(1).value
```

When the control is clicked, a dialog box opens in which you can select one or more of the values. The following statement is added to the Calibre Interactive control file if a and b are selected:

```
VARIABLE "VAR_NETS" "a" "b"
```

## Related Topics

[Configuration File Format for Calibre Interactive GUI Customization](#)

[addChoicesVariable](#)

## addOutputDirTextVariable

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to a directory used for output. The Variable statement is added to the control file. The control includes a text entry field and a browser button for selecting the directory.

---

### Note

 Variables defined as a directories may be used in the [tvf::svrf\\_var](#) command, however, such variables cannot be used to specify directories for SVRF operations and statements. See the [Variable](#) statement.

---

## Usage

### Function Syntax

```
ctrl_var = tool.addOutputDirTextVariable("name"[, "dirName"[, "tooltip"]])
```

### Object Properties

```
ctrl_var.value      = "dirName"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label      = "label"  
ctrl_var.promptText = "prompt_string"  
ctrl_var.valueQuoted = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional    = true | false  
ctrl_var.specified   = true | false  
ctrl_var.tvf         = true | false  
ctrl_var.hyperlink   = link_path  
ctrl_var.insertToFile = true | false
```

## Arguments

- **ctrl\_var**

A required argument specifying the variable name for the control.

- **name**

A required argument specifying the name of the variable for the Variable statement written to the Calibre Interactive control file. The **name** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the **name**, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- “*dirName*”

An optional argument specifying the assigned text string for the Variable statement. If “*dirName*” is not included in the function call it can be defined with the value property.

- “*tooltip*”

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (“*dirName*”) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addOutputDirTextVariable are the following:

**Table 10-26. Object Properties for addOutputDirTextVariable**

Property	Allowed Values	Description
value	“ <i>dirName</i> ” (string)	The assigned value for the Variable statement.
valueQuoted	<u>true</u>   false (boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (boolean)	Allow spaces in strings; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the value property.
optional	true   <u>false</u> (boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (boolean)	Sets the state of the checkbox when optional=true.

**Table 10-26. Object Properties for addOutputDirTextVariable (cont.)**

Property	Allowed Values	Description
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the Variable statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addOutputDirTextVariable adds a variable control with a string value to the GUI, which causes a [Variable](#) statement to be written to the Calibre Interactive control file. The string value *dirName* is expected to correspond to a directory used for output. The control includes a text entry box and a browser button. An error  is displayed next to the control if the *dirName* is not writable.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
VARIABLE name "dirName"
```

The value is quoted by default; set valueQuoted to false to not include quotes. Quotes are generally recommended.

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" {dirName}
tvf::set_global_variable "name" {dirName}
```

The property valueQuoted is not used when the tvf property is set to true.

## Examples

See the example with [addInputFileTextVariable](#), which has similar syntax.

## addOutputFileTextVariable

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI, where the string value is expected to correspond to an output file. The Variable statement is added to the control file. The control includes a text entry field, a file browser button for selecting the file, and a view file button.

### Note

 Variables defined as filenames may be used in the `tvf::svrf_var` command, however, such variables cannot be used to specify filenames for SVRF operations and statements. See the [Variable](#) statement.

---

## Usage

### Function Syntax

```
ctrl_var = tool.addOutputFileTextVariable("name"[, "fileName"[, "tooltip"]])
```

### Object Properties

```
ctrl_var.value      = "fileName"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label       = "label"  
ctrl_var.promptText = "prompt_string"  
ctrl_var.valueQuoted = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional    = true | false  
ctrl_var.specified   = true | false  
ctrl_var.tvf         = true | false  
ctrl_var.hyperlink   = link_path  
ctrl_var.insertToFile = true | false
```

## Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- ***name***

A required argument specifying the name of the variable for the Variable statement written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- “*fileName*”

An optional argument specifying the assigned text string for the Variable statement. If “*fileName*” is not included in the function call it can be defined with the value property.

- “*tooltip*”

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (“*fileName*”) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addOutputFileTextVariable are the following:

**Table 10-27. Object Properties for addOutputFileTextVariable**

Property	Allowed Values	Description
value	“ <i>fileName</i> ” (string)	The assigned value for the Variable statement.
valueQuoted	<u>true</u>   false (boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (boolean)	Allow spaces in strings; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the value property.
optional	<u>true</u>   false (boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (boolean)	Sets the state of the checkbox when optional=true.

**Table 10-27. Object Properties for addOutputFileTextVariable (cont.)**

Property	Allowed Values	Description
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the Variable statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addOutputFileTextVariable adds a variable control with a string value to the GUI, which causes a [Variable](#) statement to be written to the Calibre Interactive control file. The string value *fileName* is expected to correspond to an output file. The control includes a text entry box, a file browser button, and a view file button. An error  is displayed next to the control if the *fileName* is not writable.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
VARIABLE name "fileName"
```

The value is quoted by default; set valueQuoted to false to not include quotes. Quotes are generally recommended.

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" {fileName}
tvf::set_global_variable "name" {fileName}
```

The property valueQuoted is not used when the tvf property is set to true.

## Examples

See the example with [addInputFileTextVariable](#), which has similar syntax.

## addRealVariable

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a Variable statement with a floating-point value to the Calibre Interactive GUI. The statement is added to the control file.

### Usage

#### Function Syntax

```
ctrl_var = tool.addRealVariable("name"[, realValue[, "tooltip"]])
```

#### Object Properties

```
ctrl_var.value      = realValue
ctrl_var.description = "tooltip"
ctrl_var.pages      = page_list
ctrl_var.label       = "label"
ctrl_var.minValue    = min
ctrl_var.maxValue    = max
ctrl_var.singleStep  = step
ctrl_var.includeMinEndpoint = true | false
ctrl_var.includeMaxEndpoint = true | false
ctrl_var.editable     = true | false
ctrl_var.visible      = true | false
ctrl_var.optional      = true | false
ctrl_var.specified    = true | false
ctrl_var.tvf          = true | false
ctrl_var.hyperlink    = link_path
ctrl_var.insertToControlFile = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- **“*name*”**

A required argument specifying the name of the variable for the [Variable](#) statement that is written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- ***realValue***

An optional argument specifying the floating-point assigned value for the Variable statement. If “***realValue***” is not included in the function call it can be defined with the value property.

- “*tooltip*”

An optional argument specifying the hover text (tooltip) for the control. If “*tooltip*” is not included in the function call it can be defined with the description property.

- Object Properties

The value (*realValue*) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, value, editable, visible, and specified.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addRealDefine are the following:

**Table 10-28. Object Properties for addRealVariable**

Property	Value (type)	Description
value	<i>realValue</i> (floating-point)	The assigned value for the Variable statement.
minValue	<i>min</i> (floating-point)	The minimum value allowed for the value property.  An error notification (⚠) is displayed next to the control if the condition is not satisfied.
maxValue	<i>max</i> (floating-point)	The maximum value allowed for the value property.  An error notification (⚠) is displayed next to the control if the condition is not satisfied.
singleStep	<i>step</i> (floating-point)	Specify a <i>step</i> value in order to use a spin control in the GUI to set the value property. The properties minValue and maxValue must also be specified.  When the spin control up or down arrow is pressed, the current value is incremented or decremented by <i>step</i> . It is still possible to enter a value directly into the entry field of a spin control, as long as <i>realValue</i> satisfies the conditions dictated by minValue, maxValue, includeMinEndpoint, and includeMaxEndpoint.
includeMinEndpoint	<u>true</u>   false (Boolean)	Specifies whether to include minValue in the range of allowed values.

**Table 10-28. Object Properties for addRealVariable (cont.)**

Property	Value (type)	Description
includeMaxEndpoint	<u>true</u>   false (Boolean)	Specifies whether to include maxValue in the range of allowed values.
optional	true   <u>false</u> (Boolean)	Set to true to add a checkbox for the control. The default is false, which creates a required control.
specified	<u>true</u>   false (Boolean)	Sets the state of the checkbox when optional = true.
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the Variable statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addRealVariable adds a variable control with a floating-point value to the GUI, which causes a **Variable** statement to be written to the Calibre Interactive control file. The control includes a text entry box.

The properties minValue, maxValue, includeMinEndpoint, and includeMaxEndpoint specify minimum and maximum allowed values for *realValue* and may be specified with or without the singleStep property. Without singleStep, an error notification (⚠) is displayed next to the control if the condition is not satisfied. A spin control is included if singleStep is defined. The minValue and maxValue properties are required if singleStep is specified, and place limits on the spin control.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
VARIABLE name realValue
```

where *realValue* is replaced with the actual value.

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" realValue
tvf::set_global_variable "name" realValue
```

## Related Topics

[Configuration File Format for Calibre Interactive GUI Customization](#)

[addTextVariable](#)

## addTextBoxVariable

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI. The GUI control is a multi-line text box. The statement is added to the control file.

### Usage

#### Function Syntax

```
ctrl_var = tool.addTextBoxVariable("name"[, "textValue"[, "tooltip"]])
```

#### Object Properties

```
ctrl_var.value      = "textValue"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label       = "label"  
ctrl_var.promptText = "prompt_string"  
ctrl_var.valueQuoted = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.emptyIsValid = true | false  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional    = true | false  
ctrl_var.specified   = true | false  
ctrl_var.tvf          = true | false  
ctrl_var.hyperlink   = link_path  
ctrl_var.insertToFile = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- ***name***

A required argument specifying the name of the variable for the **Variable** statement written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- “***textValue***”

An optional argument specifying the assigned text string for the Variable statement. If “***textValue***” is not included in the function call it can be defined with the value property.

- “***tooltip***”

An optional argument specifying the hover text (tooltip) for the control. If “***tooltip***” is not included in the function call it can be defined with the description property.

- Object Properties

The value (“*textValue*”) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, and visible.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addTextVariable are the following:

**Table 10-29. Object Properties for addTextVariable**

Property	Allowed Values	Description
value	“ <i>textValue</i> ” (string)	The assigned value for the Variable statement.
valueQuoted	<u>true</u>   false (Boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (Boolean)	Allow spaces in strings; the default is false.
emptyIsValid	true   <u>false</u> (Boolean)	Allow an empty <i>textValue</i> string; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the value property.
optional	true   <u>false</u> (Boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (Boolean)	Sets the state of the checkbox when optional=true.
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.

**Table 10-29. Object Properties for addTextVariable (cont.)**

Property	Allowed Values	Description
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the Variable statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addTextBoxVariable adds a variable control with a string value to the GUI, which causes a SVRF Variable statement to be written to the Calibre Interactive control file. The control includes a multi-line text entry box, in contrast to addTextVariable, which has a single line text entry box.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
VARIABLE name "textValue"
```

The value is quoted by default; set valueQuoted to false to not include quotes.

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" {textValue}
tvf::set_global_variable "name" {textValue}
```

The property valueQuoted is not used when the tvf property is set to true.

## Examples

The following code adds an optional control to define a SVRF Variable with a string value. The code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized.

```
var v = tool.addTextBoxVariable("VAR_METAL_TEXT",
    "net1 net2 netA netB net4 net5 netC net10 net12 net13 net15",
    "specify net labels")
v.pages = ["Defines and Variables"]
v.optional = true;
v.allowSpaces = true
```

The following statement is added to the control file for Calibre Interactive:

```
VARIABLE "VAR_METAL_TEXT" "net1 net2 netA netB net4 net5 netC net10 net12
net13 net15"
```

The control looks like this:



## addTextVariable

Function call for the custom configuration file for Calibre Interactive

Adds a control to specify a Variable statement with a string value to the Calibre Interactive GUI. The statement is added to the control file.

### Usage

#### Function Syntax

```
ctrl_var = tool.addTextVariable("name"[, "textValue"[, "tooltip"]])
```

#### Object Properties

```
ctrl_var.value      = "textValue"  
ctrl_var.description = "tooltip"  
ctrl_var.pages      = page_list  
ctrl_var.label       = "label"  
ctrl_var.promptText = "prompt_string"  
ctrl_var.valueQuoted = true | false  
ctrl_var.emptyIsValid = true | false  
ctrl_var.allowSpaces = true | false  
ctrl_var.editable    = true | false  
ctrl_var.visible     = true | false  
ctrl_var.optional    = true | false  
ctrl_var.specified   = true | false  
ctrl_var.tvf          = true | false  
ctrl_var.hyperlink   = link_path  
ctrl_var.insertToFile = true | false
```

### Arguments

- ***ctrl\_var***

A required argument specifying the variable name for the control.

- “***name***”

A required argument specifying the name of the variable for the **Variable** statement written to the Calibre Interactive control file. The ***name*** can include alphanumeric and underscore (\_) characters.

Some special characters can be used in the ***name***, but require special handling. See “[Allowed Characters in Variable and Define Names](#)” on page 238.

- “***textValue***”

An optional argument specifying the assigned text string for the Variable statement. If “***textValue***” is not included in the function call it can be defined with the value property.

- “***tooltip***”

An optional argument specifying the hover text (tooltip) for the control. If “***tooltip***” is not included in the function call it can be defined with the description property.

- Object Properties

The value (“*textValue*”) and description (“*tooltip*”) properties can be defined in the function call or with a separate property definition.

See “[Option Properties](#)” on page 250 for a description of these common properties: pages, label, description, editable, and visible.

See “[Hyperlink Property to Add a Link to Custom Controls in Calibre Interactive](#)” on page 252 for a description of the hyperlink property, which makes the label of the control into a link to the specified *link\_path*.

The properties specific to addTextVariable are the following:

**Table 10-30. Object Properties for addTextVariable**

Property	Allowed Values	Description
value	“ <i>textValue</i> ” (string)	The assigned value for the Variable statement.
valueQuoted	<u>true</u>   false (Boolean)	Text strings are enclosed in quotes by default. Set valueQuoted to false to not include the quotes.
allowSpaces	true   <u>false</u> (Boolean)	Allow spaces in strings; the default is false.
emptyIsValid	true   <u>false</u> (boolean)	Allow an empty <i>textValue</i> string; the default is false.
promptText	“ <i>prompt_string</i> ” (string)	The prompt string for the text entry box. The prompt is used only if an initial value is not set with the “value” property.
optional	true   <u>false</u> (Boolean)	Places a checkbox next to the control. The default is false, resulting in a required control.
specified	<u>true</u>   false (Boolean)	Sets the state of the checkbox when optional=true.
tvf	true   <u>false</u> (Boolean)	Adds a Tcl variable to the Calibre Interactive control file when true is specified and a TVF rule file is used. The default is false if this property is not defined. If true is specified when not using a TVF rule file, the configuration command does not output any statement to the control file.

**Table 10-30. Object Properties for addTextVariable (cont.)**

Property	Allowed Values	Description
insertToControlFile	<u>true</u>   false (Boolean)	Specifies whether to write the Variable statement to the control file. The default is true; modify this property with caution.

## Description

The function call tool.addTextVariable adds a variable control with a string value to the GUI, which causes an SVRF [Variable](#) statement to be written to the Calibre Interactive control file. The control includes a text entry box.

The following statement is written to the control file if the control is required (optional = false) or if the optional checkbox is checked (optional = true and specified = true):

```
VARIABLE name "textValue"
```

The value is quoted by default; set valueQuoted to false to not include quotes.

For a TVF rule file with the tvf property set to true, the following statements are written to the control file:

```
set "name" {textValue}
tvf::set_global_variable "name" {textValue}
```

The property valueQuoted is not used when the tvf property is set to true.

## Examples

The following code adds an optional control to define a Variable statement with a string value. The code is placed within the onInitialized section of the configuration file or within a user-defined function called from onInitialized. Only the variable name is specified in the function call—the value and tooltip are specified as property definitions.

```
var v = tool.addTextVariable("VAR_METAL_TEXT")
v.pages      = ["Defines and Variables"]
v.label      = "Set VAR_METAL_TEXT"
v.description = "SVRF Variable VAR_METAL_TEXT for metal text";
v.value      = "net1";
v.optional    = true;
```

The following statement is added to the control file for Calibre Interactive:

```
VARIABLE VAR_METAL_TEXT "net1"
```

## Related Topics

[Configuration File Format for Calibre Interactive GUI Customization](#)

## Configuration File Commands to Add Buttons

You can add buttons to the Calibre Interactive GUI and specify a callback function that is executed when the button is clicked.

Command	Description
<a href="#">addAction</a>	Adds a user-defined button with a callback function to the Calibre Interactive GUI.
<a href="#">addChoicesAction</a>	Adds a user-defined button with a callback function to the Calibre Interactive GUI. The button has a dropdown list of choices.

## addAction

Function call for the custom configuration file for Calibre Interactive  
Adds a user-defined button with a callback function to the Calibre Interactive GUI.

### Usage

#### Function Syntax

*ctrl\_var* = tool.addAction("name")

#### Object Properties

*ctrl\_var.pages* = ["page\_name" ...]  
*ctrl\_var.label* = "label"  
*ctrl\_var.description* = "tooltip"  
*ctrl\_var.activated.connect(cmd)*  
*ctrl\_var.location* = Location.Footer | Location.Header | Location.Body  
| Location.HeaderAndFooter

#### Control Functions

*ctrl\_var.setEnabled*  
*ctrl\_var.setDisabled*

### Arguments

- *ctrl\_var*  
A required argument specifying the variable name for the control.
- “*name*”  
A required argument specifying the string used internally to identify the button. The *name* must be unique within the configuration file. The “*name*” is used internally.
- Object Properties

**Table 10-31. Object Properties for addAction**

Property	Allowed Values	Description
pages	list of strings	A list of the pages that the control appears on. For example: [ "Outputs", "Options" ]
label	string	The button label.
description	string	The tooltip.

**Table 10-31. Object Properties for addAction (cont.)**

Property	Allowed Values	Description
activated.connect( <i>cmd</i> )	function name	The function that is executed when the button is clicked.  The <i>cmd</i> must be a function defined within the Configuration section of the configuration file. It cannot have any arguments.
location	<u>Location.Footer</u>   Location.Header   Location.Body   Location.HeaderAndFooter	The location of the button; Location.Footer is the default. The header and footer locations are left justified. The body location is centered under the last control on the page.

- Control Functions

Control functions are built-in functions for a control.

**Table 10-32. Control Functions for addAction**

Function	Description
setEnabled	Enable the button. Buttons created by addAction are enabled by default.
setDisabled	Disable the button. A disabled button is grayed out and not active.

## Description

The addAction function adds a user-defined button to the Calibre Interactive GUI. When the button is clicked, the function specified by the activated.connect property is executed. The activated.connect property can specify a function defined in the configuration file or a user-defined command called with [launchCommand](#).

See the [connect](#) function to attach a function to a signal from the GUI. To add a callback function to the Run button, use the tool property tool.runCallback, as described in “[Example: Attach Callback Function to Run Button With tool.runCallback](#)” on page 249.

## Examples

The following example adds a button with the internal name “SVDBon” and label “SVDB turn on”. The function setSVDBOn is called when the button is clicked.

```
Configuration {
    onInitialized: {
        // add button to turn on svdb options
        var bSVDB = tool.addAction("SVDBOn");      //name is internal
        bSVDB.pages = ["Rules"];
        bSVDB.label = "SVDB turn on"; // the button label
        bSVDB.description = "turn on SVDB and all the options"; // tooltip
        bSVDB.activated.connect(setSVDBOn);
        bSVDB.location = Location.Body           // default location is Footer
    } // end onInitialized

    // define function to set SVDB options if SVDBOn button is clicked
    function setSVDBOn() {
        tool.svdb.query.value = true
        tool.svdb.asciiixref.value = true
        tool.svdb.cci.value = true
        tool.svdb.genParasiticData.value = true
        tool.svdb.pinloc.value = "PINLOC"
        tool.svdb.pinloc.specified = true
    }
} // end Configuration
```

## addChoicesAction

Function call for the custom configuration file for Calibre Interactive

Adds a user-defined button with a callback function to the Calibre Interactive GUI. The button has a dropdown list of choices.

### Usage

#### Function Syntax

```
ctrl_var = tool.addChoicesAction("name", value_list[, label_list[, tooltip_list]])
```

#### Object Properties

```
ctrl_var.pages      = page_list
ctrl_var.label     = "control_label"
ctrl_var.description = "control_tooltip"
ctrl_var.value      = "value"
ctrl_var.valueChanged.connect(cmd)
ctrl_var.location   = Location.Footer | Location.Header | Location.Body
                     | Location.HeaderAndFooter
ctrl_var.choiceAt(i).label    = "list_label"
ctrl_var.choiceAt(i).value    = "list_value"
ctrl_var.choiceAt(i).description = "list_tooltip"
ctrl_var.choiceAt(i).compatible = true | false
```

#### Control Functions

```
ctrl_var.setEnabled
ctrl_var.setDisabled
```

#### Arguments

- ***ctrl\_var***  
A required argument specifying the variable name for the control.
- **“*name*”**  
A required argument specifying the string used internally to identify the button. The ***name*** must be unique within the configuration file. The “***name***” is used internally.
- ***value\_list***  
A required argument specifying a list of possible assigned values for the control.  
For example: [“ONE”, “TWO”, “THREE”]
- ***label\_list***  
An optional argument specifying a list of labels for the control’s dropdown list.  
For example: [“one”, “two”, “three”]  
If *label\_list* is not included in the function call, the labels can be defined with the choiceAt(*i*).label property.

- *tooltip\_list*

An optional argument specifying a list of tooltips (hover text) for the control's dropdown list.

For example: [“first choice”, “second choice”, “third choice”]

If *tooltip\_list* is not included in the function call, the tooltips can be defined with the *choiceAt(i).description* property.

- Object Properties

The label (*label\_list*) and description (*tooltip\_list*) properties for the choices list can be defined in the function call or with a separate property definition using *choiceAt(i)*.

**Table 10-33. Object Properties for addChoicesAction**

Property	Allowed Values	Description
pages	list of strings	A list of the pages that the control appears on. For example: [ “Outputs”, “Options” ]
label	string	The control label. The control label is only used if location is set to Location.Body and is displayed to the left of the button.
description	string	The tooltip.
value		The value for the control. This value is passed to the callback function for the button. Set this property to one of the choice values to provide an initial value for the control.
valueChanged.connect( <i>cmd</i> )	function name	<i>cmd</i> specifies the function that is executed when a new choice is selected from the dropdown menu. The new control value ( <i>ctrl_var.value</i> ) is passed to the function.  The <i>cmd</i> must be a function defined within the Configuration section of the configuration file. It must take one argument.
location	<u>Location.Footer</u>   <u>Location.Header</u>   <u>Location.Body</u>   <u>Location.HeaderAndFooter</u>	The location of the button; Location.Footer is the default. The header and footer locations are left justified. The body location is centered under the last control on the page.

**Table 10-33. Object Properties for addChoicesAction (cont.)**

Property	Allowed Values	Description
choiceAt( <i>i</i> ).label	string	The label for the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control. Numbering starts at zero.
choiceAt( <i>i</i> ).value		The assigned value for the <i>i</i> <sup>th</sup> choice. When a new choice is selected, the choices[ <i>i</i> ].value is assigned to <i>ctrl_var.value</i> .
choiceAt( <i>i</i> ).description	string	The tooltip for the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control.
choiceAt( <i>i</i> ).compatible	<u>true</u>   false (Boolean)	Specifies whether the <i>i</i> <sup>th</sup> choice in the dropdown list of the GUI control is available (the default) or unavailable (dimmed).

- Control Functions

Control functions are built-in functions for a control.

**Table 10-34. Control Functions for addChoicesAction**

Function	Description
setEnabled	Enable the button. Buttons created by addAction are enabled by default.
setDisabled	Disable the button. A disabled button is grayed out and not active.

## Description

The addChoicesAction function adds a user-defined button to the Calibre Interactive GUI. The button has a dropdown list of choices. When the selection is changed, the following actions occur:

- The choices[*i*].value is assigned to *ctrl\_var.value*, where *i* is the selected choice.
- The function specified by the valueChanged.connect property is executed, with *ctrl\_var.value* as an argument.

When combined with appropriate logic, the choiceAt(*i*).compatible property can be used to set a choice as available or unavailable depending on other settings in the GUI.

See the [connect](#) function to attach a function to a signal from the GUI. To add a callback function to the Run button, use the tool property tool.runCallback, as described in “[Example: Attach Callback Function to Run Button With tool.runCallback](#)” on page 249.

## Examples

This example adds a button with a dropdown list of two choices.

```
Configuration {
    onInitialized: {

        var d = tool.addChoicesAction("setupAB",
            ["Case_A", "Case_B"],
            ["Setup A", "Setup B"],
            ["hcell file", "layout case, extract temp, noinstances"]);
        d.pages      = ["Rules"];
        d.label      = "Setup";
        d.description = "Choose setup A or B";

        d.value = d.choiceAt(0).value;
        d.valueChanged.connect(setupChoice);
        // call setupChoice with the value of the initial setting
        setupChoice(d.value);

    } // end onInitialized

    function setupChoice(arg) {

        switch (arg) {
            case "Case_A":
                tool.hCells.hCellsFile.value = "hcells"
                tool.hCells.hCellsFile.specified = true
                break;
            case "Case_B":
                tool.pexNetlist.noinstances.value = true
                tool.pexExtractTemp.temperatures.specified = true
                tool.layoutCase.caseSensitive.specified = true
                break;
            default:
                break;
        } // switch
    } // setupChoice
} // end Configuration
```

## Configuration File Commands for Trigger Functions

---

Several commands are available for the Calibre Interactive custom configuration file to specify pre- and post-execution trigger functions. The trigger functions can run before or after the batch Calibre run, and can run in the shell environment or the design tool environment.

See “[Configuration File Format for Calibre Interactive GUI Customization](#)” on page 240 for information on the file format and how to invoke Calibre Interactive with a custom configuration file. The trigger functions added with the configuration file are displayed in the [Triggers Page](#).

Command	Description
<a href="#">Trigger Parameters in Calibre Interactive</a>	Certain filenames and other parameters from the Calibre Interactive GUI can be passed to a trigger function using replaceable parameters. For example, the top cell name and the source netlist name can be passed to a trigger function using the replaceable trigger parameters %L and %s, respectively.
<a href="#">add ... Trigger</a>	A set of function calls to add trigger definitions to the Triggers page in the Calibre Interactive GUI. Triggers can execute in the shell environment or the design tool environment. They can execute before or after Calibre execution.

## Trigger Parameters in Calibre Interactive

Certain filenames and other parameters from the Calibre Interactive GUI can be passed to a trigger function using replaceable parameters. For example, the top cell name and the source netlist name can be passed to a trigger function using the replaceable trigger parameters %L and %s, respectively.

The parameters listed in the table are recognized for triggers and substituted with the appropriate runtime values.

**Table 10-35. Replaceable Trigger Parameters**

Runtime Replacement	Parameter	Calibre Interactive Applications
Circuit netlist	%c	LVS, PERC, and PEX
DRC RDB	%b	DRC
DRC summary file	%m	DRC
Hcell file	%h	LVS, PERC, and PEX

**Table 10-35. Replaceable Trigger Parameters (cont.)**

<b>Runtime Replacement</b>	<b>Parameter</b>	<b>Calibre Interactive Applications</b>
Layout format (GDSII, OASIS)	%F	All
Layout2 format (GDSII, OASIS)	%F2	DRC FastXOR
Layout path	%l	All
Layout2 path	%l 2	DRC FastXOR
Layout primary cell <sup>1</sup>	%L	All
Layout2 primary cell <sup>1</sup>	%L2	DRC FastXOR
Layout library	%y	All applications when started from Cadence Virtuoso
Layout2 library	%y2	DRC FastXOR when started from Cadence Virtuoso
Layout view	%v	All
Layout2 view	%v2	DRC FastXOR
LVS Report file	%m	LVS
LVS Report file	%n	PEX
PERC Report file	%m	PERC
PEX netlist	%p	PEX
PEX netlist format	%f	PEX
PEX Report file	%m	PEX
Rule file	%r	All
Run directory	%d	All
Runset filename	%R	All
Source path for primary netlist source <sup>2</sup>	%s	LVS, PERC, and PEX
Source path for all netlist sources <sup>3</sup>	%N	LVS, PERC, and PEX
Source primary cell	%S	LVS, PERC, and PEX
Termination status: 0 — Normal 1 — Aborted 2 and above — the Calibre exit code	%e	All

<sup>1</sup> Wildcards in the layout primary cell name are only expanded for a post-execution trigger run on the local host. In addition, the layout format must be GDSII, and the Calibre Interactive application must be DRC or PERC.

<sup>2</sup> For SPICE netlist input, %s returns the netlist(s) listed on the **Inputs** page. In the case of VERILOG or MIXED source input, %s returns the SPICE file that is translated from the Verilog source input.

<sup>3</sup> In addition to the source paths returned by %s, the %N trigger parameter also returns the additional SPICE netlist files specified on the **Database** page in the Library area. In the case of MIXED source input, %N also includes the SPICE netlist files listed on the **Inputs** page

## add ... Trigger

Function call for the custom configuration file for Calibre Interactive

A set of function calls to add trigger definitions to the Triggers page in the Calibre Interactive GUI. Triggers can execute in the shell environment or the design tool environment. They can execute before or after Calibre execution.

### Usage

#### Function Syntax

```
trig_var = tool.addProcessPreTrigger("run_mode", "name")  
trig_var = tool.addLayoutPreTrigger("run_mode", "name")  
trig_var = tool.addSchematicPreTrigger("run_mode", "name")  
trig_var = tool.addProcessPostTrigger("run_mode", "name")  
trig_var = tool.addLayoutPostTrigger("run_mode", "name")  
trig_var = tool.addSchematicPostTrigger("run_mode", "name")
```

#### Object Properties

```
trig_var.command = "trig_command_and_args"  
trig_var.specified = true | false
```

### Arguments

#### Function Name

- **addProcessPreTrigger**  
Adds a pre-execution trigger function that runs in the shell environment.
- **addLayoutPreTrigger**  
Adds a pre-execution trigger function that runs in the layout design tool environment.
- **addSchematicPreTrigger**  
Adds a pre-execution trigger function that runs in the schematic design tool environment.
- **addProcessPostTrigger**  
Adds a post-execution trigger function that runs in the shell environment.
- **addLayoutPostTrigger**  
Adds a post-execution trigger function that runs in the layout design tool environment.
- **addSchematicPostTrigger**  
Adds a post-execution trigger function that runs in the schematic design tool environment.

### Arguments

- ***trig\_var***

A required argument specifying the variable name for the control, which must be unique within the configuration file. The variable name is used to set the object properties.

- **“*run\_mode*”**

A required argument specifying the run mode for trigger function. The quotes are required. The choice of pre- or post-execution and execution environment is determined by the function call.

The available run modes depend on the application. In particular, the parasitic extraction applications have more than one execution step, and triggers can be defined relative to each step.

**Table 10-36. Run Modes for Trigger Functions**

run_mode	Definition	Supported Applications
“runDrc”	Execute the trigger relative to the Calibre nmDRC step.	DRC
“runLvs”	Execute the trigger relative to the Calibre nmLVS step.	LVS, PEX, xACT
“runPerc”	Execute the trigger relative to the Calibre PERC step.	PERC
“runXact”	Execute the trigger relative to the Calibre xACT step.	XACT
“runXrc”	Execute the trigger relative to the Calibre xRC parasitic extraction step.	PEX
“runPreExec”	Execute the trigger prior to any Calibre execution.	PEX
“runFmt”	Execute the trigger relative to the formatting step in Calibre xRC.	PEX
“runPostExec”	Execute the trigger after all Calibre execution.	PEX

See “[Internal Trigger Execution in Calibre Interactive](#)” on page 415 for an explanation of the timing of internal triggers in Calibre Interactive PEX and xACT.

- **“*name*”**

A required string used to identify the trigger, which must be unique within the configuration file and cannot include spaces. The “***name***” is used internally.

## Object Properties

Property	Allowed Values	Description
command	“ <i>trig_command_and_args</i> ” (string)	The command and arguments (if any) to be executed for the trigger function; must be enclosed in quotes.  Replaceable parameters may be used in the command arguments; see “ <a href="#">Trigger Parameters in Calibre Interactive</a> ” on page 390.  Special characters must be escaped using standard JavaScript methods.
specified	<u>true</u>   false (Boolean)	Specifies whether to execute the trigger function; the default is true. This property corresponds to the “Use” column in the triggers table on the Triggers page.

## Description

The add...Trigger function calls are placed in the custom configuration file and add trigger definitions to the [Triggers Page](#); such trigger definitions cannot be edited in the GUI.

The choice of function call determines whether the specified trigger function executes before or after batch Calibre (add\*PreTrigger and add\*PostTrigger); the “*run\_mode*” argument specifies whether the trigger function executes relative to the LVS or parasitic extraction step. The choice of function call also determines the execution environment for the trigger: shell (addProcess\*Trigger), layout design tool (addLayout\*Trigger), or schematic design tool (addSchematic\*Trigger).

The trigger function to be executed is defined with the command object property.

## Examples

### Process Trigger Example

This example adds a pre-DRC trigger and a post-DRC. The triggers use the Linux echo command to print information. The post-DRC trigger uses the trigger parameter %n, which is replaced with the name of the LVS report. Quotes around the argument to the echo function are escaped.

## GUI Customization for Calibre Interactive Configuration File Commands for Trigger Functions

---

```
Configuration {  
  
onInitialized: {  
    // add DRC pre-trigger  
    var trigDRCpre_1 = tool.addProcessPreTrigger("runDrc", "DRCpreTrig_1")  
    trigDRCpre_1.command = "echo starting DRC";  
  
    // add DRC post-trigger  
    var trigDRCpost_2 = tool.addProcessPostTrigger("runDrc",  
        "DRCpostTrig_2");  
    trigDRCpost_2.command = "echo \"DRC summary report %m, DRC RDB %b\"";  
    // onInitialized  
}  
//Configuration  
}
```

# Guide and Examples for Converting Tcl Customization Files to New Configuration Files

---

This section provides a conversion guide and examples to assist in converting existing Tcl-based customization files to the new configuration file format for Calibre Interactive.

Follow this general process to adapt a Tcl customization file to the new configuration file format.

1. Review “[Configuration File Format for Calibre Interactive GUI Customization](#)” on page 240 and the other topics within “[Guide and Examples for Converting Tcl Customization Files to New Configuration Files](#)” on page 397.
2. Use “[Correspondence of Tcl Customization Commands to New Configuration Commands](#)” on page 397 to determine the new configuration commands and properties that correspond to the Tcl customization commands and options in the existing Tcl customization file.

The adaptation examples demonstrate how to convert several Tcl customization files.

3. If you are manipulating GUI controls, see “[Finding Option Names to Use in the Configuration File](#)” on page 254.
4. Write the configuration file.

Use tool.transcript to add debug messages to the configuration file.

5. Load the configuration file as described in “[Loading a Configuration File in Calibre Interactive](#)” on page 230.

View the Transcript page for errors and messages printed by the tool.transcript function.

<b>Correspondence of Tcl Customization Commands to New Configuration Commands</b>	<b>397</b>
<b>Adaptation Example: #DEFINE and Variable Statements</b> .....	<b>401</b>
<b>Adaptation Example: Move and Set GUI Options</b> .....	<b>403</b>
<b>Adaptation Example: Callback Function</b> .....	<b>407</b>

## Correspondence of Tcl Customization Commands to New Configuration Commands

This section provides a table listing Tcl customization commands and the corresponding command for the new configuration file format. Some typical Tcl scripting commands and the corresponding command in the new configuration file are also given.

See “[Tool Properties](#)” on page 247 and “[Option Properties](#)” on page 250 for definitions of the tool and option properties referred to in the following table.

**Table 10-37. Correspondence for Customization and Configuration File Commands**

Tcl Customization File	New Configuration File
CUSTOM::DEFINE CUSTOM::VARIABLE	See “ <a href="#">Configuration File Commands to Add #DEFINE and #UNDEFINE Statements</a> ” on page 303 and “ <a href="#">Configuration File Functions to Add Variable Statements</a> ” on page 342.  There are different commands depending on whether the #DEFINE or Variable statement has an assigned value and the value type.  See “ <a href="#">Adaptation Example: #DEFINE and Variable Statements</a> ” on page 401.
CUSTOM::LABEL	tool.addLabel and tool.addHeader
CUSTOM::SEPARATOR	tool.addSeparator
CUSTOM::CHECK	Not available. Use check selection recipes to select the rule checks to execute.
-choices and -multichoice options	Use <a href="#">addChoicesDefine</a> , <a href="#">addChoicesVariable</a> , or <a href="#">addMultiChoicesVariable</a> .
-prompt option	“label” property
-tooltip option	“description” property
-boolean option	“optional” property. The “optional” property is not available for an addDefine control, which always has a checkbox.
-select option	“specified” property
-enable option	“editable” property
-display option	“visible” property
-multiline	Use <a href="#">addTextBoxDefine</a> and <a href="#">addTextBoxVariable</a> .
-tool option	Test for the tool.name option in order to control command flow based on the application.  <code>if (tool.name == "drc") {     //commands }</code>
-tvf option	“tvf” property

**Table 10-37. Correspondence for Customization and Configuration File Commands (cont.)**

Tcl Customization File	New Configuration File
-vartype option	The object property valueQuoted controls whether a string is quoted. The functions addOutputDir* and addOutputFile* correspond to the option -file_browser.
-vtype option	Use the appropriate version of the add*Define or add*Variable command for the desired type of value. The object property allowSpaces controls whether or not spaces are allowed in the input. The object properties minValue and maxValue set limits on the input value.
-vcmd option	The addInteger* and addReal* commands include properties minValue and maxValue. If these properties are not sufficient to verify the input value, you can use valueChanged.connect(cmd) to call a command that verifies the input value.
-master, -master_select, and -master_expr options	<i>obj_slave.addMaster(obj_master)</i> Also see addMasterSelect, addMastersWithExpr, and addMastersWithFunction.
CUSTOM::SETWINGEOM	Not available. You create and remove pages with <a href="#">addPage</a> and <a href="#">removePage</a> . You specify the visible pages with the tool.pages property. You specify the page an option is on with the tool.option.pages property.
CUSTOM::getCalibreVersionString	tool.getCalibreVersion
CUSTOM::unsetVarExists	You can test whether the option name is undefined or use the hasOwnProperty() function on the option name.  <pre>if (tool.rulesFile != undefined) {     //commands } if (tool.hasOwnProperty('rulesFile')) {     //commands } if (tool.layout.hasOwnProperty('topCell')) {     //commands }</pre>

**Table 10-37. Correspondence for Customization and Configuration File Commands (cont.)**

Tcl Customization File	New Configuration File
CUSTOM::getRunsetVarValue <pre>set topCell [CUSTOM:::getRunsetVarValue drcLayoutPrimary]</pre>	There is not a special command to get a runset value. GUI settings are available with the option property “value”.  <code>var topCell = tool.layout.topCell.value</code>
CUSTOM::setRunsetVarValue <pre>CUSTOM:::setRunsetVarValue drcStartRVE [set \$wb_startRVE]</pre>	There is not a special command to set a runset value. GUI options are set by assigning the “value” option property.  <code>tool.layout.topCell.value = TOP</code>  The controls for existing GUI options can easily be moved to a new page with the <code>tool.option.pages</code> property. The value is set as usual using the GUI.  See “ <a href="#">Adaptation Example: Move and Set GUI Options</a> ” on page 403.
CUSTOM::setOKCallback	Not directly applicable, as there is no OK button.  Use the <code>runCallback</code> tool property to specify a callback function. In GUI mode, this function is executed upon startup when a runset containing a reference to the customization file is loaded and when the Run button is pressed and a DEFINE or VARIABLE has changed. In GUI-batch mode, this function is executed at run time. See “ <a href="#">Example: Attach Callback Function to Run Button With tool.runCallback</a> ” on page 249.  Use the <code>connect()</code> function to attach function calls to a button or to a change signal from a GUI control. See “ <a href="#">Adaptation Example: Callback Function</a> ” on page 407.  Runset values defined with this function, such as the rules file, override runset changes made in the GUI. Also see <code>addAction</code> and <code>addChoicesAction</code> .
CUSTOM::setOKCallbackError	If there is an error condition, set the tool property <code>blockRun</code> to true—this halts the run when the <b>Run &lt;app&gt;</b> button is clicked and displays the <code>tool.blockReason</code> string in a popup dialog box.  When using the <code>runCallback</code> tool property, you can use the <code>runCallbackError</code> property.  See the example with <code>getCalibreVersion</code> .
\$env(MGC_HOME)	<code>tool.envVars.envValue("MGC_HOME")</code>

**Table 10-37. Correspondence for Customization and Configuration File Commands (cont.)**

Tcl Customization File	New Configuration File
info exists env (MGC_HOME)	tool.envVars.isEnvSet("MGC_HOME")
set env(MY_VAR) 123	tool.envVars.setEnv("MY_VAR", 123)
puts "Hello"	tool.transcript("Hello")
exec (for executing a shell command)	launchCommand launchCommandReturnString
proc (for defining a Tcl procedure)	function  Use the function keyword to declare a function. Function definitions should be placed in the Configuration section.

## Adaptation Example: #DEFINE and Variable Statements

The Tcl customization file uses CUSTOM:VARIABLE, CUSTOM::DEFINE with the -choices option, and CUSTOM::LABEL with the -tool option. The equivalent new configuration file uses addIntegerVariable, addChoicesDefine, and addLabel. It checks for the tool version with the tool.name property.

### Tcl Customization File for #DEFINE and Variable Statements

The Tcl customization file adds a CUSTOM:VARIABLE control and a CUSTOM::DEFINE control with a dropdown list to the customization dialog for all Calibre Interactive applications. For LVS only, a section label and a second CUSTOM::DEFINE control are added.

```
## Tcl Customization File for Classic Calibre Interactive

### General Settings
# Variable RES_TOL, initialize to 1
CUSTOM::VARIABLE -name RES_TOL -initval 1 -prompt "Variable RES_TOL" \
    -tooltip "Provide a value for Variable RES_TOL"

# DEFINE process, use a dropdown list, initialize to first value
CUSTOM::DEFINE -name process \
    -choices {{"wide metal" wmp} {"process 7" 71m}} \
    -initval wmp -prompt "Define process"
```

```
### LVS Settings
# Section label
CUSTOM::LABEL -prompt "LVS Run Settings" -tool LVS

# DEFINE MARKER_DIODE, use checkbox, and initialize to selected
CUSTOM::DEFINE -name "MARKER_DIODE" -boolean 1 -select 1 -enable 1 \
-prompt "Set MARKER_DIODE" -tool LVS
```

## New Configuration File for #DEFINE and Variable Statements

The following code is the corresponding new configuration file. The addChoicesDefine command is used as the equivalent to CUSTOM::DEFINE with the -choices option. The tool.name property is evaluated in order to add controls for only the LVS application. By default the controls are added to the Custom page.

```
// New Configuration File for Calibre Interactive
Configuration {
onInitialized: {

// General Settings

// Integer Variable RES_TOL, initial value 1
var ctrlVAR = tool.addIntegerVariable("RES_TOL", 1,
    "Provide a value for Variable RES_TOL");
// default label is "Variable RES_TOL"

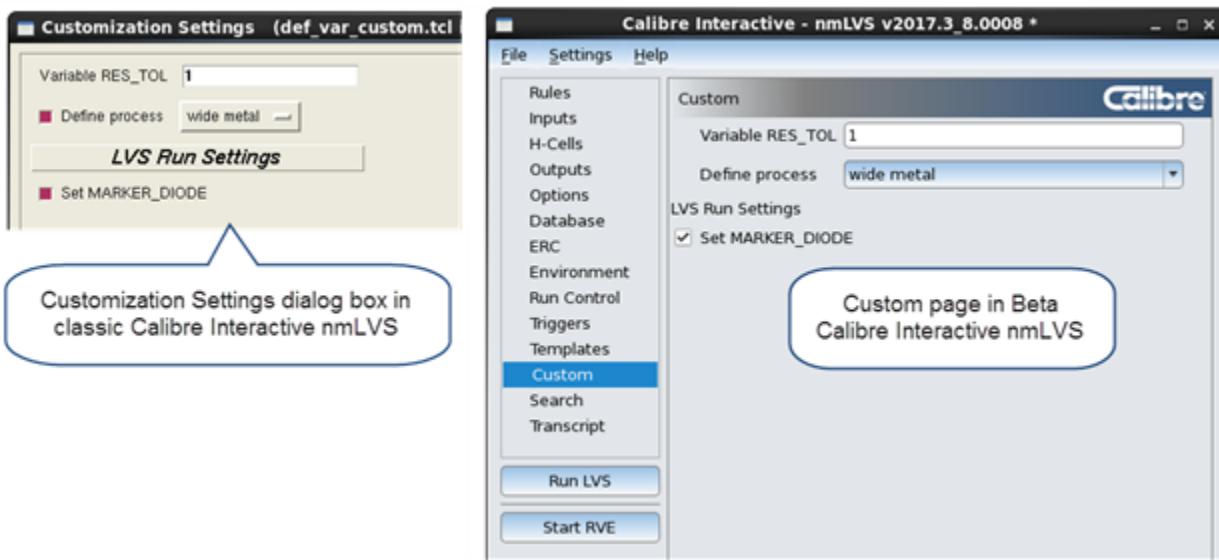
// DEFINE process, use a dropdown list, initialize to first value
var ctrlDEF = tool.addChoicesDefine("process",
    ["wmp", "71m"], // the possible values, strings must be quoted
    ["wide metal", "process 71m"]); // the dropdown list labels
// tooltips are not provided and default to the value
// initialize to first choice
ctrlDEF.value = ctrlDEF.choiceAt(0).value;
// default label is "Define process"

// LVS Settings
if (tool.name == "lvs") {
    // section label
    var lab = tool.addLabel("LVS Run Settings");

    // DEFINE MARKER_DIODE with no value, initialize to selected
    var ctrlDEF2 = tool.addDefine("MARKER_DIODE");
    ctrlDEF2.value = true; // select checkbox, default value is false
    ctrlDEF2.label = "Set MARKER_DIODE"; // provide a non-default label
    // checkbox included by default
}
} // onInitialized
} //Configuration
```

The results from each file are shown.

**Figure 10-3. #DEFINE and Variable Custom Configuration Example**



## Adaptation Example: Move and Set GUI Options

The Tcl customization file in this example uses CUSTOM:::getRunsetVarValue, CUSTOM:::setRunsetVarValue, and CUSTOM:::setOKCallback to set a runset value from the Customization GUI. The new configuration file places GUI controls on a new page to accomplish the same function.

The Tcl customization file uses CUSTOM:::setRunsetVarValue in a callback function specified by CUSTOM:::setOKCallback to set a GUI option from the Customization GUI.

There is no Customization GUI in new Calibre Interactive. However, you can create new GUI pages and place any GUI option on the new page. The GUI option is set from the GUI as with any other option, without the need for any additional commands.

### Tcl Customization File to Set GUI Options from the Customization GUI

In classic Calibre Interactive, this example places some controls for GUI settings in the Customization GUI. This can be useful to make a subset of controls available in one place. This example is for DRC, and places the settings for “Output cell errors in cell space” and “Show results in RVE” in the Customization GUI.

```
### Tcl Customization File for Classic Calibre Interactive

##### DRC settings #####
CUSTOM:::LABEL -prompt "DRC GUI Settings" -tool DRC
```

```
# Controls for GUI runset settings, initialized to current setting.  
# Need protection for runset var to prevent errors in LVS run  
if {[CUSTOM::runsetVarExists drcCellName]} {  
    set cellNameEntry [CUSTOM::DEFINE -name "cellName_var" \  
        -select [CUSTOM::getRunsetVarValue drcCellName] \  
        -prompt "Output in cell space (GUI setting)" -tool DRC]  
}  
if {[CUSTOM::runsetVarExists drcStartRVE]} {  
    set startRVEEntry [CUSTOM::DEFINE -name "startRVE_var" \  
        -select [CUSTOM::getRunsetVarValue drcStartRVE] \  
        -prompt "Start RVE (GUI setting)" -tool DRC]  
}  
  
## callback proc for OK button  
proc myOKCallback {} {  
    global cellNameEntry  
    global startRVEEntry  
  
    if {[CUSTOM::runsetVarExists drcCellName]} {  
        set wb_cellName [$cellNameEntry cget -bvar]  
        CUSTOM::setRunsetVarValue drcCellName [set $wb_cellName]  
    }  
  
    if {[CUSTOM::runsetVarExists drcStartRVE]} {  
        set wb_startRVE [$startRVEEntry cget -bvar]  
        CUSTOM::setRunsetVarValue drcStartRVE [set $wb_startRVE]  
    }  
}  
  
CUSTOM::setOKCallback myOKCallback
```

## New Configuration File to Put GUI Options on a New Page

You can easily move GUI options to a new GUI page in Calibre Interactive. You can specify new pages either with the tool.addPage function or the tool.pages property.

The location of GUI options is specified with the “pages” option property. In order to specify a new location for an option, we need to determine the option name used in the runset. For this example we need the option name corresponding to the DRC Cell Name statement and for the option to automatically start Calibre RVE. We use the search widget to find the options, then save a runset to determine the option names.

1. Open Calibre Interactive with default settings, with the environment variable defined.:

```
setenv CALIBRE_ENABLE_NEW_CI_DRC 1  
calibre -gui -drc
```

2. Enter “DRC Cell Name” in the Search text entry box at the top right corner of the GUI.

The view is automatically switched to the Search page.

Expand the DRC Results Database area on the Search page. The “Output cell errors in cell space” option is highlighted, and the tooltip shows that this is the option corresponding to DRC Cell Name.

Change the setting for “Output cell errors in cell space”. Later we will need the names of the other options, so also change the values for “Pseudocells results,” “DRC Check Text,” and “Format,” in that order. This is so the options are saved to the runset, as only non-default settings are saved.

3. Enter “RVE” in the Search text entry box. We are not sure of the name for the option to start Calibre RVE, so enter something general.

In the Search page, expand the area for “RVE Options” and change the setting for “Show results in RVE”.

4. Select **File > Save Runset** and save the runset as *drcoptions.runset*.
5. View the file *drcoptions.runset*:

```
drc.drcdb.resultsFormat.value = "GDSII"
drc.drcdb.resultsPseudoCells.specified = true
drc.drcdb.resultsCellName.value = false
drc.drcdb.resultsCheckText.specified = true
drc.rve.autoStart.value = true
```

This gives the names of the options we are interested in.

---

**Tip**

 You can save all runset options, including non-default settings, to a runset by setting the environment variable MGC\_CALIBRE\_SAVE\_ALL\_RUNSET\_VALUES to 1. However, it is not always straightforward to determine the runset option that corresponds to a particular GUI option.

---

The options we are interested in are sub-options to another option. The option resultsCellName is a sub-option to drcdb, and autoStart is a sub-option to rve. In Calibre Interactive, we must move the top-level option as a unit—we cannot move a sub-option by itself. Option names are prefaced with “tool” rather than the tool name (such as “drc”) in the configuration file, so the configuration file specifies the pages property for tool.drcdb and tool.rve. Two different methods are used to make sure the option names drcdb and rve exist.

The following new configuration file creates a new page named “Special Options.” and places the options for drcdb and rve on the new page. The drcdb options are also located on the default Outputs page. Unlike with classic Calibre Interactive, no special code is needed to enable setting the option values from the new page.

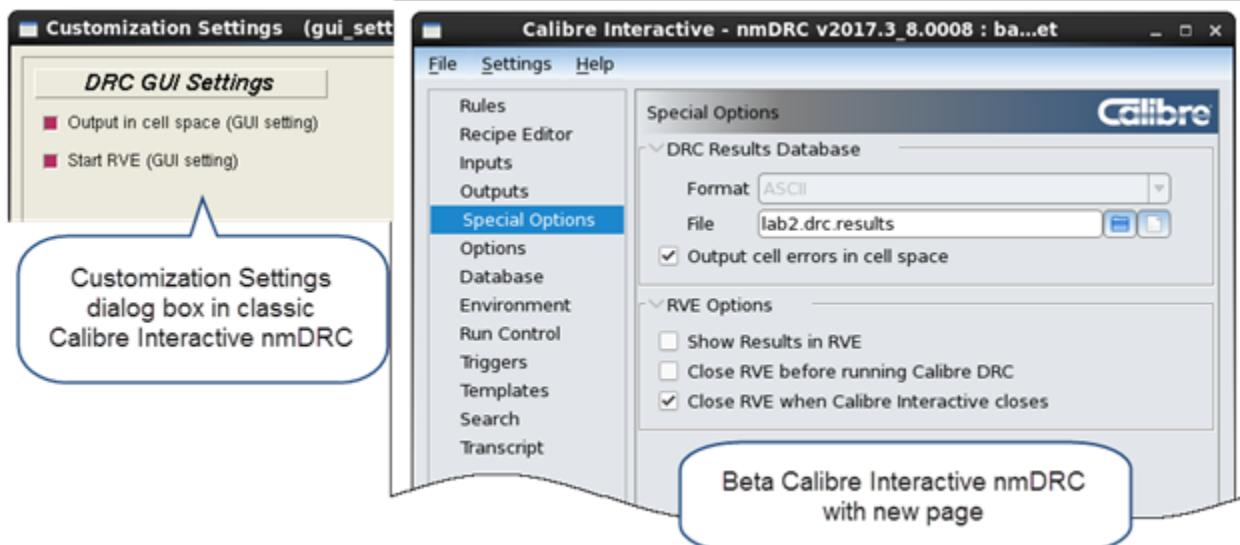
For the drcdb options, the sub-option resultsFormat is made non-editable, and the options for resultsPseudoCells and resultsCheckText are hidden. The properties for the option resultsCellName are not changed, so the option remains visible and editable.

## GUI Customization for Calibre Interactive Adaptation Example: Move and Set GUI Options

```
// New Configuration File for Calibre Interactive
Configuration {
onInitialized: {
if (tool.name == "drc") {
    // DRC Settings
    // create a new page, and place it before the Options page
    tool.addPage("Special Options", "Options");
    if (tool.drcdb != undefined) {
        // place the options for drcdb on the new page and the Outputs page
        tool.drcdb.pages = ["Special Options", "Outputs"];
        // make results format visible but not editable
        tool.drcdb.resultsFormat.editable = false;
        // hide the pseudocells and check text options
        tool.drcdb.resultsPseudoCells.visible = false;
        tool.drcdb.resultsCheckText.visible = false;
        // this code handles only the options relevant for the
        // default result format of ASCII
    }
    if (tool.hasOwnProperty('rve')) {
        // place the options for starting RVE on the new page only
        tool.rve.pages = ["Special Options"];
    }
}
} // onInitialized
} // Configuration
```

The results are shown in the following image. In Calibre Interactive, the new page has the top-level options that contain the settings for “Output cell errors in cell space” and “Show Results in RVE.”

**Figure 10-4. Moving GUI Options with the Configuration File**



## Adaptation Example: Callback Function

The Tcl customization file uses CUSTOM:::setOKCallback to associate a function with the OK button in the Customization GUI. The callback function is executed if a checkbox in the Customization GUI is checked. The new configuration file creates a button and uses the connect() function to connect a function to the button click.

The original Tcl customization file creates a checkbox and attaches a callback function to the OK button in the Customization GUI. The callback function checks the state of the checkbox.

The new configuration file makes changes directly in the Calibre Interactive GUI, so there is not an OK button, as with the Customization GUI. Two ways to obtain the same functionality are demonstrated:

- [New Configuration File to Create a Button with an Associated Function](#)
- [New Configuration File to Attach a Callback Function to a DEFINE Control](#)
- [Attach Callback Function to Run Button](#)

### Tcl Customization File with Callback Function

This example Tcl customization file is written for Calibre Interactive nmLVS. It creates one checkbox in the Customization GUI. If the checkbox is checked, all SVDB options are set. CUSTOM:::setOKCallback is used to specify a function that is executed when the OK button is clicked in the Customization GUI. The callback function sets the appropriate runset options if the checkbox is checked.

```
### Tcl Customization File for Classic Calibre Interactive
# checkbox to specify whether to set SVDB options
    set setsVDBon [CUSTOM:::DEFINE -name "setSVDB_on" \
    -select 0 -tool LVS \
    -prompt "Turn on SVDB options" ]
## callback proc for OK button

proc myOKCallback { } {
    global setsVDBon

    # use cget -bvar to get checkbox setting for setsVDBon
    set wb_setsVDBon [$setsVDBon cget -bvar]
```

```
# if checkbox is checked, set all the SVDB options
if {[set $wb_setSVDBon] } {
    if {[CUSTOM::runsetVarExists lvsCreateSVDB] } {
        CUSTOM::setRunsetVarValue lvsCreateSVDB 1
    }
    if {[CUSTOM::runsetVarExists lvsSVDBxcal] } {
        CUSTOM::setRunsetVarValue lvsSVDBxcal 1
    }
    if {[CUSTOM::runsetVarExists lvsSVDBtxref] } {
        CUSTOM::setRunsetVarValue lvsSVDBtxref 1
    }
    if {[CUSTOM::runsetVarExists lvsSVDBnopinloc] } {
        CUSTOM::setRunsetVarValue lvsSVDBnopinloc 1
    }
    if {[CUSTOM::runsetVarExists lvsSVDBcci] } {
        CUSTOM::setRunsetVarValue lvsSVDBcci 1
    }
}
}

CUSTOM::setOKCallback myOKCallback
```

## New Configuration File to Create a Button with an Associated Function

The following new configuration file accomplishes the same function as the previous code, but is applicable to the Calibre Interactive PERC and Calibre Interactive XACT applications, in addition to Calibre Interactive LVS.

The configuration file adds a button on the Custom page, which is the default location for new controls. The activated.connect() function is used to associate the setSVDBOn() function with the button. The code tests for the existence of options that are not common across the three applications.

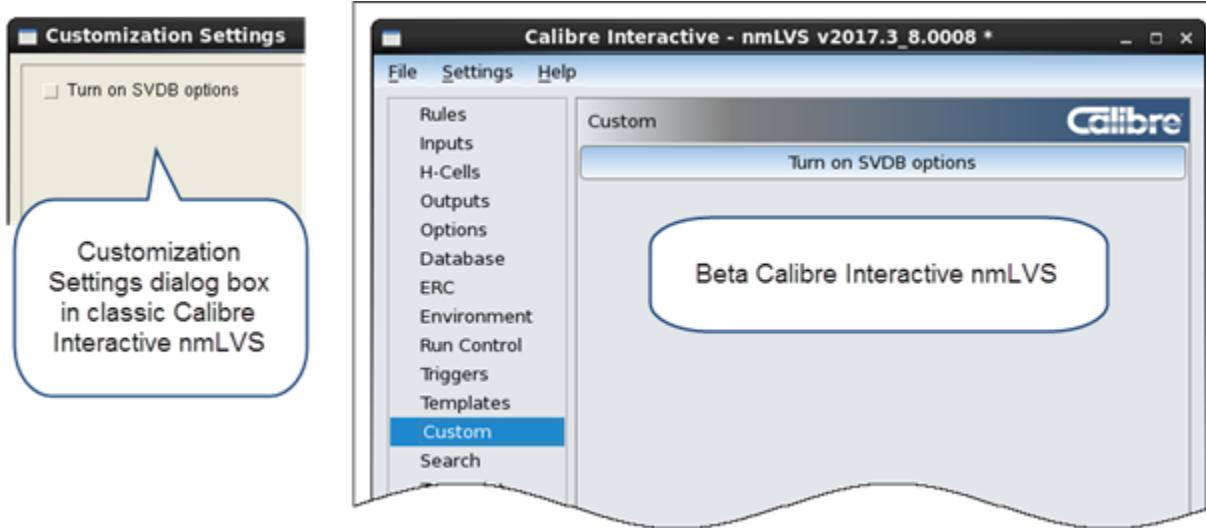
For a general procedure to find the option name corresponding to a GUI option, see “[Finding Option Names to Use in the Configuration File](#)” on page 254.

```
// New Configuration File for Calibre Interactive
Configuration {
    onInitialized: {
        tool.transcript("tool name " + tool.name);
        // add button to turn on svdb options for xact, lvs, and perc
        // "pex" is the tool.name for Calibre Interactive xACT
        if ( (tool.name == "lvs") || (tool.name == "perc")
            || (tool.name == "pex") ) {
            var bSVDB = tool.addAction("SVDBon");    //name is internal
            bSVDB.label = "Turn on SVDB options"; // the button label
            // tooltip
            bSVDB.description = "Turn on SVDB and all the options";
            bSVDB.activated.connect(setSVDBOn);
            bSVDB.location = Location.Body;    // default location is Footer
        }
    } // end onInitialized
```

```
// define function to set SVDB options if SVDBon button is clicked
function setSVDBOn() {
    tool.svdb.query.value = true;
    tool.svdb.asciixref.value = true;
    tool.svdb.cci.value = true;
    tool.svdb.pinloc.specified = true;
    tool.svdb.pinloc.value = "PINLOC";
    if ( tool.svdb.xrc != undefined ) {
        tool.svdb.xrc.value = true; // lvs and perc only
    }
    if ( tool.svdb.genParasiticData != undefined ) {
        tool.svdb.genParasiticData.value = true; // xact only
    }
    if ( tool.svdb.muwdb != undefined ) {
        tool.svdb.muwdb.value = true; // perc only
    }
}
} // end Configuration
```

The results of the two code examples are shown below.

**Figure 10-5. Configuration File Example with Callback Function**



The `connect` function can also be used to associate a function with a change signal from an property—this is shown in the next example.

### New Configuration File to Attach a Callback Function to a DEFINE Control

This example creates an `addDefine` control, which has only a checkbox. The `connect()` function specifies to execute the function `setSVDBOn` when the state of the checkbox changes. If the checkbox value is true, the function turns on several options related to the SVDB database. The variable for the `addDefine` object is declared in the Configuration section so that it has a global scope—this is necessary so that the function `setSVDBOn` can test the state of the checkbox.

```
// JavaScript Configuration File for Calibre Interactive
Configuration {
    property variant def_svdb; // make the control variable global

    onInitialized: {
        tool.transcript("tool name " + tool.name);
        // add a checkbox with a function connect to the change signal
        // from the checkbox state
        if ( (tool.name == "lvs") || (tool.name == "perc")
            || (tool.name == "pex") ) {
            def_svdb = tool.addDefine("SVDBon"); //name is internal
            def_svdb.label = "Turn on SVDB options"; // the button label
            def_svdb.description = "Turn on SVDB and all the options";
            // tooltip
            def_svdb.valueChanged.connect(setSVDBOn);
        }
    } // end onInitialized

    // define function to set SVDB options when SVDBon checkbox is changed
    // and checkbox is checked
    function setSVDBOn() {
        tool.transcript("checkbox is " + def_svdb.value);
        // only turn on options if checkbox is on (true)
        if (def_svdb.value) {
            tool.svdb.query.value = true;
            tool.svdb.asciixref.value = true;
            tool.svdb.cci.value = true;
            tool.svdb.pinloc.specified = true;
            tool.svdb.pinloc.value = "PINLOC";
            if ( tool.svdb.xrc != undefined ) {
                tool.svdb.xrc.value = true; // lvs and perc only
            }
            if ( tool.svdb.genParasiticData != undefined ) {
                tool.svdb.genParasiticData.value = true; // xact only
            }
            if ( tool.svdb.muwdb != undefined ) {
                tool.svdb.muwdb.value = true; // perc only
            }
        }
    }
}
```

## Attach Callback Function to Run Button

You can also attach a callback function to the Run button using the runCallback tool property. See “[Example: Attach Callback Function to Run Button With tool.runCallback](#)” on page 249 in the section “Tool Properties.”

# Troubleshooting Configuration Files for the Calibre Interactive GUI

There are some common errors in writing the configuration file for customization the Calibre Interactive GUI. This section outlines methods for finding and fixing these errors.

---

## Tip

 The actual error may be at the one plus the line number reported in the transcript page.

---

## Problem: Control does not appear in the GUI

Cause: Check these common programming errors:

- Define or Variable name is not entered correctly throughout the code for the control.
- Other formatting error.

Solution: Inspect the code and correct the error.

## Problem: Quotes are around the assigned value but not wanted

Cause: Quotes are used by default for the text, choices, and addListVariable controls.

Solution: Set *ctrl\_var.valueQuoted* = false.

## Problem: The text entry box does not allow spaces

Cause: Spaces are not allowed by default.

Solution: Set *ctrl\_var.allowSpaces* = true.



# Chapter 11

## Trigger Functions in Calibre Interactive

---

Calibre Interactive can execute trigger functions before and after Calibre Interactive execution and before and after the Calibre run.

<b>Trigger Overview</b> .....	<b>413</b>
<b>Internal Trigger Execution in Calibre Interactive</b> .....	<b>415</b>
<b>Internal Triggers in Calibre Interactive</b> .....	<b>417</b>
<b>External Triggers in Calibre Interactive</b> .....	<b>429</b>

## Trigger Overview

Triggers are functions that execute at specific times in your work flow. Calibre Interactive has two types of triggers: external, which execute before and after Calibre Interactive, and internal, which execute before and after the Calibre run.

The trigger types are described in the following table.

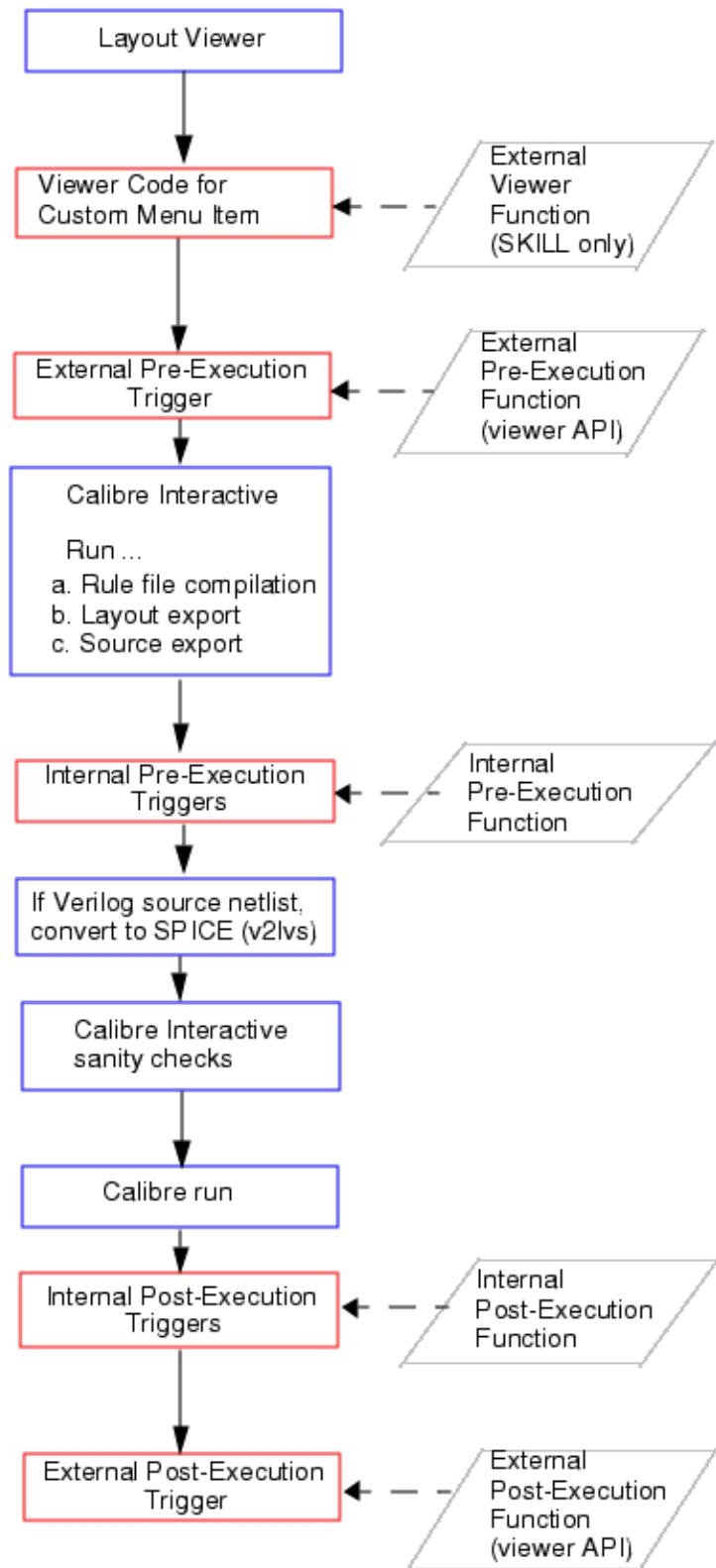
**Table 11-1. Triggers in Calibre Interactive**

Trigger Type	Characteristics
<a href="#">Internal Triggers in Calibre Interactive</a>	Internal triggers are defined within Calibre Interactive and can be saved to a runset. Internal trigger functions run in the Linux shell or a supported design tool.
<a href="#">External Triggers in Calibre Interactive</a>	External triggers operate when you start Calibre Interactive from a layout or schematic viewer that supports external triggers. They are written in the API for the supported design environment using the API language (Tcl, SKILL, or Scheme, for example).

Both Internal and External triggers can be set up as pre-execution or post-execution triggers. The following figure illustrates the order in which the various triggers are executed during a Calibre Interactive job.

**Figure 11-1. Trigger Pre- and Post-Execution Order**

1. The user opens a supported design tool, then starts Calibre Interactive from the built-in menu item or from a Custom Menu Items in Design Tools.
2. If a custom menu item is used to invoke Calibre Interactive, the optional viewer code executes in the design tool. (Cadence Virtuoso only)
3. The defined external pre-execution trigger executes.
4. The Calibre Interactive GUI opens. The user clicks "Run".  
  
The rule file must be complete for the compilation step.  
  
If exporting from a viewer, layout export, then source export (if defined), occur.
5. The defined internal pre-execution function executes before the Calibre job begins.
6. If the source netlist is Verilog, it is converted to SPICE format using v2lvs.
7. If a source netlist is used, Calibre Interactive checks it for completeness. If the input is exported from a viewer, Calibre Interactive checks for existence of the exported file.
8. The Calibre job starts.
9. The defined Internal post-execution function executes immediately after the job completes.
10. The user closes the Calibre Interactive GUI. The defined external post-execution function executes.



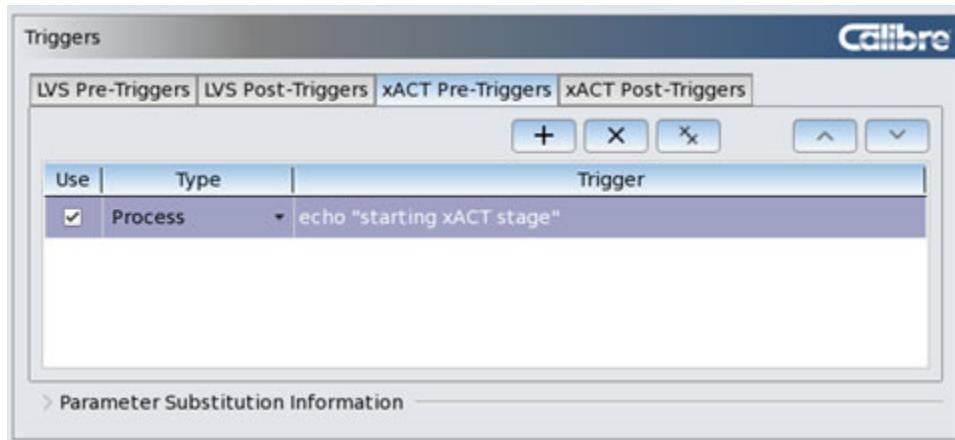
# Internal Trigger Execution in Calibre Interactive

Internal triggers are functions that run before or after the batch Calibre run. They are defined in Calibre Interactive and saved to the runset.

You define internal triggers on the **Triggers** page of the Calibre Interactive GUI; see “[Triggers Page in Calibre Interactive](#)” on page 122. You can also define triggers in a configuration file; see “[Configuration File Commands for Trigger Functions](#)” on page 390.

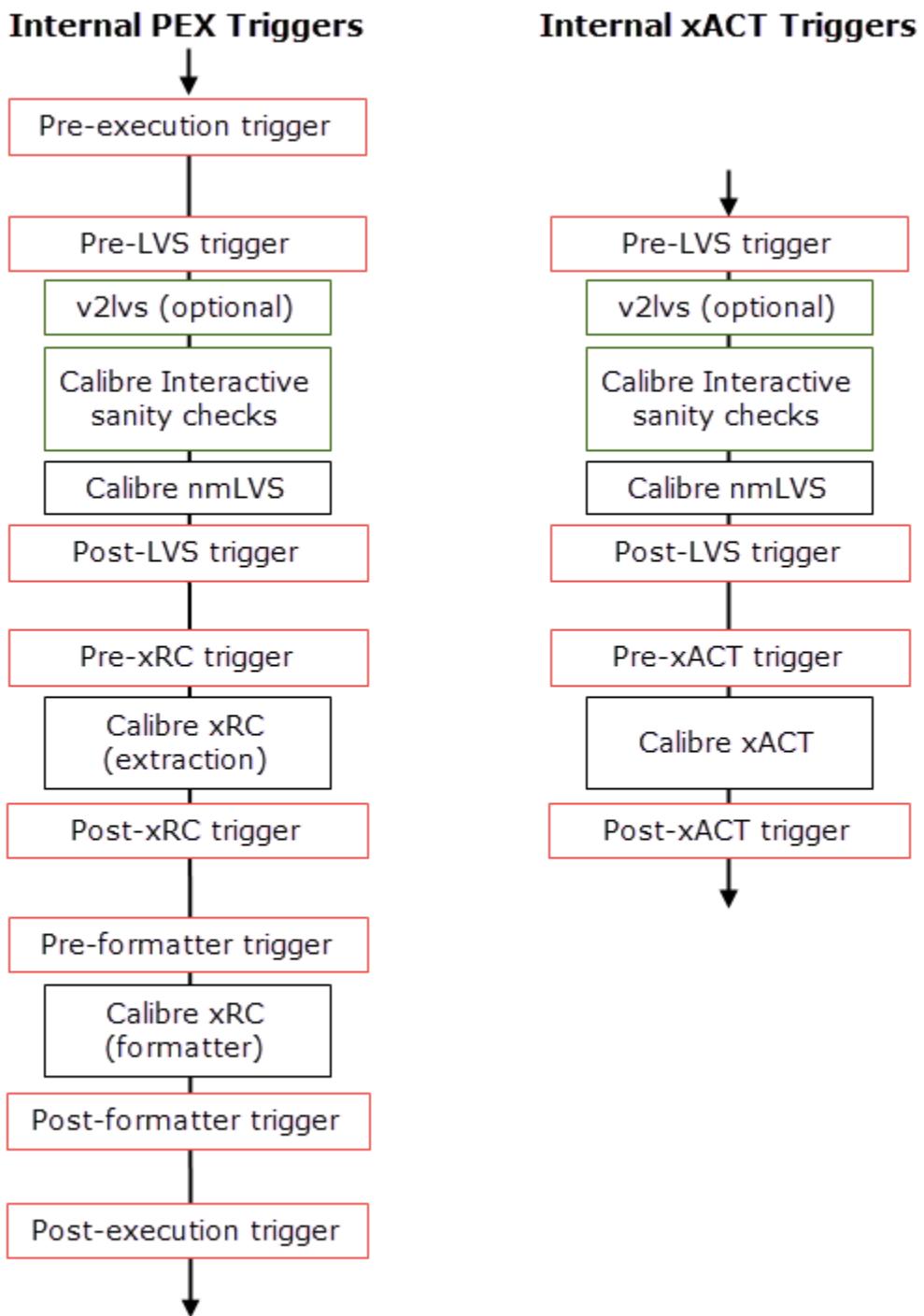
See “[Trigger Overview](#)” on page 413 for an overview of trigger execution for both external and internal triggers.

Calibre Interactive PEX and Calibre Interactive xACT support more internal triggers than the other Calibre Interactive applications. This is because a parasitic extraction run involves multiple Calibre execution steps. The following image shows the Triggers page for Calibre Interactive xACT.



The following diagram explains the timing of the internal triggers for Calibre Interactive PEX and Calibre Interactive xACT.

**Figure 11-2. Internal Trigger Timing for Parasitic Extraction Runs**



# Internal Triggers in Calibre Interactive

Calibre Interactive provides the ability to specify and run pre- and post-execution internal triggers. The trigger program is run immediately before or after batch Calibre. A trigger can be any program or script that can be executed from the Linux shell. Triggers can also run in supported design tools; in this case the trigger is written in the API language for the design tool.

Internal triggers are defined as follows:

- **Pre-execution internal trigger** — Performed before the Calibre job begins. If the internal pre-execution trigger returns an error code, the subsequent Calibre execution is canceled.

If Calibre Interactive is exporting a database from an attached design tool, then Calibre Interactive checks for the existence of the exported file after the pre-execution trigger finishes executing.

For Calibre nmLVS, Calibre PERC, and PEX runs, Calibre Interactive checks the source netlist for completeness after the pre-execution trigger finishes executing. (See “Check source netlist is complete before running LVS” in “[Miscellaneous Preferences](#)” on page 131).

- **Post-execution internal trigger** — Performed after the Calibre job ends. If the Calibre run itself fails, the internal post-execution trigger is not executed.

See “[Trigger Overview](#)” on page 413 for a diagram of the execution order of both internal and external trigger functions.

Calibre RVE for DRC offers additional options for setting triggers around database access events; see “[Setup Database Filters and Triggers Options Pane](#)” in the *Calibre RVE User’s Manual*.

These topics are covered:

<b>Trigger Parameters for Internal Triggers in Calibre Interactive.....</b>	<b>417</b>
<b>Setting Calibre Interactive Internal Triggers .....</b>	<b>419</b>
<b>Trigger Access to the Design Tool Environment .....</b>	<b>422</b>
<b>Calibre Interactive Internal Trigger Examples .....</b>	<b>424</b>

## Trigger Parameters for Internal Triggers in Calibre Interactive

Various filenames and other key values specified in the GUI can be passed as replaceable parameters that are expanded and passed to the internal trigger program.

The parameters listed in [Table 11-2](#) are recognized for internal triggers and substituted with the appropriate runtime values. The column “Calibre Interactive Applications” lists the Calibre Interactive applications for which the parameter is valid. If you use a parameter not recognized in your run type, no substitution occurs. See “[Calibre Interactive Internal Trigger Examples](#)” on page 424 for examples using replaceable parameters in internal triggers functions. See “[Setting Calibre Interactive Internal Triggers](#)” on page 419 for instructions on defining internal triggers.

**Table 11-2. Internal Trigger Parameters**

<b>Runtime Replacement</b>	<b>Parameter</b>	<b>Calibre Interactive Applications</b>
Circuit netlist	%c	LVS and PEX
DRC RDB	%b	DRC
DRC summary file	%m	DRC
Hcell file	%h	LVS, PERC, and PEX
Layout format (GDSII, OASIS, LEFDEF, or OPENACCESS)	%F	All
Layout2 format (GDSII, OASIS, LEFDEF, or OPENACCESS)	%F2	DRC FastXOR
Layout path	%l	All
Layout2 path	%l2	DRC FastXOR
Layout primary cell <sup>1</sup>	%L	All
Layout2 primary cell	%L2	DRC FastXOR
Layout library	%y	All applications when started from Cadence Virtuoso
Layout2 library	%y2	DRC FastXOR when started from Cadence Virtuoso
Layout view	%v	All
Layout2 view	%v2	DRC FastXOR
LVS Report file	%m	LVS
LVS Report file	%n	PEX
PERC Report file	%m	PERC
PEX netlist	%p	PEX
PEX netlist format	%f	PEX
PEX Report file	%m	PEX

**Table 11-2. Internal Trigger Parameters (cont.)**

Runtime Replacement	Parameter	Calibre Interactive Applications
Rule file	%r	All
Run directory	%d	All
Runset filename	%R	All
Source library	%w	LVS, PEX, and PERC
Source path for primary netlist source <sup>2</sup>	%s	LVS, PEX, and PERC
Source primary cell	%S	LVS, PEX, and PERC
Source path for all netlist sources <sup>3</sup>	%N	LVS, PEX, and PERC
Source view	%z	LVS, PEX, and PERC
Termination status: 0 — Normal 1 — Aborted 2 and above — the Calibre exit code	%e	All

<sup>1</sup> Wildcards in the layout primary cell name are only expanded for a post-execution trigger run on the local host. In addition, the layout format must be GDSII, and the Calibre Interactive application must be DRC or PERC.

<sup>2</sup> For SPICE netlist input, %s returns the netlist(s) listed in the Source Path section of the **Inputs** page. In the case of VERILOG or MIXED source input, %s returns the SPICE file that is translated from the Verilog source input.

<sup>3</sup> In addition to the source paths returned by %s, the %N trigger parameter also returns the additional SPICE netlist files specified in the Library section of the **Database** page (**Settings > Show Pages > Database**). In the case of MIXED source input, %N also includes the SPICE netlist files listed in the Source Path section of the **Inputs** page.

## Setting Calibre Interactive Internal Triggers

You can instruct Calibre Interactive to run internal triggers before and after the Calibre run. Multiple trigger functions may be specified for each stage. If you are connected to a supported design tool, you can also specify procedures that run in the design tool environment.

## Prerequisites

- For triggers that run in the Linux shell:

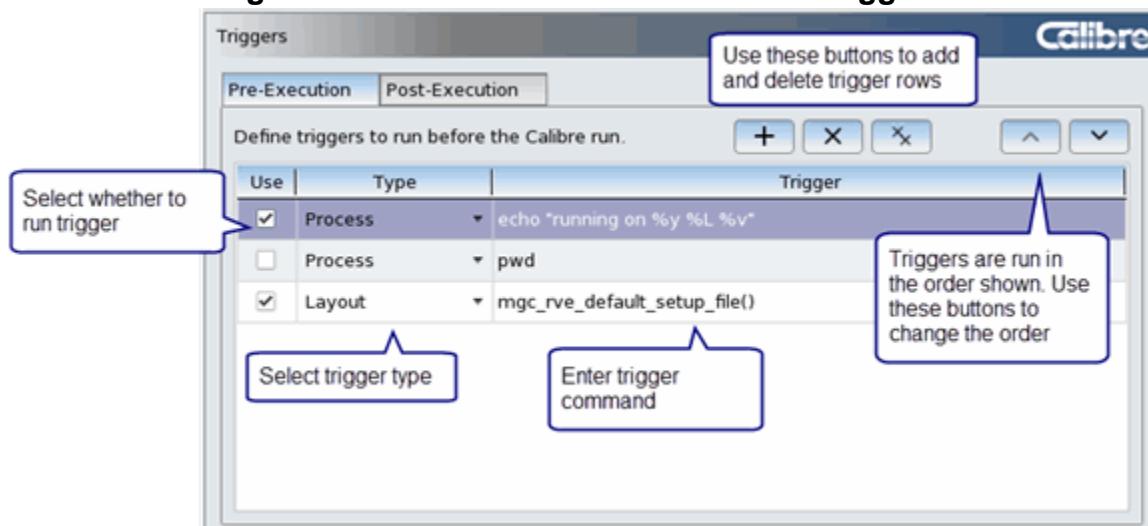
A trigger program written in the shell programming or scripting language of your choice (Awk, Sed, or Perl, for example). See “[Processing a Netlist Before the Calibre Run](#)” in the *Calibre Solutions for Physical Verification* manual for a simple script run as a trigger.
- For triggers that run in the design tool (Cadence Virtuoso, Cadence Encounter, Synopsys IC Compiler, and Calibre DESIGNrev):
  - A procedure written in SKILL or Tcl, depending on your design tool. The procedure should return the proper success or fail return code for the design environment. Calibre Interactive halts execution if the return value indicates failure.
  - Calibre Interactive was invoked from the design tool or the environment variable MGC\_CALIBRE\_ALLOW\_VIEWER\_TRIGGER is set to “always”. See “[Trigger Access to the Design Tool Environment](#)” on page 422.

For an example, see “[Internal Trigger Executed in Calibre DESIGNrev to Count Layout Cells](#)” on page 426.

## Procedure

1. Open the **Triggers** page (**Settings > Show Pages > Triggers**).
2. Click the **Pre-Execution** tab.
3. Add trigger information to the table as shown in [Figure 11-3](#). See the dialog box or “[Trigger Parameters for Internal Triggers in Calibre Interactive](#)” on page 417 for a list of available parameters.

**Figure 11-3. Calibre Interactive Internal Triggers**



Triggers are executed in the order shown; drag and drop a row to change the order.

The Type column specifies the where the trigger runs:

Trigger Type	Execution environment for trigger
Process	(default) Run as a Linux process.
Layout	Run in the layout viewer. (Cadence Virtuoso, Cadence Encounter, Synopsys IC Compiler, and Calibre DESIGNrev only)
Schematic	Run in the schematic viewer. (Cadence Virtuoso only)

By default, layout and schematic triggers can only run in the design tool session that invoked Calibre Interactive. See “[Trigger Access to the Design Tool Environment](#)” on page 422 for further information.

- Click the **Post-Execution** tab and enter trigger information as shown in [Figure 11-3](#). The following options are also available for post-execution triggers:
  - Run trigger in background** — Run the post-execution trigger as a background process.
  - Also run trigger if Calibre is terminated by user** — Run the post-execution triggers if the Calibre process is terminated by the user. By default the trigger is not run if the Calibre run is terminated.
  - Run on remote host** — Run the post-execution triggers on the same host as the main Calibre run. This setting only applies to triggers of type “process” (see [Figure 11-3](#)). By default, the post-execution trigger is run on the local host (the host running Calibre Interactive).

The following statements apply when this option is enabled:

- The post-execution trigger runs on the host specified by the “Run Calibre on” section in the **Run Control** page.
- The post-execution trigger script is added to the control script for the Calibre run.
- The control script preserves the Calibre exit code on return. The exit code of the post-execution trigger is not kept.
- For Calibre MTflex runs, the control script is added to the primary host’s script.
- If using a Post-LVS-Extraction trigger during a Calibre Interactive PEX run, the Post-LVS-Extraction trigger is added to the PHDB control script.
- If this setting is enabled, post-execution layout- and schematic type triggers are run in the design tool after the process-type triggers complete execution.

5. (Optional) If you are running Calibre Interactive PEX you can enter triggers on the **Post-LVS** tab. These triggers run following the PHDB generation step of a PEX run. For more information on enabling PEX steps, see “[Controlling PEX Step Execution](#)” on page 213.

6. Click **OK**.

## Results

If the pre-execution trigger returns an error code, the subsequent Calibre execution is canceled. If the Calibre run itself fails, the post-trigger is not executed.

If a trigger function exits with an error code, the error code is printed to the transcript. The message has the following general format:

```
Trigger function <trigger_function> exited with return code <code>.  
<error_message>  
child process exited abnormally
```

## Related Topics

[Calibre Interactive Internal Trigger Examples](#)

[External Triggers in Calibre Interactive](#)

# Trigger Access to the Design Tool Environment

The **Triggers** tab on the Setup Preferences dialog box allows you to specify triggers that run in the design tool environment (the trigger Type of “layout” or “schematic”). By default, a layout or schematic trigger can only run in the design tool session that launched the Calibre Interactive session.

For example, suppose you have two separate Cadence Virtuoso sessions open, one a layout view and one a schematic view. If you launch Calibre Interactive from the layout session, by default you can only run triggers in that session; triggers in the schematic session are not allowed to run.

The environment variable MGC\_CALIBRE\_ALLOW\_VIEWER\_TRIGGER specifies the trigger access to the design tool environment.

MGC\_CALIBRE\_ALLOW\_VIEWER\_TRIGGER can be defined as follows:

- **always** — Allow any process connected to the server socket to run trigger code in the design tool process. The value “always” is case-insensitive.
- **none** — Do not allow any process connected to the server socket to run trigger code in the design tool process. Any value other than “always” has the same effect.
- **undefined** — (default) A layout or schematic trigger can only run in the design tool session that launched the Calibre Interactive session if MGC\_CALIBRE\_ALLOW\_VIEWER\_TRIGGER is undefined.

## Related Topics

[Setting Calibre Interactive Internal Triggers](#)

## Calibre Interactive Internal Trigger Examples

Several examples are provided for Calibre Interactive internal triggers.

See “[Setting Calibre Interactive Internal Triggers](#)” on page 419 for instructions on specifying the trigger in Calibre Interactive.

You can find more examples in the chapter “[Pre- and Post-Processing for Calibre Runs](#)” in the *Calibre Solutions for Physical Verification* manual.

<b>Internal Trigger for Calling a UNIX Utility .....</b>	<b>424</b>
<b>Internal Trigger for Calling a Script .....</b>	<b>424</b>
<b>Internal Trigger to Set an SVRF Variable .....</b>	<b>424</b>
<b>Internal Trigger Executed in Calibre DESIGNrev to Count Layout Cells .....</b>	<b>426</b>
<b>Applying Triggers According to the Calibre Interactive Application .....</b>	<b>427</b>

### Internal Trigger for Calling a UNIX Utility

This example calls the “echo” utility with the trigger parameter %d as an argument. %d is replaced by the run directory at runtime.

Place the following text in the **Triggers** page as a pre-execution trigger:

```
echo %d
```

The parameter %d is replaced with the run directory at runtime, and the result of the echo command is displayed in the Calibre transcript.

### Internal Trigger for Calling a Script

This example calls a custom script with the trigger parameter %r as an argument. %r is replaced by the rule file path at runtime.

Place the following text in the **Triggers** page as a pre-execution trigger:

```
my_script %r
```

The script my\_script is executed before the Calibre run starts, and is passed the path to the rule file.

### Internal Trigger to Set an SVRF Variable

In this example an internal pre-execution trigger parses the topcell parameter and creates a file that is used as an INCLUDE file with the rule file for the Calibre run. The created file contains a Variable statement.

For this example, assume the following:

- The top cell name of a design is “<cell\_name>\_TOP”, where <cell\_name> varies from design to design.
- The cell, “<cell\_name>\_NEXT”, is always in the sub-hierarchy of <cell\_name>\_TOP. The <cell\_name> arguments are identical for each.
- There are other cell names in the design that end in “\_NEXT”.
- The following rule check is in the rule file:

```
inside_cell_test {@layer L1 inside "NEXT" cell
L1 inside cell CELLVAR
}
```

In this rule check, CELLVAR is a variable name and should specify “<cell\_name>\_NEXT”.

Note that you cannot use “\*\_NEXT” to define CELLVAR, because this would include other cells ending in “\_NEXT” that you do not want to check.

Take the following steps to specify an internal trigger that creates a file with a [Variable](#) statement and to include the created file in the Calibre Interactive run. The trigger function uses the replaceable parameter %L to write the current top cell name to the Variable statement.

1. Create a script named *my.trigger* as a pre-execution trigger for Calibre Interactive:

```
echo "VARIABLE CELLVAR '$1'" | sed "s/TOP/NEXT/" > cellvar.include
```

2. Create a file named *cellvar.include* with a Variable statement.

```
echo "VARIABLE CELLVAR 'cell_TOP'" > cellvar.include
```

This is done so that the include file exists and CELLVAR is defined when Calibre Interactive compiles the rule file for the first time; otherwise Calibre Interactive reports a compilation error. The *cellvar.include* file is overwritten with the correct cell name before Calibre execution begins.

3. In Calibre Interactive, open the **Triggers** page (**Settings > Show Pages > Triggers**) and specify “*my.trigger %L*” as a pre-execution trigger.

The *my.trigger* script uses the top cell argument (%L), replaces TOP with NEXT, and writes the following statement to the file *cellvar.include*:

```
VARIABLE CELLVAR '<cell_name>_NEXT'
```

4. Open the **Options** page (**Settings > Show Pages > Options**), then enable Include Rules Files and specify *cellvar.include*.

With this internal trigger and the created include file, Calibre Interactive uses the correct value for CELLVAR in the *inside\_cell\_test* rule check.

## Related Topics

[Setting Calibre Interactive Internal Triggers](#)

# Internal Trigger Executed in Calibre DESIGNrev to Count Layout Cells

You can specify a Calibre Interactive internal trigger that runs in Calibre DESIGNrev and counts the number of cells in the layout.

## Procedure

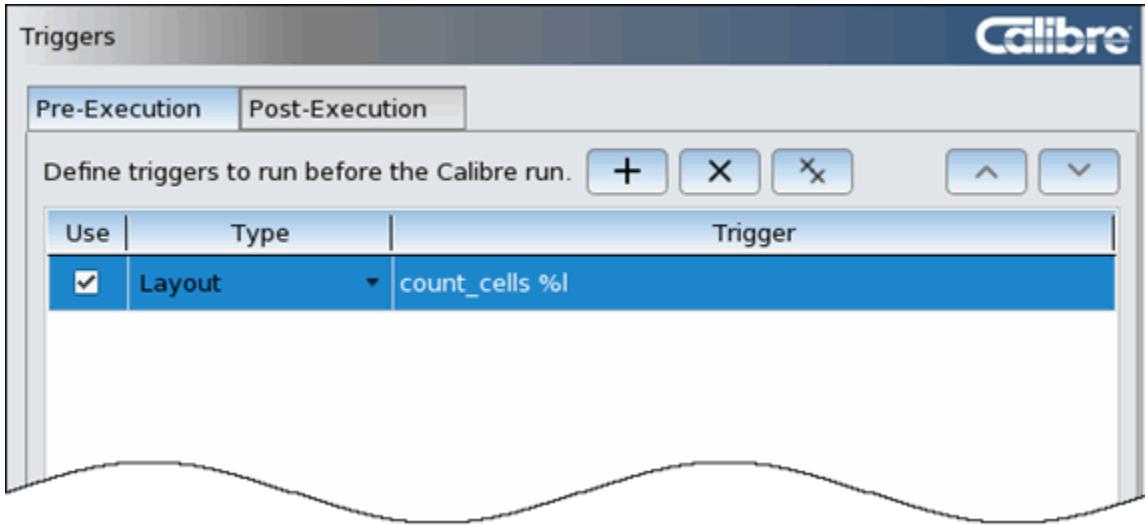
1. Add the following procedure definition to your `$HOME/.calibre_workspace/wbinit.tcl` file:

```
# called from trigger with %l as the layout file argument
proc count_cells {layout} {
    puts "cells in $layout: [layout peek $layout -cellcount]";
    return 1;
}
```

Make sure your Tcl procedure returns 1 for success; other return values indicate failure and cause Calibre Interactive to halt execution.

2. Open your design in Calibre DESIGNrev. The procedures defined in the `wbinit.tcl` file are loaded automatically.
3. Start Calibre Interactive.
4. Open the **Triggers** page (**Settings > Show Pages > Triggers**).
5. Add a trigger definition and type the following in the Trigger column to specify the `count_cells` procedure as the trigger function with `%l` as the argument. The parameter `%l` is replaced with the name of the layout file specified on the Layout Path section tab of the **Inputs** page.

```
count_cells %l
```



6. Click in the Type column and select “Layout” from the dropdown list. The “Layout” option specifies that the trigger executes in the connected layout viewer.
7. Set up the remaining Calibre Interactive options for your run.
8. Click the **Run** button in Calibre Interactive.

## Results

The text “Running viewer pre-trigger” is printed at the top of the Transcript pane in Calibre Interactive.

The following is printed to the Calibre DESIGNrev shell window for a layout database *lab2.calibre.db* with 9 cells:

```
cells in lab2.calibre.db: 9
```

## Related Topics

[Setting Calibre Interactive Internal Triggers](#)

## Applying Triggers According to the Calibre Interactive Application

In some development environments, a system administrator may define a global or project-level runset that sets options for *all* Calibre Interactive applications. In this case, it can be useful for the same runset to define internal pre- and post-execution triggers that differ according to Calibre Interactive application.

The following example defines different internal pre-execution triggers for Calibre Interactive nmLVS and Calibre Interactive PERC within a runset that defines some global options for Calibre Interactive.

## Procedure

1. Add the following lines to the runset used to define global options:

```
*lvsPreTriggers: {{lvs_netlist_script %s %S} process 1}  
*percPreTriggers: {{perc_netlist_script %s %S} process 1}
```

The trigger scripts take the source netlist (%s) and source primary cell (%S) as input.  
The parameter “process” indicates that the trigger runs as a Linux process.

2. Define an environment variable to load the multiple runsets for Calibre Interactive:

```
setenv MGC_CALIBRE_LVS_RUNSET_FILE "global_runset_lvs_runset"  
setenv MGC_CALIBRE_PERC_RUNSET_FILE "global_runset_perc_runset"
```

See “[Methods to Load a Calibre Interactive Runset](#)” on page 35 for other methods of loading multiple runsets.

## Related Topics

[Setting Calibre Interactive Internal Triggers](#)

# External Triggers in Calibre Interactive

---

Calibre Interactive external triggers operate when you start Calibre Interactive from a layout or schematic viewer that supports external triggers. External triggers are written in the API for the supported design environment using the API language (Tcl, SKILL, or Scheme, for example). The triggers can be used to run custom routines that check or set environment variables or call setup and post-processing functions.

For example, Calibre Interactive runsets and configuration files can be set by environment variables defined in an external pre-execution trigger. Refer to “[About Calibre Interactive Runsets](#)” on page 34 for more information on Runsets.

<b>External Trigger Types .....</b>	<b>429</b>
<b>External Trigger Functions and File Locations .....</b>	<b>430</b>
<b>Calibre Interactive External Trigger Examples .....</b>	<b>433</b>

## External Trigger Types

External triggers execute in the design tool environment before and after Calibre Interactive execution. There are two types of external triggers: pre-execution and post-execution.

Trigger Type	Definition
External pre-execution trigger	Executes after you choose a Calibre Interactive application from the <b>Verification</b> (Siemens viewers) or <b>Calibre</b> (other viewers) menu in your design tool. The trigger executes before Calibre Interactive starts. function name: mgc_start_calibre_trigger
External post-execution trigger	Executes when you close the Calibre Interactive window. function name: mgc_close_calibre_trigger

“[Trigger Overview](#)” on page 413 describes the order in which the various triggers are executed in a Calibre Interactive session.

The location where you define the trigger function depends on the desired scope of the trigger, as explained here:

- **User environment trigger** — The trigger is defined in the user environment and applies to all Calibre sessions for that user.
- **Local trigger** — The trigger is defined in the run directory for the design tool or directly loaded into the design environment. Depending on the design tool, this might be done with a command line argument or in a shell window for the design tool. This trigger definition takes precedence over the user environment trigger definition.

## External Trigger Functions and File Locations

The external trigger functions and the file locations depend on the design tool that is used.

External triggers are only supported for the following design tools:

<b>External Trigger Definitions in Calibre DESIGNrev.....</b>	<b>430</b>
<b>External Trigger Definitions in Cadence Virtuoso.....</b>	<b>431</b>
<b>External Trigger Definitions in Synopsys IC Compiler.....</b>	<b>432</b>

### External Trigger Definitions in Calibre DESIGNrev

You can define Calibre Interactive external pre- and post-execution triggers in Calibre DESIGNrev. This section gives the function name and the location of the file containing the function definition.

```
proc mgc_start_calibre_trigger {run_type cell} {
    <Tcl code>
}
proc mgc_close_calibre_trigger {run_type cell} {
    <Tcl code>
}
```

where the procedures are called with the following arguments:

- run\_type — The Calibre Interactive application (drc, lvs, pex, or perc)
- cell — The cell open in the viewer

The default external trigger definitions in Calibre DESIGNrev return 1.

You can define your own external trigger by providing a locally defined procedure that overrides the default definition. The following table gives the file locations for defining the external trigger procedures in Calibre DESIGNrev.

**Table 11-3. File Locations for External Trigger Definitions in Calibre DESIGNrev**

Trigger Scope	File Location
User environment	<code>~/.calibrewb_workspace/wbinit.tcl</code> See “ <a href="#">wbinit.tcl File Format</a> ” in the <i>Calibre DESIGNrev Layout View User’s Manual</i> for more information.
Local	<code>wbinit.tcl</code> file in a directory specified with the <code>MGC_CWB_CONFIG_DIRS</code> environment variable. See “ <a href="#">Multilevel Configuration</a> ” in the <i>Calibre DESIGNrev Layout View User’s Manual</i> for more information. You can also load a Tcl script with the <code>-s</code> command line option to calibre DESIGNrev.

## Related Topics

[External Trigger in Tcl](#)

## External Trigger Definitions in Cadence Virtuoso

You can define Calibre Interactive external pre- and post-execution triggers in Cadence Virtuoso. This section gives the procedure name and the location of the file containing the procedure definition.

```
procedure( mgc_start_calibre_trigger(drc_lvs_pex win)
prog( ())
    printf( "Starting %s on top cell in window %L...\n"
        upperCase(drc_lvs_pex) win)
    return(t)
)
procedure( mgc_close_calibre_trigger(drc_lvs_pex win)
prog( ())
    printf( "Closing %s on top cell in window %L...\n"
        upperCase(drc_lvs_pex) win)
    return(t)
)
```

where the procedures are called with the following arguments:

- drc\_lvs\_pex — The Calibre Interactive application (drc, lvs, pex, or perc)
- win — The ID of the window that Calibre Interactive was started from

The trigger functions are expected to return t or nil (true or false), and execution proceeds only if “t” is returned by the function.

You can define your own external trigger by providing a locally defined procedure that overrides the default trigger. The following table gives the file locations for defining the external trigger procedures in Cadence Virtuoso.

**Table 11-4. File Locations for External Trigger Definitions in Cadence Virtuoso**

Trigger Scope	File Location
User environment	.cdsinit  Add the definition of the trigger procedure after the Cadence interface has been loaded (see “ <a href="#">Creating an Interface to Cadence Virtuoso</a> ” on page 469).
Local	Consult Cadence documentation.

## Related Topics

[External Trigger in SKILL for Cadence Virtuoso](#)

## External Trigger Definitions in Synopsys IC Compiler

You can define Calibre Interactive external pre- and post-execution triggers in Synopsys IC Compiler. This section gives the procedure name and the location of the file containing the procedure definition.

```
proc mgc_start_calibre_trigger {run_type cell} {
    <Tcl code>
}
proc mgc_close_calibre_trigger {run_type cell} {
    <Tcl code>
}
```

where the procedures are called with the following arguments:

- run\_type — the Calibre Interactive application (drc, lvs, pex, or perc)
- cell — the cell open in the viewer

You can define your own external trigger by providing a locally defined procedure that overrides the default trigger. The following table gives the file locations for defining the external trigger procedures in Synopsys IC Compiler.

**Table 11-5. File Locations for External Trigger Definitions in Synopsys IC Compiler**

Trigger Scope	File Location
User environment	<code>~/.synopsys_icc_gui.tcl</code> Add the definition of the trigger procedure after <code>icc_calibre.tcl</code> has been sourced (see “ <a href="#">Loading the Calibre Interface to Synopsys IC Compiler</a> ” on page 566).
Local	Consult Synopsys IC Compiler documentation.

## Calibre Interactive External Trigger Examples

Tcl and SKILL examples are provided for Calibre Interactive external triggers.

<b>External Trigger in Tcl .....</b>	<b>433</b>
<b>External Trigger in SKILL for Cadence Virtuoso .....</b>	<b>433</b>

### External Trigger in Tcl

The following code is an example of a pre-execution external trigger in Tcl.

If the run type is DRC, it sets environment variables for the customization file and the runset. For Calibre DESIGNrev, this code can be placed in the *wbinit.tcl* file; see “[External Trigger Definitions in Calibre DESIGNrev](#)” on page 430 for complete information.

```
proc mgc_start_calibre_trigger {app cell} {
    puts "start trigger for CI-$app in cell $cell"

    # if DRC run then set customization file and runset
    if { $app eq "drc" } {
        global env
        set env(MGC_CALIBRE_CUSTOMIZATION_FILE) "custom_DRC"
        set env(MGC_CALIBRE_DRC_RUNSET_FILE) "runset_test"
        puts "runset: $env(MGC_CALIBRE_DRC_RUNSET_FILE)"
        puts "customization file: $env(MGC_CALIBRE_CUSTOMIZATION_FILE)"
    }
}
```

### External Trigger in SKILL for Cadence Virtuoso

The following example shows an external pre-execution trigger in the SKILL language for the Cadence Virtuoso design environment. This procedure can be defined in your *.cdsinit* file. The trigger defines a runset using an environment variable.

See “[External Trigger Definitions in Cadence Virtuoso](#)” on page 431 for information on defining the trigger.

```
procedure( mgc_start_calibre_trigger( run_type win)

    printf( "Starting %s run on top cell in %L\n" run_type win)

    ;Set Calibre Interactive Runset File
    setShellEnvVar("MGC_CALIBRE_DRC_RUNSET_FILE=/home/drc.runset")
    printf( "Using Calibre Interactive DRC Runset: %s/n" strcat(
        getShellEnvVar("MGC_CALIBRE_DRC_RUNSET_FILE")))

    ); end procedure
```

When the trigger is loaded, the following message appears in the CIW window:

```
Starting drc run on top cell in window: 2
Using Calibre Interactive DRC Runset: /home/drc.runset
```

# Chapter 12

## Run Control and Licensing Setup

---

The sections in this chapter provide important information about setting up Calibre jobs and choosing a licensing behavior.

<b>Specifying the Host Processor and Run Options .....</b>	<b>435</b>
<b>Configuring for Distributed (MTflex) Execution .....</b>	<b>438</b>
<b>Running With Spectrum LSF.....</b>	<b>440</b>
<b>Configuring the Remote Environment .....</b>	<b>443</b>
<b>Setting Licensing Options in Calibre Interactive .....</b>	<b>445</b>
<b>Reference for Run Control Options.....</b>	<b>447</b>
Hyperscaling Mode.....	447
Hyperscale Compare Option in Calibre Interactive nmLVS .....	447
Hyperscale Pathchk Option .....	447
Remote Data Server (RDS) Option .....	448
Hyperscale Remote Option.....	448
Backup Data Option .....	448

## Specifying the Host Processor and Run Options

Calibre Interactive enables you to run Calibre jobs on local, remote, and distributed resources. You can choose single threaded, multithreaded, or distributed execution with a choice of options such as number of CPUs and hyperscaling.

### Prerequisites

- For runs using remote hosts, configure your environment to allow rsh to execute your Calibre job remotely. This may include creating and editing a .rhosts file in your home directory.

You can also specify to use a remote shell other than rsh, as described in “[Configuring the Remote Environment](#)” on page 443.

### Procedure

- Click the **Run Control** button on the left panel of the Calibre Interactive GUI.

2. Choose the host processor or cluster option in the “Run Calibre on” dropdown list.

**Table 12-1. Run Calibre On Options on the Run Control Page**

Selection	Description
Local Host	Run Calibre on the computer currently running Calibre Interactive.
Remote Host	Run Calibre on a remote computer. <ol style="list-style-type: none"><li>1. Type the name of the remote host in the text entry field and press Enter. Basic information about the remote host is displayed in the Host Information field.</li><li>2. Configure the remote environment as described in “<a href="#">Configuring the Remote Environment</a>” on page 443.</li></ol>
Platform LSF	Use the IBM® Spectrum™ LSF® (Load Sharing Facility) distributed computing solution.
Other Cluster	Use a custom computer cluster.

3. Choose execution mode in the “Run Calibre” section, “Run Calibre <app> using”:

**Table 12-2. Run Calibre Execution Mode**

Selection	Description
Single Thread	Use single-threaded execution. This is the default. (-turbo 1)
Multiple Threads	Use multithreaded execution on a single host computer. Select the number of CPUs to use in the entry fields. (-turbo <i>num_processors</i> )
All Threads	Use multithreaded execution on a single host computer with all available CPUs. (-turbo)
Distributed (MTflex)	Use your existing computer network as a distributed computing platform with Calibre® MTflex™. Follow the setup procedure in “ <a href="#">Configuring for Distributed (MTflex) Execution</a> ” on page 438.

4. Choose other execution options in the “Run Calibre” section.

The available run options depend on the choice of execution mode.

**Table 12-3. Run Calibre Execution Options**

Option	Description
Number of CPUs to use for <app>	Specifies the number of processors in “Multiple Threads” (MT) mode (the value for the -turbo command line option).
Number of CPUs to use for LITHO (Calibre nmDRC only)	Specifies the number of processors to use for RET and MDP applications in multithreaded (MT) mode (the value for the -turbo_litho command line option).
Hyperscale (DRC, LVS, PERC)	Use the -hyper command line switch. Recommended. The hyperscale option is available for “Multiple Threads,” “All Threads,” and “Distributed (MTflex)” runs.
Hyperscale Compare (LVS)	Use hyperscaling during the comparison stage of Calibre nmLVS. (-hyper cmp)
Hyperscale PathChk (LVS)	Use hyperscaling during the pathck stage of Calibre nmLVS. (-hyper patchk).
Halt <app> if licenses cannot be obtained for number of CPUs specified	As described. Adds the -turbo_all command line option. Only available for “Multiple Threads” and “All Threads” runs.
Number of hierarchical LVS licenses for PERC (PERC only)	Specifies the “-lvs_supplement num_processors” command line option, which uses Calibre nmLVS-H licenses for the circuit extraction portion of a Calibre PERC run.  This option is only available for multithreaded and Calibre MTflex runs without “Halt PERC if licenses for number of CPUs cannot be obtained” selected.
Remote Data Server (RDS) (DRC, LVS, PERC)	Use the -remotedata command line switch. Choose <b>Default</b> (recommended) or <b>Specify</b> for the number of remote data servers.  Available for Calibre MTflex runs.
Backup Data	Use the -recoverremote command line switch to specify recovery from a remote host failure. This option is valid only when “Remote Data Server” is enabled.
Hyperscale Remote	Use the “-hyper remote” command line switch and option to improve the performance and scaling of hyperscaling. This option is valid only when “Remote Data Server” and “Hyperscaling mode” are enabled.  In Calibre nmLVS, the “Hyperscale Remote” option is not available with “Hyper Compare.”

**Table 12-3. Run Calibre Execution Options (cont.)**

Option	Description
Connect to Remote Hosts (DRC only)	<p>Specify when to connect to remote hosts in HDB (hierarchical database) construction. Only available when the processing type is set to Distributed (MTflex).</p> <ul style="list-style-type: none"><li>• At later stages of HDB construction — Uses the “-hdbflex” command line argument to specify that most of the HDB construction is to be performed by the local host, which does not connect to remotes until the later stages of HDB construction.</li><li>• In parallel with HDB construction — Uses the “-hdbflex_acquire” command line argument to specify to acquire resources in parallel with the HDB construction process.</li></ul>

## Configuring for Distributed (MTflex) Execution

Calibre® MTflex™ allows you to use your existing network as a low-cost distributed computing platform.

For more information about Calibre MTflex, see “[Calibre MTflex Processing](#)” in the *Calibre Administrator's Guide*.

### Prerequisites

- Configure your environment to allow rsh to execute your Calibre job remotely. This may include creating and editing a .rhosts file in your home directory.  
You can also specify to use a remote shell other than rsh, as described in “[Configuring the Remote Environment](#)” on page 443.
- (Optional) If necessary, set one of the following environment variables to prevent “Word too long” errors from the remote host. This error is seen when the command line length exceeds the limit set in the shell environment. By default, Calibre Interactive uses the C shell (/bin/csh) to launch the Calibre MTflex primary script. You can specify the Bash shell instead, which allows a longer command line, by defining one of the following environment variables:
  - CALIBRE\_CI\_REMOTE\_COMMAND\_USE\_BASH  
When set, use the Bash shell to launch the Calibre MTflex primary script.
  - CALIBRE\_CI\_REMOTE\_COMMAND\_LSF\_USE\_BASH  
When set, use the Bash shell to launch the Calibre MTflex primary script for Spectrum LSF runs only.

In most cases the long command line is due to a long \$path variable.

## Procedure

1. Click **Run Control** on the left panel of Calibre Interactive.
2. Select the host for Calibre execution in the “Run Calibre on” dropdown list. See “[Specifying the Host Processor and Run Options](#)” on page 435.
3. Select “Distributed (MTflex)” in the “Run Calibre <app> using” dropdown list.
4. Select Hyperscale, Remote Data Server, and other “Run Calibre” options.

These options are described in “[Specifying the Host Processor and Run Options](#)” on page 435.

“Hyperscale” is recommended. “Hyperscale Remote” is recommended for large designs. (These options are not available for Calibre Interactive PEX and xACT.)

---

### Note

 For Calibre MTflex runs, Calibre Interactive does not include the CPU count argument to the -turbo or -turbo\_litho command line arguments. This is the recommended invocation and results in Calibre using the maximum number of CPUs available.

---

5. Configure the remote hosts, depending on your selection for “Run Calibre <app> on”.
  - Running on Local Host or Remote Host — Choose the selection method for remote hosts in the Remote Host dropdown list:
    - **Specify from GUI** — Specify hosts using the Remote Hosts table. This is the default, and described in the next step.
    - **Specify File** — In “Remote Host File”, provide a configuration file to specify the Calibre MTflex remote hosts. The file is used as the parameter for the [-remotefile](#) command line argument.

For more information, see [-remotefile](#) and “[About the Configuration File](#)” in the *Calibre Administrator’s Guide*.
  - Running on Platform LSF — Fill in the Primary Host and Remote Hosts Selection tables. See “[Running With Spectrum LSF](#)” on page 440.
6. If you chose “Specify from GUI” for Remote Host, fill in the Remote Hosts table.

Add and delete hosts with the + and × buttons. The Use column specifies whether to use the host in the run. The columns in the table depend on the selection for “Run Calibre on”:

  - **Local Host or Remote Host** — The columns correspond to the parameters in the Remote Host statement; see “[REMOTE HOST](#)” in the *Calibre Administrator’s Guide*.

For Remote Host runs, if there is no entry in the RSH column, then the remote shell specified in the Remote Calibre Environment area is used for communication with the remote host. See “[Configuring the Remote Environment](#)” on page 443.

- **Platform LSF** — The columns correspond to the parameters in the Launch Cluster statement; see “[LAUNCH CLUSTER](#)” in the *Calibre Administrator’s Guide* and “[Running With Spectrum LSF](#)” on page 440.

---

**Note**

 When running DRC, if the Connect to Remote Hosts option is enabled, you must specify an explicit number for the CPUs column in the Remote Hosts table; you can not specify “All”.

---

## Running With Spectrum LSF

You can specify that your Calibre Interactive job runs on the IBM Spectrum LSF computing system.

Spectrum LSF (Load Sharing Facility) is a distributed computing solution and requires a commercial license. For complete details on configuring this environment, refer to the product documentation.

### Prerequisites

- Spectrum LSF must be installed on the host machine and on every machine that you may potentially query or submit jobs to.
- For runs using remote hosts, configure your environment to allow rsh to execute your Calibre job remotely. This may include creating and editing a .rhosts file in your home directory.

### Procedure

1. Click **Run Control** on the left panel of Calibre Interactive.
2. For “Run Calibre <app> on”, choose Platform LSF from the dropdown list.
3. Specify the “Submission Host” if it is not the same host that Calibre Interactive is running on.

If localhost is specified as the submission host, then remote submission does not take place, and the LSF job is submitted by localhost.

The submission command obeys the remote shell and remote user name settings in the Remote Calibre Environment area.

If the environment variable LSF\_ENVDIR is defined in the localhost environment or the Calibre Interactive session, then the following line is added to the script executed by Calibre Interactive:

```
. $LSF_ENVDIR/profile.lsf
```

so that LSF settings are available in the submission host.

4. Choose the setting for “Interactive Job Submission”:

- **Enabled** — (default) Submit the Spectrum LSF job with the -I option.

```
bsub -I <options>
```

- **Disabled** — Submit the Spectrum LSF job without the -I option.

```
bsub <options>
```

5. Choose Run Calibre options for your job. For details refer to [Table 12-2](#) in the topic “[Specifying the Host Processor and Run Options](#)” on page 435.

In particular, for multithreaded runs, you can specify “Synchronize Min CPUs in Host Selection table” to force Min CPUs in the host selection table to be the same as the number of CPUs specified for “Number of CPUs to use.” For DRC and DFM runs, the Min CPUs value is set to the greater of the “Number of CPUs to use” and “... for LITHO” values.

6. Specify host information.

Calibre Interactive displays the Host Selection table for multithreaded execution. For Calibre MTflex mode, a Primary Host Selection table and a Remote Hosts Selection table are displayed. The columns correspond to the parameters in the Launch Cluster statement; see “[LAUNCH CLUSTER](#)” in the *Calibre Administrator’s Guide*.

If you select Platform LSF and the LSF resource is unavailable, Calibre Interactive prints an error message to the shell and the Matching Hosts column displays 0.

- a. Use the Min CPUs, Type, and Memory columns in the Host Selection table, as shown in the following table, to define the filter criteria that you want used to determine the list of available hosts on your network. The table shows the columns displayed for multithreaded mode; a slightly different set of columns is displayed for Calibre MTflex mode.

**Table 12-4. Resource Option Variables for Spectrum LSF**

Column Name	Filters on	Corresponding Substitution Variable
Min CPUs	Minimum number of available CPUs for each host.  This entry cannot be edited if “Synchronize Min CPUs in Host Select table” is checked.	%c
Type	Machine type.	%o
Memory	Minimum amount of memory available on each host in megabytes.	%m
N/A	Primary host name. (Only available as a resource option for a remote host.)	%M

- b. After you enter your criteria in these three columns, click in the Matching Hosts table cell and click the  button to query for hosts that match your criteria. The number of matching hosts is displayed in the Matching Hosts column.

Click the  icon to view additional information about available hosts.

- c. Click in the Resource Options cell and enter any additional custom resource options that you want in addition to those already specified in the first three columns of the table. These options represent arguments for the Spectrum LSF resource requirement switch, -R.

If the Resource Options field is not specified, the host is selected based on the Min CPUs, Type, and Memory columns only; this is the default behavior.

- d. Enter options unrelated to resource settings in the Submit Options cell. For information on possible options, refer to the Spectrum LSF product documentation.

For multithreaded (MT) runs only, you can use the %b parameter in the Submit Options column. %b is substituted with the setting for “Number of CPUs to use” in the Run Calibre area; “All” is substituted for %b if “Run Calibre <app> using” is set to All Threads. The %b parameter can be used so that the number of CPUs used for the job matches what is specified for the Calibre run with the -turbo option.

When the number of CPUs is specified, the Spectrum LSF job is submitted using the -n option for both primary and remote hosts to obtain an even distribution of CPUs. The -n option should not be used in the Submit Options cell.

7. Expand the Remote Calibre Environment area and configure the remote options appropriate for your run. See “[Configuring the Remote Environment](#)” on page 443.

8. (Optional) To view the Spectrum LSF configuration, Ctrl-click the **Run <app>** button.  
The bsub command line is displayed in the Transcript page.

## Results

The LAUNCH CLUSTER command is used to specify connection parameters; see “[LAUNCH CLUSTER](#)” in the *Calibre Administrator’s Guide*.

# Configuring the Remote Environment

When running Calibre on a remote host, you can use Calibre Interactive to set parameters such as executable paths and environment variables that control remote host execution.

Calibre Interactive makes the values of the remote user name, the MGC\_HOME path, and the license file available as parameters for all Calibre MTflex operations.

## Prerequisites

- Configure your environment to allow rsh to execute your Calibre job remotely. This may include creating and editing a .rhosts file in your home directory.  
You can also specify to use a remote shell other than rsh, as described in this procedure.
- Configure Calibre to run on the remote computer. On the Run Control page, for “Run Calibre <app> on,” choose Remote Host, Other Cluster, or Platform LSF.

## Procedure

1. Click **Run Control** on the left panel of Calibre Interactive.
2. For “Run Calibre <app> on,” choose Remote Host, Other Cluster, or Platform LSF.  
This displays the “Remote Calibre Environment” section at the bottom of the Run Control page.  
Make other run control selections as described in “[Specifying the Host Processor and Run Options](#)” on page 435.
3. Expand the “Remote Calibre Environment” area at the bottom of the Run Control page.
4. For “Remote MGC\_HOME,” specify the value of MGC\_HOME on the remote host.  
Specify one of the following for “Use”:
  - **Local** — (Default) Use the same version of Calibre that is specified for Calibre runs on the local host. (The value is displayed for each platform.)
  - **Specify** — Use a user-specified MGC\_HOME. Specify the paths in the entries for each platform.
5. For “Remote MGC\_LIB\_PATH,” specify the value of MGC\_LIB\_PATH on the remote host.

Specify one of the following for “Use”:

- **Local** — Use the value that is specified for the local host.
- **Specify** — Use a user-specified MGC\_LIB\_PATH. Specify the paths in the entries for each platform.

6. Specify the remote shell with the “Use” dropdown list.

- **Default** — Uses rsh, which may require a password or be otherwise restricted.
- **Secure** — Uses ssh, which uses host fingerprinting and key pairs. Creation of an ssh private key is recommended. When using a machine without rsh, ssh is automatically chosen.

---

**Note**

 SSHASKPASS is used for ssh authentication. When using ssh, make sure the openssh-askpass package is installed; it is included with ssh. Without this package installed, host verification fails when the host is not in the known\_hosts file and there is no ssh configuration file.

---

- **Specify** — Uses a user-defined shell. When selected, enter the path to the shell.

---

**Note**

 For Calibre MTflex configurations with “Run Calibre on” specified as Local Host or Remote Host, the remote shell used for communication with each remote host is specified in the RSH column of the Remote Hosts table. If there is no entry in the RSH column, the remote shell specified with the Remote Shell entry is used. See “[Configuring for Distributed \(MTflex\) Execution](#)” on page 438.

For Calibre MTflex configurations with “Run Calibre on” specified as Local Host or Remote Host, the remote shell used for communication with each remote host is specified by the RSH keyword in the LAUNCH AUTOMATIC or REMOTE HOST statement. If RSH is not specified, the remote shell specified in the GUI is used.

---

7. Specify the remote user name with the “Use” dropdown list.

- **Current** — Use the current user name.
- **Specify** — Specify a user name.

8. Expand the “Calibre Licensing” area to set remote licensing options.

See the section “[Mentor Standard Licensing \(MSL\)](#)” in the *Calibre Administrator’s Guide* for information on the license file.

- a. Choose the License File Name.
- b. In the Use dropdown list, choose “Current” to use the current license value, or “Specify” to use a user-specified license value.

- c. If you chose “Specify” in the previous step, enter the license in the License File Value text field.

See “[Setting Licensing Options in Calibre Interactive](#)” on page 445 for information on other licensing options.

## Setting Licensing Options in Calibre Interactive

You can control licensing options using Calibre Interactive.

---

### Note

 If you are using Calibre Interactive xACT, the only licensing options available are those for licensing on remote hosts.

---

### Procedure

1. Select the Run Control page.
2. Expand the “Calibre Licensing” section.
3. Enable or disable license queue behavior with the “Licensing Modes” checkbox.

If enabled, select one of the following options:

- **Disable Queuing** — (-nowait) The Calibre run is aborted if a license is not immediately available.
- **Wait for License** — (-wait <time>) The run is placed into a queue if a license is not immediately available. Specify the wait time in minutes in the “Wait time for license” entry.
- **Retry for License** — (-lmretry loop) Retry, or loop, until a license is available, subject to the following options:
  - Retry time for license (minutes) (MAXRETRY:<minutes>)
  - Wait time between retries (seconds) (INTERVAL:<seconds>)
  - Retry even when license errors occur (RETRY\_ON\_ERROR)

When the “Licensing Modes” checkbox is disabled, the Calibre command line does not include any licensing arguments.

4. Choose application-dependent license options:
  - Calibre nmDRC and Calibre nmLVS

- **Use Calibre-CB license for RVE** — This option is available if you selected a run type of “Calibre-CB” on the Inputs page for Calibre Interactive nmDRC or Calibre Interactive nmLVS.
- **PEX**

Use these options if you are performing a flat extraction run. The Calibre CB license options for Calibre nmLVS and Calibre xRC should be selected if SOURCE names are requested in a PEX run of Calibre Interactive and you only have the Calibre Cell/Block (CB) license.

- **Use Calibre-xRC CB license** — Can only be used for flat transistor or gate-level extraction, and on designs containing up to 100,000 transistors.
- **Use Calibre-CB license for LVS** — The -flatten command line option is used in the Calibre nmLVS invocation.
- **Use Calibre-CB license for RVE** — As described.
- **Run hierarchical version of Calibre-LVS** — (default) Run Calibre nmLVS with the -hier command line option. If unchecked, the -flatten command line option is used for the Calibre nmLVS. This setting is ignored if “Use Calibre-CB license for LVS” is selected.

Also see “[Extracting a Block Using CB](#)” in the *Calibre xRC User’s Manual*.

5. If needed, set licensing options for remote hosts in the “Remote Calibre” section.

See the section “[Mentor Standard Licensing \(MSL\)](#)” in the *Calibre Administrator’s Guide* for information on the license file.

- a. Choose the License File Name.
- b. In the Use dropdown list, choose “Current” to use the current license value, or “Specify” to use a user-specified license value.
- c. If you chose “Specify” in the previous step, enter the license in the License File Value text field.

See “[Configuring the Remote Environment](#)” on page 443 for other options related to setting up the remote environment.

# Reference for Run Control Options

---

This section provides detailed information on run control options such as hyperscaling, backup data, and remote data server (RDS).

<b>Hyperscaling Mode .....</b>	<b>447</b>
<b>Hyperscale Compare Option in Calibre Interactive nmLVS .....</b>	<b>447</b>
<b>Hyperscale Pathchk Option .....</b>	<b>447</b>
<b>Remote Data Server (RDS) Option .....</b>	<b>448</b>
<b>Hyperscale Remote Option .....</b>	<b>448</b>
<b>Backup Data Option .....</b>	<b>448</b>

## Hyperscaling Mode

Hyperscaling is a mode of multithreaded operations that improves performance and scalability of CPU usage. It is enabled through the “Hyperscale” option on the **Performance** tab of the run Control pane. It enables the same functionality as the -hyper command line switch for the Calibre batch tools.

Hyperscaling functions in both multithreaded and distributed (Calibre MTflex) modes. You should use hyperscaling unless you have good reason not to. It requires no extra licenses other than what is needed for MT or Calibre MTflex execution. Additional details are available in the *Calibre Administrator’s Guide* and the *Calibre Verification User’s Manual*.

The hyperscale option is not available for PEX runs in Calibre Interactive.

## Hyperscale Compare Option in Calibre Interactive nmLVS

When “Hyperscale Compare” is enabled, Calibre nmLVS-H runs certain portions of LVS comparison concurrently with LVS circuit extraction. This reduces the run time for a hierarchical LVS run that requires both extraction and comparison.

Performance tests with this feature have reduced run times by up to 20 percent, while requiring up to 100 percent more memory. See “[LVS Comparison](#)” and “[Hierarchical Comparison with Hyperscaling Transcript](#)” in the *Calibre Verification User’s Manual* for more information.

## Hyperscale Pathchk Option

When “Hyperscale PathChk” is enabled, Calibre nmLVS-H runs certain portions of LVS comparison concurrently with Pathchk layer operations in hierarchical databases. This option is useful if Pathchk operations take the majority of time in a hyperscaling run. This option increases memory usage.

## Remote Data Server (RDS) Option

The Calibre Remote Data Server (RDS) technology is designed to reduce local (primary) memory requirements and improve Calibre MTflex performance by distributing data across remote hosts.

Remote Data Server is only available for Calibre nmDRC, Calibre nmLVS, Calibre DFM, Calibre xACT, and Calibre PERC runs; it is not available for FastXOR runs. It uses the `-remotedata` command line option.

## Hyperscale Remote Option

Hyperscale Remote is an option that improves the performance and scaling of hyperscaling. Hyperscale Remote is available in “Distributed (MTflex)” mode when the Remote Data Server (RDS) Option is enabled. These options are not available for Calibre Interactive PEX.

Hyperscale Remote mode is most useful for large designs that require more than one hour to complete. A minimum configuration of four remote hosts with 24 remote CPUs connected to a local (primary) host is required; scaling can continue beyond 100 CPUs. Additional details are available in the [Calibre Administrator’s Guide](#) and the [Calibre Verification User’s Manual](#).

## Backup Data Option

The Backup Data option enables recovery in the event of a catastrophic remote host failure; it uses the `-recoverremote` command line option. Backup Data is available in “Distributed (MTflex)” mode when the Remote Data Server (RDS) Option is enabled. These options are not available for Calibre Interactive PEX.

# Appendix A

## Interfacing Calibre Interactive to Layout and Schematic Viewers

---

The ability to communicate with industry standard design tools is an important feature of Calibre Interactive. This chapter discusses how to use Calibre Interactive with some of the design tool integrations. It also explains the setup procedure for the design tools that require custom installation and discusses special operating considerations.

The interface to several of the design tools is made automatically and requires no special setup actions or instruction. Such tools are not discussed in this chapter. Some design tools may require you to turn on toolbar or menu visibility. The complete list of design tools with a Calibre integration is given in the section “[Integration to Design Tools](#)” on page 25. For most design tools, the Calibre integration enables the interface to both Calibre RVE and Calibre Interactive.

Siemens products have a Calibre or Verification menu by default, and you use it to access Calibre Interactive and Calibre RVE. Siemens layout viewers that are not mentioned explicitly in this chapter operate similarly to Calibre DESIGNrev.

The following topics and design tools are covered in this chapter:

<b>Basic Interface</b> .....	<b>450</b>
<b>Custom Menu Items in Design Tools</b> .....	<b>453</b>
<b>Calibre DESIGNrev and WORKbench</b> .....	<b>465</b>
<b>IC Station/Pyxis</b> .....	<b>467</b>
<b>Siemens Tanner</b> .....	<b>468</b>
<b>Cadence Virtuoso</b> .....	<b>469</b>
<b>Cadence Composer</b> .....	<b>558</b>
<b>Cadence Encounter</b> .....	<b>559</b>
<b>Cadence Innovus</b> .....	<b>562</b>
<b>Synopsys IC Compiler and Synopsys IC Compiler II</b> .....	<b>564</b>
<b>Silvaco Expert</b> .....	<b>576</b>
<b>Other Viewer Integrations</b> .....	<b>577</b>

## Basic Interface

---

Some interface concepts and actions are common to all integrated layout and schematic viewers, as explained in these sections:

<b>Calibre Interactive.....</b>	<b>450</b>
<b>Setting the Socket Port in Calibre Interactive .....</b>	<b>451</b>

## Calibre Interactive

To start Calibre Interactive from an integrated viewer, choose the desired interface from the menu, depending on your viewer.

Typical menu selections are the following:

- Calibre DESIGNrev family of viewers — **Verification** menu.
- Other viewers — **Calibre** or **Calibre > Verification** menu.

Calibre Interactive communicates with the layout viewer to enable automatic export of GDSII layout databases. Calibre Interactive imports the Primary Cell according to the setting on the Inputs pane. By default, the currently open cell in the layout viewer is selected for import.

A communications socket is set up automatically when you invoke Calibre Interactive through the layout viewer. The default socket number is 9189. See “[Setting the Socket Port in Calibre Interactive](#)” on page 451 for instructions on manually setting a socket number.

- 
- Tip**  You can define the templates Calibre Interactive uses to initialize GUI field values when Calibre Interactive is started from a layout viewer. Template settings are saved as part of the runset. Refer to “[Templates for File Naming](#)” on page 100 for information on templates.
-

## Setting the Socket Port in Calibre Interactive

Calibre Interactive uses certain default port numbers for socket communication with a connected layout or schematic viewer. The port numbers can also be set manually, as described here. Individual layout and schematic viewers may have additional methods for setting the port numbers; see the section for the specific viewer.

- Setting the Calibre Interactive Socket Port for a Layout Viewer .....** [451](#)  
**Setting the Calibre Interactive Socket Port for a Schematic Viewer .....** [452](#)

### Setting the Calibre Interactive Socket Port for a Layout Viewer

You can set the socket port manually using the Calibre Interactive GUI, or using an environment variable. By default, Calibre Interactive uses socket port 9189 to communicate with the layout viewer. The tool automatically searches for an available socket if it cannot obtain the default socket. It initializes the first available socket between the numbers 5000 and 9999 and reports the socket number through a dialog box.

Some layout viewers, such as Calibre DESIGNrev, also have a GUI interface to set the socket port—see the instructions for the specific viewer.

#### Prerequisites

- Calibre Interactive must be open if you are setting the socket number using the GUI.

#### Procedure

Set the layout socket port with one of the following methods:

To set the layout socket port with ...	Do this ...
Calibre Interactive	<ol style="list-style-type: none"><li>1. Open the <b>Run Control</b> page and open the Design Tool Settings section.</li><li>2. Enter the port number in the “Socket Number” field.</li><li>3. Click <b>Connect</b>.</li></ol>
Environment variable	<p>Set the MGC_CALIBRE_LAYOUT_SERVER environment variable as shown here for the C-Shell:</p> <pre>setenv MGC_CALIBRE_LAYOUT_SERVER [&lt;host&gt;:]&lt;port&gt;</pre> <p>The &lt;host&gt; parameter is set to localhost by default, but can be set to a different host. The &lt;port&gt; parameter must be a numerical value.</p> <p>To disable socket communication to the layout viewer, set MGC_CALIBRE_LAYOUT_SERVER to “ ” (blank). The layout server then does not attempt to open the communication socket.</p>

## Related Topics

[Connection to a Design Tool \[Calibre RVE User's Manual\]](#)

[Setting the Calibre RVE Socket Port for a Layout Viewer \[Calibre RVE User's Manual\]](#)

# Setting the Calibre Interactive Socket Port for a Schematic Viewer

You can set the socket port manually using the Calibre Interactive GUI, or using an environment variable. By default, Calibre Interactive uses socket port 9199 to communicate with the schematic viewer. The tool automatically searches for an available socket if it cannot obtain the default socket. It initializes the first available socket between the numbers 5000 and 9999 and reports the socket number through a dialog box.

## Prerequisites

- Calibre Interactive must be open if you are setting the socket number through the GUI.

## Procedure

Set the schematic viewer socket port with one of the following methods:

To set the schematic socket port with ...	Do this ...
Calibre Interactive	<ol style="list-style-type: none"><li>1. Open the <b>Run Control</b> page and open the Design Tool Settings section.</li><li>2. Enter the port number in the “Socket Number” field.</li><li>3. Click <b>Connect</b>.</li></ol>
Environment variable	<p>Set the MGC_CALIBRE_SCHEMATIC_SERVER environment variable as shown here for the C-Shell:</p> <pre>setenv MGC_CALIBRE_SCHEMATIC_SERVER [&lt;host&gt;:]&lt;port&gt;</pre> <p>The &lt;host&gt; parameter is set to localhost by default, but can be set to a different host. The &lt;port&gt; parameter must be a numerical value.</p> <p>To disable socket communication to the schematic viewer, set MGC_CALIBRE_SCHEMATIC_SERVER to “ ” (blank). The schematic server then does not attempt to open the communication socket.</p>

## Related Topics

[Connection to a Design Tool \[Calibre RVE User's Manual\]](#)

[Setting the Calibre RVE Socket Port for a Schematic Viewer \[Calibre RVE User's Manual\]](#)

## Custom Menu Items in Design Tools

You can add custom, or user-defined, menu items to supported design tools. The custom menu items are added to the integration menu.

The supported design tools are given in the following table.

**Table A-1. Supported Design Tools for Custom Menu Items**

Design Tool	Integration Menu
Calibre DESIGNrev	Verification
Cadence Virtuoso	Calibre
Cadence Encounter	Calibre
Synopsys IC Compiler	Calibre

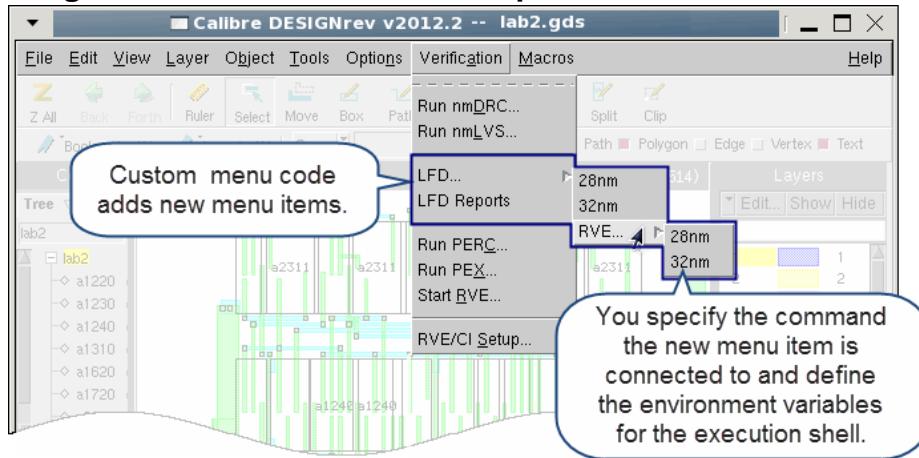
The standard items in the Calibre integration menu open Calibre Interactive, Calibre RVE, or the Calibre setup dialog box. With menu customization you can add menu items to do the following:

- Open Calibre Interactive with a particular runset
- Open Calibre RVE to a particular results database
- Start a batch Calibre run
- Run a custom script
- Specify environment variables that are in effect for the command that is executed with the menu selection

[Templates for File Naming](#) are applied when you invoke Calibre Interactive from a custom menu command.

A custom menu example is shown in [Figure A-1](#) for Calibre DESIGNrev. The menu customization file for this example is given in “[Menu Customization Example](#)” on page 463.

**Figure A-1. Custom Menu Example for Calibre DESIGNrev**



The menu customization file is written in the Tcl language and a set of menu customization commands is provided. The following topics describe how to specify the menu customization file, the file format, and the available commands:

<b>Creating Custom Menus in Design Tools .....</b>	<b>454</b>
<b>Menu Customization File .....</b>	<b>455</b>
<b>Menu Customization Commands.....</b>	<b>456</b>
<b>Calibre Integration Menu Labels and Backward Compatibility for Menu Customization Files .....</b>	<b>462</b>
<b>Menu Customization Example .....</b>	<b>463</b>

## Creating Custom Menus in Design Tools

The following procedure describes how to add custom menu items to the supported design tools.

See “[Custom Menu Items in Design Tools](#)” on page 453 for a list of the supported design tools.

### Procedure

1. Write the menu customization file using the syntax and commands described in “[Menu Customization File](#)” on page 455 and “[Menu Customization Commands](#)” on page 456.

The following simple example adds one menu item which starts a Calibre Interactive batch run:

```
# This menu customization file adds a menu item to
# start a Calibre Interactive DRC run in batch mode

mgc_calibre_add_menu_item -type command -after "Run nmDRC" \
-command_type DRC -label "Run DRC GUI batch" \
-command "calibre -gui -drc -runset drc_runset -batch"
```

2. Set the environment variable MGC\_CALIBRE\_VIEWER\_MENU\_CMDS to the pathname for the menu customization file.
3. If desired, set MGC\_CALIBRE\_ECHO\_VIEWER\_MENU\_CMDS to echo custom menu commands to the layout viewer's transcript.
4. If desired, set MGC\_CALIBRE\_VIEWER\_MENU\_CMDS\_SINGLE\_TOOL to make custom menu items behave the same as built-in menu items when selected multiple times. When set, custom menu items redirect you to a previous window instead of opening a new window.
5. Start your design tool.

## Results

The new menu items are included in the **Verification** or **Calibre** menu when the design tool starts up.

## Related Topics

[Menu Customization File](#)

[Menu Customization Commands](#)

[Menu Customization Example](#)

# Menu Customization File

Input for: [Custom Menu Items in Design Tools](#)

The menu customization file must be valid Tcl syntax.

The following syntax rules apply:

- Comment lines start with '#'.
- Characters special to Tcl which appear in string literals, such as '\$', must be escaped.
- Blank lines are ignored.

The location of the menu customization file is specified with the environment variable MGC\_CALIBRE\_VIEWER\_MENU\_CMDS.

## Related Topics

[Tcl Library in Calibre \[Calibre Administrator's Guide\]](#)

## Menu Customization Commands

Several commands are available for creating custom menus in supported design tools.

The syntax conventions used in the command descriptions are given in [Table 1-4](#) on page 30.  
The following menu customization commands are supported:

Command	Description
<a href="#">mgc_calibre_get_viewer_name</a>	Returns the name of the design tool reading the menu customization file.
<a href="#">mgc_calibre_add_menu_item</a>	Adds a menu item to the indicated menu (by default, the root <b>Calibre</b> or <b>Verification</b> menu). Pre-execution code that executes in the design tool environment can also be specified for a command menu item.
<a href="#">mgc_calibre_delete_menu_item</a>	Removes an item from the indicated menu (by default, the root <b>Calibre</b> or <b>Verification</b> menu).

Also see “[Calibre Integration Menu Labels and Backward Compatibility for Menu Customization Files](#)” on page 462.

## **mgc\_calibre\_get\_viewer\_name**

Returns the name of the design tool reading the menu customization file.

### **Usage**

**mgc\_calibre\_get\_viewer\_name**

### **Arguments**

None.

### **Description**

The command returns the value of the MGC\_CALIBRE\_LAYOUT\_SERVER\_NAME environment variable set by the design tool. The possible values are the same as those allowed for the runset variable cmnLayoutServerType. The list of supported design tools for menu customization and the corresponding return value for mgc\_calibre\_get\_viewer\_name is given in the following table.

**Table A-2. Return Value for mgc\_calibre\_get\_viewer\_name**

Design Tool	Return Value
Calibre DESIGNrev	Mentor:Calibre
Cadence Virtuoso	Cadence:Virtuoso
Cadence Encounter	Cadence:Encounter
Synopsys IC Compiler	Synopsys:ICCompiler

### **Examples**

The command may be used to combine viewer-specific menu customization code within a single file in the following manner:

```
if {[mgc_calibre_get_viewer_name] eq "Mentor:Calibre"} {  
    # DesignRev-specific menu modifications  
} else {  
    # modifications for some other viewer  
}
```

## **mgc\_calibre\_add\_menu\_item**

Adds a menu item to the indicated menu (by default, the root **Calibre** or **Verification** menu). Pre-execution code that executes in the design tool environment can also be specified for a command menu item.

### **Usage**

Add menu item tied to a command

```
mgc_calibre_add_menu_item -type command -label "label"  
  -command "command"  
  -command_type { DRC | LVS | DFM | PEX | PERC | XACT | RVE | CUSTOM }  
  [-code "viewer_code"]  
  [-view_type {layout | schematic}]  
  [-after "label" | -before "label"] [-parent "label"]  
  [{-env "name=value"} ...] [{-unsetenv "name"} ...]
```

Add parent menu item

```
mgc_calibre_add_menu_item -type menu -label "label"  
  [-view_type {layout | schematic}]  
  [-after "label" | -before "label"] [-parent "label"]
```

Add separator line

```
mgc_calibre_add_menu_item -type separator  
  [-view_type {layout | schematic}]  
  [-after "label" | -before "label"] [-parent "label"]
```

### **Arguments**

- **-type {command | menu | separator}**

A required argument and parameter which specifies the type of the menu item, where the allowed types are described as follows:

- **command** — Adds a menu item connected to a command.
- **menu** — Adds a menu item, such as a parent menu, not connected to a command. The **-command**, **-command\_type**, and **-env** parameters are not used.  
  
This option is not available for Synopsys IC Compiler. Use “-type command” with the **-parent** option to specify a parent menu; the parent menu is created if it does not exist.
- **separator** — Adds a separator line.

- **-label "label"**

The **-label** parameter indicates the label for the new menu item.

You may place the ampersand character (&) before a character in the menu label to indicate that it is a keyboard shortcut and should be underlined. The ampersand is not required when

referring to the label in the -before or -after option. This feature is not supported in Cadence Virtuoso versions prior to 6.0.

- **-command “*command*”**

Specifies the path to a command to be invoked when the menu item is selected, including parameters.

Any token beginning with '\$' is taken to be the name of an environment variable, delimited by any of '/', '#', ':', or''. If it matches any token given as a name in a -env parameter, the value of the environment variable specified in the -env parameter will replace it when the command is invoked.

By default, when custom menu items are selected multiple times, a new window appears instead of redirecting you to the previously opened window. If desired, set the environment variable MGC\_CALIBRE\_VIEWER\_MENU\_CMDS\_SINGLE\_TOOL to make custom menu items behave the same as built-in menu items.

See the description section for important information about invoking Calibre RVE.

- **-code “*viewer\_code*”**

Optional argument and parameter that specifies code to execute in the viewer environment before invoking the command tied to the menu item. See “[Example 3](#)” on page 461.

The -env and -unsetenv options are not applied during execution of *viewer\_code*. If the **command** is an empty string (-command "") or -command is omitted, only the *viewer\_code* is executed.

- **-command\_type { DRC | LVS | DFM | PEX | PERC | RVE | XACT | CUSTOM }**

Specifies the type of command being issued so that the correct Calibre Interactive cell, library, and view templates are used.

- **DRC, LVS, DFM, PEX, PERC, XACT, or RVE** — Command is for the indicated Calibre Interactive application or for Calibre RVE.
- **CUSTOM** — Command is of arbitrary type.

- **-view\_type {layout | schematic}**

Optional argument and parameter specifying the viewer type for Cadence Virtuoso. The default is layout if this option is not specified.

- **{-after “*label*” | -before “*label*”}**

Optional argument and parameter which specifies the insertion method and insertion location for the new menu item. If neither argument is present, the new menu item is added at the end of the parent menu, or at the end of the root Calibre menu if -parent is not specified.

The *label* parameter must match the label of an existing menu item; it is not necessary, however, to include the trailing ellipses (...). The *label* parameter is case-sensitive. If *label* is not found, the new menu item is added at the end of the parent menu, or at the end of the root Calibre menu if -parent is not specified.

“Run DRC” and “Run LVS” are allowed abbreviations, as explained in “[Calibre Integration Menu Labels and Backward Compatibility for Menu Customization Files](#)” on page 462.

- **-env “name=value”**

An optional argument and parameter which adds a name/value pair to the environment of the invoked command. It may be specified multiple times.

If MGC\_HOME is set with the -env argument, CALIBRE\_HOME should also be set with -env. CALIBRE\_HOME takes precedence if both MGC\_HOME and CALIBRE\_HOME exist in the environment in which the new process is invoked; therefore, if only MGC\_HOME is set with the -env argument, the version of Calibre run in the new process may not be what is expected.

- **-unsetenv “name”**

An optional argument and parameter which unsets the environment variable given by *name* in the environment of the invoked command. It may be specified multiple times.

When used with Cadence Virtuoso, this option is only supported for version 6.1.5 and later.

- **-parent “label”**

An optional argument and parameter that specify which menu the new item is added to. The *label* parameter can specify a hierarchy of labels separated by '/' and must exactly match the label of an existing menu item.

If this argument is omitted, the new item is added to the root **Calibre** or **Verification** menu.

## Description

This command adds a menu item to the indicated menu. If the parent menu (-parent argument) and location (-before and -after arguments) are not specified, the menu item is added at the end of the root **Calibre** or **Verification** menu. If the -before or -after label is specified but not found, the menu item is added at the end of the **Calibre** or **Verification** menu, or the end of the parent menu.

When calling Calibre RVE from a custom menu, the syntax should be the following:

```
\$MGC_HOME/bin/calibre -gui -rve <other_options>
```

where the -gui option is needed when invoking Calibre RVE from another program.

[Templates for File Naming](#) are applied when you invoke Calibre Interactive from a custom menu command.

## Examples

### Example 1

This example adds a menu item that can be used as a parent menu.

```
mgc_calibre_add_menu_item -type menu -after "Run LVS" -label "LFD..."
```

### Example 2

This example adds a menu item that starts Calibre RVE and opens a specific database. The -gui switch is needed when invoking Calibre RVE from another application.

```
mgc_calibre_add_menu_item -type command \
    -parent "LFD..." -label "32nm" \
    -command_type RVE \
    -command "\$MGC_HOME/bin/calibre -gui -rve -dfm 32nmDB"
```

### Example 3

This example specifies code to execute in the Cadence Virtuoso environment before starting Calibre Interactive PEX.

```
# Add a menu item to start Calibre Interactive PEX with a runset.
# Use the -code option to call the SKILL command
# mgc_rve_default_setup_file before executing the command
# tied to menu item.

mgc_calibre_add_menu_item -type command -label "pex_with_code" \
    -command_type PEX -command "calibre -gui -pex -runset adder.runset" \
    -code "mgc_rve_default_setup_file()"
```

### Example 4

This example adds and deletes menu items from the schematic view in Cadence Virtuoso. The label for the menu item uses the ampersand (&) character to designate a keyboard shortcut.

```
mgc_calibre_delete_menu_item -label "Run LVS" -view_type schematic

mgc_calibre_add_menu_item -type command -label "PERC &source based" \
    -view_type schematic -after "Run PERC" \
    -command_type PERC \
    -command "calibre -gui -perc -runset perc.runset.source"
```

## Related Topics

[Menu Customization Example](#)

[Calibre Integration Menu Labels and Backward Compatibility for Menu Customization Files](#)

## mgc\_calibre\_delete\_menu\_item

Removes an item from the indicated menu (by default, the root **Calibre** or **Verification** menu).

### Usage

```
mgc_calibre_delete_menu_item -label "label"  
[-view_type {layout | schematic}] [-parent "label"]
```

### Arguments

- **-label “label”**  
A required argument and parameter which specify the label of the menu item to be deleted.
- **-view\_type {layout | schematic}**  
Optional argument and parameter specifying the viewer type for Cadence Virtuoso. The default is layout if this option is not specified.
- **-parent “label”**  
An optional argument and parameter which specifies the parent menu of the item to remove. The *label* parameter can specify a hierarchy of labels separated by '/' and must exactly match the label of an existing menu item.

If the -parent argument is omitted, the item is deleted from the root **Calibre** or **Verification** menu.

### Examples

```
mgc_calibre_delete_menu_item -label "Run DFM..."
```

### Related Topics

[Calibre Integration Menu Labels and Backward Compatibility for Menu Customization Files](#)

## Calibre Integration Menu Labels and Backward Compatibility for Menu Customization Files

The menu labels for the Calibre integration changed in some design tools with the 2014.2 release. Menu customization files written for previous releases are still read correctly if they use the earlier menu label in the -before and -after options to specify the placement of a custom menu item.

Starting with the 2014.2 release, the **Calibre** or **Verification** menu items for the Cadence Virtuoso, Cadence Encounter, Synopsys IC Compiler, and Calibre DESIGNrev integrations are the same for each design tool. Menu items that were previously “Run DRC” or “Run LVS” are now “Run nmDRC...” or “Run nmLVS...”, respectively. In addition, the order of the menu items is now the same in all of these integrations.

It is not necessary to update menu customization files written prior to the 2014.2 release to correct the menu labels used in the -before and -after options to the `mgc_calibre_add_menu_item` and `mgc_calibre_delete_menu_item` commands. The menu customization process correctly resolves the labels used in earlier releases, as shown in the following table:

These menu labels:	Match this in 2014.2 and later
Run DRC	Run nmDRC...
Run DRC...	
Run nmDRC	
Run LVS	Run nmLVS...
Run LVS...	
Run nmLVS	

In addition, trailing ellipsis marks (...) are ignored when matching labels using the -before and -after options. For example, this command:

```
mgc_calibre_add_menu_item -type separator -before "Start RVE"
```

adds a separator line before “Start RVE...”. Trailing ellipsis marks are also ignored when matching custom menu items added with the `mgc_calibre_add_menu_item` command.

## Menu Customization Example

This customization file works with any supported design tool. It deletes the **DFM** and **3DSTACK** menu items, and adds two new menu items and a submenu.

The results are shown in [Figure A-1](#).

```
# menu customization file
#
mgc_calibre_delete_menu_item -label "Run DFM"
mgc_calibre_delete_menu_item -label "Run 3DSTACK"

# add the LFD... menu item to root menu
# use & to designate F as a keyboard shortcut
mgc_calibre_add_menu_item -type menu -after "Run nmLVS" -label "L&FD..."
mgc_calibre_add_menu_item -type command -parent "LFD..." -label "28nm" \
    -command_type DFM \
    -command "\$MGC_HOME/bin/calibre -gui -dfm -runset 28nmrunset" \
    -env "OA_HOME=/oa/v2"
mgc_calibre_add_menu_item -type command -parent "LFD..." -label "32nm" \
    -command_type DFM \
    -command "\$MGC_HOME/bin/calibre -gui -dfm -runset 32nmrunset" \
    -env "OA_HOME=/oa/v1"

# add RVE submenu to new LFD... menu
mgc_calibre_add_menu_item -type menu -label "RVE..." -parent "LFD..."
mgc_calibre_add_menu_item -type command \
    -parent "LFD.../RVE..." -label "28nm" \
    -command_type RVE \
    -command "\$MGC_HOME/bin/calibre -gui -rve -dfm 28nmDB"
mgc_calibre_add_menu_item -type command \
    -parent "LFD.../RVE..." -label "32nm" \
    -command_type RVE \
    -command "\$MGC_HOME/bin/calibre -gui -rve -dfm 32nmDB"

# add LFD Reports menu item to root menu
mgc_calibre_add_menu_item -type command \
    -label "LFD Reports" -after "LFD" \
    -command_type CUSTOM \
    -command "/local/bin/RunLFDReport /path_to_report_dir"
mgc_calibre_add_menu_item -type separator -before "LFD"
mgc_calibre_add_menu_item -type separator -after "LFD Reports"
```



### Tip

You can omit trailing ellipsis marks (...) from labels specified in the -before and -after options.

---

## Related Topics

[Creating a One-Click Custom Menu Command With Skill Pre-Processing \[Calibre Solutions for Physical Verification\]](#)

[Calibre Integration Menu Labels and Backward Compatibility for Menu Customization Files](#)

## Calibre DESIGNrev and WORKbench

Calibre DESIGNrev and Calibre WORKbench include a Verification menu for starting Calibre Interactive, Calibre RVE, and some Calibre Pattern Matching applications.

The Verification menu includes the following selections:

**Table A-3. Verification Menu in Calibre DESIGNrev and WORKbench**

Item	Description
Run nmDRC	Opens the corresponding Calibre Interactive application.
Run DFM	
Run nmLVS	
Run PERC	
Run PEX	
Run xACT	
Run 3DSTACK	
Run PM Capture	Opens Calibre Pattern Matching applications. See these topics in the <i>Calibre Pattern Matching User's Manual</i> :
Run Pattern Matching	<a href="#">“Capturing Patterns From a Layout”</a> <a href="#">“Running Pattern Matching From a Calibre Layout Viewer”</a>
Start RVE	Starts Calibre RVE.
RVE/CI Setup	Opens a dialog for setting Calibre RVE and Calibre Interactive setup parameters, such as socket number, highlighting layers, highlight line width, and cell mapping.
RealTime	See the <a href="#">Calibre RealTime Custom User's Manual</a> .

You can add custom menu items to the **Verification** menu; see “[Custom Menu Items in Design Tools](#)” on page 453.

<b>Setting Socket Connections with Calibre DESIGNrev or Calibre WORKbench . . . . .</b>	<b>465</b>
<b>Calibre Interactive with Calibre DESIGNrev or WORKbench . . . . .</b>	<b>466</b>

## Setting Socket Connections with Calibre DESIGNrev or Calibre WORKbench

By default, Calibre Interactive and Calibre RVE use socket port 9189 for communication. The tool automatically searches for an available socket if it cannot obtain the default socket.

You can manually specify a port number using the environment variable MGC\_CALIBRE\_LAYOUT\_SERVER, as described in “[Setting the Calibre Interactive Socket Port for a Layout Viewer](#)” on page 451.

You can also specify the socket port by choosing **Verification > RVE/CI Setup** from Calibre DESIGNrev or Calibre WORKbench. Enter the socket number and click **Start**.

## Calibre Interactive with Calibre DESIGNrev or WORKbench

To start Calibre Interactive from Calibre DESIGNrev or Calibre WORKbench, choose Verification and choose the desired interface.

Refer to “[Invoking the Calibre Interactive GUI](#)” on page 58 for more information on how to invoke Calibre Interactive from Calibre DESIGNrev.

You can also define a trigger for Calibre Interactive in Calibre DESIGNrev; see “[External Trigger Definitions in Calibre DESIGNrev](#)” on page 430.

## IC Station/Pyxis

The IC Station product family changed its name to Pyxis with the v10.0 release of Pyxis, and this manual therefore refers to the Pyxis product. Pyxis has a **Calibre** pulldown menu with selections to open Calibre Interactive and Calibre RVE.

The **Calibre** menu in Pyxis is similar to the **Verification** menu for Calibre DESIGNrev, which is shown in [Table A-3](#).

**Setting the software tree** — Calibre Interactive or Calibre RVE is called from in Pyxis by setting the CALIBRE\_HOME variable to the desired software path. Pyxis will search for environment variables as follows:

1. If CALIBRE\_HOME is defined, it is searched for the Calibre tree. If there is no Calibre tree in CALIBRE\_HOME, then the MGC\_HOME defined for the Pyxis session is searched. If still no Calibre tree can be found, you should specify the path to Calibre.
2. If CALIBRE\_HOME is not defined, then the MGC\_HOME defined for the Pyxis session is searched. If there is no Calibre tree in MGC\_HOME, then you should specify the path to Calibre.

**Setting Socket Connections with Pyxis** ..... [467](#)

**Using Calibre Interactive with Pyxis** ..... [467](#)

## Setting Socket Connections with Pyxis

By default, Calibre Interactive and Calibre RVE use socket port 9189 to communicate with Pyxis Layout. The tool automatically searches for an available socket if it cannot obtain the default socket.

You can manually specify a port number using the environment variable MGC\_CALIBRE\_LAYOUT\_SERVER, as described in “[Setting the Calibre Interactive Socket Port for a Layout Viewer](#)” on page 451.

You can also use following command from within Pyxis Layout to specify a different port without resetting the MGC\_CALIBRE\_LAYOUT\_SERVER environment variable and restarting Pyxis Layout:

**mgc\_rve\_init\_socket <port\_number>**

Type the command in Pyxis Layout to initialize the specified socket port for communication.

## Using Calibre Interactive with Pyxis

Select the desired option from the **Calibre** menu to open the corresponding Calibre Interactive application.

## Siemens Tanner

Calibre Interactive is integrated to the viewers Siemens Tanner S-Edit, a schematic viewer, and L-Edit, a layout design tool.

For more information on using these integrations with Calibre, refer to *Tanner and Linux-Based Tools Setup* in the Tanner documentation. For details on specific integrations, see the *L-Edit User's Manual* or the *S-Edit User's Manual*.

# Cadence Virtuoso

---

The following sections describe the Calibre interface to Cadence Virtuoso:

<b>Creating an Interface to Cadence Virtuoso.....</b>	<b>469</b>
<b>The Calibre Menu in Cadence Virtuoso .....</b>	<b>471</b>
<b>Layout Export in Cadence Virtuoso .....</b>	<b>474</b>
<b>Direct Read of Cadence Virtuoso OpenAccess Database.....</b>	<b>482</b>
<b>Netlist Export in Cadence Composer.....</b>	<b>484</b>
<b>Using the Calibre - Virtuoso Interface.....</b>	<b>489</b>
<b>Setting Socket Connections with Cadence Virtuoso.....</b>	<b>491</b>
<b>Calling SKILL Functions from Calibre Interactive.....</b>	<b>492</b>
<b>Creating a Calibre View .....</b>	<b>493</b>
<b>Setting the Bus Delimiter for Schematic Netlists .....</b>	<b>550</b>
<b>Mapping Schematic Nets to Calibre View Nets for Simulation and Plotting.....</b>	<b>552</b>
<b>Backannotation of Parasitic Values to the Schematic in Cadence Virtuoso .....</b>	<b>555</b>
<b>Automating Calibre Interactive Runsets in the Cadence Design Environment .....</b>	<b>556</b>

## Creating an Interface to Cadence Virtuoso

Calibre Interactive and Calibre RVE support Cadence Virtuoso versions 5.1, 6.1, 6.18, 12.1, 12.2, 12.3, and 18.1.

Beginning with the 2013.3 release, Calibre supports OpenAccess version oa22.43p006 by default. OpenAccess version 22.50 can be set with an environment variable. See “[Providing OpenAccess Input with Calibre Interactive](#)” on page 73 and “[FDI Environment Variables](#)” in the *Calibre Layout Comparison and Translation Guide*.

---

### Note

 Cadence Virtuoso 6.18 and 18.1 are supported for layout export and highlighting only. These versions create OpenAccess v22.60 databases, which cannot be read directly by Calibre.

---

### Procedure

1. Set the environment variable CALIBRE\_HOME or MGC\_HOME to point to the Calibre software tree; see “[Setting the CALIBRE\\_HOME Environment Variable](#)” in the *Calibre Administrator’s Guide*. Setting CALIBRE\_HOME is preferred.

Instructions in this section assume that CALIBRE\_HOME is set; substitute MGC\_HOME for CALIBRE\_HOME if that is how your software tree is defined.

2. Depending on whether you use CALIBRE\_HOME or MGC\_HOME to point to the Calibre software tree, make sure that \$CALIBRE\_HOME/tmp or \$MGC\_HOME/tmp exists. The tmp directory is required.

If the tmp directory does not exist, define the environment variable MGC\_TMPDIR to point to a directory that can be used by the installation process.

The installation process searches first for \$CALIBRE\_HOME/tmp or \$MGC\_HOME/tmp, then searches for \$MGC\_TMPDIR if neither of those directories are found. If no tmp directory is found, a warning is issued and the installation process aborts.

3. Review the README file located at

`$CALIBRE_HOME/shared/pkgs/icv/tools/queryskl/skillREADME`

This file provides complete instructions on installing the *calibre.skl* file that resides at the same directory level. (Links to *calibre.skl* and *calibre.4.3.skl* are available at \$CALIBRE\_HOME/lib in the software tree.)

4. Edit your *.cdsinit* file as directed in the *skillREADME* file to automatically load the Calibre integration.

---

**Note**

 Also see “[Requirements and Setup for Calibre RealTime with Cadence Virtuoso](#)” in the *Calibre RealTime User’s Manual* if you are using Calibre RealTime.

---

The **Calibre** menu is added by the SKILL Interface as a user menu trigger. Whenever a layout window (of type maskLayout, maskLayoutXL, maskLayoutGXL, maskLayoutEAD, or maskLayoutEXL) is opened, the user menu trigger is executed. The SKILL Interface then installs any other user menus that may have been defined (either by you or by other applications), and then installs the **Calibre** menu as the last menu.

If other applications also install menus in layout windows, you should load *calibre.skl* after you load these other applications’ SKILL files. If you want to load the **Calibre** menu before other applications’ menus, you need to be sure that the other applications’ SKILL code ensures that the **Calibre** menu also gets loaded.

- You can also load the Calibre integration from the CIW as follows:

Standard installation:

```
load( strcat( getShellEnvVar("CALIBRE_HOME")
    "/shared/pkgs/icv/tools/queryskl/calibre.skl" ) )
```

Cadence Virtuoso 4.3 installation:

```
load( strcat( getShellEnvVar("CALIBRE_HOME")
    "/shared/pkgs/icv/tools/queryskl/calibre.4.3.skl" ) )
```

5. (Optional) Define the window type for the **Calibre** menu if you are not using one of the supported layout views of maskLayout, maskLayoutXL, maskLayoutGXL, maskLayoutEAD, or maskLayoutEXL.

Do this by setting the SKILL variable mgcCalibreMenuViewType to include the other view type before you load *calibre.skl*. This can be done in the *.cdsinit* file as shown in the following code section, where myLayout is the unsupported layout view:

```
mgcCalibreMenuViewType = list("maskLayout" "maskLayoutXL"  
    "maskLayoutGXL" "maskLayoutEAD" "maskLayoutEXL" "myLayout")  
load( strcat( getShellEnvVar("CALIBRE_HOME")  
    "/shared/pkgz/icv/tools/queryskl/calibre.skl" ) )
```

In order to preserve support of the standard view types, define mgcCalibreMenuViewType as a list of supported view types, as shown. The **Calibre** menu is only displayed for the view types defined by mgcCalibreMenuViewType.

---

**Note**

 The mgc\_load\_calibre\_menu() SKILL command is also provided. This command installs the **Calibre** menu in the active window. You can tie this command to a bind key if desired.

---

## Related Topics

- [Basic Interface](#)
- [Custom Menu Items in Design Tools](#)
- [Using the Calibre - Virtuoso Interface](#)
- [The Calibre Menu in Cadence Virtuoso](#)
- [Cadence Virtuoso \[Calibre RVE User's Manual\]](#)

## The Calibre Menu in Cadence Virtuoso

The **Calibre** menu in Cadence Virtuoso layout and schematic windows (windows of type maskLayout or schematic by default) has the selections to start Calibre Interactive, start Calibre RVE, set up options, and map schematic nets.

You can also add custom menu items to the Calibre integration menu; see “[Custom Menu Items in Design Tools](#)” on page 453.

The following tables describe the menu contents:

- Calibre Menu ([Table A-4](#)) — The menu items available from a layout or schematic window.
- Calibre Setup Menu ([Table A-5](#)) — The items in the **Calibre > Setup** submenu in a layout or schematic window.

In addition, there is a Calibre menu in the Cadence Virtuoso Analog Design Environment (ADE) window; see “[Mapping Schematic Nets to Calibre View Nets with Multiple Extracted Views for Simulation and Plotting](#)” on page 553.

**Table A-4. Calibre Menu in Cadence Virtuoso**

Menu Item	Description
Run nmDRC	Starts Calibre Interactive DRC. The classic version of Calibre Interactive is loaded. (not available in schematic view)
Run DFM	Starts Calibre Interactive DFM. The classic version of Calibre Interactive is loaded. (not available in schematic view)
Run nmLVS	Starts Calibre Interactive LVS. The classic version of Calibre Interactive is loaded.
Run PERC	Starts Calibre Interactive PERC. The classic version of Calibre Interactive is loaded.
Run PEX	Starts Calibre Interactive PEX. The classic version of Calibre Interactive is loaded. (not available in schematic view)
Run xACT	Starts Calibre Interactive xACT.
Start RVE	Starts Calibre RVE.
New Calibre Interactive	Opens a separate menu with the option to start Calibre Interactive DRC, LVS, PERC, and PEX in the new Calibre Interactive GUI.
Clear Highlights	Clears Calibre RVE highlights in the current cell.
Setup	See <a href="#">Table A-5</a> .
OPC Workbench	Contains a submenu for using Calibre WORKbench, if installed.
RealTime DRC	See the <a href="#">Calibre RealTime Custom User's Manual</a> .
Map Schematic Net	Only available in the schematic view. Opens the dialog box “Map Schematic Nets to CalibreView Nets.” See the sections “ <a href="#">Mapping Schematic Nets to Calibre View Nets for Simulation and Plotting</a> ” on page 552 and “ <a href="#">Backannotation of Parasitic Values to the Schematic in Cadence Virtuoso</a> ” on page 555.

The **Calibre > Setup** menu has the following items:

**Table A-5. Calibre Setup Menu in Cadence Virtuoso**

Menu Item	Description
Layout Export	Displays a dialog that controls options used while exporting layout databases to Calibre. See “ <a href="#">Layout Export in Cadence Virtuoso</a> ” on page 474.
Netlist Export	Controls options used while exporting Composer schematics to Calibre. See “ <a href="#">Netlist Export in Cadence Composer</a> ” on page 484.
Calibre View	Sets Calibre View options for Calibre parasitic extraction. Environment variables and SKILL variables can be used as part of a filename. See “ <a href="#">Calibre View Setup Dialog Box</a> ” on page 520.
RVE	Sets up your Cadence Library for Calibre RVE and your Highlight Layers. See “ <a href="#">Setup RVE Dialog Box in Cadence Virtuoso</a> ” in the <i>Calibre RVE User’s Manual</i> .
Socket	<p>Specifies the socket that Calibre Interactive and Calibre RVE use to communicate with Cadence Virtuoso. The socket is not initialized until needed.</p> <p>If you want to initialize the socket during the loading of calibre.skl, set the environment variable MGC_RVE_INIT_SOCKET_AT_STARTUP to any value before loading calibre.skl. The socket number is displayed in the CIW window.</p> <p>The socket port number may be specified by the MGC_CALIBRE_LAYOUT_SERVER environment variable, as described in “<a href="#">Setting the Calibre Interactive Socket Port for a Layout Viewer</a>” on page 451.</p>

## Layout Export in Cadence Virtuoso

You can control the layout export from Cadence Virtuoso with the Calibre Layout Export Setup dialog box and the Cadence Virtuoso template file. You can also use the Cadence Virtuoso streamOutKeys or oasisOutKeys variables.

The [Calibre Layout Export Setup Dialog Box in Cadence Virtuoso \(Calibre > Setup > Layout Export\)](#) does not provide control over all the various options available for layout export in Cadence Virtuoso. However, the export procedure utilizes the template file if one is specified. The recommended flow is to set up all the applicable options for export through the Cadence Virtuoso File > Export > Stream dialog box, then save these options in a template file. See [“Using a Template File for Layout Export with Cadence Virtuoso”](#) on page 478 for instructions on specifying the template file for use by Calibre Interactive.

You can load the template file in the **Calibre > Setup > Layout Export** dialog box or use the SKILL function `mgc_rve_load_export_layout_template_file`. The complete procedure is described in the section [“Using the Calibre - Virtuoso Interface”](#) on page 489. You can specify that the Calibre Layout Export Setup dialog box is opened before each export process by enabling “Show dialog before export” in the dialog. This is useful if you need to change layout export settings for each run.

When exporting a layout to GDSII format, pipo strmout or strmout is used, depending on whether you are using CDBA-based Cadence Virtuoso (version 5.1.4 and earlier) or the Open Access version (version 6.1 and later). Several options are available to control the layout export. These options are stored within Cadence Virtuoso as the SKILL variable streamOutKeys.

When exporting to OASIS format, the oasisout command is used. OASIS export is not supported in CDBA-based Virtuoso. The options for layout export are stored in the SKILL variable oasisOutKeys.

The streamOutKeys or oasisOutKeys variable may be loaded directly in the CIW, using a trigger function or a template file.

For pipo strmout with CDBA-based Cadence Virtuoso, Calibre Interactive determines whether the streamOutKeys variable has been set within the Cadence Virtuoso editor before instructing Cadence Virtuoso to export the layout. If the streamOutKeys variable is set, Calibre Interactive creates a text file containing the values. However, if the variable is not set, Calibre Interactive initializes the variable to default values. The created text file is saved in the Run Directory and is named *CGI.streamOut.<cell\_name>*.

If you are using Cadence Virtuoso 6.1 and later, you can stream out the design from memory, rather than having to save the design to disk prior to running Calibre. Choose **Calibre > Setup > Layout Export** to open the Calibre Layout Export Setup dialog box, then enable “Export From Virtual Memory.” All options in the Stream Out form are filled in by Calibre.

Lower and uppercase settings are supported. You can set the streamOutKeys->caseSensitivity or oasisOutKeys->case variable to upper, lower, or preserve before exporting the layout.

Calibre Interactive alters only two options set by the existing streamOutKeys or oasisOutKeys within Cadence Virtuoso—the case sensitivity and primary cell options. The case sensitivity option is set to preserve, and the primary cell option is set to the value specified in the Top Cell field on the Inputs pane.

The layout export process displays a dialog if the cell being exported has been modified. The dialog allows you to optionally save the cell before exporting it. Note that only the top-level cell is checked for modifications.

For export to GDS format, if the layout export process renames the top cell because another cell has the same name, the following warning is issued, and calibre is not run.

```
Duplicate cell name found in design. Renaming 'PATH:path' to 'path_2'.  
### Calibre will not proceed due to above warning(s)!###
```

You can disable the check for duplicate cell names by setting the environment variable MGC\_CALIBRE\_NO\_PIPO\_WARNING\_CHECK to any value. By default, MGC\_CALIBRE\_NO\_PIPO\_WARNING\_CHECK is not set. This environment variable does not apply for OASIS export.

See “[Cadence Virtuoso](#)” on page 469 for general information about the Calibre Interactive interface to Cadence. You may also want to see these topics:

- “[Automatically Saving the Express Pcell Cache](#)” on page 482
- “[Direct Read of Cadence Virtuoso OpenAccess Database](#)” on page 482

See following topics for specific information related to layout export:

<b>Calibre Layout Export Setup Dialog Box in Cadence Virtuoso .....</b>	<b>475</b>
<b>Using a Template File for Layout Export with Cadence Virtuoso .....</b>	<b>478</b>
<b>SKILL Functions for Layout Export.....</b>	<b>480</b>

## Calibre Layout Export Setup Dialog Box in Cadence Virtuoso

The Calibre Layout Export Setup dialog box sets options that control layout export from Cadence Virtuoso to Calibre. Environment variables and SKILL variables can be used in filenames. From Cadence, choose **Calibre > Setup > Layout Export** to open the layout export dialog box.

You can also have a similar dialog box, the Calibre Layout Export dialog box, open automatically before the export process starts; this enables you to change settings before the

run. To do this, enable “Show dialog before export” in the Calibre Layout Export Setup dialog box. The settings in the two dialog boxes are slightly different.

The following table lists the settings in both the Calibre Layout Export Setup and the Calibre Layout Export dialog box; a note is included if a setting is only present in one of the dialog boxes. For complete details on the settings and the related streamOutKeys and oasisOutKeys variables, consult the Cadence documentation.

**Table A-6. Calibre Layout Export Setup Dialog Box Contents**

Setting	Description
Template File  (Layout Export Setup dialog box only)	Specify a template file. Click <b>Load</b> to load the settings into the dialog box.  Specifying a template file creates a streamOutKeys or oasisOutKeys variable and sets the SKILL variable mgc_calibre_export_layout_template_file (GDS) or mgc_calibre_export_layout_oasis_template_file (OASIS).  Also see “ <a href="#">Using a Template File for Layout Export with Cadence Virtuoso</a> ” on page 478, which includes instructions for setting the template file using a SKILL variable.
Library Name  (Layout Export dialog box only)	The library name, taken from the “Library Name” field on the Inputs pane of Calibre Interactive.
Top Cell Name  (Layout Export dialog box only)	The top cell name, taken from the “Top Cell” field on the Inputs pane of Calibre Interactive.
View Name	Sets the Cadence view of the cell. For the Layout Export dialog box, this is taken from the “View Name” field on the Inputs pane of Calibre Interactive.
Stream File/OASIS File  (Layout Export dialog box only)	The filename for the exported layout file, taken from the “Layout File” field on the Inputs pane of Calibre Interactive. This setting cannot be changed.

**Table A-6. Calibre Layout Export Setup Dialog Box Contents (cont.)**

<b>Setting</b>	<b>Description</b>
Run Directory	<p>Specify the run directory used during layout export. Relative pathnames specified in the export dialog box are qualified with the run directory pathname. The default is the current directory.</p> <p>Tip: The log file location can be controlled by setting the run directory and by specifying a relative pathname for the log file.</p>
Export Log File  (Layout Export dialog box only)	Saves the Cadence log file for the export to the run directory with the specified filename (default: PIPO.LOG.<cellname> or OASIS.LOG.<cellname>)
Export Summary File  (Layout Export dialog box only)	Saves the Cadence summary file for the export to the run directory with the specified filename (default: PIPO.SUM.<cellname> or OASIS.SUM.<cellname>)
Layer Map File	Specify a layer map file to be used during layout export.
Object Map File	<p>Specify a object mapping file to be used during layout export. If you do not specify object mapping, Calibre uses Cadence Virtuoso defaults.</p> <p>This setting is only available if you are using an OpenAccess-based version of Cadence Virtuoso (5.1 or 6.1).</p>
User Skill File	Specify a SKILL file to be used during layout export.
Export From Virtual Memory  (only available in OpenAccess version of Cadence Virtuoso)	Specify to stream out from virtual memory using the Cadence Virtuoso XStream Out or XOasis Out form instead of the strmout or oasisout command. All options in the form are filled in by Calibre.
Enable Coloring  (only available in Cadence Virtuoso IC 12)	<p>Use the pre-coloring selections defined in the Cadence Virtuoso design. Coloring is handled by the Virtuoso streamout or oasisout command.</p> <p>You can set the environment variable MGC_CI_EXPORT_ENABLE_COLORING to have this option set automatically. By default, coloring information is not exported.</p>

**Table A-6. Calibre Layout Export Setup Dialog Box Contents (cont.)**

Setting	Description
Options for Format	Specify GDSII or OASIS for the exported database format. This setting cannot be changed from the Calibre Layout Export dialog box.
Show dialog before export  (Layout Export Setup dialog box only)	Display the Calibre Layout Export dialog box every time before export.

## Using a Template File for Layout Export with Cadence Virtuoso

You can use a template file to control the layout export when using Calibre Interactive with Cadence Virtuoso. The template file is specified with the global variable `mgc_calibre_export_layout_template_file` (GDSII) or `mgc_calibre_export_layout_oasis_template_file` (OASIS) or using the GUI.

The template file is typically defined in your `.cdsinit` file or other startup routine. If specified, Calibre Interactive uses the template file as is and does not perform any error checking on the settings. Certain settings in the template file are overwritten by Calibre Interactive at runtime based on Calibre Interactive settings, the open cell view, and the settings in the Calibre Layout Export dialog box (**Calibre > Setup > Layout Export**). Default streamout settings are used if a template file is not specified.

If a template file is specified as part of the Cadence Virtuoso startup process, the settings are loaded into the Calibre Layout Export dialog box when the Calibre integration is loaded. You can also load the template file settings at any time using the [`mgc\_rve\_load\_export\_layout\_template\_file`](#) command or the Load button in the dialog box.

### Prerequisites

- A Cadence Virtuoso template file for layout streamout.

## Procedure

Define the template file with one of these methods:

To define using ...	Do this ...
SKILL variable:	<p>Define one of the following variables with the path to the template file:</p> <ul style="list-style-type: none"> <li>• GDS: <code>mgc_calibre_export_layout_template_file</code></li> <li>• OASIS: <code>mgc_calibre_export_layout_oasis_template_file</code></li> </ul> <p>(Optional) If you defined the template file manually rather than as part of a startup procedure, call <code>mgc_rve_load_export_layout_template_file("GDSII")</code> or <code>mgc_rve_load_export_layout_template_file("OASIS")</code> to load the settings into the Calibre Layout Export Setup dialog box.</p>
GUI setting:	<p>In Cadence Virtuoso, do the following:</p> <ol style="list-style-type: none"> <li>1. Select <b>Calibre &gt; Setup &gt; Layout Export</b>.</li> <li>2. In “Options for Format”, select GDSII or OASIS for the exported layout format.</li> <li>3. Fill in “Template File” with the file path.</li> <li>4. Click <b>Load</b> to read the settings into the Calibre Layout Export Setup dialog box.</li> <li>5. Make other settings as desired.</li> <li>6. Click <b>OK</b>.</li> </ol>

## Results

Calibre Interactive uses the template file to control layout export. Certain settings in the template file are overwritten by Calibre Interactive at runtime based on Calibre Interactive settings, the open cell view, and the settings in the Calibre Layout Export dialog box (**Calibre > Setup > Layout Export**).

If you used the GUI to specify the template file, this defines `mgc_calibre_export_layout_template_file` or `mgc_calibre_export_layout_oasis_template_file`, depending on whether GDS or OASIS export is selected.

## Examples

The following example specifies a layout template file for GDS export:

```
mgc_calibre_export_layout_template_file = "my_layout_template"
```

When placed in a .cdsinit file or other startup routine, the settings are automatically loaded into the Calibre Layout Export dialog box when the Calibre integration is loaded.

## SKILL Functions for Layout Export

Layout export functions are provided for Cadence Virtuoso.

The following functions are provided:

- [mgc\\_rve\\_export\\_layout\\_cmd](#)
- [mgc\\_rve\\_load\\_export\\_layout\\_template\\_file](#)

The commands `mgc_rve_load_export_layout_template_file("GDSII")` and `mgc_rve_load_export_layout_template_file("OASIS")` are executed upon loading the Calibre interface.

### [mgc\\_rve\\_export\\_layout\\_cmd](#)

Returns a string value consisting of the layout export command used by Calibre Interactive.

This command is provided for historical reasons for those wanting to customize the layout export process.

#### Syntax

```
mgc_rve_export_layout_cmd(libName cellName viewName fileName  
@optional (format "GDSII") )
```

#### Arguments

- libName — string; the library name.
- cellName — string; the top cell name.
- viewName —string; the cellview name.
- fileName — string; the layout database filename, taken from the File entry field on the Inputs pane of Calibre Interactive.
- format — An optional argument specifying the format of the exported layout; allowed values are the “GDSII” and “OASIS”. The default is “GDSII”.

#### Return value

Returns a string value consisting of the layout export command.

#### Related Topics

[mgc\\_rve\\_load\\_export\\_layout\\_template\\_file](#)

[Layout Export in Cadence Virtuoso](#)

[Calibre Layout Export Setup Dialog Box in Cadence Virtuoso](#)

## **mgc\_rve\_load\_export\_layout\_template\_file**

SKILL command to load the layout template file specified by the SKILL variable mgc\_calibre\_export\_layout\_template\_file or mgc\_calibre\_export\_layout\_oasis\_template\_file.

### Syntax

```
mgc_rve_load_export_layout_template_file(@optional (format "GDSII"))
```

### Arguments

- format — An optional argument specifying the format of the exported layout; allowed values are the following:
  - GDSII — (default) Loads the file specified by the mgc\_calibre\_export\_layout\_template\_file variable.
  - OASIS — Loads the file specified by the mgc\_calibre\_export\_layout\_oasis\_template\_file variable.

### Return value

Returns “t” on success or an error message on failure.

### Examples

The following example loads a layout template file for GDS export:

```
mgc_calibre_export_layout_template_file = "my_layout_template"  
mgc_rve_load_export_layout_template_file()
```

The following example loads a layout template file for OASIS export:

```
mgc_calibre_export_layout_oasis_template_file = "my_oasis_template"  
mgc_rve_load_export_layout_template_file("OASIS")
```

---

### Note

-  If a template file is specified as part of the Cadence Virtuoso startup process, the settings are loaded into the Calibre Layout Export dialog box when the Calibre integration is loaded.  
See “[Using a Template File for Layout Export with Cadence Virtuoso](#)” on page 478.
-

## Direct Read of Cadence Virtuoso OpenAccess Database

---

Calibre can read the design data directly from the OpenAccess database. This is called Direct Read, in contrast to Layout Export, in which Cadence Virtuoso converts the design to GDS format and streams it out to Calibre Interactive.

See “[Providing OpenAccess Input with Calibre Interactive](#)” on page 73 for instructions on specifying Direct Read of an OpenAccess database.

You can enable Calibre Interactive to save the Express Pcell cache, as described in “[Automatically Saving the Express Pcell Cache](#)” on page 482.

The following additional topics may be of interest:

“[Layout Export in Cadence Virtuoso](#)” on page 474

“[Creating an Interface to Cadence Virtuoso](#)” on page 469

**Automatically Saving the Express Pcell Cache..... 482**

## Automatically Saving the Express Pcell Cache

Calibre Interactive automatically saves the Express Pcell cache when performing direct read of Cadence Virtuoso OpenAccess databases if the following conditions are met.

### Prerequisites

- Using the OpenAccess-based version of Cadence Virtuoso, and the environment variables CDS\_INST\_ROOT or CDS\_INST\_DIR, and PATH are set correctly. CDS\_INST\_ROOT takes precedence if both CDS\_INST\_ROOT and CDS\_INST\_DIR are defined.
- Make sure LD\_LIBRARY\_PATH includes the libstdc++.so.6.0.8 compiler libraries. This is true by default for Linux machines with RHEL 4u6 or later.

Here is an example if you need to set LD\_LIBRARY\_PATH:

```
setenv LD_LIBRARY_PATH
${CDS_INST_DIR}/share/oa/lib/linux_rhel40_gcc44x_64/opt:
${CDS_INST_DIR}/tools/cdsSkillPcell/lib/64bit:
${CDS_INST_DIR}/tools/lib/64bit:${LD_LIBRARY_PATH}
```

### Procedure

1. Make sure the pcell definition is loaded in the .cdsinit file. For example, your local .cdsinit file might look like this:

```
; local .cdsinit
load("~/cdsinit")
load("mypcells.il")
; ... rest of code
```

2. Open Calibre Interactive.
3. In the Layout Path section in the **Inputs** page, set “Layout Format” to “OPENACCESS.”
4. Open the **OA/LEFDEF** page and do the following:
  - a. Select the OpenAccess Environment section.
  - b. Enable “Set OA\_HOME when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”
  - c. Enable “Set CALIBRE\_READDB\_LD\_LIBRARY\_PATH when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”
  - d. Enable “Set PATH when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”
  - e. If you are using Express Pcells, enable “Set CDS\_EXP\_PCELL\_DIR when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”
  - f. If you are using Express Pcells, enable “Set CDS\_EXP\_ALL\_PARAMS when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”
  - g. If you are using Express Pcells, enable “Set CDS\_ENABLE\_EXP\_PCELL when CDS\_INST\_ROOT or CDS\_INST\_DIR is set.”

## Results

The Pcell cache is automatically saved in the directory specified by CDS\_EXP\_PCELL\_DIR.

## Netlist Export in Cadence Composer

Many aspects of netlist export from Cadence Composer are controlled with the Calibre Netlist Export Setup dialog box. The netlist export routines use the Simulation Interface system (si -batch) to translate the schematic to CDL when performing netlist export.

The Calibre Interactive GUI for LVS, PEX, and PERC supports automatic import of Composer netlists as the source database. You can specify CDL netlisting template parameters in the [Calibre Netlist Export Setup Dialog Box in Cadence Composer](#). You can load the template file using the dialog box or the SKILL function [mgc\\_rve\\_load\\_export\\_netlist\\_template\\_file](#).

The source database format is set to CDL when Calibre Interactive is invoked from Cadence Virtuoso. Consequently, the netlist export routines use the Simulation Interface system (si -batch) to translate the schematic to CDL. The si command can do one of three things: set up the environment, create a netlist, or run a simulation. In the case of Calibre Interactive, you are creating a netlist. The command is:

```
si -batch -command netlist
```

When using CDL views for netlisting, si reads the si.env file created by Calibre Interactive at runtime, generates a netlist, and performs as expected.

When using auCdl views, si once again reads the si.env file, but also checks the .simrc file to see what variables are set. If the variables cdlSimViewList or cdlSimStopList are set, they overwrite what you specified in the si.env file. If they are the same, there is no problem; however, if they are different, netlisting fails.

The settings in the Calibre Netlist Export Setup dialog box are similar to CDL export settings and you can set them through the cdlOutKeys global SKILL variable. This variable is found in the si.env file. You can load this variable in the CIW, in a trigger function, or by specifying the mgc\_calibre\_export\_netlist\_template\_file SKILL variable to point to a template file.

The netlist export process displays a dialog if the cell being exported has been modified. The dialog allows you to optionally save the cell before exporting it. Note that only the top-level cell is checked for modifications.

During netlist export, the *si.log* file is created in the same directory as the netlist file. You can use the SKILL variable mgc\_rve\_globals->siLogFile to specify a different name for the file.

See “[Cadence Virtuoso](#)” on page 469 for general information about the Calibre Interactive interface to Cadence.

See following topics for specific information related to netlist export:

<b>Calibre Netlist Export Setup Dialog Box in Cadence Composer .....</b>	<b>485</b>
<b>SKILL Functions for Netlist Export .....</b>	<b>488</b>

## Calibre Netlist Export Setup Dialog Box in Cadence Composer

From Cadence, choose **Calibre > Setup > Netlist Export** to open the netlist export dialog box.

The dialog box sets options that control the export of Composer schematics to Calibre. Environment variables and SKILL variables can be used in filenames.

The Calibre Netlist Export Setup dialog has the settings shown in the following table. You may need to scroll down to see all of the entries. For complete details on the settings and the related cdlOutKeys variable, consult the Cadence documentation

**Table A-7. Calibre Netlist Export Setup Dialog Box Contents**

Setting	Description
Template File	Specify a template file to load or save. Saving creates a cdlOutKeys variable.  You can also load the template file by specifying it with the SKILL variable <code>mgc_calibre_export_netlist_template_file</code> , then calling the SKILL function <a href="#"><code>mgc_rve_load_export_netlist_template_file</code></a> .
View name	Sets the Cadence view of the cell.
Simulator	Sets which tool does the netlisting (cdl or audCdl are typical choices).
View List	The view list (also call “view switch list”) controls hierarchy traversal during netlist export.  You can set the environment or SKILL variable <code>CDS_Netlisting_Mode</code> to pre-populate the View List and Stop List fields.
Stop List	The stop list (also call “stopping point view list”) controls when expansion stops during hierarchy traversal for netlist export.  You can set the environment or SKILL variable <code>CDS_Netlisting_Mode</code> to pre-populate the View List and Stop List fields.
Resistor Threshold Value	Sets the value for the *RESI command.
Resistor Model Name	Produces the *.RESI command in the netlist.
Equivalents	Produces *.EQUIV statement in netlist

**Table A-7. Calibre Netlist Export Setup Dialog Box Contents (cont.)**

Setting	Description
Connects	Produces *.CONNECT statement in netlist. Specify the nets with an equals sign (=) between them.  For example, specifying the following in the dialog entry field:  VDD=VDD! VCC=VCC!=VCC2 produces the following statements in the netlist: *.CONNECT VDD VDD! *.CONNECT VCC VCC! VCC2
Include File	Produces *.INCLUDE statement in netlist
Check Resistors	When set, include resistor information in the output netlist.
Check Capacitors	When set, include capacitor information in the output netlist.
Check Diodes	When set, include diode information in the output netlist.
Scale	Select the scale for the output netlist; the default is meter. This produces a *SCALE METER or *SCALE MICRON command in the netlist.
Shrink Factor for width(w) and length(l)	Indicates a scaling percentage.
Check LDD	Produces *.LDD statement in netlist.
Display Pin Information	Produces *.PININFO statement in netlist
Map Bus Name From <> To []	Control mapping of bus pins in netlist export. Enabled — [] Disabled — <> (default) Also see “ <a href="#">Automatically Convert Bus Name Delimiter for Calibre Interactive and Calibre RVE</a> ” on page 550.
Global Power Signals	A list of global power signals, which are indicated with a :P in the netlist.
Global Ground Signals	A list of global ground signals, which are indicated with a :G in the netlist.

**Table A-7. Calibre Netlist Export Setup Dialog Box Contents (cont.)**

Setting	Description
Analog Netlisting Type (The “Simulator” option must be set to audCdl)	Selects how netlisting is done when the simulator is audCdl. <ul style="list-style-type: none"><li>• Connection By Order auCdlDefNetlistProc = "ansCdlSubcktCall" (also known as implicit netlisting)</li><li>• Connection By Name auCdlDefNetlistProc = "ansCdlHnlPrintInst" (also known as explicit netlisting)</li></ul>
Show dialog before export	As described.

## SKILL Functions for Netlist Export

Netlist export functions are provided for Cadence Virtuoso.

The following functions are provided:

- [mgc\\_rve\\_export\\_netlist\\_cmd](#)
- [mgc\\_rve\\_load\\_export\\_netlist\\_template\\_file](#)

The command `mgc_rve_load_export_netlist_template_file()` is executed upon loading the Calibre interface.

### [mgc\\_rve\\_export\\_netlist\\_cmd](#)

Returns a string value consisting of the netlist export command used by Calibre Interactive.

This command is provided for historical reasons for those wanting to customize the netlist export process.

#### Syntax

`mgc_rve_export_netlist_cmd(libName cellName viewName fileName)`

#### Arguments

- `libName` — string; the library name.
- `cellName` — string; the top cell name.
- `viewName` — string; the cellview name.
- `fileName` — string; the netlist filename, taken from the File entry field on the Inputs pane of Calibre Interactive.

#### Return value

Returns a string value consisting of the netlist export command.

#### Related Topics

[mgc\\_rve\\_load\\_export\\_netlist\\_template\\_file](#)

[Netlist Export in Cadence Composer](#)

[Calibre Netlist Export Setup Dialog Box in Cadence Composer](#)

### [mgc\\_rve\\_load\\_export\\_netlist\\_template\\_file](#)

Loads the netlist template file specified by the SKILL variable `mgc_calibre_export_netlist_template_file`.

## Syntax

```
mgc_rve_load_export_netlist_template_file()
```

### Example

```
mgc_calibre_export_netlist_template_file = "my_netlist_template"  
mgc_rve_load_export_netlist_template_file
```

## Related Topics

[Netlist Export in Cadence Composer](#)

[Calibre Netlist Export Setup Dialog Box in Cadence Composer](#)

# Using the Calibre - Virtuoso Interface

The Calibre Interactive interface allows you to run Calibre on a Cadence Virtuoso cell. You can specify settings that control the layout export process.

The procedure below is written for the Calibre Interactive nmDRC application, but similar steps are used for other Calibre Interactive applications.

## Prerequisites

- The Calibre - Virtuoso interface is installed, as described in “[Creating an Interface to Cadence Virtuoso](#)” on page 469.
- (Recommended) A Cadence Virtuoso layout export template file is defined in the .cdsinit or other startup procedure. Use the SKILL variable `mgc_calibre_export_layout_template_file` (GDSII) or `mgc_calibre_export_layout_oasis_template_file` (OASIS), as explained in “[Using a Template File for Layout Export with Cadence Virtuoso](#)” on page 478.
- (Optional) If you are exporting a schematic, set up all the applicable options for netlist export using the Cadence Virtuoso File > Export > CDL dialog box, then save these options in a template file.
- Cadence Virtuoso is open to the desired cell, and the connection is made to Calibre Interactive. See Basic Interface - “[Calibre Interactive](#)” on page 450, “[Setting the Socket Port in Calibre Interactive](#)” on page 451, and “[Setting Socket Connections with Cadence Virtuoso](#)” on page 491.
- Calibre Interactive is open, and GUI settings are correct.

## Procedure

1. In Calibre Interactive, in the Layout Path section of the **Inputs** page, verify that the “Export from layout viewer” option is selected; this directs Calibre Interactive to communicate with the SKILL Interface before the Calibre run.

If you are exporting a schematic, also verify that “Export from source viewer” is selected.

2. (Recommended) If you are using the Open Access version of Cadence Virtuoso and CDS\_INST\_ROOT or CDS\_INST\_DIR is set in your environment, you may want to set the following options in Calibre Interactive.
  - a. Open the **OA/LEFDEF** page.
  - b. Under the OpenAccess Environment section, enable the following:
    - o Set OA\_HOME when CDS\_INST\_ROOT or CDS\_INST\_DIR is set
    - o Set CALIBRE\_READDB\_LD\_LIBRARY\_PATH when CDS\_INST\_ROOT or CDS\_INST\_DIR is set
    - o Set PATH when CDS\_INST\_ROOT or CDS\_INST\_DIR is set
    - o Set CDS\_EXP\_PCELL\_DIR when CDS\_INST\_ROOT or CDS\_INST\_DIR is set
    - o Set CDS\_EXP\_ALL\_PARAMS when CDS\_INST\_ROOT or CDS\_INST\_DIR is set
    - o Set CDS\_ENABLE\_EXP\_PCELL when CDS\_INST\_ROOT or CDS\_INST\_DIR is set

The settings automatically set the indicated environment variable based on the value of CDS\_INST\_ROOT or CDS\_INST\_DIR. CDS\_INST\_ROOT takes precedence if both CDS\_INST\_ROOT and CDS\_INST\_DIR are defined. See the “OpenAccess Environment Options” table in “[OA/LEFDEF and OPENACCESS Pages in Calibre Interactive](#)” on page 112 for more information.

3. In Cadence Virtuoso, choose **Calibre > Setup > Layout Export** to open the Calibre Layout Export Setup dialog box and verify the settings are correct. See “[Calibre Layout Export Setup Dialog Box in Cadence Virtuoso](#)” on page 475.

---

**Tip**

 You can enable the “Show dialog before export” option on the Calibre Layout Export Setup or Calibre Netlist Export Setup dialog box if you want to make changes to the options each time there is an export request from Calibre Interactive.

---

4. (Optional) If you are exporting a schematic, choose **Calibre > Setup > Netlist Export** to open the Calibre Netlist Export Setup dialog box.
  - a. (Optional) Load a CDL template file.
  - b. Adjust settings as needed, as described in “[Calibre Netlist Export Setup Dialog Box in Cadence Composer](#)” on page 485.

5. In Calibre Interactive, click **Run DRC** on the left panel, or the appropriate button for your Calibre application.

The SKILL Interface exports the layout cell in stream format using strmout, oasisout, or pipo strmout, as appropriate.

## Related Topics

[Layout Export in Cadence Virtuoso](#)

[Netlist Export in Cadence Composer](#)

[Creating a Calibre View](#)

[Automating Calibre Interactive Runsets in the Cadence Design Environment](#)

[External Trigger Definitions in Cadence Virtuoso](#)

## Setting Socket Connections with Cadence Virtuoso

The Calibre SKILL Interface uses a TCP socket to communicate with Calibre RVE and Calibre Interactive. When you load the calibre.skl file, the tool attempts to initialize a server socket at port 9189. If that port is being used, the tool searches for a free socket port in the range 5000 to 9999.

If the default port number is used, the following message is displayed in the CIW transcript:

```
// Calibre layout-server initialized successfully at socket 9189.
```

If a different port is found after a search for free ports, the following messages are displayed in the CIW transcript:

```
// Could not initialize Calibre layout-server socket at port 9189. Trying
to find free socket . . .
// Calibre layout-server initialized successfully at socket 5000.
```

To find out what your socket is, enter the following in the CIW window:

```
mgc_rve_globals->socket_number
```

If necessary, you can control the port number used by setting the environment variable MGC\_CALIBRE\_LAYOUT\_SERVER as described in “[Setting the Calibre Interactive Socket Port for a Layout Viewer](#)” on page 451.

You can also use following command from within Cadence Virtuoso to specify a different port without resetting the MGC\_CALIBRE\_LAYOUT\_SERVER environment variable and restarting Cadence Virtuoso:

```
mgc_rve_init_socket <port_number>
```

Type the command in Cadence Virtuoso to initialize the specified socket port for communication.

## Calling SKILL Functions from Calibre Interactive

It is often useful to call a SKILL procedure before or after your Calibre run.

The following topics discuss methods for doing this:

- “[External Trigger Definitions in Cadence Virtuoso](#)” on page 431
- “[Setting Calibre Interactive Internal Triggers](#)” on page 419, using a trigger type of “layout” or “schematic”
- “[mgc\\_calibre\\_add\\_menu\\_item](#)” on page 458, the -code option when adding a custom menu to a design tool

## Creating a Calibre View

A Calibre View of a cell is a schematic view of a cell you can simulate in the Cadence Analog Design Environment. This schematic view contains the connectivity and intentional devices the Calibre LVS tool extracts, and, optionally, parasitic devices the Calibre xRC tool extracts.

You can create the Calibre View using Calibre Interactive PEX or the Calibre Interactive nmLVS and Calibre RVE tools. You can access Calibre Interactive from the **Calibre** pulldown menu in Cadence Virtuoso. An OpenAccess database is required. When displaying the Calibre View, Cadence Virtuoso uses a cellmap file to map the layout device to the corresponding schematic symbol.

You can also create the Calibre View in batch mode, without Calibre Interactive. Use the SKILL functions `mgc_rve_load_setup_file()` or `mgc_rve_create_cellview()` to create the Calibre View. See “[SKILL Functions for Creating a Calibre View](#)” on page 540.

There are global variables and environment variables that affect Calibre View generation; see “[Global Variables for Calibre View Generation](#)” on page 542 and “[Environment Variables for Calibre View](#)” on page 585.

These topics are covered:

<b>What You Need Before Creating a Calibre View</b> .....	<b>494</b>
<b>What a Calibre View Produces</b> .....	<b>495</b>
<b>Automatic OpenAccess Version Detection in Calibre View</b> .....	<b>495</b>
<b>Device Property Calculation in Calibre View Generation</b> .....	<b>496</b>
<b>Handling of Nets With Inherited Connectivity</b> .....	<b>497</b>
<b>Bus Delimiter in Calibre View</b> .....	<b>498</b>
<b>Identifying Devices with a Cellmap File</b> .....	<b>498</b>
<b>Creating the Cellmap File</b> .....	<b>498</b>
<b>Syntax for Cellmap Statements</b> .....	<b>500</b>
<b>Cellmap Examples</b> .....	<b>510</b>
<b>Setting the Calibre View Cellmap File in Calibre Interactive</b> .....	<b>517</b>
<b>Importing Schematic Properties for Calibre View</b> .....	<b>518</b>
<b>Auto-Mapping Behavior, Control, and Debug in Calibre View Generation</b> .....	<b>519</b>
<b>Calibre View Setup Dialog Box</b> .....	<b>520</b>
<b>Creating a Post-LVS Calibre View</b> .....	<b>523</b>
<b>Creating a Post-Parasitic Extraction Calibre View</b> .....	<b>526</b>
<b>Creating a Gate-Level Calibre View</b> .....	<b>529</b>
<b>Creating Calibre Views in a Multiple Corner Extraction Run</b> .....	<b>532</b>
<b>Creating a SPECTRE Netlist from the Calibre View Netlist</b> .....	<b>533</b>

<b>Creating a CalibreView Setup File.....</b>	<b>534</b>
<b>Showing Parasitic Polygons in Calibre View .....</b>	<b>535</b>
<b>CalibreView Setup File Format .....</b>	<b>536</b>
<b>SKILL Functions for Creating a Calibre View.....</b>	<b>540</b>
<b>Global Variables for Calibre View Generation .....</b>	<b>542</b>
<b>Creating a Calibre View in Batch Mode .....</b>	<b>543</b>
<b>Calibre View and IC Manage Revision Control.....</b>	<b>544</b>
<b>Calibre View Generation Warnings.....</b>	<b>544</b>

## What You Need Before Creating a Calibre View

Producing a Calibre View requires the correct product licenses, input files, and integration setup.

- Calibre SKILL Interface — For setup instructions, refer to the README file at `./shared/pkgs/icv/tools/querieskl/skillREADME` in your Siemens software tree.
- Rule File — A valid Standard Verification Rule Format (SVRF) rule file for the design. For parasitic extraction, the SVRF rule file must contain, at a minimum, valid calibrated SVRF Capacitance statements and, if required, SVRF Resistance statements.
- Cadence Virtuoso IC6 or IC12 — If you are using Cadence Virtuoso IC5, please contact your Siemens representative.
- Layout Database — An OpenAccess database.
- Layout Netlist — A CDL or Calibre Interactive tool-produced SPICE source netlist if you require source names in the output for debugging.
- Licenses — When creating a Calibre View, you must have the following license features:
  - A Calibre RVE license.
  - A Calibre Interactive license if the Calibre Interactive GUI is opened.
  - A Calibre nmLVS-H license if using source names or creating a post-LVS Calibre View.
  - A Calibre Connectivity Interface license if creating a post-LVS Calibre View.

Refer to [Calibre Administrator's Guide](#) for complete licensing information.

- Cellmap File — A user-created ASCII file mapping the layout device the Calibre tool extracts to the corresponding schematic symbol. Multiple cellmap files may be specified. See “[Identifying Devices with a Cellmap File](#)” on page 498.

- Library Definitions — If a *cds.lib* file exists in the working directory, it is used as the default library definition file. The *cds.lib* is converted to a temporary *lib.defs* file and saved in \$MGC\_TMPDIR. If both *cds.lib* and *lib.defs* exist in the working directory, *lib.defs* is ignored. The *lib.defs* file is used if there is not a *cds.lib* file in the working directory.

## What a Calibre View Produces

You can produce the Calibre Views after an LVS run or after a parasitic extraction run.

The different run methods have the following results:

- Post-LVS View — generates intentional devices for analysis. See “[Creating a Post-LVS Calibre View](#)” on page 523.
- Post-Parasitic Extraction View — generates intentional devices and parasitics for analysis. See “[Creating a Post-Parasitic Extraction Calibre View](#)” on page 526 and “[Creating a Gate-Level Calibre View](#)” on page 529.

This file is produced after the Calibre View is generated:

- *calibreview.setup* — A file in the [CalibreView Setup File Format](#). This file can be loaded into the Calibre View Setup dialog box to populate the dialog box with settings for the run. Browse for the file or enter the filename into the “Calibreview Setup File” field and click the **Load** button. The file can also be used as input to the SKILL function `mgc_rve_load_setup_file` to create a Calibre View in batch mode.

The setup parameters used to create a Calibre View are saved with the cell view. You can view the Calibre View setup parameters using the following command:

```
mgc_eview_display_setup_settings("lib_name" "cell_name" "view_name")
```

## Automatic OpenAccess Version Detection in Calibre View

When Calibre View uses OpenAccess, it must be a compatible version to the version of Cadence Virtuoso being used. Calibre View automatically detects which OpenAccess version to use and warns the user if the version they specified does not match.

The OpenAccess version is automatically detected based on the version of Virtuoso being used and from the \$CDS\_ROOT and \$CDS\_HOME directory content. The version detected by the Virtuoso version is determined as follows:

Virtuoso Version	OpenAccess Version
IC6.1.2*, IC6.12*, IC6.13*, IC6.14*	22.04
IC6.15*	22.41
IC6.16*, ICAD/12.1*	22.43

Virtuoso Version	OpenAccess Version
IC6.17*, ICAD/12.20.*, ICAD/12.30*	22.50
IC6.18*, ICADVM181/18.10.*	22.60

If the version detected based on the Virtuoso version does not match the version detected based on the \$CDS\_ROOT and \$CDS\_HOME directories, then the following warning is issued to tell you which version is being used:

WARNING: The OA version "22.60" detected from the Virtuoso version is different from the OA version "22.50" retrieved from the "CDS\_ROOT"/"CDS\_HOME" environment variable. Using "22.50" or set the required version using the "MGC\_FDI\_OA\_VERSION" environment variable.

In the above case, the version specified by the \$CDS\_ROOT and \$CDS\_HOME directories was used.

You can also specify the OpenAccess version with the MGC\_FDI\_OA\_VERSION environment variable. When this specified version does not match the automatically detected version, the following warning is issued to tell you which version is being used:

WARNING: The automatic detected OA version "22.43" is different from the user-specified OA version "22.50". Using "22.50".

In the above case, the version specified with the MGC\_FDI\_OA\_VERSION environment variable was used.

When the detected OpenAccess version is 22.04 or 22.41, the following warning is printed and version 22.43 is used:

WARNING: For "22.04" and "22.41" OA versions "22.43" OA version is used.

## Device Property Calculation in Calibre View Generation

Several internal steps are involved when device properties are calculated during Calibre View generation.

1. Layout properties are calculated and included in the Calibre View format netlist after running Calibre. The device property calculations are based on statements in the Calibre LVS rule file.

This step depends on keywords in the PEX Netlist statement, such as SOURCENAMES, LAYOUTNAMES, SOURCEBASED, and SCHEMATICONLY.

2. Process properties that only exist in the schematic as follows:

For each device instance, do one of the following:

- o If importSchemProp is specified for the device in the cellmap file, import the schematic property if importSchemProp is true and skip the property if importSchemProp is false.
- o If importSchemProp is *not* specified for the device in the cellmap file, import the schematic property if the global variable mgc\_eview\_globals->importSchematicProperties is set to t (the default), otherwise skip the property.

---

**Note**

-  Properties in the layout netlist always have precedence over schematic properties.  
Schematic properties are imported according to the logic stated in this step and only if the layout property does not exist.
- 

3. Execute callbacks if mgc\_eview\_globals->executeCallbacks is set to t (for true).  
Callbacks are typically used for property calculations done with equations and may overwrite the device properties calculated in previous steps.

## Related Topics

[Importing Schematic Properties for Calibre View](#)

[Global Variables for Calibre View Generation](#)

## Handling of Nets With Inherited Connectivity

The Calibre View generated view does not show nets with inherited connectivity that do not exist in schematic.

Inherited connectivity is handled in this manner:

1. If there is a net expression in a netset property, it is only copied to the corresponding netset property in the generated view device. This behavior always occurs.
2. If the net expression in the netset property needs to be created in the view to form an inherited connection, enable the env var CALIBRE\_FDIBA\_ENABLE\_NETEXPR\_COPY. The net expression is not created by default.

---

**Note**

-  The environment variable CALIBRE\_FDIBA\_ENABLE\_NETEXPR\_COPY returns the behavior to that found in 2019.2 and earlier releases.
-

## Bus Delimiter in Calibre View

The generated CalibreView netlist always uses <> as the bus delimiter to avoid conflicts with the standard Cadence schematic netlist, which uses [] as the bus delimiter. If you generate a CalibreView from a netlist which uses [] as the bus delimiter, pin and net names ending in [] are automatically converted to use <> as the bus delimiter.

## Identifying Devices with a Cellmap File

Before creating a Calibre View, you must create a cellmap file. A cellmap file is an ASCII mapping file used for mapping the layout device to the corresponding schematic symbol. Both intentional devices and parasitic devices can be mapped with the cellmap file.

Multiple cellmap files may be specified as a space-separated list in the Calibre View Setup dialog box or the Calibre View setup file. Multiple files are concatenated in the order specified.

### Related Topics

[Creating the Cellmap File](#)

[Calibre View Setup Dialog Box](#)

[Creating a CalibreView Setup File](#)

## Creating the Cellmap File

The cellmap file is a text file that is used when creating a Calibre View.

### Procedure

1. In an ASCII text editor, create a unique entry for each intentional device.

See “[Cellmap Syntax for Intentional Devices](#)” on page 500.

Note: you may use an include file in the cellmap file.

2. Add entries for parasitic devices, if needed.

See “[Cellmap Syntax for Parasitic Devices](#)” on page 507.

3. Add layer mapping entries, if needed. This step is generally recommended.

See these topics for additional information:

- “[Layer Mapping Syntax](#)” on page 506
- “[Layer Map Statements to Optimize the Cellmap File](#)” on page 514
- “[Identifying Cadence Virtuoso Layers for Mapping](#)” on page 515

4. Save the cellmap file in the directory containing the design.

**Tip**

 If you are importing schematic properties, see “[Importing Schematic Properties for Calibre View](#)” on page 518 for needed modifications to the SVRF rule file.

---

## Related Topics

[Syntax for Cellmap Statements](#)

[Full Syntax for Property Specification in the Cellmap File](#)

[Cellmap Examples](#)

## Syntax for Cellmap Statements

The cellmap file is needed for creating a Calibre View.

See “[Creating a Calibre View](#)” on page 493 information on Calibre View. For examples of specific cellmap statements and comments on how to solve common mapping issues, see “[Cellmap Examples](#)” on page 510.

The syntax conventions used in the cellmap descriptions are given in [Table 1-4](#) on page 30. You should enter literal text, which is not in italics, exactly as shown. Parentheses are literal in the syntax definitions. The semicolon (;) is the comment character for cellmap files. Text after the semicolon is not parsed.

This section includes syntax definitions for the following cellmap statements:

<b>Cellmap Syntax for Intentional Devices .....</b>	<b>500</b>
<b>Cellmap Syntax for Cells.....</b>	<b>502</b>
<b>Full Syntax for Property Specification in the Cellmap File.....</b>	<b>504</b>
<b>Layer Mapping Syntax .....</b>	<b>506</b>
<b>Cellmap Syntax for Parasitic Devices .....</b>	<b>507</b>

## Cellmap Syntax for Intentional Devices

The cellmap statement for intentional devices instructs the Calibre View process on how to map the device.

### Syntax

```
(device_model_name_in_layout
  (CDN_library_name CDN_cell_name CDN_view_name)
  (
    ({calibre_device_pin_name | nil} {CDN_cell_pin_name | nil})
    ...
  )
  [
    ([nil] calibre_property_name CDN_property_name)
    ...
    [importSchemProp {t | nil}])
  ]
)
```

### Parameters

- ***device\_model\_name\_in\_layout***  
The Calibre device to be mapped. Black-boxed devices are handled automatically; it is not necessary to include them in the cellmap file.
- ***CDN\_library\_name CDN\_cell\_name CDN\_view\_name***

Map device\_model\_name\_in\_layout to an instance of this cellview.

- ( { *calibre\_device\_pin\_name* | nil } { *CDN\_cell\_pin\_name* | nil } )

Maps device pins to cellview pins. At least one pin mapping instruction must be included.

- **nil** — The parameter nil is used to indicate that the pin does not exist. See “[Mapping with Fewer Device Pins](#)” on page 511 for an example of mapping when the device has fewer pins than the cellview.

When a device pin is mapped to a cellview terminal that doesn't exist, a property will be created on the cellview instance with its value set to *CDN\_cell\_pin\_name*. See “[Mapping Extra Device Pins as Nets Mapped to Cell Properties](#)” on page 512 for an example of mapping a Calibre device to a cellview with fewer pins.

Bus pins may be mapped as follows:

( *calibre\_bus\_name*<*start\_pin:end\_pin*> *CDN\_cell\_bus\_name*<*start\_pin:end\_pin*> )

The order and number of bus pins must be the same.

Bus pins may also be mapped individually with the notation *bus\_name*<*pin*>.

- ([nil] *calibre\_property\_name* *CDN\_property\_name* ) ...

Maps a Calibre property to a Cadence cell view property. You can map a Calibre property to multiple Cadence cell view properties. This is done by specifying each property mapping separately.

The property mapping section is optional. See the section “[Full Syntax for Property Specification in the Cellmap File](#)” on page 504 for the complete syntax of the property mapping section, including the use of expressions.

- (importSchemProp { t | nil })

An optional parameter and value that specifies whether or not schematic properties are imported for this device

t — import schematic properties

nil, or any value other than t — do not import schematic properties

If importSchemProp is specified for a device, it overrides the global flag *mgc\_eview\_globals->importSchematicProperties*. If importSchemProp is not specified for a device, the global flag is used. The default value of *mgc\_eview\_globals->importSchematicProperties* is t.

See the section “[Importing Schematic Properties for Calibre View](#)” on page 518 for important information on case sensitivity related to importing schematic properties.

## Usage Notes

You can have multiple devices with the same modelname, however each device should have different pin counts. No errors or warnings occur if you map a Calibre device to different Cadence cellviews with the same number of pins, but this is not recommended. See the example “[Specifying Devices with the Identical Modelname](#)” on page 510.

Black-boxed devices are handled automatically; it is not necessary to include them in the cellmap file. See “[Auto-Mapping Behavior, Control, and Debug in Calibre View Generation](#)” on page 519.

## Examples

This cellmap example specifies the mapping for device *n* with properties specified:

```
(n
  (gpdk nmos symbol)
  (
    (d D)
    (g G)
    (s S)
    (b B)
  )
  (
    (w W)
    (l L)
  )
)
```

This example maps a bus.

```
(ElementB
  (libB ElementB symbol)
  (
    (VDD VDD)
    (test3<7:0> test3<7:0>)
    (BUS2<7:0> BUS2<7:0>)
  )
)
```

## Related Topics

[Full Syntax for Property Specification in the Cellmap File](#)

[Cellmap Syntax for Parasitic Devices](#)

[Cellmap Examples](#)

## Cellmap Syntax for Cells

The cellmap statement instructs the Calibre View process on how to map the cell.

## Syntax

```
(cell_name
  (CDN_library_name CDN_cell_name CDN_view_name)
  (
    ( { calibre_pin_name | nil } { CDN_cell_pin_name | nil } )
    ...
  )
)
```

## Description

The cellmap entry for cells is used for a gate-level CalibreView; see “[Creating a Gate-Level Calibre View](#)” on page 529.

## Parameters

- *cell\_name*

The cell name to be included in the gate-level CalibreView. Black-boxed devices are handled automatically; it is not necessary to include them in the cellmap file.

- *CDN\_library\_name CDN\_cell\_name CDN\_view\_name*

Map *cell\_name* to an instance of this cellview.

- ( { *calibre\_pin\_name* | *nil* } { *CDN\_cell\_pin\_name* | *nil* } )

Maps device pins to cellview pins. At least one pin mapping instruction must be included.

- *nil* — The parameter *nil* is used to indicate that the pin does not exist. See “[Mapping with Fewer Device Pins](#)” on page 511 for an example of mapping when the device has fewer pins than the cellview.

When a pin is mapped to a cellview terminal that doesn't exist, a property will be created on the cellview instance with its value set to *CDN\_cell\_pin\_name*. See “[Mapping Extra Device Pins as Nets Mapped to Cell Properties](#)” on page 512 for an example of mapping a Calibre device to a cellview with fewer pins.

## Examples

Suppose you have a cell named One\_Bit\_Adder with this source HSPICE definition:

```
.subckt One_Bit_Adder a b Sum Carry
  <circuit_details>
.ends
```

Suppose you have a symbol named “Adder1Bit” with pins A, B, SUM, and CARRY in the Cadence schematic library named “MyLib”.

The cellmap entry for the cell One\_Bit\_Adder is written as follows:

```
(One_Bit_Adder
  (MyLib Adder1Bit symbol)
  (
    (a A)
    (b B)
    (Sum SUM)
    (Carry CARRY)
  )
)
```

## Full Syntax for Property Specification in the Cellmap File

You can add property specifications to the cellmap file that determine how property information is passed from Calibre to the Calibre View.

You can use the cellmap file to specify the following property behaviors:

- Specify expressions based on Calibre property values. Properties are evaluated in the order that they appear in the cellmap file.
- Enable and disable CDF callback functions.
- Ignore properties from the schematic when creating the Calibre View.
- Ignore properties from the layout when creating the Calibre View.
- Specify which property values have priority for properties common to both schematic and layout.

You can use the formats in the following table to specify property behaviors in the cellmap file. See the section “[Importing Schematic Properties for Calibre View](#)” on page 518 for important information on case sensitivity related to importing schematic properties.

In the table, *x* is a Calibre property name and *y* is a Cadence property name.

**Table A-8. Defining Properties in the Cellmap File**

Property Syntax	Description	Example
( <i>x</i> "")	Calibre property <i>x</i> is ignored when Calibre View is created.	(w "")
(nil <i>x</i> )	Property <i>x</i> is not copied from the schematic cell view instance.	(nil w)
(nil <i>x</i> <i>y</i> )	<i>x</i> property is set to <i>y</i> in the Calibre View regardless of its value in the schematic.	(nil SUB gnd)

**Table A-8. Defining Properties in the Cellmap File (cont.)**

Property Syntax	Description	Example
(x y (expression))	Cadence property y is set to the value of the expression. The only parameter available in the expression is the Calibre device property x.	(w W (w * 1e-06)) Also see “ <a href="#">Using Expressions to Specify Properties in the Cellmap File</a> ” on page 513.
((x y) ... (importSchemProp t   nil))	importSchemProp is an optional parameter that specifies whether or not schematic properties are imported for the device.  If importSchemProp is specified for a device, it overrides the global flag <i>mgc_eview_globals-&gt;importSchematicProperties</i> . If importSchemProp is not specified for a device, the global flag is used.	See “ <a href="#">Example Cellmap with Property Mapping Including Fingers</a> ” on page 512.
(x(y t   nil))	x is the Calibre property name y is the Cadence property name Second element of list indicates whether the CDF callback function will be executed when setting the value of property y.  t: (default) The CDF callback is executed.  nil: The CDF callback is not executed.  The following syntax is equivalent: (x (y t)) (x (y)) (x y)  Note: Both the t option and the global control must be set in order to execute callbacks. For the global control, you can set <i>mgc_eview_globals-&gt;executeCallbacks</i> to t or check the option “Execute Callbacks” in the Calibre View Setup dialog box. The default for the global option is nil.	(w (w t))

**Table A-8. Defining Properties in the Cellmap File (cont.)**

Property Syntax	Description	Example
(x (y t   nil) (expression))	Cadence property y is set to the value of the expression, which is calculated based on the value of the Calibre x property; execution of the callback associated with property y (if a callback exists) is determined by rules stated in the previous row description.	(w (W t) (w * 1e-06))

You can map a Calibre property to multiple Cadence cell view properties. This is done by specifying each property mapping separately, for example:

```
(w WA) ;Map Calibre property w to Cadence cell view property WA  
(w WB) ;Map Calibre property w to Cadence cell view property WB
```

See “[Using Expressions to Specify Properties in the Cellmap File](#)” on page 513 for a similar example using property expressions.

See the command [PEX BA Mapfile](#) for a description of the LPE parameter, which is included by default in Calibre View netlists.

## Related Topics

[Cellmap Syntax for Intentional Devices](#)

[Cellmap Syntax for Parasitic Devices](#)

[Layer Mapping Syntax](#)

[Cellmap Examples](#)

[Global Variables for Calibre View Generation](#)

## Layer Mapping Syntax

Each layer that is imported to a Calibre View requires a mapping statement to identify the corresponding Cadence layer name.

### Layer Mapping Syntax

```
mgc_layer_map( list(  
    list( "calibre_layer_name" ' (CDS_layer_name CDS_layer_purpose)  
    ...  
)
```

## Example

```
mgc_layer_map( list(
    list( "NWL"  '(NWELL drawing))
    list( "OX"   '(OXIDE drawing))
    list( "PO1"  '(POLY1 drawing))
    list( "CO"   '(CONTACT drawing))
    list( "M1"   '(METAL1 drawing))
)
)
```

See “[Layer Map Statements to Optimize the Cellmap File](#)” on page 514 for information on what layers to map.

See “[Identifying Cadence Virtuoso Layers for Mapping](#)” on page 515 for information on how to match the Calibre layer to the Cadence layer.

## Cellmap Syntax for Parasitic Devices

This section describes the cellmap syntax for parasitic devices.

The following parasitic devices are included:

Parasitic Capacitor

Parasitic Self-inductor

Parasitic Resistor

Parasitic coupling-element

### Parasitic Capacitor

#### Syntax

```
((p cap capacitance_property_name)
  (library_name symbol_name symbol)
  (
    (symbol_pin)
    (symbol_pin)
  )
)
```

#### Example

```
((p cap c)
  (analog_lib cap symbol)
  (
    (PLUS)
    (MINUS)
  )
)
```

## Parasitic Resistor

### Syntax

```
((p res resistance_property_name)
  (library_name symbol_name symbol)
  (
    (symbol_pin)
    (symbol_pin)
  )
)
```

You can also specify extra cell pins—see the section “[Mapping with Fewer Device Pins](#)” on page 511.

### Example

```
((p res r)
  (analog_lib res symbol)
  (
    (PLUS)
    (MINUS)
  )
)
```

## Parasitic Self-inductor

### Syntax

```
((p ind inductance_property_name)
  (library_name symbol_name view_name)
  (
    (symbol_pin)
    (symbol_pin)
  )
)
```

### Example

```
((p ind l)
  (analogLib ind symbol)
  (
    (MINUS)
    (PLUS)
  )
)
```

---

#### Note

---

 Use the pin order shown in the example.

---

## Parasitic coupling-element

### Syntax

```
((p mutk coupling_element_property_name)
  (library_name symbol_nameview_name)
  (
    (parameter1)
    (parameter2)
  )
)
```

### Example

```
((p mutk k)
  (analogLib mind symbol)
  (
    (ind1)
    (ind2)
  )
)
```

In this example, ind1 and ind2 are parameters of the mutual inductor and CalibreView will treat them exclusively as parameters, not as pins.

### Related Topics

[Cellmap Syntax for Intentional Devices](#)

[Full Syntax for Property Specification in the Cellmap File](#)

[Layer Mapping Syntax](#)

[Cellmap Examples](#)

## Cellmap Examples

The following cellmap file examples are included in this section:

<b>Specifying Devices with the Identical Modelname .....</b>	<b>510</b>
<b>Mapping with Fewer Device Pins.....</b>	<b>511</b>
<b>Mapping Extra Device Pins as Nets Mapped to Cell Properties.....</b>	<b>512</b>
<b>Example Cellmap with Property Mapping Including Fingers .....</b>	<b>512</b>
<b>Using Expressions to Specify Properties in the Cellmap File .....</b>	<b>513</b>
<b>Layer Map Statements to Optimize the Cellmap File .....</b>	<b>514</b>
<b>Identifying Cadence Virtuoso Layers for Mapping .....</b>	<b>515</b>

### Specifying Devices with the Identical Modelname

You can have multiple devices with the same modelname, however each device should have different pin counts.

For example, the two statements in the following example are valid mappings for device type mp.

No errors or warnings occur if you map a Calibre device to different Cadence cellviews with the same number of pins, but this is not recommended.

## Example

```
(mp ; 4-terminal pfet
  ( sample pfet4 symbol )
  (
    (g G)
    (s S)
    (d D)
    (b B)
  )
  (
    (w WIDTH) (l LENGTH)
  )
)
(mp ; 6-terminal pfet
  ( sample pfet6 symbol )
  (
    (g G)
    (s S)
    (d D)
    (b B)
    (b0 B0)
    (b1 B1)
  )
  (
    (w WIDTH) (l LENGTH)
  )
)
```

## Mapping with Fewer Device Pins

You can map a Calibre device to a Cadence cell containing a greater number of pins. For example, the following cellmap entry would map a 2-pin Calibre device to a 3-pin device:

```
(cap
  ( sample cap3 symbol )
  (
    (plus PLUS)
    (minus MINUS)
    (nil BULK "GND")
  )
)
```

In the generated Calibre View, the BULK pin of the cap3 cell is connected to the GND net in any placement of the capacitor device extracted by the Calibre tool.

If pin BULK does not exist, property BULK is placed on the cellview instance with value “GND”.

Another option is possible, allowing you to connect Cadence pins together:

```
(nil SUB (D)) ; Connect Cadence pin SUB to net connected to pin D
```

If pin SUB does not exist on the cellview, a property is created on the cellview instance. Note that pin D has to appear before pin SUB in the pin mapping specification.

## Mapping Extra Device Pins as Nets Mapped to Cell Properties

You can map nets connecting to a pin of a device the Calibre tool extracts to a property on a cell the Calibre View instantiates.

Instead of specifying a terminal of the cell in the cellmap file, you can specify the name of the property the Calibre tool will create on the cell. You assign the net name the value of this property. For example, you can map a 4-pin Calibre device to a 3-terminal cell with a property on the fourth pin's net.

The following cellmap file statement will map the net connected to pin b of the Calibre pfet device to the BULK property on the 3-terminal pmos cell instance.

```
;;
; map 4-terminal pfet Calibre device to 3-terminal pmos cell instance
;
(pfet
  ( sample pmos symbol ) ; 3-terminals on pmos cell
  (
    (g G)
    (s S)
    (d D)
    (b BULK) ; net connected to b will attach with value of BULK
  )
)
```

## Example Cellmap with Property Mapping Including Fingers

The schematic property “fingers” is set to the number of fingers of the device. Since each individual finger in the layout needs fingers=1, we need to set this in the cellmap file, or the schematic property value will be applied to each finger of the device.

Use this syntax to set the property “fingers” in the schematic:

```
(nil fingers 1) ;Property "fingers" is not in the layout
                  ;but is in the schematic. Want to set fingers
                  ;to a specific value
```

For example:

```
(n
  (tsmc13lg nmos1v symbol)
  (
    (b B)
    (d D)
    (g G)
    (s S)
  )
  (
    (w fw) ; map layout property "w" to schematic property "fw"
    (nil fingers 1) ; Property "fingers" is not in the layout
                      ; so set its value in the schematic
    (importSchemProp t) ; import schematic properties for this device
  )
)
(p
  (tsmc13lg pmos1v symbol)
  (
    (b B)
    (d D)
    (g G)
    (s S)
  )
  (
    (w fw) ; map layout property "w" to schematic property "fw"
    (nil fingers 1) ; Property "fingers" is not in the layout
                      ; so set its value in the schematic
    (importSchemProp t) ; import schematic properties for this device
  )
)
```

Please note that if you attempt to set the “fw” property to the value of the “w” property for all devices by using the Reset Properties field in the Calibre View Setup dialog box (**Calibre > Setup > Calibre View**), it will not work properly because the Reset Properties field performs simple assignments. In performing these assignments the Reset properties field is not doing any variable or property de-referencing or expression evaluation. To get the value of a layout property assigned to schematic property for all devices, you should use the property mapping statement in each device, as shown in this cellmap example.

## Using Expressions to Specify Properties in the Cellmap File

This example uses expressions to map a Calibre property to Cadence properties.

The Calibre *w* property value is mapped to two different Cadence properties (*wscale1* and *wscale2*) using an expression. A conditional expression is used to map the Calibre *lpe* property. See the command **PEX BA Mapfile** for a description of the LPE parameter, which is included by default in Calibre View netlists.

```
(p
  ( mgc_p4 pmos4 symbol)
  ( (b B) (d D) (g G) (s S) )
  ( (l l) (w w) (m m)
    (w wscale1 (w*17))
    (w wscale2 (w*125))
    (lpe lpe_a (if lpe==0 || lpe==1 then lpe_a=1 else lpe_a=0))
  )
)
```

## Layer Map Statements to Optimize the Cellmap File

To create an optimized cellmap file, you should map the Connect statement layers and Device statement layers.

Inspect your LVS rule file for the following layers:

- Connect statement layers

For example, if you have the following statement in your LVS rule file, you will need to map the metal1, metal2, and via1 layers to their corresponding Cadence layers:

```
Connect metal1 metal2 by via1
```

- Device layers

You need to include both seed layers and pin layers because the seed layer is the layer that enables you to recognize the device and the pin layers will most likely be included in the connect statements.

For example, if you have the following statement in your LVS rule file, you will need to map the ngate, poly, nsd and psub layers to their corresponding Cadence Layers.

```
DEVICE MN(nch) ngate poly(G) nsd(S) nsd(D) psub(B)
```

By mapping these two types of layers, you can generate a mask-Layout Calibre View as output of both Calibre LVS and Calibre xRC.

---

### Tip

 Layer mapping statements are required in the cellmap file when backannotating net geometries to the Calibre View. See the section “[Specifying Extracted Netlist Annotation of Net Geometries to Calibre View](#)” on page 210.

---

**Note**

 Calibre LVS and xRC write layers used for either device definition or to define circuit connectivity to the standard verification database (SVDB). Some original or intermediate derived layers may not be available to import since they were never written to the SVDB.

If a layer is in the SVDB and has a cell mapping statement, but does not have a Cadence layer purpose pair (LPP) defined, Calibre will attempt to create that LPP and import the geometries.

---

## Related Topics

[Creating the Cellmap File](#)

[Layer Mapping Syntax](#)

[Identifying Cadence Virtuoso Layers for Mapping](#)

## Identifying Cadence Virtuoso Layers for Mapping

Calibre layers can be mapped Cadence drawn layers in the cellmap file, and you need to identify the corresponding Calibre layer and Cadence drawn layer.

As explained in [Layer Map Statements to Optimize the Cellmap File](#), certain Calibre layers should be mapped to Cadence drawn layers in the cellmap file. These steps illustrate how to match the Calibre layers to the Cadence layers. See “[Layer Mapping Syntax](#)” on page 506 for a definition of the layer mapping statement.

## Procedure

1. In Cadence Virtuoso, choose **Tools > Technology File Manager** to display the Technology File Toolbox.
2. Click **Dump**. The Dump Technology File dialog box displays.
3. Select a “Technology Library” name from the dropdown menu; enable the “layerRules” options; enter a name for the file into the ASCII Technology File field; click **OK**.

A Cadence technology file is output containing the layer names and their GDS numbers as shown in the following figure.

**Figure A-2. Cadence Technology File**

```

; Technology File sphinx
; Generated on Aug 29 17:33:33 2006
; with @(#)CDS: icfb.exe version 5.1.0 06/24/2004 17:30 (intelibm5.Cadence)
; COM) $ 

*****+
+ LAYER RULES
*****+
layerRules<

streamLayers(
;( layer      streamNumber    dataType      translate )
;(
("diff_ment" "P0") 2          0           t        )
(("NP_ment" "P0") 3          0           t        )
(("PP_ment" "P0") 4          0           t        )
(("NIELL_ment" "P0") 1        0           t        )
(("DNL" "P0") 5          0           t        )
(("DC" "P0") 6          0           t        )
(("GT" "P0") 30         0           l        )
(("HRP" "P0") 39         0           t        )
(("SN" "P0") 7          0           t        )
(("SP" "P0") 8          0           t        )
(("MVP" "P0") 44         0           t        )
(("MWN" "P0") 45         0           t        )
(("SAB" "P0") 48         0           t        )
(("CT" "P0") 9          0           l        )
(("MIM" "P0") 58         0           t        )
(("M1_ment" "P0") 10        0           t        )
(("M2_ment" "P0") 11        0           t        )
(("M3_ment" "P0") 12        0           t        )
(("M4_ment" "P0") 13        0           t        )
(("M5_ment" "P0") 14        0           t        )
(("M6_ment" "P0") 15        0           t        )
(("Vial" "annotate") 16        0           t        )
(("Vla2" "annotate") 17        0           t        )
(("Vla3" "annotate") 10        0           t        )
(("Vla4" "annotate") 19        0           t        )
(("Vla5" "annotate") 20        0           t        )
(("NHN" "annotate") 85        0           t        )
(("HRPDJUN" "annotate") 92        0           t        )
(("Rcs_NU" "annotate") 95        0           t        )
(("Res_P1" "annotate") 96        0           t        )
(("Res_OA" "annotate") 97        0           t        )
(("UID" "annotate") 300        0           t        )
(("MAP" "annotate") 131        1           t        )
(("RTTM" "annotate") 134        0           t        )
(("CAPRP" "annotate") 137        0           t        )
(("DSTR" "annotate") 138        0           t        )
(("M1Res_ment" "annotate") 171        0           t        )
(("M2Res_ment" "annotate") 172        0           t        )
(("M3Res_ment" "annotate") 173        0           t        )
(("M4Res_ment" "annotate") 174        0           t        )
1,1           Top

```

4. Identify the device you want to map.
5. Determine the layers that device is composed of using the LVS rule file. The following excerpt shows the layers that make up the MN(nch) device.

```

// LVS rule file

Layer active 1
Layer nplus 2
Layer poly 3           <---- poly has a GDS number of 3

// NMOS Device
ndiff = active and nplus
ngate = poly and ndiff   <---- device contains poly
nsd = ndiff not ngate

DEVICE MN(nch) ngate poly(G) nsd(S) nsd(D)

```

6. Determine the mapping of device layers to drawn layers.

For example, the layer ngate in the above rule file maps to the drawn layer poly, which has GDS layer number 3.

7. Check the Cadence technology file and find the Cadence layer names that corresponds to each GDS layer number you plan to map.

In the example rule file from Step 5, we want to find the Cadence layer name for layer 3. From [Figure A-2](#) (or the following excerpt), you can see that Cadence layer name PO has a GDS number of 3.

```
streamLayers(
; ( Layer           streamNumber dataType translate )
; ( -----          ----- ----- ----- )
( ("act" "drawing")      1           0       t      )
( ("PO"  "drawing")      3           0       t      )
```

8. In the cellmap file, map the Calibre layers to the Cadence layers you identified in the technology file.

In this example, derived Calibre layer ngate corresponds to drawn layer poly, which is mapped to the Cadence layer PO.

```
;Layer mapping part of cell map file

mgc_layer_map( list(
list( "ngate" '(PO drawing))
list( "poly"  '(PO drawing))
list( "nsd"   '(act drawing))
)
)
```

9. Map the remaining device layers using the steps in this procedure.

In the cellmap except shown with Step 8, Calibre layer poly is mapped to Cadence layer PO, and Calibre derived layer nsd is mapped to Cadence layer act.

## Setting the Calibre View Cellmap File in Calibre Interactive

You can specify the cellmap file for Calibre View generation using Calibre Interactive PEX or Calibre Interactive xACT. At run time, the Calibre View cellmap file is also entered in the Cadence Virtuoso “Calibre View Setup” dialog box.

---

### Note

 If the “Calibre View Setup” dialog box is already open at run time, the cellmap setting in the dialog box takes precedence over the setting in Calibre Interactive—the setting in the Calibre View Setup dialog box is used during the run and the dialog box setting is not updated with the Calibre Interactive setting.

In addition, the cellmap file defined in the CIW with mgc\_eview\_globals->cellMapFile takes precedence over the cellmap file defined in Calibre Interactive.

---

## Prerequisites

- A design is open in Cadence Virtuoso.

- The [Calibre View Setup Dialog Box](#) is not open.
- Calibre Interactive PEX or Calibre Interactive xACT is open and connected to the Cadence Virtuoso session.
- Calibre Interactive is set up to generate a Calibre View netlist.

For Calibre Interactive PEX, see “[Creating a Calibre View](#)” on page 493. For an example, see “[Creating a Post-Parasitic Extraction Calibre View](#)” on page 526.

## Procedure

1. Open the **Options** page (**Settings > Show Pages > Options**).
2. Check the “CALIBREVIEW Cellmap Files” checkbox.
3. Enter one or more cellmap files.

## Results

The specified Calibre View cellmap file is used for Calibre View generation. If the Calibre View Setup dialog box pops up during the run, the Calibre Interactive setting for the cellmap file is present in the dialog box.

If multiple cellmap files are specified the files are concatenated in the order specified. If there are two cellmap statements for the same device, the last statement is used.

## Importing Schematic Properties for Calibre View

Because the Cadence Analog Design framework treats net and instance names in schematics case sensitively, you must add the Source Case statement to your SVRF rule file when creating a Calibre View.

For example:

**Source Case YES**

Adding this statement enables highlighting by the Calibre RVE tool in the Cadence schematics and the importation of schematic properties by the Calibre View generator.

You may also need to use the SVRF statements Layout Case and LVS Compare Case, depending on your design.

The global variable mgc\_eview\_globals->importSchematicProperties controls whether schematic properties are imported; the default is t (or true). You can specify importSchemProp for a particular device in the cell map file in order to override the global setting. See these relevant topics:

- “[Global Variables for Calibre View Generation](#)” on page 542
- “[Cellmap Syntax for Intentional Devices](#)” on page 500

- “[Device Property Calculation in Calibre View Generation](#)” on page 496

You can specify `mgc_eview_globals->debug_instance_properties_sources` to print information to the log file about whether a parameter is from the schematic or layout.

When searching for a schematic instance in order to import properties, if an exact match by name is not found, then alternative schematic instances names are searched for. In particular, if no hierarchical schematic instance can be retrieved, properties are copied from the parent instance if it exists. For example, if the “I0/p0/\_MP” instance is not placed in schematic, then properties are retrieved from “I0/p0”, if it exists.

Also see “[Creating a Calibre View](#)” on page 493.

## Auto-Mapping Behavior, Control, and Debug in Calibre View Generation

Intentional devices and cells without mapping information in the cellmap file are automatically mapped (auto-mapped) and instantiated in the generated Calibre View. By default, the mapping uses case-sensitive matching of pin names followed by order-based mapping. You can specify a different auto-map behavior with the environment variable

`MGC_FDI_PIN_AUTOMAP_METHOD`. You can also specify that information about the auto-map process be printed to the transcript with the environment variable

`MGC_FDI_ENABLE_AUTOMAP_LOGGING`.

Auto-mapping often takes place in the gate-level extraction flow using LVS Box. The contents of the boxed cells are not extracted and do not appear in the Calibre View netlist. However, the boxed cells appear in the netlist as a device.

### Auto-Map Control

Auto-mapping is controlled with the environment variable

`MGC_FDI_PIN_AUTOMAP_METHOD`. The allowed values are described in the following table.

**Table A-9. MGC\_FDI\_PIN\_AUTOMAP\_METHOD Values**

Value	Auto-Map Behavior
0 (default)	Apply case-sensitive pin name matching first, then apply order-based mapping.
1	Apply case-insensitive pin name matching first, then apply order-based mapping.
2	Apply case-sensitive pin name matching only.
3	Apply case-insensitive pin name matching only.
4	Apply order-based pin mapping only.

## Auto-Map Behavior

If auto-mapping fails for a device, the device placement is skipped. Auto-mapping fails in these situations:

- The pin count in the Calibre View netlist is greater than the pin count in the master for the matching schematic instance. (Warning FDI3046)
- More than one Calibre View netlist pin maps to the same pin in the Cadence schematic view. (Warning FDI3047)
- A mapping cannot be determined. (Warning FDI3044)

For example, if case-sensitive name matching 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.

Devices are auto-mapping with a warning in this case:

- 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. (Warning FDI3045)

## Debug Output

Define the environment variable MGC\_FDI\_ENABLE\_AUTOMAP\_LOGGING to enable informational messages about the auto-mapping process. These messages are printed to the Calibre View log file (*calview.log* by default).

The following messages are possible:

- INFO: Auto mapping cell *<var>* successfully done according to the netlist.  
This message is printed when all pins are mapped by name.
- INFO: Auto mapping cell *<var>*, the following mapping is done: *<pin mapping>*  
This message is printed when mapping only by name does not succeed and order-based mapping is used.

## Related Topics

[Calibre View Generation Warnings](#)

## Calibre View Setup Dialog Box

Open with: **Calibre > Setup > Calibre View** from Cadence Virtuoso

The Calibre View Setup dialog box has settings for several aspects of the Calibre View creation process.

**Table A-10. Calibre View Setup Dialog Box Contents**

Option	Description
CalibreView Setup File	<p>Specifies a CalibreView Setup File. Click <b>View</b> to view the file and <b>Load</b> to load the parameters into the Calibre View Setup dialog box. The parameters are not used unless you click <b>Load</b>.</p> <p>This filename is also used as the name of the setup file when you click the <b>Save</b> button.</p> <p>See “<a href="#">Creating a CalibreView Setup File</a>” on page 534 for information on the setup file.</p>
CalibreView Netlist File	The pathname of an existing CalibreView netlist file. This field is filled in by the tool when the dialog box is opened automatically during a Calibre Interactive run.
Output Library	The library for the design.
Schematic Library	The library for the schematic.
Cellmap File	<p>The cellmap or iCellMap filename. See “<a href="#">Identifying Devices with a Cellmap File</a>” on page 498 and “<a href="#">Syntax for Cellmap Statements</a>” on page 500.</p> <p>Multiple cellmap files may be specified; enter multiple files as a space-separated list. Files are concatenated in the order specified. If there are two cellmap statements for the same device, the last statement is used.</p> <p><b>i Tip:</b> You can enter the cellmap file in Calibre Interactive PEX. See “<a href="#">Setting the Calibre View Cellmap File in Calibre Interactive</a>” on page 517. Only one cellmap file may be specified in Calibre Interactive.</p>
Log File	The name of the log file for Calibre View creation.
Calibre View Name	The view name for the Calibre View.
Calibre View Type	The view type, which can be “maskLayout” or “schematic.” A Calibre View Type of “maskLayout” permits both geometries and schematic symbols to be drawn together.
Create Terminals	Selects the mode for creating terminals: <ul style="list-style-type: none"> <li>• When matching terminal exists on symbol</li> <li>• Create all terminals</li> </ul>
Preserve Device Case	Specifies whether to preserve device case in netlist files. Select this option to ensure that mixed case netlist files have their capitalization maintained throughout Calibre processing. The case of pin names is always ignored, regardless of this setting.

**Table A-10. Calibre View Setup Dialog Box Contents (cont.)**

Option	Description
Execute Callbacks	Specifies whether to execute CDF callback functions. Also see “ <a href="#">Full Syntax for Property Specification in the Cellmap File</a> ” on page 504 and <code>mgc_eview_globals-&gt;executeCallbacks</code> in “ <a href="#">Global Variables for Calibre View Generation</a> ” on page 542.
Suppress Notes	Specifies whether informational notes are output to the transcript.
Reset Properties	Specify properties to reset to a specific value after being copied from the schematic. Only simple assignment is done; you cannot assign to a variable or another property value.
Magnify Instances By <sup>1</sup>	Magnification factor for intentional devices and parasitics. The magnification factor is useful to prevent overlapping of instances when the Calibre View is generated in maskLayout view. Specify a magnification factor less than one to reduce overlap.
Device Placement	Specifies method for device placement: <ul style="list-style-type: none"> <li>• Layout Location</li> <li>• Arrayed</li> </ul>
Parasitic Placement	Specifies method for placement of parasitic elements: <ul style="list-style-type: none"> <li>• Layout Location</li> <li>• Arrayed</li> </ul> Parasitic placement is only performed if “Calibre View Type” is set to “maskLayout.” “Parasitic Placement” should be set to “Layout Location” if net geometries are included in the Calibre View.
Show Parasitic Polygons	Specifies whether to show polygons for the parasitic elements. If this setting is enabled, then “Parasitic Placement” must be set to “Layout Location” and “Calibre View Type” must be set to “maskLayout.” “Show Parasitic Polygons” should be enabled if net geometries are included in the Calibre View.
Open Calibre CellView	Specifies the mode for opening the Calibre View: <ul style="list-style-type: none"> <li>• Read-mode</li> <li>• Edit-mode</li> <li>• Don’t Open</li> </ul>

**Table A-10. Calibre View Setup Dialog Box Contents (cont.)**

Option	Description
Generate SPECTRE Netlist	Specifies whether to generate a SPECTRE netlist in addition to the Calibre View netlist. See “ <a href="#">Creating a SPECTRE Netlist from the Calibre View Netlist</a> ” on page 533.
Always Show Dialog	Specifies whether to automatically open the Calibre View Setup dialog box during Calibre View creation.
<b>Buttons</b>	
OK	Create a CalibreView using the settings in the dialog box.
Cancel	Discard any modifications made in the dialog box and close it.
Save	Save the parameters specified in the dialog box to a setup file and close the dialog box. If a filename is specified in the “CalibreView Setup File” field, that filename is used, otherwise the default filename of <i>calibreview.setup</i> is used. In addition, the parameters are used the next time you open the Calibre View Setup dialog box, and when <a href="#"><code>mgc_rve_create_cellview</code></a> is called.

<sup>1</sup> For Cadence Virtuoso CDBA-based databases (Cadence Virtuoso version 5.1.4 and earlier) the dialog box has the entries “Magnify Devices By” and “Magnify Parasitics By”.

## Related Topics

[Creating a CalibreView Setup File](#)

[Showing Parasitic Polygons in Calibre View](#)

[Specifying Extracted Netlist Annotation of Net Geometries to Calibre View](#)

## Creating a Post-LVS Calibre View

Calibre Interactive and Calibre RVE generate a Calibre View for the top-level cell, regardless of the cell context set in the tools. The tools generate net and device names in the Calibre View from the source (or schematic) name space, unless nets and devices were unmatched during LVS comparison. In the latter case, the tool prefixes layout names with a “\_Layout”.

### Prerequisites

- Ensure you have the necessary input files. See “[What You Need Before Creating a Calibre View](#)” on page 494.
- If importing schematic properties, make sure Source Case, Layout Case, and LVS Compare Case SVRF statements are in your SVRF rule file.

Because the Cadence Analog Design framework treats net and instance names in schematics case sensitively, you must use the following statement in your SVRF rule file:

#### Source Case YES

Adding this statement enables highlighting by the Calibre RVE tool in the Cadence schematics and the importation of schematic properties by the Calibre View generator. Also see “[Importing Schematic Properties for Calibre View](#)” on page 518.

- Create the cellmap file. See “[Creating the Cellmap File](#)” on page 498.
- (Optional) Set the environment variable CALIBRE\_DISABLE\_PPAR\_WARNINGS to disable pPar evaluation warnings.

### Procedure

1. Open the layout in Cadence Virtuoso.
2. Choose **Calibre > Run nmLVS**.

This invokes Calibre Interactive nmLVS.

---

#### Note

 Leave the “Export from layout viewer” button checked on the **Layouts** tab of the Inputs pane.

---

3. Open the **Outputs** page and do the following:
  - a. Enable “Mask SVDB.”
  - b. Enter the “SVDB Directory” name.
  - c. Enable the “Generate Calibre Connectivity Interface Data” option.
4. In the Run Control page, enable “Start RVE after LVS finishes.” section
5. Set all other required parameters for the **Rules**, **Inputs**, **Outputs** and **Options** pages, as described in “[Setting Up a Calibre Interactive nmLVS Run](#)” on page 173.
6. Click **Run LVS**.

Upon completion of the Calibre nmLVS-H run, Calibre RVE automatically opens with the results of the run.

7. In Calibre RVE, choose **Tools > Export CalibreView**.

---

#### Tip

 If you did *not* open Calibre RVE from Calibre Interactive, choose **Highlight > Setup Cell Map** before creating the Calibre View and confirm that the cell mapping is correct. See “[Map Calibre RVE Cells to Different Cadence Library for Highlighting](#)” in the *Calibre RVE User’s Manual*.

---

8. Click **OK** in the Export Calibre View dialog box.
9. Make selections in the Calibre View Setup dialog box. If the Calibre View Setup dialog box does not open automatically, choose **Calibre > Setup > Calibre View** and enable “Always Show Dialog.” Most dialog settings are self-explanatory; refer to “[Calibre View Setup Dialog Box](#)” on page 520 or see below for tips on setting some of the fields:
  - **CalibreView Setup File** — Enter a CalibreView setup file then click **Load** to populate the dialog box with the settings. If the setup file includes all the necessary settings, proceed to Step 10. See “[Creating a CalibreView Setup File](#)” on page 534.
  - **CalibreView Netlist File** — Enter the name of an existing CalibreView netlist file. This field is filled in automatically when the dialog box is opened automatically during a Calibre Interactive run.
  - **Output Library** — Enter the library name for the design.
  - **Cellmap File** — Enter the cellmap filename. See “[Cellmap Syntax for Intentional Devices](#)” on page 500 for cellmap creation procedures. Multiple cellmap files may be specified; enter multiple files as a space-separated list. Files are concatenated in the order specified. You may also specify an iCellMap file.
  - **Calibre View Type** — Set the Calibre View Type to “maskLayout.” This view permits both geometries and schematic symbols to be drawn together.
  - **Preserve Device Case** — Select this option to ensure that mixed case netlist files have their capitalization maintained throughout Calibre processing. The case of pin names is always ignored, regardless of this setting.
  - **Reset Properties** — Specify properties you want reset to a specific value after being copied from the schematic. Only simple assignment is done; you cannot assign to a variable or another property value.
  - **Magnify Instances By** — Enter the magnification value the tool applies to intentional and parasitic devices. Specify a value less than one to prevent overlap of instances when generating a maskLayout Calibre View.
  - **Generate SPECTRE Netlist** — Generate a SPECTRE netlist in addition to the Calibre View netlist. See “[Creating a SPECTRE Netlist from the Calibre View Netlist](#)” on page 533.
10. Click **OK** to create the CalibreView and save the settings in the dialog box.

---

**Note**

 Wire stubs with net name labels are available in maskLayout view type. Similar to Schematics view type, the net names associated with the wires should also be created.

To make the wires selectable in CalibreView, choose **Options > Display**. In the Options dialog box, set both the X Snap Spacing and Y Snap Spacing fields to smaller values

---

(e.g. 0.001); verify that the layer purpose pair (“wire” “drawing”) is both visible and selectable in the Layer Selection Window.

---

## Results

The Calibre View netlist is created.

In addition, the file *calibreview.setup* is created in the run directory with the current Calibre View settings. This file is in the [CalibreView Setup File Format](#). The file can be loaded into the Calibre View Setup dialog box to populate the dialog box with settings for the run. Browse for the file or enter the filename into the “Calibreview Setup File” field and click the **Load** button. It can also be used as input to the SKILL function [mgc\\_rve\\_load\\_setup\\_file](#) to create the Calibre View in batch mode.

## Related Topics

[Mapping Schematic Nets to Calibre View Nets for Simulation and Plotting](#)

[Backannotation of Parasitic Values to the Schematic in Cadence Virtuoso](#)

[Showing Parasitic Polygons in Calibre View](#)

[Creating a Calibre View in Batch Mode](#)

## Creating a Post-Parasitic Extraction Calibre View

The post-parasitic extraction Calibre View includes parasitic elements as well as intentional devices.

### Prerequisites

- Ensure you have the necessary input files. See “[What You Need Before Creating a Calibre View](#)” on page 494.
- If importing schematic properties, make sure Source Case, Layout Case, and LVS Compare Case SVRF statements are in your SVRF rule file.

Because the Cadence Analog Design framework treats net and instance names in schematics with case sensitivity, you must use the following statement in your SVRF rule file:

**Source Case YES**

This statement enables highlighting by Calibre RVE in the Cadence schematics and the importation of schematic properties by the Calibre View generator. Also see “[Importing Schematic Properties for Calibre View](#)” on page 518.

- Create the cellmap file. See “[Creating the Cellmap File](#)” on page 498.
- (Optional) Set the environment variable CALIBRE\_DISABLE\_PPAR\_WARNINGS to disable pPar evaluation warnings.

## Procedure

1. Open the layout in Cadence Virtuoso.

2. Choose **Calibre > Run Pex**.

**Note**

 Leave “Export from layout viewer” enabled on the **Layout** tab of the Inputs pane.

---

3. On the **Rules** page, and enter the filename for the SVRF rule file.

4. Click the **Load** button () to load the rule file.

5. Open the **Inputs** page and fill in the fields as described in “[Inputs for a PEX Run](#)” on page 202.

6. Open the **Outputs** page.

7. Select the PEX Netlist section. For Netlist Format, select CALIBREVIEW from the dropdown button list.

8. Select “Source Names” for “Use Names From.”

This specifies SOURCENAMES in PEX Netlist.

9. Set all other required parameters for the Outputs pane and for PEX Options.

Also see “[Showing Parasitic Polygons in Calibre View](#)” on page 535 for instructions on setting parameters to show parasitic polygons.

10. Click **Run PEX**.

The tool creates the post-parasitic extraction view and automatically begins importing the Calibre View into the Cadence Environment.

11. Make selections in the Calibre View Setup dialog box. If the Calibre View Setup dialog box does not open automatically, choose **Calibre > Setup > Calibre View** from Cadence Virtuoso and enable “Always Show Dialog.” Most dialog settings are self-explanatory; refer to “[Calibre View Setup Dialog Box](#)” on page 520 or see below for tips on setting some of the fields:

- **CalibreView Setup File** — Enter a CalibreView setup file then click **Load** to populate the dialog box with the settings. If the setup file includes all the necessary settings, proceed to Step 12. See “[Creating a CalibreView Setup File](#)” on page 534.
- **CalibreView Netlist File** — Enter the name of an existing CalibreView netlist file. This field is filled in automatically when the dialog box is opened during a Calibre Interactive run.
- **Output Library** — Enter the library name for the design.

- **Cellmap File** — Enter the cellmap filename. See “[Cellmap Syntax for Intentional Devices](#)” on page 500 for cellmap creation procedures. Multiple cellmap files may be specified; enter multiple files as a space-separated list. Files are concatenated in the order specified. You may also specify an iCellMap file.  
You can also specify the cellmap file in Calibre Interactive; see “[Setting the Calibre View Cellmap File in Calibre Interactive](#)” on page 517.
- **Calibre View Type** — Set the Calibre View Type to “maskLayout.” This view permits both geometries and schematic symbols to be drawn together.
- **Preserve Device Case** — Select this option to ensure that mixed case netlist files have their capitalization maintained throughout Calibre processing. The case of pin names is always ignored, regardless of this setting.
- **Reset Properties** — Specify properties you want reset to a specific value after being copied from the schematic. Only simple assignment is done; you cannot assign to a variable or another property value.
- **Magnify Instances By** — Enter the magnification value the tool applies to intentional and parasitic devices. Specify a value less than one to prevent overlap of instances when generating a maskLayout Calibre View.
- **Generate SPECTRE Netlist** — Generate a SPECTRE netlist in addition to the Calibre View netlist. See “[Creating a SPECTRE Netlist from the Calibre View Netlist](#)” on page 533.

---

**Tip**

 Symbols are generally smaller when viewed on top of a layout, so magnifying them is recommended.

---

12. Click **Save** to save the settings and exit the Calibre View Setup dialog box. The CalibreView is created during the Calibre run.

The **OK** button creates the Calibre View immediately.

## Results

The CalibreView netlist is created.

In addition, the file *calibreview.setup* is created in the run directory with the current Calibre View settings. This file is in the [CalibreView Setup File Format](#). The file can be loaded into the Calibre View Setup dialog box to populate the dialog box with settings for the run. Browse for the file or enter the filename into the “Calibreview Setup File” field and click the **Load** button. It can also be used as input to the SKILL function `mgc_rve_load_setup_file` to create the Calibre View in batch mode.

## Related Topics

[Mapping Schematic Nets to Calibre View Nets for Simulation and Plotting](#)

[Backannotation of Parasitic Values to the Schematic in Cadence Virtuoso](#)

[Showing Parasitic Polygons in Calibre View](#)

[Creating a Calibre View in Batch Mode](#)

## Creating a Gate-Level Calibre View

A gate-level Calibre View includes parasitics from the top-level routing plus symbols for the cells.

### Prerequisites

- Ensure you have the necessary input files. See “[What You Need Before Creating a Calibre View](#)” on page 494.
- If importing schematic properties, make sure Source Case, Layout Case, and LVS Compare Case SVRF statements are in your SVRF rule file.

Because the Cadence Analog Design framework treats net and instance names in schematics with case sensitivity, use the following statement in your SVRF rule file:

**Source Case YES**

This statement enables highlighting by Calibre RVE in the Cadence schematics and the importation of schematic properties by the Calibre View generator. Also see “[Importing Schematic Properties for Calibre View](#)” on page 518.

- Create the cellmap file. See “[Creating the Cellmap File](#)” on page 498.

For a gate-level Calibre View, the cellmap file needs to include entries for each cell; these entries follow the syntax given in “[Cellmap Syntax for Cells](#)” on page 502.

- (Optional) Set the environment variable CALIBRE\_DISABLE\_PPAR\_WARNINGS to disable pPar evaluation warnings.

### Procedure

1. Open the layout in Cadence Virtuoso.
2. Choose **Calibre > Run PEX**.

---

**Note**

 Leave the “Export from layout viewer” option enabled on the **Layout** tab of the **Inputs** pane.

---

3. Open the **Rules** page and enter the filename for the SVRF rule file.
4. Click the **Load** button () to load the rule file.
5. Open the **Inputs** page. Fill in the fields as described in “[Inputs for a PEX Run](#)” on page 202.

6. Open the **Outputs** page and do the following:
  - a. Under the Extraction Type section, for “Level,” specify “Gate Level.”
  - b. Under the PEX Netlist section:
    - i. For “Netlist Format,” select “CALIBREVIEW” from the dropdown list.
    - ii. For “Use Names From,” select “Source Names” from the dropdown list. This specifies SOURCENAMES in PEX Netlist.
7. Under the PEX Netlist section, do the following:
8. Set all other required parameters for the **Outputs** and **Options** pages.

See “[Showing Parasitic Polygons in Calibre View](#)” on page 535 for instructions in setting parameters to show parasitic polygons.

9. Click **Run PEX**.

The tool creates the post-parasitic extraction view and automatically begins importing the Calibre View into the Cadence environment.

10. Make selections in the Calibre View Setup dialog box. If the Calibre View Setup dialog box does not open automatically, choose **Calibre > Setup > Calibre View** from Cadence Virtuoso and enable “Always Show Dialog.” Most dialog settings are self-explanatory; refer to “[Calibre View Setup Dialog Box](#)” on page 520 or see below for tips on setting some of the fields:
    - **CalibreView Setup File** — Enter a CalibreView setup file then click **Load** to populate the dialog box with the settings. If the setup file includes all the necessary settings, proceed to Step 11. See “[Creating a CalibreView Setup File](#)” on page 534.
    - **CalibreView Netlist File** — Enter the name of an existing CalibreView netlist file. This field is filled in automatically when the dialog box is opened during a Calibre Interactive run.
    - **Output Library** — Enter the library name for the design.
    - **Cellmap File** — Enter the cellmap filename. See “[Cellmap Syntax for Intentional Devices](#)” on page 500 for cellmap creation procedures. Multiple cellmap files may be specified; enter multiple files as a space-separated list. Files are concatenated in the order specified. You may also specify an iCellMap file.
- You can also specify the cellmap file in Calibre Interactive; see “[Setting the Calibre View Cellmap File in Calibre Interactive](#)” on page 517.
- **Calibre View Type** — Set the Calibre View Type to “maskLayout.” This view permits both geometries and schematic symbols to be drawn together.

- **Preserve Device Case** — Select this option to ensure that mixed case netlist files have their capitalization maintained throughout Calibre processing. The case of pin names is always ignored, regardless of this setting.
- **Reset Properties** — Specify properties you want reset to a specific value after being copied from the schematic. Only simple assignment is done; you cannot assign to a variable or another property value.
- **Magnify Instances by** — Enter the magnification value the tool applies to intentional and parasitic devices. Specify a value less than one to prevent overlap of instances when generating a maskLayout Calibre View.
- **Generate SPECTRE Netlist** — Generate a SPECTRE netlist in addition to the Calibre View netlist. See “[Creating a SPECTRE Netlist from the Calibre View Netlist](#)” on page 533.

---

**Tip**  Symbols are generally smaller when viewed on top of a layout, so magnifying them is recommended.

---

11. Click **Save** to save the settings and exit the Calibre View Setup dialog box. The CalibreView is created during the Calibre run.

Note: The **OK** button creates the Calibre View immediately.

## Results

The CalibreView netlist is created.

In addition, the file *calibreview.setup* is created in the run directory with the current Calibre View settings. This file is in the [CalibreView Setup File Format](#). The file can be loaded into the Calibre View Setup dialog box to populate the dialog box with settings for the run. Browse for the file or enter the filename into the “Calibreview Setup File” field and click the **Load** button. It can also be used as input to the SKILL function [`mgc\_rve\_load\_setup\_file`](#) to create the CalibreView in batch mode.

## Related Topics

- [Mapping Schematic Nets to Calibre View Nets for Simulation and Plotting](#)
- [Backannotation of Parasitic Values to the Schematic in Cadence Virtuoso](#)
- [Showing Parasitic Polygons in Calibre View](#)
- [Creating a Calibre View in Batch Mode](#)

## Creating Calibre Views in a Multiple Corner Extraction Run

When you generate a Calibre View with multiple corners selected for extraction, a Calibre View is created for each corner.

### Prerequisites

- Ensure you have the necessary input files. See “[What You Need Before Creating a Calibre View](#)” on page 494.
- Calibre Interactive PEX is open.

### Procedure

1. In the Outputs page, check the “Corners” checkbox and enter the corners to extract.
2. Set up a Calibre Interactive run for Calibre View generation:
  - “[Creating a Post-Parasitic Extraction Calibre View](#)” on page 526
  - “[Creating a Gate-Level Calibre View](#)” on page 529
3. Click **Run PEX**.

The tool creates the post-parasitic extraction view and automatically begins importing the Calibre View into the Cadence environment.

4. Make selections in the Calibre View Setup dialog box. See the procedures noted in Step 2 for details on the dialog box entries.  
  
A dialog box opens for each corner that is extracted. The view name and netlist file are named automatically as follows, where the examples assume corners are named typical, MIN, and MAX.

Calibre View name: calibre\_<corner>

Example: calibre\_typical, calibre\_MIN, calibre\_MAX

Calibre View netlist name:

- For netlists with a period in the name, insert “\_<corner>” before the last period, however, do not change the netlist name for a corner named typical.  
  
Example, for an original netlist name of *opamp.pex.netlist*: *opamp.pex.netlist*, *opamp.pex\_MIN.netlist*, *opamp.pex\_MAX.netlist*
- For netlist without a period in the name, the Calibre View netlist name is <netlist>\_<corner>.<netlist\_minus\_last\_character>, however , do not change the netlist name for a corner named typical.

Example, for an original netlist name of *opamp\_netlist*: *opamp\_netlist*, *opamp\_netlist\_MINUS.opamp\_netlist*, *opamp\_netlist\_MAX.opamp\_netlist*

Click **OK** to close each Calibre View Setup dialog box.

## Creating a SPECTRE Netlist from the Calibre View Netlist

You can generate a SPECTRE netlist from the Calibre View netlist using the Cadence Analog Design Environment (ADE). Cellmap information is used in the netlist generation, unlike the case when generating a SPECTRE netlist from Calibre xRC, in which source names are used.

### Prerequisites

- Ensure you have the necessary input files. See “[What You Need Before Creating a Calibre View](#)” on page 494.
- If importing schematic properties, then manually add Source Case, Layout Case, and LVS Compare Case SVRF statements to your SVRF rule file.

Because the Cadence Analog Design framework treats net and instance names in schematics case sensitively, you must add the following statement to your SVRF rule file:

**Source Case YES**

Adding this statement enables highlighting by the Calibre RVE tool in the Cadence schematics and the importation of schematic properties by the Calibre View generator. Also see “[Importing Schematic Properties for Calibre View](#)” on page 518.

- Create the cellmap file. See “[Creating the Cellmap File](#)” on page 498.
- (Optional) Set the environment variable CALIBRE\_DISABLE\_PPAR\_WARNINGS to disable pPar evaluation warnings.

### Procedure

1. Set up to produce a Calibre View netlist as described in “[Creating a Post-LVS Calibre View](#)” on page 523 or “[Creating a Post-Parasitic Extraction Calibre View](#)” on page 526.
2. In the Calibre View Setup dialog box, enable “Generate SPECTRE Netlist.”  
If the Calibre View Setup dialog box does not open automatically, choose **Calibre > Setup > Calibre View** from Cadence Virtuoso.
3. Click **Save** to save the settings and close the Calibre View Setup dialog box.  
The Map Calibre Device dialog box opens automatically.
4. Make appropriate selections in the Map Calibre Device dialog box, then close it.  
The Cadence Virtuoso Analog Design Environment window opens, and then a window with the SPECTRE netlist opens.
5. In the window with the SPECTRE netlist, choose **File > Save As** to save the netlist.

**Tip**

 Once a Calibre View netlist has been created, you can create the SPECTRE netlist from the CIW with the following command:

```
mgc_eview_run_netlister("lib_name" "cell_name" "view_name")
```

## Creating a CalibreView Setup File

The CalibreView setup file contains all the settings necessary to generate a CalibreView. You can use the file as input to the SKILL function `mgc_rve_load_setup_file` to create the CalibreView in batch mode, without opening the Calibre View Setup dialog box. The file can also be loaded into the Calibre View Setup dialog box in order to populate the dialog box with saved settings.

**Note**

 The file `calibreview.setup` is automatically created in the run directory after Calibre View generation. The file contains the Calibre View settings and is in the [CalibreView Setup File Format](#). You can edit this file to create a setup file with different parameters.

You can also create a setup file using the Calibre View Setup dialog box.

## Procedure

- Do one of the following, depending on whether you have generated a Calibre View:

If ...	Do this ...
You have generated a Calibre View and want to use most of the same parameters	Copy <code>calibreview.setup</code> to another filename, such as <code>newcalibreview.setup</code> .
You have <i>not</i> generated a Calibre View and want a default setup file	<ol style="list-style-type: none"> <li>Execute the SKILL function: <code>mgc_rve_default_setup_file</code> This creates the file <code>.calibreview_default_setup_file</code> in the directory you started Cadence Virtuoso from.</li> <li>Copy <code>.calibreview_default_setup_file</code> to another filename, such as <code>newcalibreview.setup</code>.</li> </ol>
You want to enter parameters for the setup file from the Calibre View Setup dialog box	<ol style="list-style-type: none"> <li>From Cadence Virtuoso, select <b>Calibre &gt; Setup &gt; Calibre View</b>.</li> <li>Enter the filename for the setup file in the “CalibreView Setup File” field, or leave it blank to use the default filename of <code>calibreview.setup</code>.</li> <li>Enter parameters in the dialog box.</li> <li>Click the <b>Save</b> button at the bottom of the dialog box.</li> </ol>

2. Edit your CalibreView setup file (*newcalibreview.setup* for example) and provide required file names, library settings, and other parameters. See “[CalibreView Setup File Format](#)” on page 536 for details.

This step is not necessary if you created the setup file using the Calibre View Setup dialog box and entered all the desired parameters.

3. Save and exit the file.

---

**Note**

 See “[Creating a Calibre View in Batch Mode](#)” on page 543 for instructions on creating the CalibreView in batch mode using the CalibreView setup file.

You can also load the file into the Calibre View Setup dialog box—browse to the file and click **Load** to populate the dialog box with the settings.

---

## Showing Parasitic Polygons in Calibre View

You can use the display and annotation of parasitic polygons to more easily analyze design parasitics.

See “[Specifying Extracted Netlist Annotation of Net Geometries to Calibre View](#)” on page 210.

## CalibreView Setup File Format

Input for: The SKILL function `mgc_rve_load_setup_file` and the Calibre View Setup dialog box  
The CalibreView setup file allows you to specify the parameters for CalibreView creation without opening the Calibre View Setup dialog box.

See “[Creating a CalibreView Setup File](#)” on page 534 for instructions on creating a setup file. The file `calibreview.setup` is automatically created in the run directory after Calibre View generation. The file contains the Calibre View settings.

### Format

- ASCII text file. Blank lines are ignored, and white space at the beginning of a line is ignored.
- Comments are not allowed.
- Keywords and values are not case-sensitive.
- Only one keyword/value pair is allowed on each line.

### File Contents

Each line has the following format:

`Keyword: value`

where the allowed keywords and values are given in the next section.

### Parameters

The allowed keywords and values for the CalibreView setup file are given in the following table.

**Table A-11. CalibreView Setup File Keywords and Values**

Keyword	Required or Optional	Allowed Values	Default Value
<code>calibre_view_netlist_file</code>	Required	A valid pathname for a CalibreView netlist.	None
<code>output_library</code>	Required	A valid Cadence library.	None
<code>schematic_library</code>	Required	A valid Cadence library.	None

**Table A-11. CalibreView Setup File Keywords and Values (cont.)**

Keyword	Required or Optional	Allowed Values	Default Value
cell_name	Optional	The cell name of the generated CalibreView.  This field is filled in by the tool when the <i>calibreview.setup</i> file is created during a run. The field is not required for successful batch generation of the Calibre View.	None
cellmap_file	Required	A valid CalibreView cellmap file.  Multiple cellmap files may be specified; enter multiple files as a space-separated list. Files are concatenated in the order specified.	None
calibreview_log_file	Optional	A valid path to the log file.	./calview.log
calibreview_name	Optional	A valid name.	calibre
calibreview_type	Optional	<ul style="list-style-type: none"> <li>• maskLayout</li> <li>• schematic</li> </ul> calibreview_type must be set to maskLayout to include net geometries in the CalibreView.	maskLayout
create_terminals	Optional	<ul style="list-style-type: none"> <li>• if_matching</li> <li>• all</li> </ul>	if_matching
preserve_device_case	Optional	<ul style="list-style-type: none"> <li>• on</li> <li>• off</li> </ul>	off
execute_callbacks	Optional	<ul style="list-style-type: none"> <li>• on</li> <li>• off</li> </ul>	off
suppress_notes	Optional	<ul style="list-style-type: none"> <li>• on</li> <li>• off</li> </ul>	off
reset_properties	Optional	A valid list of properties in this format:  (prop=value,prop=value,...)	(m=1)

**Table A-11. CalibreView Setup File Keywords and Values (cont.)**

Keyword	Required or Optional	Allowed Values	Default Value
magnify_devices_by	Optional	<p>A valid positive value.</p> <p>For OpenAccess databases<sup>1</sup>, this keyword affects magnification of both intentional and parasitic devices, and corresponds to “Magnify Instances By” in the Calibre View Setup dialog box.</p> <p>The magnification factor is useful to prevent overlapping of instances when the Calibre View is generated in maskLayout view. Specify a magnification factor less than one to prevent overlap.</p>	1
magnify_parasitics_by	Optional	<p>A valid positive value.</p> <p>Not used with OpenAccess databases<sup>1</sup>.</p>	1
device_placement	Optional	<ul style="list-style-type: none"> <li>• layout_location</li> <li>• arrayed</li> </ul>	layout_location
parasitic_placement	Optional	<ul style="list-style-type: none"> <li>• layout_location</li> <li>• arrayed</li> </ul> <p>Parasitic placement is only performed if calibreview_type is set to maskLayout.</p> <p>parasitic_placement should be set to layout_location if net geometries are included in the CalibreView.</p>	arrayed

**Table A-11. CalibreView Setup File Keywords and Values (cont.)**

Keyword	Required or Optional	Allowed Values	Default Value
show_parasitic_polygons	Optional	<ul style="list-style-type: none"> <li>• on</li> <li>• off</li> </ul> <p>If this value is on, then parasitic_placement must be set to layout_location and calibreview_type must be set to maskLayout.</p> <p>show_parasitic_polygons should be set to on if net geometries are included in the CalibreView.</p>	off
open_calibreview	Optional	<ul style="list-style-type: none"> <li>• read_mode</li> <li>• edit_mode</li> <li>• don't_open</li> </ul>	don't_open
generate_spectre_netlist	Optional	<ul style="list-style-type: none"> <li>• on</li> <li>• off</li> </ul>	off

<sup>1</sup>For CDBA-based databases, the keywords magnify\_devices\_by and magnify\_parasitics\_by correspond to “Magnify Devices by” and “Magnify Parasitics by” in the Calibre View Setup dialog box.

## Examples

The following is an example of a CalibreView setup file:

```

calibre_view_netlist_file : ./adder_4.pex.netlist
output_library : ADDER
schematic_library : ADDER
cell_name : adder_4
cellmap_file : ./calview.cellmap
calibreview_log_file : ./calview.log
calibreview_name : calibre
calibreview_type : maskLayout
create_terminals : if_matching
preserve_device_case : off
execute_callbacks : off
reset_properties : (m=1)
magnify_devices_by : 1
magnify_parasitics_by : 1
device_placement : layout_location
parasitic_placement : arrayed
show_parasitic_polygons : off
open_calibreview : don't_open
generate_spectre_netlist : off

```

## Related Topics

- [Creating a CalibreView Setup File](#)
- [Identifying Devices with a Cellmap File](#)

## SKILL Functions for Creating a Calibre View

Several functions are provided that allow you to create a Calibre View from the command interpreter without opening the Calibre Interactive GUI or Calibre RVE.

<a href="#">mgc_rve_default_setup_file</a>	<a href="#">mgc_rve_load_setup_file</a>
<a href="#">mgc_eview_display_setup_settings</a>	<a href="#">mgc_eview_run_netlister</a>
<a href="#">mgc_eview_export_from_layout</a>	<a href="#">mgc_rve_create_cellview</a>
<a href="#">mgc_eview_export_from_schematic</a>	<a href="#">mgc_rve_run_pex</a>
<a href="#">mgc_eview_post_execution_trigger</a>	

### **mgc\_rve\_default\_setup\_file**

#### **mgc\_rve\_default\_setup\_file**

Output a file named *.calibreview\_default\_setup\_file* to the directory Cadence Virtuoso was invoked from. The file *.calibreview\_setup\_file* is in the correct format for the CalibreView setup file and has default settings for most parameters. See the function [mgc\\_rve\\_load\\_setup\\_file\(\)](#) and “[Creating a CalibreView Setup File](#)” on page 534.

### **mgc\_eview\_display\_setup\_settings**

#### **mgc\_eview\_display\_setup\_settings("lib\_name" "cell\_name" "view\_name")**

Display the Calibre View setup settings that were used to create the specified cell view.

### **mgc\_eview\_export\_from\_layout**

#### **mgc\_eview\_export\_from\_layout( "libName" "cellName" "viewName")**

Export the netlist from the layout using the specified library name, cell name, and view name.

### **mgc\_eview\_export\_from\_schematic**

#### **mgc\_eview\_export\_from\_schematic( "libName" "cellName" "viewName")**

Export the netlist from the schematic using the specified library name, cell name, and view name.

### **mgc\_eview\_post\_execution\_trigger**

```
mgc_eview_post_execution_trigger("layLibName" "layCellName" "layViewName"  
"schLibName" "schCellName" "schViewName" "calView")
```

A trigger function that is called after the view is created and saved for the Calibre View, and after the SPECTRE netlist is generated. The trigger function is called with the indicated arguments, where *calView* is the handle to the new view. Define this function in your Cadence environment in order to use it.

### **mgc\_rve\_load\_setup\_file**

```
mgc_rve_load_setup_file("calibreview_setup_file")
```

Create a CalibreView using the parameters in the *calibreview\_setup\_file*. You can use `mgc_rve_default_setup_file()` to create a default file with the correct format. See “[CalibreView Setup File Format](#)” on page 536.

### **mgc\_eview\_run\_netlister**

```
mgc_eview_run_netlister("lib_name" "cell_name" "view_name")
```

Generate a SPECTRE netlist from the Calibre View netlist with the specified library, cell, and view. Also see “[Creating a SPECTRE Netlist from the Calibre View Netlist](#)” on page 533.

### **mgc\_rve\_create\_cellview**

```
mgc_rve_create_cellview(@optional "calibreview_netlist")
```

Create the Calibre View using the specified netlist. The Calibre View Setup dialog box opens in order to provide necessary Calibre View generation information. The netlist name is filled in if provided.

### **mgc\_rve\_run\_pex**

```
mgc_rve_run_pex("batch_script")
```

Executes the shell commands in the file *batch\_script*.

---

#### **Tip**

 Also see “[Global Variables for Calibre View Generation](#)” on page 542.

---

## **Related Topics**

[Creating a Calibre View](#)

[Creating a Calibre View in Batch Mode](#)

[Creating a CalibreView Setup File](#)

## Global Variables for Calibre View Generation

Several global variables are provided that are used in the creation of a Calibre View.

**Table A-12. Global Variables for Calibre View Generation**

Variable	Definition
mgc_eview_globals->cellMapFile	Path to cell map file
mgc_eview_globals->debug_instance_properties_sources	Specifies whether to print information to the log file about whether a parameter is from the schematic or layout. Allowed values are t or nil.
mgc_eview_globals->executeCallbacks	Specifies whether to execute callbacks. The default value is nil. Set this variable to t in order for callbacks to be executed.
mgc_eview_globals->extViewName	Specifies the extracted view name to search in when mapping schematic nets.  See “ <a href="#">Mapping Schematic Nets to Calibre View Nets for Simulation and Plotting</a> ” on page 552.
mgc_eview_globals->extViewType	View type
mgc_eview_globals->importSchematicProperties	Specifies whether to import schematic properties. Allowed values are t or nil. The default is t.  The global setting can be overridden for a particular device with the importSchemProp setting in the cellmap file; see “ <a href="#">Cellmap Syntax for Intentional Devices</a> ” on page 500.  Also see “ <a href="#">Device Property Calculation in Calibre View Generation</a> ” on page 496.
mgc_eview_globals->outputLibrary	Output library
mgc_eview_globals->pathSeparator	The path separator.
mgc_eview_globals->schematicLibrary	Schematic library. The default setting is outputLibrary.
mgc_eview_globals->showCalviewDlg	Specifies whether or not to automatically open the Calibre View Setup dialog box. Set to t or nil.

## Creating a Calibre View in Batch Mode

The SKILL function `mgc_rve_load_setup_file()` uses the parameters in the CalibreView setup file to create a CalibreView without opening the Calibre View Setup dialog box.

### Prerequisites

- “[What You Need Before Creating a Calibre View](#)” on page 494

### Procedure

1. Run Calibre, either from the command interpreter or as a batch run from the shell.  
To run calibre from the command interpreter, use the commands defined in the section “[SKILL Functions for Creating a Calibre View](#)” on page 540.
2. Create the CalibreView setup file as described in “[Creating a CalibreView Setup File](#)” on page 534.  
If the CalibreView includes net geometries make sure the CalibreView setup File parameters are set accordingly and that the following conditions are met:
  - The cellmap file includes layer mapping
  - The query\_results file generated by Calibre RVE exist in the run directory
3. Create the Calibre View by calling this function from the command interpreter or a SKILL file:

```
mgc_rve_load_setup_file("calibreview_setup")
```

where *calibreview\_setup* is the CalibreView setup file created in Step 2.

### Results

The following behavior takes place if errors are encountered while reading the CalibreView setup file, depending on whether the parameter is required or optional (see the parameter definitions in “[CalibreView Setup File Format](#)” on page 536):

- **Required keyword** — Exit with an error message.
- **Optional keyword** — Use the default value, issue a warning, and continue.

The Calibre View is written to the file specified in the CalibreView setup file if no errors are encountered.

The termination message from Calibre View is printed to the CDS.LOG file as follows:

```
Calibre View generation completed with <num> WARNINGS and <num> ERRORS.  
Please consult the CIW transcript for messages.
```

### Related Topics

[Importing Schematic Properties for Calibre View](#)

[Showing Parasitic Polygons in Calibre View](#)

### Specifying Extracted Netlist Annotation of Net Geometries to Calibre View

## Calibre View and IC Manage Revision Control

Calibre View creation is designed to work with the IC Manage integration to Cadence Virtuoso. New or updated Calibre Views are created in the add mode and not checked in automatically; checkin must be done by the user.

- **New Calibre View** — The new view is created in the “Open for Add” state. The view must be checked in to save the view; the checkin is not done automatically.
- **Existing Calibre View** — If the Calibre View already exists in IC Manage the view is checked out and a new version is created in “Open for Edit” mode. The view must be checked in to save the view; the checkin is not done automatically.

See “[Creating a Calibre View](#)” on page 493 for further information.

## Calibre View Generation Warnings

This section describes warnings from Calibre View generation.

The number of warnings printed to the transcript can be controlled with the environment variable FDI\_DBDIFF\_SUPPRESS\_VALUES. See “[FDI Environment Variables](#)” in the *Calibre Layout Comparison and Translation Guide*. For example, to limit the number of warnings for FDI3034 to 20, set FDI\_DBDIFF\_SUPPRESS\_VALUES to “FDI3034 20”.

**Table A-13. Calibre View Warnings**

Warning #	Warning Text (Explanation)
FDI3009	WARNING: [FDI3009] Unsupported parasitic type <type> received. (Occurs when an unsupported parasitic type is specified in netlist file.)
FDI3013	WARNING: [FDI3013] Unable to connect instance <inst> to net <net>. (Occurs when it is not possible to connect an instance to the net.)
FDI3014	WARNING: [FDI3014] Could not find cell mapping for device <dev>. Ignoring instance <inst>. (Occurs when there is no cell mapping for a device in the cellMap file.)
FDI3017	WARNING: [FDI3017] Could not find schematic view of cell <cell> to copy properties from. (Occurs when it is not possible to open the schematic view of a cell.)
FDI3018	WARNING: [FDI3018] Library <lib> not found for device <dev>. Ignoring instance <inst>. (Occurs when the library is not found for a device.)

**Table A-13. Calibre View Warnings (cont.)**

<b>Warning #</b>	<b>Warning Text (Explanation)</b>
FDI3020	WARNING: [FDI3020] Could not find symbol or schematic for cell <cell>. All terminals in view <view> will be of type inputOutput. (Occurs when it is not possible to open symbol or schematic views for a cell.)
FDI3021	WARNING: [FDI3021] Creating an inputOutput terminal <term> in (<cell> <view>) since matching terminal was not found on <symView_or_schView>. (Occurs when -createUnmatchedTerminals option is specified.)
FDI3022	WARNING: [FDI3022] Not creating terminal <term> in (<cell> <view>) since matching terminal was not found on <symView_or_schView>. (Occurs when the corresponding terminal does not exist in the symbol view and the -createUnmatchedTerminals option is not set.)
FDI3023	WARNING: [FDI3023] Additional entry for parasitic in cellmap. ignoring parameter <param>. (Occurs when an additional entry is defined for the parasitic devices in cellmap file.)
FDI3024	WARNING: [FDI3024] Can't find layermap entry for layer <layer> in cellmap file. (This warning occurs when a layer from netlist file does not exist in the cellmap file.)
FDI3025	WARNING: [FDI3025] Invalid layer-purpose pair <layer> <purpose>. (Occurs when an invalid layer-purpose pair specified in the cellmap file.)
FDI3026	WARNING: [FDI3026] lpe construction <expr> is not supported. (Occurs when the syntax of lpe construction is not correct, if its value is not numeric.)
FDI3027	WARNING: [FDI3027] lpe value for device <dev> is not specified in netlist file. (Occurs when the lpe value is not specified for the device in the Calibre View netlist file.)
FDI3028	WARNING: [FDI3028] Failed to open cellview <lib>/<cell>/<view> because cellView database does not exist. (Occurs when the cellview database does not exist.)
FDI3029	WARNING: [FDI3029] Failed to open schematic cellview <lib>/<cell>/<view> from SKILL. (Occurs when it is not possible to open a schematic cellview from SKILL.)

**Table A-13. Calibre View Warnings (cont.)**

Warning #	Warning Text (Explanation)
FDI3030	WARNING: [FDI3030] Could not evaluate property <prop> for the instance: <inst>. (Occurs when a property value is not specified for the instance.)
FDI3031	WARNING: [FDI3031] Cellview <cell> (<view>) does not exist in library <lib>. (Occurs when the cellview does not exist in the library.)
FDI3032	WARNING: [FDI3032] Could not create instances of parasitic device “<dev>”: Cell mapping information not specified. (Occurs when the cell mapping information is not specified for an instance of a parasitic device.)
FDI3033	WARNING: [FDI3033] Schematic instance <inst> not found. (Occurs when a schematic instance from the netlist file is not found.)
FDI3034	WARNING: [FDI3034] Schematic instance <inst> not found, use found instance <inst> instead. (Occurs when the schematic instance is not found.)
FDI3035	WARNING: [FDI3035] Terminal <term> does not exist on cells: (<cells>). Nets will be added as <prop> property values. (Occurs if the terminal does not exist in the cell but does exist in the cdf properties list.)
FDI3036	WARNING: [FDI3036] Terminal <term> does not exist on cells: (<cells>). (Occurs when the cell does not contain the required terminal for connection.)
FDI3037	WARNING: [FDI3037] Copying properties from schematic instance of <schLib:schCell:schView> to instance of <lib:cell:view>. (Occurs when copying properties from a schematic view to a view if the master cells are different.)
FDI3038	WARNING: [FDI3038] Wrong cellmap syntax detected. Block is partly ignored. (Occurs when unknown cellmap syntax is detected. In this case the entire cell's block is ignored.)
FDI3039	WARNING: [FDI3039] Failed to evaluate expression <expr>. Leaving value <val>. (Occurs when an expression evaluation fails and the value from the netlist file is mapped as is.)

**Table A-13. Calibre View Warnings (cont.)**

<b>Warning #</b>	<b>Warning Text (Explanation)</b>
FDI3044	<p>WARNING: [FDI3044] Failed to create mapping for device &lt;<i>deviceName</i>&gt;. Placement is skipped.</p> <p>(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.)</p>
FDI3045	<p>WARNING: [FDI3045] Netlist for &lt;<i>deviceName</i>&gt; has fewer pins than schematic view. The following pins are missing and will be connected to noConn device: &lt;<i>pin_list</i>&gt;.</p> <p>(Occurs during auto-mapping fails if 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.)</p>
FDI3046	<p>WARNING: [FDI3046] Failed to create mapping for device &lt;<i>deviceName</i>&gt;. Netlist for &lt;<i>deviceName</i>&gt; has more pins than schematic view.</p> <p>(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.)</p>
FDI3047	<p>WARNING: [FDI3047] Failed to create mapping for device &lt;<i>deviceName</i>&gt;. More than one netlist pins mapped to one pin from schematic view.</p> <p>(Occurs when auto-mapping fails because more than one Calibre View netlist pin maps to the same pin in the Cadence schematic view.)</p>
FDI3048	<p>WARNING: [FDI3048] Incorrect mapping of bus pins detected in cellmap file. Block is ignored.</p> <p>(Occurs if the order of bus pins in the cellmap file are different or the number of bus pins are not the same.)</p>
FDI3050	<p>WARNING: [FDI3050] NET GEOMETRY: &lt;<i>file_name_line_number</i>&gt;: Invalid statement &lt;<i>statement_name</i>&gt;.</p> <p>(Occurs when the <i>query_results</i> file has an invalid specification statement.)</p>
FDI3051	<p>WARNING: [FDI3051] NET GEOMETRY: The attached technology library was not found.</p> <p>(Occurs when the design has no attached technology file.)</p>
FDI3052	<p>WARNING: [FDI3052] NET GEOMETRY: Could not find layer &lt;<i>layer_name</i>&gt;. Skipping objects on Calibre layer &lt;<i>calibre_layer_name</i>&gt; in the generated layout in Calibre View.</p> <p>(Occurs when the mapped layer is missing from the technology library.)</p>

**Table A-13. Calibre View Warnings (cont.)**

Warning #	Warning Text (Explanation)
FDI3053	WARNING: [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	WARNING: [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.)
FDI3056	WARNING: [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	WARNING: [FDI3059] NET GEOMETRY: Could not find net < <i>net_name</i> >. (Occurs when the net name in the mgc_rve_net statement does not exist in the generated view.)
FDI3060	WARNING: [FDI3060] NET GEOMETRY: Could not find “query_results” file. (Occurs when running MaskLayout with Net Geometry flow if the <i>query_results</i> file has not been created within one minute of invoking the flow.)
FDI3061	WARNING: [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	WARNING: [FDI3062] The device < <i>device_name</i> > with <i>pins_count</i> 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	WARNING: [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	WARNING: [FDI3064] Failed to open cellview <i>lib/cell/view</i> from SKILL. (Occurs when the generating view fails to open from SKILL.)
FDI3065	WARNING: [FDI3065] Failed to close cellview from SKILL. (Occurs when the generating view fails to close from SKILL.)

**Table A-13. Calibre View Warnings (cont.)**

Warning #	Warning Text (Explanation)
FDI3066	WARNING: [FDI3066] Failed to create pCell instance on SKILL side. (Occurs when creating pCell instances using SKILL procedures fails.)
FDI3067	WARNING: [FDI3067] Failed to get pCell schematic instance <i>&lt;instance_name&gt;</i> from SKILL. (Occurs when retrieving the pCell schematic instance using Skill procedures fails.)
FDI3068	WARNING: [FDI3068] <i>&lt;file_name_line_number&gt;</i> Invalid Point2Point statement: <i>&lt;statement_text&gt;</i> (Occurs when a RESISTANCE statement is not valid.)

## Related Topics

[Auto-Mapping Behavior, Control, and Debug in Calibre View Generation](#)

## Setting the Bus Delimiter for Schematic Netlists

The delimiter for bus names may be `<>` or `[]`, depending on the convention used in the schematic netlist. You can convert the bus delimiter, if needed.

**Convert Bus Name Delimiter During Schematic Netlist Export..... 550**

**Automatically Convert Bus Name Delimiter for Calibre Interactive and Calibre RVE 550**

### Convert Bus Name Delimiter During Schematic Netlist Export

You can instruct the tool to convert the bus name delimiter from `<>` to `[]` when the schematic netlist is exported for a Calibre Interactive run.

#### Procedure

1. Open the schematic in Cadence Composer.
2. Choose **Calibre > Setup > Netlist Export**.
3. Enable “Map Bus Name From `<>` to `[]`.” You may need to scroll down in the dialog box to see the selection.

### Automatically Convert Bus Name Delimiter for Calibre Interactive and Calibre RVE

In some cases Calibre Interactive or Calibre RVE is using `[]` as the bus name delimiter but the tool is interfacing with a Cadence schematic view that uses `<>` as the bus name delimiter. You can convert the bus name delimiter so that net selection and highlighting is handled correctly.

#### Procedure

1. Do one of the following to instruct the tool to automatically convert between the two conventions for the bus name delimiter:
  - Define the environment variable `MGC_CALIBRE_MAP_BUS_NAME` in the shell environment.
  - Set `MGC_CALIBRE_MAP_BUS_NAME` to a non-nil value in the SKILL environment.
2. Start Calibre Interactive and Calibre RVE.

#### Results

If `MGC_CALIBRE_MAP_BUS_NAME` is set, Calibre Interactive and Calibre RVE have the following behavior:

- Calibre Interactive Behavior

If you select a net in an attached schematic to populate a GUI field, the bus name delimiter is converted from <> to [].

For instance, in Calibre Interactive PEX you can enter nets to be excluded by selecting nets in the attached Cadence schematic view. If the net is named IN<2> in the schematic view, it will appear as IN[2] in the Calibre Interactive field if MGC\_CALIBRE\_MAP\_BUS\_NAME is set.

- Calibre RVE Behavior

If you are using Calibre RVE to examine an SVDB in which [] is used for the bus name delimiter, it will automatically convert to <> for the bus name delimiter when highlighting nets in an attached Cadence schematic view if MGC\_CALIBRE\_MAP\_BUS\_NAME is set.

# Mapping Schematic Nets to Calibre View Nets for Simulation and Plotting

A typical step after parasitic extraction is the simulation of an extracted Calibre View in the Cadence Analog Design Environment. The method you use depends on whether you have multiple extracted Calibre Views or just one.

Mapping Schematic Nets to Calibre View Nets with a Single Extracted View for Simulation and Plotting .....	552
Mapping Schematic Nets to Calibre View Nets with Multiple Extracted Views for Simulation and Plotting.....	553
Annotating Point to Point Resistance in Calibre View.....	554

## Mapping Schematic Nets to Calibre View Nets with a Single Extracted View for Simulation and Plotting

A typical step after parasitic extraction is the simulation of an extracted Calibre View in the Cadence Analog Design Environment. You can select nets to plot using the selection **Calibre > Map Schematic Net** from the Cadence Virtuoso Schematic Editor.

### Prerequisites

- A Calibre View; see “[Creating a Post-Parasitic Extraction Calibre View](#)” on page 526.
- Simulation in the Cadence environment is set up to use the extracted Calibre View. Consult the Cadence documentation for details.
- A single extracted CalibreView is used.

### Procedure

1. Open your schematic in Cadence Composer.
2. Choose **Calibre > Map Schematic Net** to open the Map Schematic Nets to CalibreView Nets dialog box.
3. If necessary, use the **Clear Selection** button to clear the net selection.
4. In the CalibreView Selection area, select the library, cell, and view name for the extracted Calibre View.
5. Click the **Select Schematic Net** button, then click in a net in the Composer schematic.

The following changes occur in the Map Schematic Nets to CalibreView Nets dialog box:

- The schematic net name is entered into the “Schematic Net” field at the top of the dialog box.

- The CalibreView net name is entered in the “CalibreView Net” field.
- Values are displayed in the “Net R,C Totals” field.

If the CalibreView net name is not displayed and no values appear in “Net R,C Totals,” make sure the entries for library, cell, and view are correct.

6. (Optional) If you selected a bus, the “CalibreView Net” field includes a dropdown list in which you can select a specific net in the bus.
7. Click the **Select** button next to “Plot Selected Nets” and select an item from the dropdown list. This sends the net to the Outputs pane in the Cadence Analog Design Environment. Actions are logged to the CIW.
8. If desired, click the **Select** button next to “Calculator Expression” and select an expression. This opens the calculator with the selected net. For some expressions, you may need to select a pin in the Net Pins area. Actions are logged to the CIW.
9. Click **Apply** and continue with simulation.

## Related Topics

[Mapping Schematic Nets to Calibre View Nets with Multiple Extracted Views for Simulation and Plotting](#)

# Mapping Schematic Nets to Calibre View Nets with Multiple Extracted Views for Simulation and Plotting

A typical step after parasitic extraction is the simulation of an extracted Calibre View in the Cadence Analog Design Environment (ADE). After you select nets to plot in the Cadence Analog Design Environment, you can convert between CalibreView names and schematic names with the selections **Calibre > Map Schematic Nets to CalibreView Nets** and **Calibre > Map CalibreView Nets to Schematic Nets**. The text file *extviews.txt* provides the mapping between the extracted views and the cells.

## Prerequisites

- A Calibre View; see “[Creating a Post-Parasitic Extraction Calibre View](#)” on page 526.
- Simulation in the Cadence environment is set up to use the extracted Calibre View. Consult the Cadence documentation for details.
- The text file *extviews.txt* must exist in the directory containing the CalibreView netlist. Each line in *extviews.txt* should have one of the following formats:

```
extracted_viewName cellName
library extracted_viewName cellName
```

Use the second format when the extracted Calibre View is not in the same library as the schematic, and specify the library name for the Calibre View.

For multiple extracted views within the same design, mapping is done by specifying all view names for every cell in the *extviews.txt* file. Calibre View searches within these views and performs the mapping. When mapping is done, hierarchical nets such as A1/IO/net2 are renamed to A1\_IO\_net2 in the CalibreView. If no extracted view is found, the net name does not change.

For example:

```
calibreRCC      buffer2
ref_calibreRCC buffer2
libCV          calview2 buf3
```

## Procedure

1. Open your schematic in Cadence Composer.
  2. Launch Cadence Virtuoso ADE.
  3. Populate the Outputs table in Cadence Virtuoso ADE with the nets to plot.
  4. Switch between CalibreView and schematic net naming with the following steps:
    - a. Choose **Calibre > Map Schematic Nets to CalibreView Nets** or **Calibre > Map CalibreView Nets to Schematic Nets**.
    - b. Click the **Netlist and Run** button.
- The net names in the Outputs table are converted as requested. Net names are not mapped until you click **Netlist and Run**.
5. Continue with the simulation.

## Results

Net names are converted between schematic and CalibreView naming conventions as requested, using the mapping information in the *extviews.txt* file.

For instance, suppose you are looking at cellA and descend into instance I1 which has net midA. When you select the net to plot it is entered in the Outputs table with schematic naming as I1/midA. After converting to CalibreView naming the net appears as I1\_midA in the Outputs table.

## Related Topics

[Mapping Schematic Nets to Calibre View Nets with a Single Extracted View for Simulation and Plotting](#)

## Annotating Point to Point Resistance in Calibre View

When generating a Point to Point report to be viewed in the Map Schematic Nets to CalibreView Nets dialog box, you can include details on the resistances for specified nets. When enabled, information on each individual resistor is displayed.

This feature can only be used if the PEX netlist format is CalibreView. To use this feature, perform one of the following steps:

- In the Calibre Interactive GUI, navigate to **Outputs > PEX Report Point2Point** and enable the CALIBREVIEW checkbox.
- When using the **PEX Report Point2Point** statement, include the CALIBREVIEW keyword with SOURCENAMES.

The resistance information is annotated in the Net R,C Totals field in the Map Schematic Nets to CalibreView Nets dialog box.

When the resistance value is missing for a selected net, the following warning is printed:

WARNING: No Point2Point value is found for "net" net.

## Backannotation of Parasitic Values to the Schematic in Cadence Virtuoso

When debugging parasitics in the design it is sometimes useful to backannotate the schematic with the extracted parasitic values. This can be done using the dialog box “Map Schematic Nets to CalibreView Nets”. Backannotation can also be deleted.

### Prerequisites

- A Calibre View; see “[Creating a Post-Parasitic Extraction Calibre View](#)” on page 526.

### Procedure

1. Open your schematic in Cadence Composer.
2. (Optional) If you are mapping hierarchical nets, specify the extracted view name to search in when doing the mapping. This step automates the handling of hierarchical nets and is useful if many nets are being mapped.

Create a text file *extviews.txt* in the directory containing the CalibreView netlist. Each line in *extviews.txt* should have one of the following formats:

```
extracted_viewName cellName
library extracted_viewName cellName
```

Use the second format when the extracted Calibre View is not in the same library as the schematic, and specify the library name for the Calibre View.

For multiple extracted views within the same design, mapping is done by specifying all view names for every cell in the *extviews.txt* file. Calibre View searches within these views and performs the mapping. When mapping is done, hierarchical nets such as A1/IO/net2 are renamed to A1\_IO\_net2 in the CalibreView. If no extracted view is found, the net name does not change.

3. Choose **Calibre > Map Schematic Nets to CalibreView Nets** to open the Map Schematic Nets to CalibreView Nets dialog box.
4. In the Calibreview Selection area, select the library, cell, and view name for the extracted Calibre View.
5. Enable the checkbox “Show/Hide Annotation for Net selected in Schematic.”
6. In the Composer schematic, click each net that you want to backannotate. If necessary, use the **Clear Selection** button to clear the net selection.

The total resistance, lumped capacitance, and total cross-coupled capacitance values are annotated into the schematic.

If you click in a net with existing annotation, the annotation is deleted.
7. Uncheck the “Show/Hide Annotation for Net selected in Schematic” checkbox when done annotating.

## Automating Calibre Interactive Runsets in the Cadence Design Environment

Environment variables can be set by custom scripts, or with Calibre Interactive pre-execution triggers for use within the Cadence Design Environment. The following script shows how Calibre Interactive execution triggers can be used within the Cadence Design Environment to automatically specify a Calibre Interactive runset based on the technology library attached to the design.

For example:

```

procedure(mgc_start_calibre_trigger(drc_or_lvs win)
prog( (
    topCell
    designlibId
    techlibId
    techlibName
)

printf( "Starting %s on top cell in window %L...\n"
    upperCase(drc_or_lvs)
    win )
topCell=win->topCellView
designlibId=topCell->lib
techlibId=techGetTechFile(designlibId)
techlibName=techlibId->libName

case( upperCase(drc_or_lvs)
    ("DRC"
        if( techlibName=="tech1"
            then
                setShellEnvVar(
                    "MGC_CALIBRE_DRC_RUNSET_FILE=tech1.drc.runset")
                println( "Calibre Interactive runset tech1.drc.runset")
            else
                csh("unsetenv MGC_CALIBRE_DRC_RUNSET_FILE")
                println( "No runset file set for this technology...")
        )
    )
    ("LVS"
        if( techlibName=="tech1"
            then
                setShellEnvVar(
                    "MGC_CALIBRE_LVS_RUNSET_FILE=tech1.lvs.runset")
                println( "Calibre Interactive runset tech1.lvs.runset")
            else
                csh("unsetenv MGC_CALIBRE_LVS_RUNSET_FILE")
                println( "No runset file set for this technology...")
        )
    )
    ("PEX"
        if( techlibName=="tech1"
            then
                setShellEnvVar(
                    "MGC_CALIBRE_PEX_RUNSET_FILE=tech1.pex.runset")
                println( "Calibre Interactive runset tech1.pex.runset")
            else
                csh("unsetenv MGC_CALIBRE_PEX_RUNSET_FILE")
                println( "No runset file set for this technology...")
        )
    )
)
return(t)
))

```

## Cadence Composer

The Cadence Composer integration is similar to that for Cadence Virtuoso.

See the section “[Cadence Virtuoso](#)” on page 469 for general information about the Calibre Interface to Cadence tools.

In Cadence Composer, after selecting a net or multiple nets in the schematic viewer for entry into a Calibre Interactive or Calibre RVE GUI field, you must enter Ctrl-C on the keyboard to end the net selection process and enter the selected nets.

See “[Map Cadence Netlist Names to Schematic Names for Highlighting from Calibre RVE](#)” in the *Calibre RVE User’s Manual* for specific information about highlighting schematic elements.

---

### Note

---

 For Calibre 2012.3 and earlier releases, use the Esc key to end the net selection process. For Calibre 2012.4 and later releases, use Ctrl-C to end the net selection process.

---

## Cadence Encounter

Calibre includes software that enables seamless integration of Calibre Interactive and Calibre RVE within the Cadence Encounter platform. A **Calibre** menu is activated when the supplied software is loaded into the Encounter environment, usually through the cal\_enc.tcl file.

Calibre Interactive and Calibre RVE supports Cadence Encounter up to version 14.1.

You can add custom menu items to the Calibre integration menu; see “[Custom Menu Items in Design Tools](#)” on page 453.

<b>Creating an Interface to Cadence Encounter .....</b>	<b>559</b>
<b>Importing saveNetlist Parameters for Verilog Export in Cadence Encounter .....</b>	<b>560</b>

## Creating an Interface to Cadence Encounter

The following script can be inserted into the enc.tcl file to load the Calibre software:

```
proc load_calibre {} {
    global env
    if {[!info exists env(CALIBRE_HOME)] || $env(CALIBRE_HOME) == ""} {
        if {[!info exists env(MGC_HOME)] || $env(MGC_HOME) == ""} {
            puts "Environment variable CALIBRE_HOME not set."
            puts "Calibre interface NOT loaded."
            return
        } else {
            puts "Environment variable CALIBRE_HOME not set, using MGC_HOME."
            set env(CALIBRE_HOME) $env(MGC_HOME)
        }
    }

    set etclf [file join $env(CALIBRE_HOME) lib cal_enc.tcl]
    if {[!file readable $etclf]} {
        puts "Could not find Calibre initialization files."
        puts "Calibre interface not loaded."
        return
    }
    if {[catch {source $etclf} msg]} {
        puts "ERROR while loading Calibre interface: $msg"
    }
}
load_calibre
```

This script first checks to see if the CALIBRE\_HOME environment variable is set. If so, it loads the cal\_enc.tcl file located in the lib subdirectory of the CALIBRE\_HOME directory. The rest of the files accessed by cal\_enc.tcl are located in the \$CALIBRE\_HOME/shared/pkgs/icv/tools/queryenc subdirectory. Collectively, these files are referred to as the Calibre Encounter Interface. The Calibre Encounter Interface adds a **Calibre** menu to the Encounter application’s banner menu.

You can also use MGC\_HOME instead of CALIBRE\_HOME to define the location of your software tree; see “[Setting the CALIBRE\\_HOME Environment Variable](#)” in the *Calibre Administrator’s Guide*.

The following menu items are available in the **Calibre** menu:

**Table A-14. Calibre Menu in Cadence Encounter**

Menu Item	Description
Run nmDRC	Starts Calibre Interactive DRC on the current cell
Run DFM	Starts Calibre Interactive DFM on the current cell
Run nmLVS	Starts Calibre Interactive LVS on the current cell
Run PERC	Starts Calibre Interactive PERC on the current cell
Run PEX	Starts Calibre Interactive PEX on the current cell
Start RVE	Starts Calibre RVE
Clear Highlights	Clears Calibre RVE highlights in the current cell
Setup	Open a dialog box to set parameters for the following: <ul style="list-style-type: none"><li>• GDS Export</li><li>• DEF Export</li><li>• Verilog Export (also see “<a href="#">Importing saveNetlist Parameters for Verilog Export in Cadence Encounter</a>” on page 560)</li><li>• RVE</li><li>• Socket</li></ul>
About	Displays information about the Calibre Encounter Interface

For general information on the socket communication between a layout viewer and Calibre Interactive and Calibre RVE, see “[Setting the Socket Port in Calibre Interactive](#)” on page 451.

## Importing saveNetlist Parameters for Verilog Export in Cadence Encounter

You can populate the **Calibre > Setup > Verilog Export** dialog box with the parameters used by the saveNetlist command in Cadence Encounter.

### Procedure

1. Create a text file that contains the desired saveNetlist parameters. For example:

```
% cat exportParamsFile
-excludeLeafCell -flat -includeBumpCell
```

2. Define the environment variable MGC\_CALIBRE\_VERILOG\_EXPORT\_PARAM\_FILE to point to the location of the file created in Step 1. For example:

```
setenv MGC_CALIBRE_VERILOG_EXPORT_PARAM_FILE exportParamsFile
```

3. Start Cadence Encounter.

The saveNetlist parameters specified in the file pointed to by MGC\_CALIBRE\_VERILOG\_EXPORT\_PARAM\_FILE are used to populate the **Calibre > Setup > Verilog Export** dialog box and are used when the saveNetlist command is invoked.

## Cadence Innovus

Calibre Interactive and Calibre RVE are integrated with the Cadence® Innovus™ Implementation System. The Calibre menu is added when the integration code is loaded.

The integration is supported for Cadence Innovus versions 16.22, 16.23, and 16.24.

**Creating an Interface to Cadence Innovus .....** ..... **562**

## Creating an Interface to Cadence Innovus

Integration code must be loaded to enable the Calibre integration to Cadence Innovus. The integration is supported for Calibre RVE and all Calibre Interactive applications except Calibre Interactive 3DSTACK.

The integration is supported for Cadence Innovus versions 16.22, 16.23, 16.24, and 17.11.

The Tcl integration script is found at the following location:

```
$CALIBRE_HOME/lib/cal_enc.tcl
```

### Procedure

Load the integration script with one of the following methods:

- At the Cadence Innovus command prompt

Invoke Cadence Innovus as usual, then source the integration script at the command prompt. For example:

```
innovus 1> source $::env(CALIBRE_HOME)/lib/cal_enc.tcl
```

- At invocation with the -files argument

You can load the integration script using the -files command line option when invoking Cadence Innovus. For example, at the shell prompt:

```
innovus -files "$CALIBRE_HOME/lib/cal_enc.tcl enc_setup.tcl" -win  
-log innovus.log
```

- With automatic loading

Consult the Cadence documentation regarding automatically loaded initialization files such as *enc.tcl* or *innovus.tcl*. You can add the Tcl integration code to such a file, or create a symbolic link to *\$CALIBRE\_HOME/lib/cal\_enc.tcl*.

## Results

Cadence Innovus is invoked and the Calibre menu is added with the following items:

**Table A-15. Calibre Menu in Cadence Innovus**

Menu Item	Description
Run nmDRC	Starts Calibre Interactive nmDRC on the current cell
Run DFM	Starts Calibre Interactive DFM on the current cell
Run nmLVS	Starts Calibre Interactive nmLVS on the current cell
Run PERC	Starts Calibre Interactive PERC on the current cell
Run PEX	Starts Calibre Interactive PEX on the current cell
Run xACT	Starts Calibre Interactive xACT on the current cell
Start RVE	Starts Calibre RVE
Clear Highlights	Clears Calibre RVE highlights in the current cell
Setup	Opens a dialog box to set parameters for the following: <ul style="list-style-type: none"> <li>• GDS Export</li> <li>• DEF Export</li> <li>• Verilog Export (also see “<a href="#">Importing saveNetlist Parameters for Verilog Export in Cadence Encounter</a>” on page 560)</li> <li>• RVE</li> <li>• Socket</li> </ul>
RealTime DRC	<ul style="list-style-type: none"> <li>• <b>Run DRC</b> — Starts a DRC run using Calibre RealTime Digital.</li> <li>• <b>Options</b> — Opens the Calibre RealTime Options dialog box.</li> </ul> <p>See “<a href="#">Calibre RealTime Digital in Cadence Innovus</a>” in the <i>Calibre RealTime Digital User’s Manual</i>.</p>
About	Displays the document “About the Calibre Encounter/Innovus Interface.” The interface to Calibre is the same for Cadence Encounter and Cadence Innovus.

For general information on the socket communication between a layout viewer and Calibre Interactive and Calibre RVE, see “[Setting the Socket Port in Calibre Interactive](#)” on page 451.

# Synopsys IC Compiler and Synopsys IC Compiler II

---

Calibre provides an integration of Calibre Interactive and Calibre RVE to Synopsys IC Compiler and Synopsys IC Compiler II. The integration makes it easy to run Calibre flows in the Synopsys IC Compiler environment and debug the results.

Code shipped in the Calibre software tree enables the integration, which adds a **Calibre** menu to the ICC and ICC2 GUI (see [Figure A-3](#)).

**Supported version** — The Calibre—IC Compiler interface supports Synopsys IC Compiler versions through 2014.9 and Synopsys IC Compiler II version M-2016.12-SP2.

You can do the following using the Calibre interface to Synopsys IC Compiler:

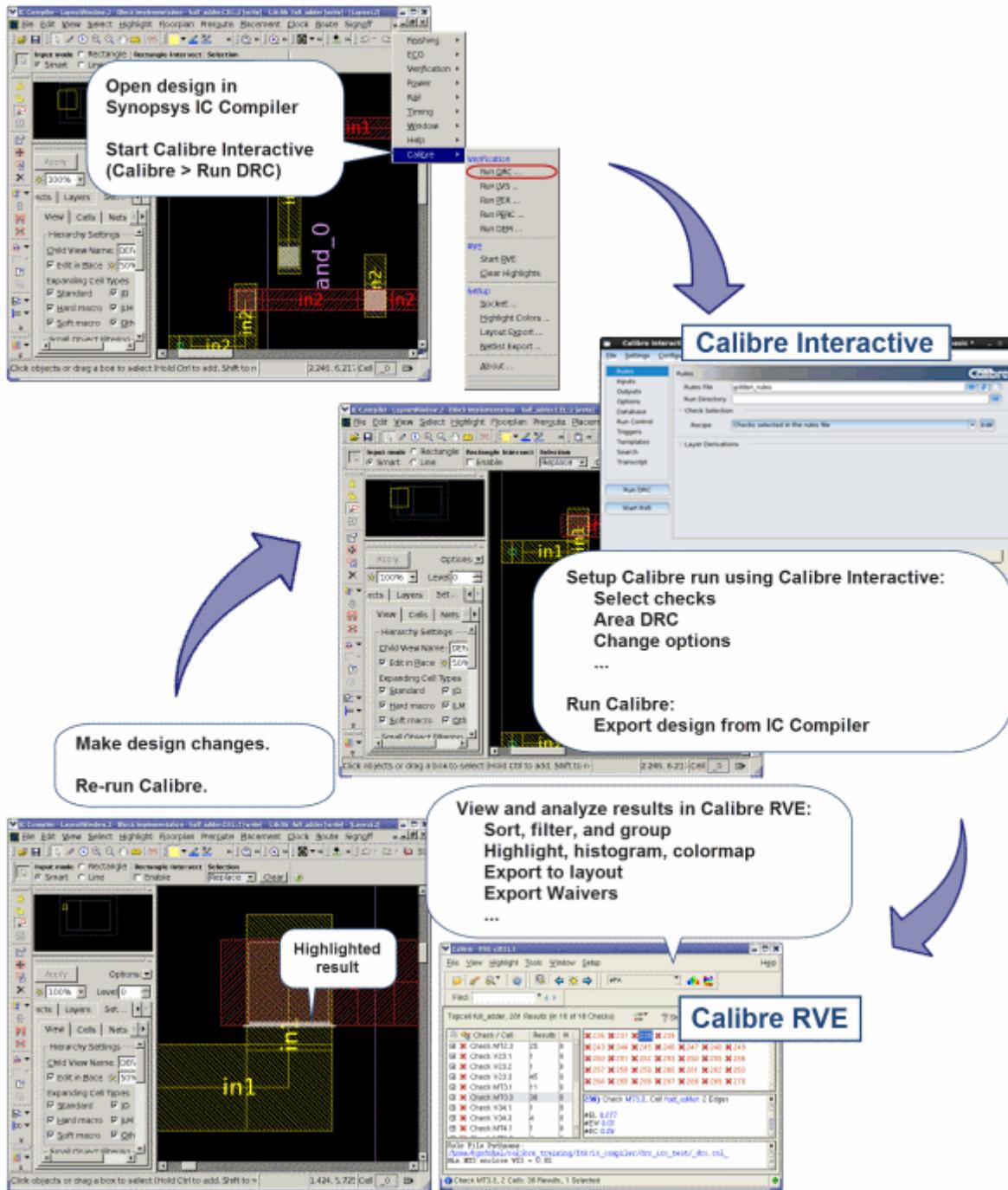
- Run Calibre Interactive nmDRC, nmLVS, PERC, PEX, and DFM:
  - Use any Calibre rule file.
  - Adjust rule file settings using the Calibre Interactive GUI.
  - Select DRC checks using the Calibre Interactive GUI.
  - Run Calibre LFD, FastXOR, Calibre PERC LDL, dummy fill, Calibre Pattern Matching, Calibre DP, and other applications.
- View and debug Calibre results using Calibre RVE (Results Viewing Environment):
  - View DRC, LVS, PERC, PEX, and DFM results.
  - Highlight results in layout and schematic.
  - Group, sort, and filter results.
  - Perform interactive short isolation using Calibre RVE for LVS.
- Add custom menu items to the **Calibre** menu in Synopsys IC Compiler. See “[Custom Menu Items in Design Tools](#)” on page 453.
- Convert results from an ASCII format Calibre results database (RDB) to error markers and route guides in Synopsys IC Compiler. This allows you to use familiar tools when debugging errors and correcting the design. See “[Calibre RDB Conversion: Working with Calibre Results in Place and Route Tools](#)” in the *Calibre RVE User’s Manual*.

<b>Calibre Flow with Synopsys IC Compiler .....</b>	<b>565</b>
<b>Calibre Interface to Synopsys IC Compiler .....</b>	<b>566</b>
<b>Running Calibre Interactive with Synopsys IC Compiler.....</b>	<b>570</b>
<b>Setting Socket Connections with Synopsys IC Compiler and IC Compiler II.....</b>	<b>573</b>
<b>Troubleshooting the Calibre Interface with Synopsys IC Compiler .....</b>	<b>574</b>

## Calibre Flow with Synopsys IC Compiler

The interactive flow with Calibre Interactive, Calibre RVE, and Synopsys IC Compiler allows you to easily run Calibre and debug results. The flow with Calibre nmDRC is illustrated.

**Figure A-3. Interactive Calibre Flow with Synopsys IC Compiler**



Although the figure shows Calibre nmDRC, the Calibre nmLVS, Calibre PERC, DFM, and PEX applications are also supported with Synopsys IC Compiler.

## Calibre Interface to Synopsys IC Compiler

The Calibre interface to Synopsys IC Compiler supports the Calibre nmDRC, Calibre nmLVS, Calibre PERC, PEX, and DFM applications for both Calibre Interactive and Calibre RVE. The **Calibre** menu is available after the interface is loaded. Synopsys IC Compiler II is also supported.

<b>Loading the Calibre Interface to Synopsys IC Compiler.....</b>	<b>566</b>
<b>Loading the Calibre Interface to Synopsys IC Compiler II.....</b>	<b>567</b>
<b>README File for the Calibre-IC Compiler Integration.....</b>	<b>569</b>
<b>Calibre Menu in Synopsys IC Compiler .....</b>	<b>569</b>

## Loading the Calibre Interface to Synopsys IC Compiler

The Calibre interface can be loaded automatically each time you start Synopsys IC Compiler, or you can manually install the interface. The Calibre—IC Compiler interface supports Synopsys IC Compiler versions through 2014.9. Synopsys IC Compiler II version M-2016.12-SP2 is supported.

### Prerequisites

- The environment variable CALIBRE\_HOME or MGC\_HOME points to the Calibre software tree. See “[Setting the CALIBRE\\_HOME Environment Variable](#)” in the *Calibre Administrator’s Guide*.

### Procedure

1. Copy the following code into a file *load\_calibre\_interface.tcl*:

```
if {! [info exists ::env(CALIBRE_HOME)] && [info exists ::env(MGC_HOME)]} {  
    puts "Environment variable CALIBRE_HOME not set, using MGC_HOME"  
    set ::env(CALIBRE_HOME) $::env(MGC_HOME)  
}  
if {! [info exists ::env(CALIBRE_HOME)]} {  
    puts "ERROR, environment variable CALIBRE_HOME or MGC_HOME not set"  
    puts "Calibre interface not loaded"  
} else {  
    source [file join $::env(CALIBRE_HOME) lib icc_calibre.tcl]  
}
```

2. Use one of the following methods to install the interface:

- **Installation with a startup script and -f command-line option**
  - i. Source the installation code from a startup command script which is specified with the -f command-line option. For example, place this line in your startup command script *startup\_cmd.tcl*:

```
source load_calibre_interface.tcl
```

where *load\_calibre\_interface.tcl* is the file created in Step 1. Alternatively, you can copy the installation code directly into the startup script.

- ii. Start Synopsys IC Compiler as follows:

```
% $SYNOPSYS/bin/icc_shell -gui -f startup_cmd.tcl
```

- **Automatic installation with *.synopsys\_icc\_gui.tcl* file**

Source the installation code in your *\$HOME/.synopsys\_icc\_gui.tcl* file. The interface is loaded each time you start Synopsys IC Compiler in gui mode.

- **Manual Installation at the *icc\_shell* command prompt**

Type the following command at the command prompt for Synopsys IC Compiler:

```
icc_shell> source load_calibre_interface.tcl
```

## Results

When the Calibre interface to Synopsys IC Compiler is successfully loaded, the **Calibre** menu is added to the ICC menu bar. The **Calibre** menu is described in the section “[Calibre Menu in Synopsys IC Compiler](#)” on page 569.

In addition, a banner similar to the one below is displayed:

```
//  
//          Calibre IC Compiler Interface * (v2014.3_16.15) *  
//  
//          Copyright Siemens 1996 - 2020  
//          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.  
//
```

## Loading the Calibre Interface to Synopsys IC Compiler II

The Calibre interface can be loaded automatically each time you start Synopsys IC Compiler II, or you can manually install the interface. Synopsys IC Compiler II version M-2016.12-SP2 is supported.

### Prerequisites

- The environment variable **CALIBRE\_HOME** or **MGC\_HOME** points to the Calibre software tree. See “[Setting the \*\*CALIBRE\\_HOME\*\* Environment Variable](#)” in the *Calibre Administrator’s Guide*.

- Do one of the following, depending on whether you want automatic installation:

- **For automatic installation with `~/.synopsys_icc2.setup` file:**

Copy the following code into the file `$HOME/.synopsys_icc2.setup`:

```
if {! [info exists ::env(CALIBRE_HOME)] && [info exists ::env(MGC_HOME)]} {  
    puts "Environment variable CALIBRE_HOME not set, using MGC_HOME"  
    set ::env(CALIBRE_HOME) $::env(MGC_HOME)  
}  
if {! [info exists ::env(CALIBRE_HOME)]} {  
    puts "ERROR, environment variable CALIBRE_HOME or MGC_HOME not set"  
    puts "Calibre interface not loaded"  
} else {  
    set_app_options -name gui.custom_setup_files \  
        -value [file join $::env(CALIBRE_HOME) lib icc_calibre.tcl]  
}
```

- **For non-automatic installation:**

Copy the following code into the file `load_calibre_interface.tcl`:

```
if {! [info exists ::env(CALIBRE_HOME)] && [info exists ::env(MGC_HOME)]} {  
    puts "Environment variable CALIBRE_HOME not set, using MGC_HOME"  
    set ::env(CALIBRE_HOME) $::env(MGC_HOME)  
}  
if {! [info exists ::env(CALIBRE_HOME)]} {  
    puts "ERROR, environment variable CALIBRE_HOME or MGC_HOME not set"  
    puts "Calibre interface not loaded"  
} else {  
    source [file join $::env(CALIBRE_HOME) lib icc_calibre.tcl]  
}
```

## Procedure

Use one of the following methods to install the interface:

- **Automatic installation with `$HOME/.synopsys_icc2.setup` file**

The interface is automatically loaded each time you start Synopsys IC Compiler II in gui mode. For example:

```
% $SYNOPSYS/bin/icc2_shell -gui <options>
```

- **Installation with a startup script and the `-f` command-line option**

- i. Source the installation code from a startup command script which is specified with the `-f` command-line option. For example, place this line in your startup command script `startup_cmd.tcl`:

```
source load_calibre_interface.tcl
```

where `load_calibre_interface.tcl` is the file created in the Prerequisites section. Alternatively, you can copy the installation code directly into the startup script.

- ii. Start Synopsys IC Compiler II as follows:

```
% $SYNOPSYS/bin/icc2_shell -gui -f startup_cmd.tcl
```

- **Manual installation at the icc\_shell command prompt**

- i. Start Synopsys IC Compiler II:

```
% $SYNOPSYS/bin/icc2_shell -gui <options>
```

- ii. Type the following at the command prompt for Synopsys IC Compiler II:

```
icc_shell> source load_calibre_interface.tcl
```

## Results

When the Calibre interface to Synopsys IC Compiler II is successfully loaded, the **Calibre** menu is added to the menu bar. The **Calibre** menu is described in the section “[Calibre Menu in Synopsys IC Compiler](#)” on page 569.

In addition, a banner similar to the one below is displayed:

```
//  
//          Calibre IC Compiler Interface * (v2019.3) *  
//  
//          Copyright Siemens 1996 - 2020  
//          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.  
//
```

## README File for the Calibre-IC Compiler Integration

The README file for the Calibre -IC Compiler integration is at the following location:

```
$CALIBRE_HOME/shared/pkgs/icv/tools/querytcl/ICCompilerREADME
```

The file includes these instructions for installing the **Calibre** menu in Synopsys IC Compiler and further details about the Calibre—IC Compiler Interface.

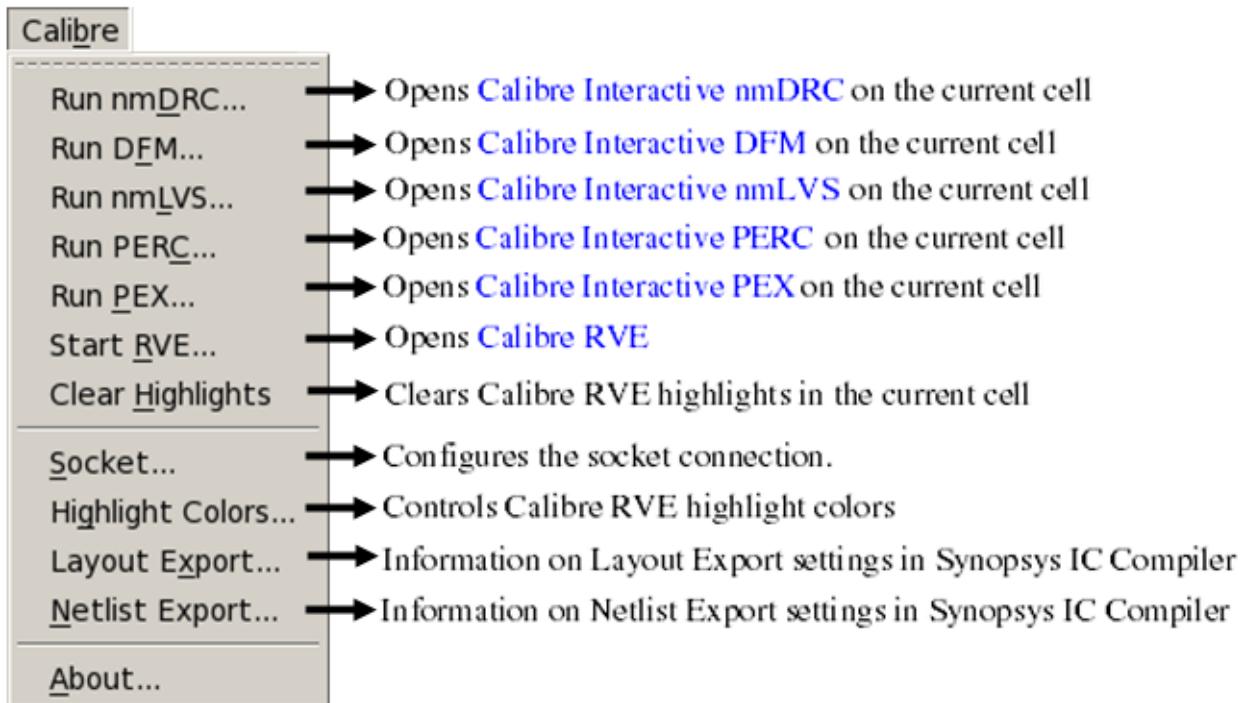
## Calibre Menu in Synopsys IC Compiler

The **Calibre** menu is available after Calibre interface is loaded. The menu has selections to start Calibre Interactive, Calibre RVE, and to selection setup options.

The **Calibre** menu in Synopsys IC Compiler has the selections shown in [Figure A-4](#). See “[Loading the Calibre Interface to Synopsys IC Compiler](#)” on page 566 if you do not see the **Calibre** menu in the ICC GUI.

You can also add custom menu items to the Calibre integration menu; see “[Custom Menu Items in Design Tools](#)” on page 453.

**Figure A-4. Calibre Menu Contents in Synopsys IC Compiler**



## Running Calibre Interactive with Synopsys IC Compiler

All Calibre Interactive applications are supported: Calibre nmDRC, Calibre nmLVS, DFM, Calibre PERC, and PEX.

You can use Calibre Interactive to easily change options for your Calibre run, since Calibre Interactive settings override settings in the rule file. You save all your Calibre Interactive settings in a runset for later use.

### Prerequisites

- “[Loading the Calibre Interface to Synopsys IC Compiler](#)” on page 566 has been completed.

### Procedure

1. Launch Synopsys IC Compiler:

```
icc_shell -gui
```

You should see the Calibre banner displayed in the ICC command window; if not, see “[Loading the Calibre Interface to Synopsys IC Compiler](#)” on page 566.

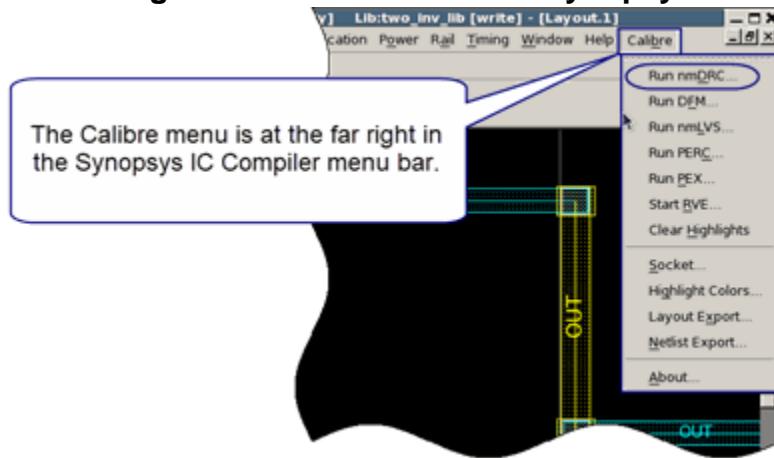
2. Open your library

```
icc_shell> open_mw_lib my_milkyway_lib
```

```
icc_shell> open_mw_cel top_cell
```

3. Choose **Calibre > Run nmDRC**, or the desired Calibre Interactive application.

**Figure A-5. Calibre Menu in Synopsys IC Compiler**



Calibre Interactive opens and displays the Inputs pane with the current cell selected.

**Note**

 Calibre Pattern Matching, Calibre DP, and dummy fill are run using the **Run DRC** selection. Calibre SmartFill is run using the **Run DFM** selection.

4. (Optional) Load a runset with **File > Load Runset**; a runset includes all Calibre Interactive settings. See “[About Calibre Interactive Runsets](#)” on page 34 for information.

If your runset includes all the desired options, you may be ready to start a run; if so, skip to Step 13.

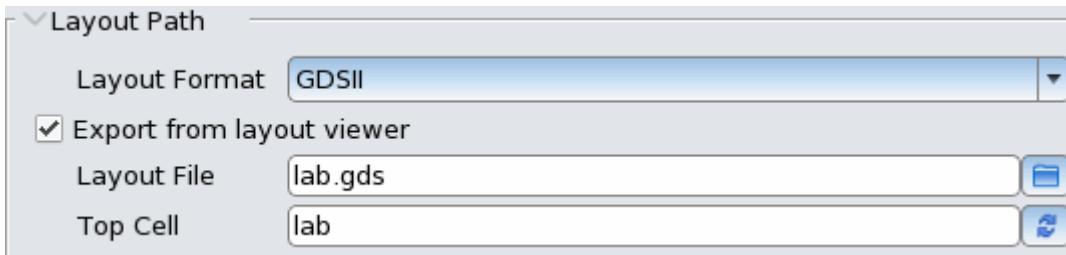
5. Specify the input source as “Layout Export” and the desired format of the exported design.

These steps instruct Synopsys IC Compiler to export the open design to Calibre. Synopsys IC Compiler controls the conversion to the output format.

- In the **Inputs** page, set “Layout Format” to GDSII, OASIS, or LEFDEF.
- Enable “Export from layout viewer.”
- Verify the filename for the exported layout database in the “Layout File” field.

An intermediate file is extracted and saved to the file named in the “Layout File” field during layout export. This filename is generated automatically based on the setting in Inputs section of the Templates page

(**Settings > Show Pages > Templates**); see “[Templates for File Naming](#)” on page 100.



- d. Set streamout options in Synopsys IC Compiler. Consult the Synopsys IC Compiler documentation, or view **Calibre > Layout Export** and **Calibre > Netlist Export** in Synopsys IC Compiler for information.
6. View the **Rules** and **Outputs** pages and fill in the required settings.
7. (Optional) You can select the checks that are run in Calibre Interactive nmDRC, DFM, and Calibre nmLVS (for ERC checks). You can select checks by groups, by layer, and other criteria.  
See “[Specifying the Checks to Execute in a Calibre Interactive nmDRC Run](#)” on page 140 for more information:
8. (Optional) Calibre Interactive supports trigger functions which can execute at several points in your flow. Trigger functions are useful for customizing your flow and doing pre- and post-processing of design input and result output.  
Open the Triggers page (**Settings > Show Pages > Triggers**).  
See the following topics for information:
  - “[Trigger Functions in Calibre Interactive](#)” on page 413 — Complete information.
  - “[External Trigger Definitions in Synopsys IC Compiler](#)” on page 432 — External triggers, which execute before and after Calibre Interactive execution.
  - “[Internal Triggers in Calibre Interactive](#)” on page 417 — Internal triggers, which execute before and after the Calibre run.
9. Click **Run Control** on the left panel and review the settings. Performance on large designs can be improved by enabling the multithreaded (MT) and hyperscaling processing modes. See “[Run Control and Licensing Setup](#)” on page 435 for more information.
10. (Optional) You can run Calibre on a selected region of the design when using Calibre Interactive nmDRC.  
See “[Running with Area DRC](#)” on page 141 for instructions.

11. (Optional) View the control file for the Calibre Interactive run by clicking **Ctrl-Run DRC**. The control file contains the settings that control the Calibre run; control file settings take precedence over rule file settings if there is a conflict. See “[About Control Files](#)” on page 44 for more information.
12. Choose **File > Save Runset** to save your Calibre Interactive settings; see “[About Calibre Interactive Runsets](#)” on page 34 for information.
13. Click **Run DRC** (or your selected application) on the left panel to start the Calibre run.  
Calibre Interactive switches the view to the **Transcript** page, where you can view the progress of the Calibre run and any run-time errors and warnings. Right-click the transcript to save it.
14. If Calibre RVE did not open automatically, choose **Calibre > Start RVE** in Synopsys IC Compiler to view the Calibre results. See “[Calibre RVE with Synopsys IC Compiler](#)” in the *Calibre RVE User’s Manual* for more information.

## Related Topics

[Troubleshooting the Calibre Interface with Synopsys IC Compiler](#)

# Setting Socket Connections with Synopsys IC Compiler and IC Compiler II

Calibre Interactive and Calibre RVE communicate with Synopsys IC Compiler through a socket port. Socket communication is established automatically when you start Calibre Interactive or Calibre RVE from the **Calibre** menu in Synopsys IC Compiler. By default, Calibre Interactive and Calibre RVE use socket port 9189 for communication, and the tool automatically searches for an available socket if it cannot obtain the default socket.

Synopsys IC Compiler II is also supported.

You can also manually configure the socket, as described below, although this is not needed in most situations.

## Prerequisites

- “[Loading the Calibre Interface to Synopsys IC Compiler](#)” on page 566

## Procedure

Do one of the following:

To set the socket using ...	Do this ...
Synopsys IC Compiler	Choose <b>Calibre &gt; Setup &gt; Socket</b> in Synopsys IC Compiler.

To set the socket using ...	Do this ...
Environment variable	Set the environment variable MGC_CALIBRE_LAYOUT_SERVER, as described in “ <a href="#">Setting the Calibre Interactive Socket Port for a Layout Viewer</a> ” on page 451.

## Results

The socket is initialized when needed.

---

### Tip

 You can also specify that the socket is started when Synopsys IC Compiler starts. Set the environment variable MGC\_RVE\_INIT\_SOCKET\_AT\_STARTUP to any value; when the Calibre integration code is read into Synopsys IC Compiler, the server socket will be started.

---

# Troubleshooting the Calibre Interface with Synopsys IC Compiler

Several common problem may occur in the Calibre Interface with Synopsys IC Compiler.

The following issues are discussed:

- Problem: Some rule file settings are not used in the Calibre Interactive run
- Tip: Debugging Calibre Interactive operation

## Problem: Some rule file settings are not used in the Calibre Interactive run

Cause: Calibre Interactive settings override rule file settings.

Solution: Do one of the following, depending on your situation.

- Enable the desired setting in the GUI. You can save the GUI settings to a runset with **File > Save Runset**; then load the runset later with **File > Load Runset**. Also see “[Methods to Load a Calibre Interactive Runset](#)” on page 35.
- If you want to load all rule file settings into the GUI, click **Load** after specifying the rule file on the Rules pane. Any GUI changes that you make after loading the rule file override the corresponding setting in the rule file. Also see “[Automatically Loading a Rule File](#)” on page 70.

**Tip**

 You can view the control file for the Calibre Interactive run by holding the Ctrl key while clicking **Run app** where *app* is the Calibre Interactive application you are running. The control file contains the settings that control the Calibre run; control file settings take precedence over rule file settings if there is a conflict. See “[About Control Files](#)” on page 44 and “[How Rule Files Relate to Control Files](#)” on page 51 for more information.

---

### Tip: Debugging Calibre Interactive operation

- Save the Calibre Interactive transcript by choosing **Transcript > Echo to File**. This makes it easier to compare the transcript output to other runs, or to the transcript from a command line run.
- View the control file for the Calibre Interactive run by holding the Ctrl key while clicking **Run app**. The control file contains the settings that control the Calibre run; control file settings take precedence over rule file settings if there is a conflict. See “[About Control Files](#)” on page 44 and “[How Rule Files Relate to Control Files](#)” on page 51 for more information.
- Perform a command line Calibre run and compare transcript and results output to that from the Calibre Interactive run.

## Silvaco Expert

---

The Silvaco Expert design tool includes integrations to Calibre Interactive nmDRC, Calibre Interactive nmLVS, Calibre RVE, and a custom interface to launch a batch Calibre nmDRC job. You can use these interfaces to launch a Calibre verification run from Silvaco Expert, then use Calibre RVE to highlight the errors in the open design.

Also see “[Silvaco](#)” on the Calibre Interfaces web page for additional information.

<b>Creating the Interface to Silvaco Expert .....</b>	<b>576</b>
<b>Setting Socket Connections with Silvaco Expert .....</b>	<b>576</b>
<b>Calibre Menu in Silvaco Expert .....</b>	<b>576</b>

## Creating the Interface to Silvaco Expert

Silvaco Expert requires the MGC\_HOME environment variable to be set and pointing to the location of the Calibre installation. It is good practice to also define CALIBRE\_HOME in the same manner.

No additional setup is needed to enable the Calibre integration. The integration automatically adds the **Verification > Calibre** menu items to Silvaco Expert; these menu selections are used to start a Calibre job. See “[Calibre Menu in Silvaco Expert](#)” on page 576 for more information.

### Related Topics

[Setting the CALIBRE\\_HOME Environment Variable \[Calibre Administrator's Guide\]](#)

## Setting Socket Connections with Silvaco Expert

Calibre Interactive and Calibre RVE use certain default port numbers for socket communication with a connected design tool. By default, Calibre Interactive and Calibre RVE use socket port 9189 to communicate with a layout editor. The tool automatically searches for an available socket if it cannot obtain the default socket. When Silvaco Expert successfully opens a socket, it reports the socket number to the log file.

You can manually set the socket number with the MGC\_CALIBRE\_LAYOUT\_SERVER environment variable. See “[Setting the Calibre Interactive Socket Port for a Layout Viewer](#)” on page 451 for additional information.

## Calibre Menu in Silvaco Expert

The **Calibre** menu in Silvaco Expert is at **Verification > Calibre**.

The following table describes the menu selections and provides basic instructions for running Calibre.

**Table A-16. Calibre Menu in Silvaco Expert**

Menu Selection	Description
Verification > Calibre > Run DRC	<p>Opens Calibre Interactive nmDRC.</p> <p>You can enable “Show results in RVE” on the Outputs pane to have Calibre RVE open automatically when the run is finished, then highlight DRC results in Expert.</p> <p>See “<a href="#">Calibre Interactive nmDRC</a>” on page 137 and “<a href="#">Using Calibre RVE for DRC</a>” in the <i>Calibre RVE User’s Manual</i> for more information.</p>
Verification > Calibre > Run LVS	<p>Opens Calibre Interactive nmLVS.</p> <p>You can enable “Start RVE after LVS finishes” on the Outputs pane to have Calibre RVE open automatically when the run is finished, then highlight errors in Expert.</p> <p>See “<a href="#">Calibre Interactive nmLVS</a>” on page 173 and “<a href="#">Using Calibre RVE for LVS</a>” in the <i>Calibre RVE User’s Manual</i> for more information.</p>
Verification > Calibre > Batch DRC	<p>Opens a dialog box to start a batch Calibre nmDRC run.</p> <p>Fill in parameters, then click <b>OK</b> to start the Calibre run.</p> <p>To inspect the errors after the run finishes, choose <b>Verification &gt; DRC &gt; Errors &gt; Load Errors</b> in Expert.</p>

## Other Viewer Integrations

Some viewer integrations are not discussed here because the integration is straightforward and does not require any special setup actions or custom commands for operation.

The full list of design tools with a Calibre integration is given in “[Integration to Design Tools](#)” on page 25. If the design tool is listed but not discussed in detail in this appendix, then no special setup is needed.



# Appendix B

## Environment Variables

Calibre Interactive and Calibre View generation read a number of environment variables that specify run-time behavior. Some environment variables apply to more than one application.

The *Calibre Administrator's Guide* lists general and application specific Calibre environment variables in the appendix “[Calibre Environment Variables](#)”.

<b>Environment Variables for Calibre Interactive .....</b>	<b>579</b>
<b>Environment Variables for Calibre View .....</b>	<b>585</b>

## Environment Variables for Calibre Interactive

Several environment variables are available to control the behavior of Calibre Interactive.

The following table lists environment variables used in Calibre Interactive. The Type column indicates whether the environment variable is available in the shell environment (env) or in SKILL scripts.

**Table B-1. Calibre Interactive Environment Variables**

Variable	Type	Description
CALIBRE_CI_REMOTE_COMMAND _USE_BASH	env	When set, use the Bash shell to launch the MTflex primary script.
CALIBRE_CI_REMOTE_COMMAND _LSF_USE_BASH	env	When set, use the Bash shell to launch the MTflex primary script for Spectrum LSF runs only.
CALIBRE_FX_DB_DIFF_OPTIONS	env	Sets DBdiff options for a Calibre FastXOR run. See “ <a href="#">Fast XOR Page</a> ” on page 152.
CALIBRE_FX_PASS_THROUGH_LA YER_ENABLE	env	Specifies handling for unique layers in a Calibre FastXOR run. See “ <a href="#">Fast XOR Page</a> ” on page 152.
CALIBRE_HOME	env	Sets the location of the software tree for Calibre tools; see “ <a href="#">Setting the CALIBRE_HOME Environment Variable</a> ” in the <i>Calibre Administrator's Guide</i> .
CDS_NETLISTING_MODE	env or SKILL	Sets netlisting export mode in Cadence Virtuoso

**Table B-1. Calibre Interactive Environment Variables (cont.)**

Variable	Type	Description
MGC_CALGUI_RELEASE_LICENSE_TIME	env	Sets the time after which the Calibre Interactive license is released if there is no activity. The default time unit is hours.
MGC_CALIBRE_ALLOW_VIEWER_TRIGGERS	env or SKILL	Specifies access level for triggers running in the design tool.
MGC_CALIBRE_CGIDB_DIR	env	Sets location of preference files.
MGC_CALIBRE_CUSTOMIZATION_FILE	env	Sets location of a classic Calibre Interactive customization file; see “ <a href="#">Loading a Classic Tcl Customization File in Calibre Interactive</a> ” on page 232.
MGC_CALIBRE_DB_READ_OPTIONS	env	Sets database translation options for OpenAccess and LEF/DEF database. This environment variable must be set when using a Calibre Interactive control file to perform a command line Calibre run if the layout database format is OpenAccess or LEF/DEF.  See “ <a href="#">OA/LEFDEF and OPENACCESS Pages in Calibre Interactive</a> ” on page 112.
MGC_CALIBRE_DRC_CONFIG_FILE	env	Specifies one or more configuration files to load in Calibre Interactive nmDRC. Specify multiple files as a space-separated list enclosed in quotes. See “ <a href="#">Loading a Configuration File in Calibre Interactive</a> ” on page 230.
MGC_CALIBRE_DRC_RUNSET_FILE	env	Sets the Calibre Interactive DRC runset (“ <a href="#">Environment Variables for Calibre Interactive Runsets</a> ” on page 38).
MGC_CALIBRE_DRC_RUNSET_LIST	env	Populates Calibre Interactive DRC runsets in the Load Runset File dialog box and opens the Load Runset dialog at startup (“ <a href="#">Populating the Load Runset Files Dialog Box and Recent Files Lists</a> ” on page 39).
MGC_CALIBRE_EXPORT_LAYOUT_TEMPLATE_FILE	SKILL	Sets layout export template file location in Cadence Virtuoso.
MGC_CALIBRE_EXPORT_NETLIST_TEMPLATE_FILE	SKILL	Sets netlist template file location in Cadence Virtuoso.

**Table B-1. Calibre Interactive Environment Variables (cont.)**

Variable	Type	Description
MGC_CALIBRE_LAYOUT_SERVER	env	<p>Specifies a hostname and communications socket port number for Calibre Interactive and Calibre RVE to use for communication with a connected layout viewer.</p> <p>The format is: [&lt;host&gt;:]&lt;port&gt;</p> <p>The optional &lt;host&gt; parameter is set to localhost by default, but can be set to a different host. The &lt;port&gt; parameter must be a numerical value.</p> <p>To disable socket communication to the layout viewer, set MGC_CALIBRE_LAYOUT_SERVER to “ ” (blank). The layout server then does not attempt to open the communication socket.</p>
MGC_CALIBRE_LAYOUT_SERVER_NAME	env	This environment variable is set automatically by the tool to the type of layout viewer.
MGC_CALIBRE_LAYOUT_TMP_FILE	env	<p>Sets the name of the intermediate file used in layout editor communication.</p> <p>When set, the filename is used for the “Highlight Data File” setting on the Run Control page, Design Tool Settings area. The default filename is <i>query_results</i>.</p>
MGC_CALIBRE_LVS_CONFIG_FILE	env	<p>Specifies one or more configuration files to load in Calibre Interactive nmLVS. Specify multiple files as a space-separated list enclosed in quotes. See <a href="#">“Loading a Configuration File in Calibre Interactive” on page 230</a>.</p>
MGC_CALIBRE_LVS_RUNSET_FILE	env	<p>Sets the Calibre Interactive LVS runset (<a href="#">“Environment Variables for Calibre Interactive Runsets” on page 38</a>).</p>
MGC_CALIBRE_LVS_RUNSET_LIST	env	<p>Populates Calibre Interactive LVS runsets in the Load Runset File dialog box and opens the Load Runset dialog at startup (<a href="#">“Populating the Load Runset Files Dialog Box and Recent Files Lists” on page 39</a>).</p>

**Table B-1. Calibre Interactive Environment Variables (cont.)**

Variable	Type	Description
MGC_CALIBRE_MAP_BUS_NAME	env or SKILL	When interfacing with a Cadence schematic view, automatically converts between the [] bus name delimiter for Calibre Interactive and Calibre RVE and the <> delimiter used by Cadence.
MGC_CALIBRE_MRU_SIZE	env	Sets the list length of the “most recently used” lists in Calibre Interactive and Calibre RVE. The environment variable also controls the list length for the Recent Runsets and Recent Text Files menu selections.  The default list length is 20. The maximum length is 50.
MGC_CALIBRE_PERC_CONFIG_FILE	env	Specifies one or more configuration files to load in Calibre Interactive PERC. Specify multiple files as a space-separated list enclosed in quotes. See <a href="#">“Loading a Configuration File in Calibre Interactive” on page 230</a> .
MGC_CALIBRE_PERC_RUNSET_FILE	env	Sets the Calibre Interactive PERC runset ( <a href="#">“Environment Variables for Calibre Interactive Runsets” on page 38</a> ).
MGC_CALIBRE_PERC_RUNSET_LIST	env	Populates Calibre Interactive PERC runsets in the Load Runset File dialog box and opens the Load Runset dialog at startup ( <a href="#">“Populating the Load Runset Files Dialog Box and Recent Files Lists” on page 39</a> ).
MGC_CALIBRE_PEX_CONFIG_FILE	env	Specifies one or more configuration files to load in Calibre Interactive PEX. Specify multiple files as a space-separated list enclosed in quotes. See <a href="#">“Loading a Configuration File in Calibre Interactive” on page 230</a> .
MGC_CALIBRE_PEX_RUNSET_FILE	env	Sets the Calibre Interactive PEX runset ( <a href="#">“Environment Variables for Calibre Interactive Runsets” on page 38</a> ).

**Table B-1. Calibre Interactive Environment Variables (cont.)**

Variable	Type	Description
MGC_CALIBRE_PEX_RUNSET_LIST	env	Populates Calibre Interactive PEX runsets in the Load Runset File dialog box and opens the Load Runset dialog at startup (“ <a href="#">Populating the Load Runset Files Dialog Box and Recent Files Lists</a> ” on page 39).
MGC_CALIBRE_SAVE_ALL_RUNSET_VALUES	env	Controls whether you save both default and non-default data to a runset file (“ <a href="#">Environment Variables for Calibre Interactive Runsets</a> ” on page 38).
MGC_CALIBRE_SCHEMATIC_SERVER	env	<p>Specifies a hostname and communications socket port number Calibre Interactive and Calibre RVE to use for communication with a connected schematic viewer.</p> <p>The format is: [&lt;host&gt;:]&lt;port&gt;</p> <p>The optional &lt;host&gt; parameter is set to localhost by default, but can be set to a different host. The &lt;port&gt; parameter must be a numerical value.</p> <p>To disable socket communication to the layout viewer, set MGC_CALIBRE_SCHEMATIC_SERVER to “ ” (blank). The schematic server then does not attempt to open the communication socket.</p>
MGC_CALIBRE_SCHEMATIC_SERVER_NAME	env	This environment variable is set automatically by the tool to the type of schematic viewer.
MGC_CALIBRE_VERILOG_EXPORT_PARAM_FILE	env	<p>Specifies the path to a file with saveNetlist parameters for Cadence Encounter.</p> <p>See “<a href="#">Importing saveNetlist Parameters for Verilog Export in Cadence Encounter</a>” on page 560.</p>
MGC_CALIBRE_XACT_RUNSET_FILE	env	<p>Used in Calibre Interactive xACT to pre-populate the Recent Runsets and Recent Text Files lists in the File menu, and the Load Runset Files dialog box.</p> <p>See “<a href="#">Environment Variables for Calibre Interactive Runsets</a>” on page 38.</p>

**Table B-1. Calibre Interactive Environment Variables (cont.)**

Variable	Type	Description
MGC_CALIBRE_XACT_RUNSET_LIST	env	Populates Calibre Interactive xACT runsets in the Load Runset File dialog box and opens the Load Runset dialog at startup (“ <a href="#">Populating the Load Runset Files Dialog Box and Recent Files Lists</a> ” on page 39).
MGC_CI_EXPORT_ENABLE_COLORING	env	When set, specifies to export coloring information during streamout from Cadence Virtuoso. The option “Enable coloring” is checked in the Calibre Layout Export Setup dialog box. By default, coloring information is not exported.
MGC_FDI_OA_VERSION	env	Sets the Open Access version for FDI and DBdiff utilities. The allowed values are 22.43 and 22.50. The default is 22.43.  If you specify the OA version in the fdi2gds or fdi2oasis utilities, the environment variable is ignored.  Also see “ <a href="#">Providing OpenAccess Input with Calibre Interactive</a> ” on page 73 and “ <a href="#">FDI Environment Variables</a> ” in the <i>Calibre Layout Comparison and Translation Guide</i> .
MGC_RVE_INIT_SOCKET_AT_STARTUP	env	Sets socket initiation preference for Cadence Virtuoso. Used for both Calibre Interactive and Calibre RVE.
MGCCALIBREMENUVIEWTYPE	SKILL	Specifies the window type in which to load a <b>Calibre</b> menu. Use this when you are not using one of the supported Cadence layout views.  See “ <a href="#">Creating an Interface to Cadence Virtuoso</a> ” on page 469.

**Table B-1. Calibre Interactive Environment Variables (cont.)**

<b>Variable</b>	<b>Type</b>	<b>Description</b>
OA_HOME <sup>1</sup>	env	<p>Calibre uses the OpenAccess library at \$CALIBRE_HOME/shared/pkgs/icv_o. 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.</p> <p>Note: As of the 2013.3 release, the supported OpenAccess version in Calibre is oa22.43p006. Also see footnote.</p>
PEX_FMT_HP_PORT_MAP_MODE	env	Specifies the use of templates in Calibre Interactive PEX. When this environment variable is defined, the templates directory in the SVDB is not deleted before a run. See “ <a href="#">Templates</a> ” in the <i>Calibre xRC User’s Manual</i> .

1. If this variable is set, then the specified library must be version oa22.43p006 or later. For Linux platforms, 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.

## Environment Variables for Calibre View

Several environment variables are available to control Calibre View generation.

The following table lists environment variables used in Calibre View generation. The Type column indicates whether the environment variable is available in the shell environment (env) or in SKILL scripts.

Also see “[Global Variables for Calibre View Generation](#)” on page 542.

**Table B-2. Calibre View Environment Variables**

Variable	Type	Description
CALIBRE_HOME	env	Specifies the software tree invoked when Calibre Interactive or Calibre RVE is called from Pyxis (“ <a href="#">Using Calibre Interactive with Pyxis</a> ” on page 467). Also sets the location of the software tree for Calibre tools; see “ <a href="#">Setting the CALIBRE_HOME Environment Variable</a> ” in the <i>Calibre Administrator’s Guide</i> .
CALIBRE_DISABLE_PPAR_WARNINGS	env	Set this environment variable to disable pPar evaluation warnings.
CALIBRE_FDIBA_ENABLE_NETEXPR_CO PY	env	See “ <a href="#">Handling of Nets With Inherited Connectivity</a> ” on page 497.
MGC_RVE_INIT_SOCKET_AT_STARTUP	env	Sets socket initiation preference for Cadence Virtuoso (“ <a href="#">Calibre Setup Menu in Cadence Virtuoso</a> ” on page 473) and Synopsys IC Compiler (“ <a href="#">Setting Socket Connections with Synopsys IC Compiler and IC Compiler II</a> ” on page 573).
MGC_RVE_RELEASE_LICENSE_TIME	env	Sets the time in hours after which the Calibre RVE license is released if there is no activity ( <a href="#">License Timeout in Calibre RVE</a> ” in the <i>Calibre RVE User’s Manual</i> ). If the environment variable MGC_CALGUI_RELEASE_LIC ENSE_TIME is not set, then MGC_RVE_RELEASE_LICENS E_TIME also sets the license timeout for Calibre Interactive. (“ <a href="#">License Timeout in Calibre Interactive</a> ” on page 60).
MGCCALIBREMENUVIEWTYPE	SKILL	Specifies window type in which to load a <b>Calibre</b> menu (“ <a href="#">Creating an Interface to Cadence Virtuoso</a> ” on page 469).

**Table B-2. Calibre View Environment Variables (cont.)**

Variable	Type	Description
OA_HOME <sup>1</sup>	env	Sets OpenAccess library version to one other than the default. The OpenAccess default library that ships with Calibre is located at ./shared/pkgs/icv_o in your Calibre software tree.  Note: As of the 2013.3 release, the supported OpenAccess version in Calibre is oa22.43p006. Also see footnote.

1. If this variable is set, then the specified library must be version oa22.43p006 or later. For Linux platforms, 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.



# Appendix C Runset Options

---

A Calibre Interactive runset is a text file created by Calibre Interactive to store the settings you specify in the GUI. You can load a runset into Calibre Interactive to restore the GUI interface to the saved settings. This appendix gives the runset file format and option names.

<b>Calibre Interactive Runset File Format</b> .....	<b>590</b>
<b>Common Runset Options</b> .....	<b>591</b>
<b>DRC Runset Options</b> .....	<b>624</b>
<b>LVS Runset Options</b> .....	<b>650</b>
<b>PERC Runset Options</b> .....	<b>684</b>
<b>PEX Runset Options</b> .....	<b>711</b>
<b>Trigger Functions in the Calibre Interactive Runset</b> .....	<b>798</b>
<b>Template Definitions</b> .....	<b>799</b>
<b>Waiver Setup Runset Options</b> .....	<b>805</b>

# Calibre Interactive Runset File Format

Input for: Calibre Interactive

Created by: Calibre Interactive

Calibre Interactive runset files are built automatically using the Calibre Interactive GUI.  
Although not recommended, it is possible to manually edit the options in a runset file.

## Format

Most runset options are set by assigning the “value” or “specified” property. For example:

```
pex.rulesFile.value = "rules"
pex.runDir.value = "project"
cmn.applyTemplates.value = true
cmn.turboCommand.runHow.value = "mtflex"

cmn.layoutTextFile.specified = true
cmn.layoutTextFile.value = "layoutText.svrf"

cmn.showOptionPages.value = [ "Options", "Preferences" ]
```

- Option names are prefixed by either cmn or pex.
- String values are quoted, although the quotes may not be included in the list of “Available Values” in the runset option tables. The values true and false are not quoted.
- Some settings have both a value and a checkbox that enables use of the setting. To use the option, the “specified” property must be set to true, as in the preceding example for the layoutTextFile runset option.
- Some values are lists, which are enclosed in [] brackets.
- Some options are set by defining the “parameters” property (not shown in the example). The lists of runset options give the property name.

Some runset entries are functions, such as the following, which defines an environment variable to use during the run:

```
pex.envVars.setEnv("name", "value")
```

Template definitions are also saved to the runset as functions. Trigger definitions are saved to the runset as an object declaration and command definition.

## Parameters

The entries for the runset file are described in these sections:

- [Common Runset Options](#)

Options that are common to all applications.

- [DRC Runset Options](#)  
Options that are used in Calibre Interactive nmDRC.
- [LVS Runset Options](#)  
Options that are used in Calibre Interactive nmLVS.
- [PERC Runset Options](#)  
Options that are used in Calibre Interactive PERC.
- [PEX Runset Options](#)  
Options that are used in Calibre Interactive PEX.
- [Trigger Functions in the Calibre Interactive Runset](#)  
Options and functions that are used to define Calibre Interactive Internal triggers.
- [Template Definitions](#)  
The function call for defining a template setting.
- [Waiver Setup Runset Options](#)  
Options that are used for Calibre Auto-Waivers setup in Calibre Interactive nmDRC and for ERC runs Calibre Interactive nmLVS.

## Common Runset Options

The common runset options have the prefix cmn.

The common options apply to all Calibre Interactive applications unless otherwise noted in the Name column.

**Table C-1. General Runset Options for Calibre Interactive**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.applyTemplates.value	Boolean that controls whether templates are applied when Calibre Interactive is started from a design tool. <b>Templates</b> page	[true false] (Boolean) {true}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.cmnRunControl.clusterCommand.value	The command used to submit a job to the remote cluster. Used when cmn.cmnRunControl.runWhere = “cluster”.  <b>Run Control</b> page, “Run Calibre On” section, “Other Cluster”.	[] () { }
cmn.cmnRunControl.lsfHostName.value (not xACT)	Specify the name of the host used to submit jobs to the LSF queue.  <b>Run Control</b> page, “Submission Host”	[] () {localhost}
cmn.cmnRunControl.lsfInteractiveSubmission.value (not xACT)	Specify whether LSF submission is interactive.  <b>Run Control</b> page, “Interactive job Submission”	[true false] (Boolean) {true}
cmn.cmnRunControl.remoteClusterSync.value	Checkbox for “Synchronous” in Calibre runs on a remote cluster (cmn.cmnRunControl.runWhere = “cluster”).  <ul style="list-style-type: none"> <li>• true — Calibre Interactive waits for the Calibre run to complete. Calibre RVE and reports are displayed when the run is finished if appropriate options are set.</li> <li>• false — Calibre Interactive executes the queue command as a background process. The run transcript, Calibre RVE, and reports are not displayed when the run finishes.</li> </ul> <b>Run Control</b> page, “Run Calibre On” section, “Synchronous”.	[false true] (Boolean) {false}
cmn.cmnRunControl.remoteHostName.value	The name of the remote host for the Calibre run. Used when cmn.cmnRunControl.runWhere = “remote”.  <b>Run Control</b> page, “Run Calibre On” section, “Host Name”.	[] () {localhost}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.cmnRunControl.runWhere.value	<p>Specifies the host processor for the Calibre run.</p> <ul style="list-style-type: none"> <li>• local — Run on the host that is running Calibre Interactive.</li> <li>• remote — Run on the remote host specified with cmn.cmnRunControl.remoteHostName.value.</li> <li>• cluster — Run on a computer cluster. The launch command is specified with cmn.cmnRunControl.clusterCommand.value.</li> <li>• lsf — Run on IBM Spectrum LSF. Not available in Calibre Interactive xACT.</li> </ul> <p><b>Run Control</b> page, “Run Calibre On” section, “Run Calibre xACT on”</p>	[local remote lsf cluster] ( ) {local}
cmn.customPrefs.customFile.specified	Boolean controlling cmn.customPrefs.customFile.	[false true] (Boolean) {false}
cmn.customPrefs.customFile.value	Specifies the name of the customization file.	[] ( ) { }
cmn.disablePrefs.disableEditRecipes.value (DRC)	<p>Disables editing of check selection recipes.</p> <p><b>Preferences</b> page, Disable, “Disable check selection recipe editing”</p>	[false true] (Boolean) {false}
cmn.disablePrefs.disableFXCombineUsage.value (DRC)	<p>When true, the Use Layer Map File and Use Base Layers options are mutually exclusive.</p> <p><b>Preferences</b> page, Disable, “Disable Fast XOR layer map and base layers combined usage”</p>	[true false] (Boolean) {true}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.disablePrefs.disableFXMaxDiffPref.value (DRC)	When true, the “Max Diff” control on the Fast XOR page is not displayed. <b>Preferences</b> page, Disable, “Disable Fast XOR max diff count control”	[true false] (Boolean) {true}
cmn.disablePrefs.disableFXOptions.value (DRC)	When true, disables the Options area on the Fast XOR page. <b>Preferences</b> page, Disable, “Disable Fast XOR options”	[true false] (Boolean) {true}
cmn.disablePrefs.disableFXRuleCustomName.value (DRC)	When true, disables the option to specify a custom name for the generated rule file in a FastXOR run. <b>Preferences</b> page, Disable, “Disable Fast XOR generated rules file custom name”	[true false] (Boolean) {true}
cmn.disablePrefs.disableSelectChecks.value (DRC)	When true, disables changing of the DRC check selection recipe. <b>Preferences</b> page, Disable, “Disable check selection”	[false true] (Boolean) {false}
cmn.ipcSettings.layoutHostName.value	The name of the host running the layout viewer. <b>Run Control</b> page, “Design Tool Settings” section	[] () {localhost}
cmn.ipcSettings.layoutSocketNumber.value	The socket port number of the layout viewer. Enter a value from 1024 to 65535. <b>Run Control</b> page, “Design Tool Settings” section	[] (integer) {9189}
cmn.ipcSettings.layoutTmpFile.specified	Boolean controlling cmn.ipcSettings.layoutTmpFile. <b>Run Control</b> , “Design Tool Settings” section	[false true] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.ipcSettings.layoutTmpFile.value	The name of the intermediate file to use for sending layout highlight data to the layout tool.  This setting is used if cmn.ipcSettings.layoutTmpFile.specified = true.  <b>Run Control</b> , “Design Tool Settings” section	[] () {query_results}
cmn.ipcSettings.schematicHostName.value	The name of the host running the schematic viewer.  This setting is used if cmn.ipcSettings.schematicUseLayoutIPC.value = false.  <b>Run Control</b> page, “Design Tool Settings” section	[] () {localhost}
cmn.ipcSettings.schematicSocketNumber.value	The socket port number of the schematic viewer. Enter a value from 1024 to 65535.  This setting is used if cmn.ipcSettings.schematicUseLayoutIPC.value = false.  <b>Run Control</b> page, “Design Tool Settings” section	[] (integer) {9199}
cmn.ipcSettings.schematicUseLayoutIPC.value	Boolean that specifies if the schematic viewer uses the same socket port as the layout viewer.  <b>Run Control</b> page, “Design Tool Settings” section	[true false] (Boolean) {true}
cmn.layout.defDirs.value (DRC, LVS, and PERC)	Specify one or more directories containing DEF files.  <b>Inputs</b> page, Layout Path, “DEF Directories”	[] () {}
cmn.layout.defFiles.value (DRC, LVS, and PERC)	Specify one or more DEF files.  <b>Inputs</b> page, Layout Path, “DEF Files”	[] () {}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.layout.layoutLibraryFDI.value	The name of the OPENACCESS library definitions file. This could be the path to either the <i>cds.lib</i> or <i>lib.defs</i> library definitions file.  Used only when the layout format is OPENACCESS.  <b>Inputs</b> page, Layout Path	[] () {}
cmn.layout.lefDirs.value (DRC, LVS, and PERC)	Specify either a single or multiple directories containing LEF files.  <b>Inputs</b> page, LEF Directories	[] () {}
cmn.layout.lefFiles.value (DRC, LVS, and PERC)	Specify either a single or multiple LEF files.  <b>Inputs</b> page, LEF Files	[] () {}
cmn.layout.lefTechFiles.value (DRC, LVS, and PERC)	Specify either a single or multiple LEF technology files.  <b>Inputs</b> page, LEF Technology Files	[] () {}
cmn.layout.topCellLibraryFDI.value	The library name for the top cell in an OPENACCESS database.  Used only when the layout format is OPENACCESS.  <b>Inputs</b> page, Layout Path	[] () {}
cmn.layout.topCellViewFDI.value	The view name for the top cell in an OPENACCESS database.  Used only when the layout format is OPENACCESS.  <b>Inputs</b> page, Layout Path	[] () {}
cmn.layoutTextFile.specified	Boolean controlling cmn.layoutTextFile  <b>Options</b> page, “Include Layout Text File”	[false true] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.layoutTextFile.filePath.value	Specifies a file that contains Layout Text objects.  <b>Options</b> page, “Include Layout Text File” section	[] () { }
cmn.licenseTimeout.specified	Boolean controlling cmn.licenseTimeout  <b>Run Control</b> page, “Calibre Interactive License Timeout”	[false true] (Boolean) {false}
cmn.licenseTimeout.timeout.value	The number of hours of inactivity after which to release the license.  <b>Run Control</b> page, “Calibre Interactive License Timeout” section	[] () {1}
cmn.licensing.loopIntervalTime.specified (not xACT)	Boolean controlling cmn.licensing.loopIntervalTime  Used when cmn.licensing.mode.value = LMRETRY.  <b>Run Control</b> page, Calibre Licensing, “Wait time between retries (seconds)” checkbox	[false true] (Boolean) {false}
cmn.licensing.loopIntervalTime.value (not xACT)	The wait time in seconds between retries for a Calibre license. (INTERVAL)  Used when cmn.licensing.mode.value = LMRETRY.  <b>Run Control</b> page, Calibre Licensing, “Wait time between retries (seconds)”	[] (positive integer) {60}
cmn.licensing.loopRetryOnError.value (not xACT)	Specify whether to retry for the license when errors occur. (RETRY_ON_ERROR)  Used when cmn.licensing.mode.value = LMRETRY.  <b>Run Control</b> page, Calibre Licensing, “Retry even when license errors occur”	[false true] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.licensing.loopTime.specified (not xACT)	Boolean controlling cmn.licensing.loopTime Used when cmn.licensing.mode.value = LMRETRY.  <b>Run Control</b> page, Calibre Licensing, “Retry time for license (minutes)” checkbox	[false true] (Boolean) {false}
cmn.licensing.loopTime.value (not xACT)	The total time in minutes to retry for a Calibre license. (MAXRETRY) Used when cmn.licensing.mode.value = LMRETRY.  <b>Run Control</b> page, Calibre Licensing, “Retry time for license (minutes)”	[] (positive integer) {180}
cmn.licensing.mode.specified (not xACT)	Boolean controlling cmn.licensing.mode Checkbox that specifies whether to use Calibre licensing options. When this option is set to false, the Calibre command line does not include any licensing arguments.  <b>Run Control</b> page, Calibre Licensing, “Licensing Modes” checkbox	[true false] (Boolean) {true}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.licensing.mode.value (not xACT)	<p>Specifies the Calibre license queuing mode. The command line option is given in parentheses.</p> <ul style="list-style-type: none"> <li>• NOWAIT — Do not wait for a license. (-nowait)</li> <li>• WAIT — Queue for a license. The wait time is given by cmn.licensing.waitTime.value. (-wait &lt;time&gt;)</li> <li>• LMRETRY — Retry, or loop, for a license. (-lmretry loop)</li> </ul> <p><b>Run Control</b> page, Calibre Licensing, “Licensing Modes” dropdown list</p>	[NOWAIT WAIT LMRETRY] () {NOWAIT}
cmn.licensing.remoteLICENSEFILE Name.value	<p>For runs on remote hosts or computer clusters, the name of the licensing environment variable to use on the remote host.</p> <p><b>Run Control</b> page, “Calibre Licensing” section, “License File Name”</p>	[LM_LICENSE _FILE MGLS_LICENS E_FILE] () {LM_LICENSE _FILE}
cmn.licensing.remoteLICENSEFILE UseCurrent.value	<p>Specifies what license file value to use on remote hosts:</p> <ul style="list-style-type: none"> <li>• CURRENT — Use the current value.</li> <li>• SPECIFY — Specify the license file value (set with cmn.licensing.remoteLICENSEFILEValue.value).</li> </ul> <p><b>Run Control</b> page, “Calibre Licensing” section, “Use”</p>	[CURRENT SPECIFY] () {CURRENT}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.licensing.remoteLICENSEFILE Value.value	The value of the remote licensing environment variable to use when cmn.licensing.remoteLICENSEFILE UseCurrent.value is set to SPECIFY.  <b>Run Control</b> page, “Calibre Licensing” section, “License File Value”	[] () {}
cmn.licensing.useCBForRVE.value (DRC, LVS, PEX)	Specifies whether to use the Calibre-CB license for Calibre RVE.  Only available when the run mode is Calibre CB.  <b>Run Control</b> page, “Calibre Licensing” section, “Use Calibre-CB license for RVE”	[true false] (Boolean) {true}
cmn.licensing.waitTime.value (not xACT)	The time in minutes to wait for a license. (-wait <time>).  Used when cmn.licensing.mode.value = WAIT.  <b>Run Control</b> page, “Calibre Licensing” section, “Wait time for license (minutes)”	[] (positive integer) {0}
cmn.lsfMasterHosts.parameters (not xACT)	LSF Master Hosts	[] (list) {}
cmn.lsfMultiThreadHosts.parameters (not xACT)	LSF Multi Thread Hosts	[] (list) {}
cmn.lsfSingleThreadHosts.parameters (not xACT)	LSF Single Thread Hosts	[] (list) {}
cmn.lsfSlaveHosts.parameters (not xACT)	LSF Slave Hosts	[] (list) {}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.miscPrefs.ignoreExportInBatch.value	<p>Boolean that specifies how to handle layout and schematic export settings in a batch run.</p> <ul style="list-style-type: none"> <li>• true — Ignore export settings</li> <li>• false — Use export settings</li> </ul> <p>By default, export settings are ignored in batch runs.</p> <p><b>Preferences</b> page, “Misc” section, “Ignore layout and source export in batch mode”</p>	[true false] (Boolean) {true}
cmn.miscPrefs.overwriteLayoutPref.value	<p>Boolean that controls whether a dialog is shown when the layout database file is about to be overwritten during layout export.</p> <p><b>Preferences</b> page, “Misc” section, “Warn before overwriting layout database”</p>	[true false] (Boolean) {true}
cmn.miscPrefs.overwriteSourcePref.value (LVS, PERC, PEX, xACT)	<p>Boolean that controls whether a dialog is shown when the schematic source file is about to be overwritten during schematic export.</p> <p><b>Preferences</b> page, “Misc” section, “Warn before overwriting source database”</p>	[true false] (Boolean) {true}
cmn.miscPrefs.spiceInstPrefix.value (PEX, xACT)	<p>The prefix that the netlister adds to each instance in the schematic. Calibre Interactive adds this prefix to instance names obtained by clicking in the schematic.</p> <p><b>Preferences</b> page, “Misc” section, “SPICE instance prefix”</p>	[] () {X}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.miscPrefs.verifyCheckSelected.value (DRC)	Checkbox that specifies whether to verify that at least one rule check is selected before running Calibre DRC. <b>Preferences</b> page, “Misc” section, “Verify checks are selected before running DRC”	[true false] (Boolean) {true}
cmn.miscPrefs.verifySourceNetlist.value (LVS, PERC, PEX, xACT)	Boolean that controls whether to check the source netlist for errors before running Calibre xACT. <b>Preferences</b> page, “Misc” section, “Check source netlist is complete before running <app>”	[true false] (Boolean) {true}
cmn.miscPrefs.waiverSetupFileName.specified (DRC, LVS)	Boolean controlling cmn.miscPrefs.waiverSetupFileName <b>Preferences</b> page, “Misc” section, “Specify waiver setup file name” checkbox	[false true] (Boolean) {false}
cmn.miscPrefs.waiverSetupFileName.value (DRC, LVS)	The name of the waiver setup file created by Calibre Interactive when creating or using waivers. This value is used when cmn.miscPrefs.waiverSetupFileName.specified is set to true. <b>Preferences</b> page, “Misc” section, “Specify waiver setup file name”	[] () {}
cmn.miscPrefs.warnLayoutFormatsDiff.value	Boolean that controls whether to warn when the format of a library layout file differs from the primary layout format. This setting applies only when the primary layout format is GDS or OASIS. <b>Preferences</b> page, “Misc” section, “Warn when library layout file format differs from primary layout format”	[true false] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.oaEnvOptions.setEnableExpPCell.value	Boolean that controls the “Set CDS_ENABLE_EXP_PCELL ...” option. <b>OPENACCESS</b> page, “OpenAccess Environment” section	[true false] (Boolean) {false}
cmn.oaEnvOptions.setExpAllParams.value	Boolean that controls the “Set CDS_EXP_ALL_PARAMS ...” option. <b>OPENACCESS</b> page, “OpenAccess Environment” section	[true false] (Boolean) {false}
cmn.oaEnvOptions.setExpPCellDir.value	Boolean that controls the “Set CDS_EXP_PCELL_DIR ...” option. <b>OPENACCESS</b> page, “OpenAccess Environment” section	[true false] (Boolean) {false}
cmn.oaEnvOptions.setLdLibPath.value	Boolean that controls the “Set CALIBRE_READDB_LD_LIBRARY_PATH ...” option. <b>OPENACCESS</b> page, “OpenAccess Environment” section	[true false] (Boolean) {false}
cmn.oaEnvOptions.setOaHome.value	Boolean that controls the “Set OA_HOME when ...” option. <b>OPENACCESS</b> page, “OpenAccess Environment” section	[true false] (Boolean) {false}
cmn.oaEnvOptions.setPath.value	Boolean that controls the “Set PATH when ...” option. <b>OPENACCESS</b> page, “OpenAccess Environment” section	[true false] (Boolean) {false}
cmn.oaLefDefAdditionalFiles.cellGDSDirectories.value	Cell GDS directory list used in database conversion. Used when cmn.oaLefDefAdditionalFiles.useCellGDSFiles.value = true. <b>OPENACCESS</b> page, “Additional Files” section	[] () {}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.oaLefDefAdditionalFiles.cellGDSFiles.value	Cell GDS file list used in database conversion. Used when cmn.oaLefDefAdditionalFiles.useCellGDSFiles.value = true.  <b>OPENACCESS</b> page, “Additional Files” section	[] () { }
cmn.oaLefDefAdditionalFiles.useCellGDSFiles.value	Boolean that controls the “Use Cell GDS Files” checkbox.  <b>OPENACCESS</b> page, “Additional Files” section	[true false] (Boolean) {false}
cmn.oaLefDefMappingFiles.cellMapFile.specified	Boolean controlling cmn.oaLefDefMappingFiles.cellMapFile  <b>OPENACCESS</b> page, “Mapping Files” section	[false true] (Boolean) {false}
cmn.oaLefDefMappingFiles.cellMapFile.value	The cell map file used in database conversion.  <b>OPENACCESS</b> page, “Mapping Files” section	[] () { }
cmn.oaLefDefMappingFiles.inputExceptionFile.specified	Boolean controlling cmn.oaLefDefMappingFiles.inputExceptionFile  <b>OPENACCESS</b> page, “Mapping Files” section	[false true] (Boolean) {false}
cmn.oaLefDefMappingFiles.inputExceptionFile.value	The input exception file used in database conversion.  <b>OPENACCESS</b> page, “Mapping Files” section	[] () { }
cmn.oaLefDefMappingFiles.layerMapFile.specified	Boolean controlling cmn.oaLefDefMappingFiles.layerMapFile  <b>OPENACCESS</b> page, “Mapping Files” section	[false true] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.oaLefDefMappingFiles.layerMapFile.value	The layer map file used in layout database conversion. <b>OPENACCESS</b> page, “Mapping Files” section	[] () {}
cmn.oaLefDefMappingFiles.netPropertiesFile.specified	Boolean controlling cmn.oaLefDefMappingFiles.netPropertiesFile <b>OPENACCESS</b> page, “Mapping Files” section	[false true] (Boolean) {false}
cmn.oaLefDefMappingFiles.netPropertiesFile.value	The net properties file used in layout database conversion. <b>OPENACCESS</b> page, “Mapping Files” section	[] () {}
cmn.oaLefDefMappingFiles.objectMapFile.specified	Boolean controlling cmn.oaLefDefMappingFiles.objectMapFile <b>OPENACCESS</b> page, “Mapping Files” section	[false true] (Boolean) {false}
cmn.oaLefDefMappingFiles.objectMapFile.value	The object map file used in layout database conversion. <b>OPENACCESS</b> page, “Mapping Files” section	[] () {}
cmn.oaLefDefOtherOptions.oaViewLists.value	The view list for OpenAccess libraries, <b>OPENACCESS</b> page, “Other Options” section	[] () {}
cmn.oaLefDefOtherOptions.writeLayerNamesFile.specified	Boolean controlling cmn.oaLefDefOtherOptions.writeLayerNamesFile <b>OPENACCESS</b> page, “Other Options” section	[false true] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.oaLefDefOtherOptions.writeLayerNamesFile.value	The output layer name file for “Write Layer Names File”. Used in layout database conversion. <b>OPENACCESS</b> page, “Other Options” section	[] () {}
cmn.oaLefDefOtherOptions.writeLogFile.specified	Boolean controlling cmn.oaLefDefOtherOptions.writeLogFile <b>OPENACCESS</b> page, “Other Options” section	[false true] (Boolean) {false}
cmn.oaLefDefOtherOptions.writeLogFile.value	The output log file for “Write Log File”. Used in layout database conversion. <b>OPENACCESS</b> page, “Other Options” section	[] () {}
cmn.oaLefDefOtherOptions.writeMapInfoFile.specified	Boolean controlling cmn.oaLefDefOtherOptions.writeMapInfoFile <b>OPENACCESS</b> page, “Other Options” section	[false true] (Boolean) {false}
cmn.oaLefDefOtherOptions.writeMapInfoFile.value	The output map info file for “Write Map InfoFile” Used in layout database conversion. <b>OPENACCESS</b> page, “Other Options” section	[] () {}
cmn.oaLefDefReadOptions.abortOnReadEmptyPCells.value	Boolean that controls whether to abort upon encountering empty PCells when reading an OpenAccess library. <b>OPENACCESS</b> page, “Read Options” section	[true false] (Boolean) {true}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.oaLefDefReadOptions.instanceNamesAsProperty.specified	Boolean controlling cmn.oaLefDefReadOptions.instanceNamesAsProperty <b>OPENACCESS</b> page, “Read Options” section	[false true] (Boolean) {false}
cmn.oaLefDefReadOptions.instanceNamesAsProperty.value	Value of the property name used for “Read Instance Name as Property”. <b>OPENACCESS</b> page, “Read Options” section	[] () {}
cmn.oaLefDefReadOptions.netNamesAsProperty.specified	Boolean controlling cmn.oaLefDefReadOptions.netNamesAsProperty <b>OPENACCESS</b> page, “Read Options” section	[false true] (Boolean) {false}
cmn.oaLefDefReadOptions.netNamesAsProperty.value	Value of the property name used for “Read Net Names as Property”. <b>OPENACCESS</b> page, “Read Options” section	[] () {}
cmn.oaLefDefReadOptions.netNamesAsText.specified	Boolean controlling cmn.oaLefDefReadOptions.netNamesAsText <b>OPENACCESS</b> page, “Read Options” section	[false true] (Boolean) {false}
cmn.oaLefDefReadOptions.netNamesAsText.value	Value set for “Read Net Names as Text”. <ul style="list-style-type: none"><li>• ALL — (default) Add net name annotations in all cells</li><li>• TOP — Add net name annotations in the top cell only</li></ul> This option is off by default. <b>OPENACCESS</b> page, “Read Options” section	[TOP ALL] () {ALL}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.oaLefDefReadOptions.netTypeAsProperty.specified	Boolean controlling cmn.oaLefDefReadOptions.netTypeAsProperty <b>OPENACCESS</b> page, “Read Options” section	[false true] (Boolean) {false}
cmn.oaLefDefReadOptions.netTypeAsProperty.value	Value of the property name used for “Read Net Type as Property”. <b>OPENACCESS</b> page, “Read Options” section	[] () {}
cmn.oaLefDefReadOptions.objects.value	Specifies the objects that are read when converting a third-party database to GDS or OASIS. Specify a comma-separated list enclosed in brackets. <ul style="list-style-type: none"> <li>• TEXTS — Read text objects</li> <li>• PINS — Read pin shapes.</li> <li>• BLOCKAGES — Read blockages for OpenAccess databases.</li> <li>• INSTANCEPINS — Read instance pin shapes.</li> </ul> Default: [ “TEXTS”, “PINS”] <b>OPENACCESS</b> page, “Read Options” section	[TEXTS PINS BLOCKAGES INSTANCEPINS] (list) {}
cmn.oaLefDefReadOptions.pinNamesAsProperty.specified	Boolean controlling cmn.oaLefDefReadOptions.pinNamesAsProperty <b>OPENACCESS</b> page, “Read Options” section	[false true] (Boolean) {false}
cmn.oaLefDefReadOptions.pinNamesAsProperty.value	Value of the property name used for “Read Pin Names as Property”. <b>OPENACCESS</b> page, “Read Options” section	[] () {}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.oaLefDefReadOptions.pinNamesAsText.value	Boolean controlling the “Read Pin Names as Text” checkbox. <b>OPENACCESS</b> page, “Read Options” section	[true false] (Boolean) {false}
cmn.remoteEnv.remoteMGCHOME.UseCurrent.value	Specifies what value of MGC_HOME to use for launching Calibre on the remote host. <ul style="list-style-type: none"> <li>• LOCAL — Use the localhost value of MGC_HOME</li> <li>• SPECIFY — Specify the value of MGC_HOME</li> </ul> <b>Run Control</b> page, Remote Calibre Environment section	[LOCAL SPECIFY] () {LOCAL}
cmn.remoteEnv.remoteMGCHOME._aoi.value	The value of MGC_HOME to use for launching Calibre on an aoi remote host. Used when cmn.remoteEnv.remoteMGCHOME.UseCurrent=SPECIFY. <b>Run Control</b> page, Remote Calibre Environment section	[] () {}
cmn.remoteEnv.remoteMGCHOME._aoj.value	The value of MGC_HOME to use for launching Calibre on an aoj remote host. Used when cmn.remoteEnv.remoteMGCHOME.UseCurrent=SPECIFY. <b>Run Control</b> page, Remote Calibre Environment section	[] () {}
cmn.remoteEnv.remoteMGCHOME._ixl.value	The value of MGC_HOME to use for launching Calibre on an ixl remote host. Used when cmn.remoteEnv.remoteMGCHOME.UseCurrent=SPECIFY. <b>Run Control</b> page, Remote Calibre Environment section	[] () {}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.remoteEnv.remoteMGCLIBPA THUseCurrent.value	<p>Specifies what value of MGC_LIB_PATH to use for Calibre on the remote host.</p> <ul style="list-style-type: none"> <li>• LOCAL — Use the localhost value of MGC_LIB_PATH</li> <li>• SPECIFY — Specify the value of MGC_LIB_PATH</li> </ul> <p><b>Run Control</b> page, Remote Calibre Environment section</p>	[LOCAL SPECIFY] ( {LOCAL}
cmn.remoteEnv.remoteMGCLIBPA TH_aoi.value	<p>The value of MGC_LIB_PATH to use for Calibre on an aoi remote host. Used when cmn.remoteEnv.remoteMGCLIBPAT HUseCurrent=SPECIFY.</p> <p><b>Run Control</b> page, Remote Calibre Environment section</p>	[] ( {}
cmn.remoteEnv.remoteMGCLIBPA TH_aoj.value	<p>The value of MGC_LIB_PATH to use for Calibre on an aoj remote host. Used when cmn.remoteEnv.remoteMGCLIBPAT HUseCurrent=SPECIFY.</p> <p><b>Run Control</b> page, Remote Calibre Environment section</p>	[] ( {}
cmn.remoteEnv.remoteMGCLIBPA TH_ixl.value	<p>The value of MGC_LIB_PATH to use for Calibre on an ixl remote host. Used when cmn.remoteEnv.remoteMGCLIBPAT HUseCurrent=SPECIFY.</p> <p><b>Run Control</b> page, Remote Calibre Environment section</p>	[] ( {}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.remoteEnv.remoteShellDefault.value	<p>Specifies which shell to use for launching Calibre on the remote host.</p> <ul style="list-style-type: none"> <li>• DEFAULT — Use the default platform-specific shell</li> <li>• SECURE — Use the secure shell. Password authentication is disabled and public key authentication is recommended.</li> <li>• SPECIFY — Specify the path to the shell executable.</li> </ul> <p><b>Run Control</b> page, Remote Calibre Environment section</p>	[DEFAULT SECURE SPECIFY] () {DEFAULT}
cmn.remoteEnv.remoteShellName.value	<p>The path to the shell to use for launching Calibre on the remote host. Used when cmn.remoteEnv.remoteShellDefault=SPECIFY.</p> <p><b>Run Control</b> page, Remote Calibre Environment section</p>	[] () {}
cmn.remoteEnv.remoteUserIsCurrentUser.value	<p>Specify what user name to use for the remote Calibre run.</p> <ul style="list-style-type: none"> <li>• CURRENT — Use the current user name</li> <li>• SPECIFY — Specify the user name</li> </ul> <p><b>Run Control</b> page, Remote Calibre Environment section</p>	[CURRENT SPECIFY] () {CURRENT}
cmn.remoteEnv.remoteUserName.value	<p>The user name for the remote Calibre run. Used when cmn.remoteEnv.remoteUserIsCurrentUser=SPECIFY.</p> <p><b>Run Control</b> page, Remote Calibre Environment section</p>	[] () {}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.rhdb.mode.value	Specifies whether to save or restore a reusable hierarchical database (RHDB) file.  <b>Outputs</b> page, Reusable Hierarchical Database section	[SAVE RESTORE] () {SAVE}
cmn.rulesPrefs.alwaysCompile.value	Boolean that controls the option “Always compile the rules file when Run button is pressed”.  <b>Preferences</b> page, Rules section	[true false] (Boolean) {false}
cmn.rulesPrefs.batchModeCompatibleControlFile.value	Boolean that controls the option “Write control file suitable for command line run”.  <b>Preferences</b> page, Rules section	[true false] (Boolean) {true}
cmn.rulesPrefs.checkForModifiedRules.value	Boolean that controls the option “Automatically load rules file”.  When enabled, the rule file is automatically loaded 1) when the runset is loaded, and 2) at runtime if the rules file has been changed.  <b>Preferences</b> page, Rules section	[true false] (Boolean) {false}
cmn.rulesPrefs.compileOnly.value	Boolean that controls the option “Only compile the rules file when Run button is pressed”  <b>Preferences</b> page, Rules section	[true false] (Boolean) {false}
cmn.rulesPrefs.controlFile.specified	Boolean controlling cmn.rulesPrefs.controlFile  <b>Preferences</b> page, Rules section	[false true] (Boolean) {false}
cmn.rulesPrefs.controlFile.value	The name of the Calibre Interactive control file.  This setting is not used if Calibre Interactive is invoked from a design tool and a template for the control file is specified.  <b>Preferences</b> page, Rules section	[] () {}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.rulesPrefs.readIOFromRules.value	Boolean that controls the option “Read Inputs/Outputs fields when loading rules file”. <b>Preferences</b> page, Rules section	[true false] (Boolean) {true}
cmn.rulesPrefs.rulesAfterMain.value	Boolean that controls the option “Include additional rules files and statements after main rules file”. <b>Preferences</b> page, Rules section	[true false] (Boolean) {false}
cmn.rulesPrefs.saveTvfToSvrf.value	Boolean that controls the option “Save TVF Rules to SVRF before running Calibre”. The generated SVRF is named <i>rules.svrf</i> , where <i>rules</i> is the name of the TVF rule file on the <b>Rules</b> page. This adds the -E option to the calibre command line. <b>Preferences</b> page, Rules section	[true false] (Boolean) {false}
cmn.rulesPrefs.showModifiedRulesPrompt.value	Boolean that controls the option “Show prompt”. Enable this to display a prompt before loading the rule file.  This option takes effect only if cmn.rulesPrefs.checkForModifiedRules.value is set to true. <b>Preferences</b> page, Rules section	[true false] (Boolean) {true}
cmn.rulesPrefs.transcriptIncludedFiles.value	Select this setting to display contents of included rules files in the transcript when Calibre is run.  Boolean that controls the option “Display text of included rules files in transcript”. <b>Preferences</b> page, Rules section	[true false] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.runsetPrefs.processLocalLinks.value	Boolean that controls whether links are created in the directory of the runset. <b>Preferences</b> page, Runset section	[true false] (Boolean) {true}
cmn.runsetPrefs.runOnOpen.value	Boolean that controls the option “Run Calibre when runset is opened”. <b>Preferences</b> page, Runset section	[true false] (Boolean) {false}
cmn.runsetPrefs.saveOnClose.value	Setting for the option “On Close Runset,” which determines behavior when the GUI is closed: <ul style="list-style-type: none"> <li>• PROMPT — Prompt to save the runset if it has changed.</li> <li>• AUTOSAVE — Automatically save the runset.</li> <li>• NOSAVE — Do not save the runset and do not prompt.</li> </ul> <b>Preferences</b> page, Runset section	[PROMPT AUTOSAVE NOSAVE] () {PROMPT}
cmn.runsetPrefs.saveOnRun.value	Boolean that controls the option “Save runset each time Calibre is run”. <b>Preferences</b> page, Runset section	[true false] (Boolean) {false}
cmn.runsetPrefs.saveRunsetToCurDirOnRun.value	Boolean that controls the option “Save runset to run directory each time Calibre is run”. <b>Preferences</b> page, Runset section.	[true false] (Boolean) {false}
cmn.rve.autoClose.value	Boolean that controls the option “Close RVE before running Calibre <app>”. <b>Run Control</b> page, RVE Options section	[true false] (Boolean) {true}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.rve.closeOnExit.value	Boolean that controls the option “Close RVE when Calibre Interactive closes”. <b>Run Control</b> page, RVE Options section	[true false] (Boolean) {true}
cmn.rve.runOnLocalHost.value	Boolean that controls the option “Run RVE on local host”. If set to false, RVE is run using the settings in the “Run Calibre On” option. <b>Run Control</b> page, RVE Options section	[true false] (Boolean) {true}
cmn.showOptionPages.value	The list of pages to show. The default is to show all optional pages. Default: [ “Database”, “Environment”, “Options”, “Preferences”, “Templates”, “Triggers” ] <b>Settings &gt; Show Pages</b> menu item	[] (list) { }
cmn.transcriptIssuesPrefs.specified	Checkbox for “Errors and Warnings” on the <b>Preferences</b> page. Enable this to automatically save the errors and warnings. Controls the cmn.transcriptIssuesPrefs group of options. <b>Preferences</b> page, Errors and Warnings section	[true false] (Boolean) {false}
cmn.transcriptIssuesPrefs.appendOrReplace.value	Selection for the “Mode” option. Specifies whether to replace or add to an existing errors and warnings file when cmn.transcriptIssuesPrefs.specified = true. <b>Preferences</b> page, Errors and Warnings section	[REPLACE APPEND] () {REPLACE}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.transcriptIssuesPrefs.errorEchoFile.value	The name of the saved errors and warnings file when cmn.transcriptIssuesPrefs.specified = true.  <b>Preferences</b> page, Errors and Warnings section, “File Path”	[] () {}
cmn.transcriptPrefs.specified	Checkbox for “Transcript” on the <b>Preferences</b> page. Enable this to automatically save the transcript. Controls the cmn.transcriptPrefs group of options.  <b>Preferences</b> page, Transcript section	[true false] (Boolean) {false}
cmn.transcriptPrefs.appendOrReplace.value	Selection for the “Mode” option. Specifies whether to replace or add an existing transcript file when cmn.transcriptPrefs.specified = true.  <b>Preferences</b> page, Transcript section	[REPLACE APPEND] () {REPLACE}
cmn.transcriptPrefs.transcriptEchoFile.value	The name of the saved transcript when cmn.transcriptPrefs.specified = true.  <b>Preferences</b> page, Transcript section, “File Path”	[] () {}
cmn.turboCommand.backupRDS.value (DRC, LVS, PERC)	Boolean that controls the “Backup Data” option for Calibre MTflex runs with “Remote Data Serve (RDS)” enabled. (-recoverremote)  <b>Run Control</b> page, Run Calibre section	[true false] (Boolean) {false}
cmn.turboCommand.hdbFlex.specified	Boolean that controls the “Connect to Remote Hosts” option for Calibre MTflex runs.  <b>Run Control</b> page, Run Calibre section	[false true] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.turboCommand.hdbFlex.value	Specifies whether to connect to remote stages at a later stage of or in parallel with HDB construction. <b>Run Control</b> page, Run Calibre section	[LATERHDB ACQUIREHDB ] ( {LATERHDB }
cmn.turboCommand.mTFlexHostFile.value	The path to remote host configuration file for Calibre MTflex runs. (-remotefile) <b>Run Control</b> page, Run Calibre section, “Remote Host File”	[] ( {}
cmn.turboCommand.numLithoProcessors.value (DRC)	Specifies the number of processors to use for RET and MDP applications in multithreaded (MT) mode. (-turbo_litho <num>) <b>Run Control</b> page, Run Calibre, “Number of CPUs to use for LITHO”	[] (integer) {2}
cmn.turboCommand.numLvsSupplement.specified (PERC)	Boolean controlling cmn.turboCommand.numLvsSupplement <b>Run Control</b> page, Run Calibre section, “Number of hierarchical LVS licenses for PERC” checkbox	[false true] (Boolean) {false}
cmn.turboCommand.numLvsSupplement.value (PERC)	Specifies the “-lvs_supplement <num>” command line option, which uses Calibre nmLVS-H licenses for the circuit extraction portion of a Calibre PERC run.  Available for multithreaded runs without -turbo_all. <b>Run Control</b> page, Run Calibre section, “Number of hierarchical LVS licenses for PERC”	[] (integer) {2}
cmn.turboCommand.numProcessors.value	The number of processors to use in multithreaded runs. (-turbo) <b>Run Control</b> page, Run Calibre section, “Number of CPUs to use”	[] (integer) {2}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.turboCommand.rdsCount.value (DRC, LVS, PERC)	The number of Remote Data Servers allocated to each remote machine for Calibre MTflex runs. (-remotedata <rdscount>)  Used when cmn.turboCommand.rdsCountHow.value = SPECIFYCOUNT.  <b>Run Control</b> page, Run Calibre section, “Count” under “Remote Data Server (RDS)”	[] (integer) {1}
cmn.turboCommand.rdsCountHow.s pecified (DRC, LVS, PERC)	Boolean controlling cmn.turboCommand.rdsCountHow Specifies whether to use Remote Data Servers. (-remotedata [<rdscount>])  <b>Run Control</b> page, Run Calibre section, “Remote Data Server (RDS)” checkbox	[false true] (Boolean) {false}
cmn.turboCommand.rdsCountHow.v alue (DRC, LVS, PERC)	Specifies how to determine the number of Remote Data Servers. <ul style="list-style-type: none"> <li>• DEFAULTCOUNT — Use the internally-determined default. (recommended)</li> <li>• SPECIFYCOUNT — Specify a count with cmn.turboCommand.rdsCount.value.</li> </ul> <b>Run Control</b> page, Run Calibre section, “Remote Data Server (RDS)” dropdown list	[DEFAULTCO UNT SPECIFYCOUN T] ( {DEFAULTCO UNT}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.turboCommand.remoteHostsHo w.value	Select how to specify the remote hosts for a Calibre MTflex run.  In Calibre Interactive xACT, fromFile is the only available selection. <ul style="list-style-type: none"> <li>• fromFile — Specify a remote host file with cmn.turboCommand.mTFlexHost File.value. (-remotefile)</li> <li>• fromGUI — Specify remote hosts in the GUI. See cmn.mtflexSlaveHosts.parameters</li> </ul> <b>Run Control page, Run Calibre section, “Remote Hosts”</b>	[fromFile fromGUI] () {fromGUI}
cmn.turboCommand.runHow.value	Specify the Calibre execution mode: <ul style="list-style-type: none"> <li>• single — Single-threaded (-turbo 1)</li> <li>• multi — Multithreaded (-turbo num)</li> <li>• all — Multithreaded with all available processors (-turbo)</li> <li>• mtflex — Calibre MTflex (-turbo -remotefile file)</li> </ul> <b>Run Control page, Run Calibre section, “Run Calibre &lt;app&gt; using”</b>	[single multi all mtflex] () {multi}
cmn.turboCommand.runHyper.value (DRC, LVS, PERC)	Specifies whether to use hyperscaling. (-hyper)  <b>Run Control page, Run Calibre section, “Hyperscale”</b>	[false true] (Boolean) {false}
cmn.turboCommand.runHyperComp are.value (LVS)	Specify whether to use hyperscaling during the comparison portion for Calibre nmLVS (-hyper cmp)  <b>Run Control page, Run Calibre section, “Hyperscale Compare”</b>	[false true] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.turboCommand.runHyperPathC hk.value (LVS)	Specify whether to use hyperscaling during the pathchk portion of ERC in Calibre nmLVS (-hyper pathchk) <b>Run Control</b> page, Run Calibre section, “Hyperscale PathChk”	[false true] (Boolean) {false}
cmn.turboCommand.runHyperSlave. value (DRC, LVS, PERC)	Enables remote pseudohierarchical database technology. (-hyper remote) Available when using Calibre MTflex with hyperscaling and Remote Data Server. <b>Run Control</b> page, Run Calibre section, “Hyperscale Remote”	[false true] (Boolean) {false}
cmn.turboCommand.syncCPUNum. value (not xACT)	Specifies whether to automatically update Min CPUs in Host Selection table by the Maximum number of CPUs specified. Available only for multithreaded runs on IBM Spectrum LSF. <b>Run Control</b> page, Run Calibre section, “Synchronize Min CPUs in Host Selection table”	[false true] (Boolean) {false}
cmn.turboCommand.turboAll.value	Boolean control for “Halt Calibre <app> if licenses cannot be obtained for number of CPUs specified”. (-turbo_all) <b>Run Control</b> page, Run Calibre section	[true false] (Boolean) {false}
cmn.units.overrideLayoutPrecision.v alue	Checkbox control for “Override layout value”. The control is active if “Precision” is specified. If not enabled, a compiler error results if the database precision does not match the rule file precision. <b>Database</b> page, Units section	[true false] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.units.precision.specified	Boolean controlling cmn.units.precision <b>Database</b> page, Units section	[false true] (Boolean) {false}
cmn.units.precision.value	Specifies the precision, which is the ratio of database units to user units. This setting provides the value for the <b>Precision</b> statement. It can be a single integer value, or a list of two integers. <b>Database</b> page, Units section	[] () {1000}
cmn.units.resolution.specified	Boolean controlling cmn.units.resolution <b>Database</b> page, Units section	[false true] (Boolean) {false}
cmn.units.resolution.value	Specifies the layout (or user) grid step size in database units. This setting provides the value for the <b>Resolution</b> statement. It can be a single integer value, or a list of two integers to provide the step size in the x and y directions separately. <b>Database</b> page, Units section	[] () {1}
cmn.virtualConnect.colon.specified	Boolean controlling cmn.virtualConnect.colon <b>Options</b> page, Virtual Connect section	[false true] (Boolean) {false}
cmn.virtualConnect.colon.value	Value for the “Virtual Connect Colon” option. Specifies whether to virtually connect physically disjoint nets whose names agree up to the first colon character found in the names. <b>Options</b> page, Virtual Connect section	[NO YES] () {NO}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.virtualConnect.netNames.value	The list of net names to connect by name for the Virtual Connect Name statement. The wildcard character “?” matches zero or more characters.  This option is used when cmn.virtualConnect.vconnect.value=“SPECIFIEDNAMES” (the “Virtual Connect Name” option).  <b>Options</b> page, Virtual Connect section	[] (list) { }
cmn.virtualConnect.report.specified	Boolean controlling cmn.virtualConnect.report  <b>Options</b> page, Virtual Connect section	[false true] (Boolean) {false}
cmn.virtualConnect.report.value	The setting for the “Virtual Connect Report” option, which controls warnings produced when a virtual connection is made. <ul style="list-style-type: none"> <li>• NO — Do not report connections.</li> <li>• YES — Report connections.</li> <li>• YES UNSATISFIED — Only report virtual connections that are not physically connected higher in the hierarchy.</li> </ul> <b>Options</b> page, Virtual Connect section	[“NO” “YES” “YES UNSATISFIED”] () {NO}
cmn.virtualConnect.reportMaximum .specified	Boolean controlling cmn.virtualConnect.reportMaximum  <b>Options</b> page, Virtual Connect section	[false true] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.virtualConnect.reportMaximum.value	<p>Specifies whether to limit the maximum number of virtual connect warnings issued when virtual connect reporting is enabled.</p> <ul style="list-style-type: none"> <li>• MAXIMUMALL — Report all</li> <li>• MAXIMUMCOUNT — Limit reporting to the number specified by cmn.virtualConnect.reportMaximumCount.value.</li> </ul> <p><b>Options</b> page, Virtual Connect section</p>	[MAXIMUMALL MAXIMUMCOUNT] ( {MAXIMUMALL})
cmn.virtualConnect.reportMaximumCount.value	<p>Specifies the upper limit on the number of virtual connect messages. Used when cmn.virtualConnect.reportMaximum = MAXIMUMCOUNT.</p> <p><b>Options</b> page, Virtual Connect section</p>	[] (positive integer) {0}
cmn.virtualConnect.vconnect.specified	<p>Boolean controlling cmn.virtualConnect.vconnect. This option controls the “Virtual Connect Name” checkbox, which determines whether to use the statement <a href="#">Virtual Connect Name</a>.</p> <ul style="list-style-type: none"> <li>• false — Do not use. Disjoint nets with the same name are treated as separate nets.</li> <li>• true — Use. Disjoint nets are connected by name with the setting specified by cmn.virtualConnect.vconnect.value.</li> </ul> <p><b>Options</b> page, Virtual Connect section</p>	[false true] (Boolean) {false}

**Table C-1. General Runset Options for Calibre Interactive (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.virtualConnect.vconnect.value	<p>Specifies how to form virtual connections of named nets.</p> <ul style="list-style-type: none"> <li>• ALLNAMES — Virtually connect disjoint nets with identical names.</li> <li>• SPECIFIEDNAMES — Virtually connect disjoint nets that are in a specified list. The list is defined by cmn.virtualConnect.netNames.value.</li> </ul> <p><b>Options</b> page, Virtual Connect section</p>	<p>[ALLNAMES SPECIFIEDNAMES] ( {ALLNAMES})</p>

## DRC Runset Options

DRC Runset Options.

**Table C-2. DRC Runset Options**

Name	Description	[Available Values] (Type restrictions) {default}
drc.analyzeMode.value	<p>Adds the -analyze option to the command line, to get error distribution results data for analysis of areas of a design that require attention.</p> <p><b>Inputs</b> page, “Analyze”</p>	<p>[false true] (Boolean) {false}</p>
drc.areaDrc.applyScaling.value	<p>Boolean controlling whether area DRC coordinates are scaled to take LAYOUT MAGNIFY and DRC RESULTS DATABASE PRECISION settings into account.</p> <p><b>Inputs</b> page, Area DRC section, “Apply coordinate scaling”</p>	<p>[true false] (Boolean) {true}</p>

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.areaDrc.autoHalo.value	<p>Choose whether the Area DRC halo is calculated automatically or set by the user.</p> <ul style="list-style-type: none"> <li>• <b>AUTO</b> — The halo size is the larger of the width or length of the area's extent.</li> <li>• <b>EXPLICIT</b> — The halo size is set with drc.areaDrc.haloWidth.value.</li> </ul> <p><b>Inputs</b> page, Area DRC section, “Automatic Halo”</p>	[AUTO EXPLICIT] ( {AUTO})
drc.areaDrc.clipToArea.value	<p>Boolean that specifies whether to remove results from the halo region after running DRC (for ASCII format results database and Side RDBs only).</p> <p><b>Inputs</b> page, Area DRC section, “Remove results from halo region after Area DRC”</p>	[true false] (Boolean) {true}
drc.areaDrc.haloWidth.value	<p>The Area DRC halo size in user-units. Used when drc.areaDrc.autoHalo.value = EXPLICIT.</p> <p><b>Inputs</b> page, Area DRC section, “Halo Size”</p>	[] ( {0})
drc.areaDrc.layoutWindow.value	<p>The coordinates of the window to check for Area DRC. Specify two user-unit coordinate pairs for the lower left and top right corners of the window.</p> <p><b>Inputs</b> page, Area DRC section, “Layout Window”</p>	[] ( {})
drc.areaDrc.specified	<p>Boolean controlling drc.areaDrc</p> <p><b>Inputs</b> page, Area DRC checkbox</p>	[false true] (Boolean) {false}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.configPrefs.configFiles.value	The file names of the Configuration Files.  <b>Preferences</b> page, Configuration section, “Configuration Files”	[] ( {}
drc.configPrefs.specified	Boolean controlling drc.configPrefs  <b>Preferences</b> page, Configuration checkbox	[false true] (Boolean) {false}
drc.createFXRuleFile.value	Checkbox that specifies whether to run DBdiff to create the XOR rules file before running Calibre Fast XOR.  This option must be set to true to enable the drc.fxFormatOptions and drc.fxMappingFiles options.  <b>Fast XOR</b> page, “Create XOR rules file”	[false true] (Boolean) {false}
drc.ctoFile.generateCategories.specified	Boolean controlling drc.ctoFile.generateCategories  <b>Outputs</b> page, “Create CTO File” section, “Generate Categories from” checkbox	[false true] (Boolean) {false}
drc.ctoFile.generateCategories.value	Specifies categories to include in the generated CTO file.  <b>Outputs</b> page, “Create CTO File” section, “Generate Categories from” dropdown list	[Groups Layers Recipes] ( {Groups}
drc.ctoFile.layerMapFile.specified	Boolean controlling drc.ctoFile.layerMapFile  <b>Outputs</b> page, “Create CTO File” “Create CTO File” section, Viewer Layer Map File checkbox	[false true] (Boolean) {false}
drc.ctoFile.layerMapFile.value	The Viewer Layer Map File to use during CTO file creation.  <b>Outputs</b> page, “Create CTO File” section, Viewer Layer Map File text entry	[] ( {}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.ctoFile.specified	Boolean controlling drc.ctoFile <b>Outputs</b> page, “Create CTO File” checkbox	[false true] (Boolean) {false}
drc.ctoFile.warnOverwrite.value	Display a warning before overwriting the CTO file.  <b>Outputs</b> page, “Create CTO File” section, “Warn before overwriting CTO file”	[false true] (Boolean) {false}
drc.dfmDefaults.resultsCellPrefix.specified	Boolean controlling drc.dfmDefaults.resultsCellPrefix. Adds the PREFIX keyword.  <b>Database</b> page, DFM Defaults RDB area, “Cell prefix” checkbox	[false true] (Boolean) {false}
drc.dfmDefaults.resultsCellPrefix.value	A string giving the cell prefix for the PREFIX keyword in DFM Defaults RDB.  Used when drc.dfmDefaults.resultsCellPrefix.specified = true.  <b>Database</b> page, DFM Defaults RDB area, Cell prefix text box	[] () {}
drc.dfmDefaults.resultsCellSuffix.specified	Boolean controlling drc.dfmDefaults.resultsCellSuffix. Adds the APPEND keyword.  <b>Database</b> page, DFM Defaults RDB area, Cell suffix checkbox	[false true] (Boolean) {false}
drc.dfmDefaults.resultsCellSuffix.value	A string giving the cell suffix for the APPEND keyword in DFM Defaults RDB.  Used when drc.dfmDefaults.resultsCellSuffix.specified = true.  <b>Database</b> page, DFM Defaults RDB area, Cell suffix text box	[] () {}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.dfmDefaults.resultsFile.value	The default filename for DFM RDB {GDS   OASIS} commands. This filename is used with the FILE keyword in DFM Defaults RDB {GDS   OASIS}.  <b>Database</b> page, DFM Defaults RDB area, File text box	[] () {}
drc.dfmDefaults.resultsFormat.value	The format of the DFM RDB that the DFM Defaults statements apply to.  <b>Database</b> page, DFM Defaults RDB area, Format	[GDS OASIS] () {GDS}
drc.dfmDefaults.resultsPseudoCells.specified	Boolean controlling drc.dfmDefaults.resultsPseudoCells. The default behavior is the USERMERGED keyword.  <b>Database</b> page, DFM Defaults RDB area, “Pseudocells results” checkbox	[false true] (Boolean) {false}
drc.dfmDefaults.resultsPseudoCells.value	Specifies to use the TOP keyword in the DFM Defaults statement.  Used when drc.dfmDefaults.resultsPseudoCells.specified = true.  <b>Database</b> page, DFM Defaults RDB area, “Pseudocells results” dropdown list	[TOP] () {TOP}
drc.dfmDefaults.specified	Boolean controlling drc.dfmDefaults. Set to true to use DFM Defaults RDB statements from the GUI.  <b>Database</b> page, DFM Defaults RDB checkbox	[false true] (Boolean) {false}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.drcMaximumResults.count.value	<p>Specifies the maximum result count for an individual rule check in the DRC MAXIMUM RESULTS statement.</p> <p>Used when drc.drcMaximumResults.upperLimit.value = MAXIMUMCOUNT.</p> <p><b>Options</b> page, DRC Maximum Results, “Maximum Result Count”</p>	[ ] (positive integer) {1000}
drc.drcMaximumResults.specified	<p>Boolean controlling drc.drcMaximumResults</p> <p><b>Options</b> page, DRC Maximum Results checkbox</p>	[true false] (Boolean) {true}
drc.drcMaximumResults.upperLimit.value	<p>Specifies whether to use the ALL keyword or an integer count in the DRC MAXIMUM RESULTS statement.</p> <ul style="list-style-type: none"> <li>• <b>MAXIMUMCOUNT</b> — Specify a maximum result count with drc.drcMaximumResults.count.value.</li> <li>• <b>MAXIMUMALL</b> — Use the ALL keyword.</li> </ul> <p><b>Options</b> page, DRC Maximum Results, “Upper Limit”</p>	[MAXIMUMCOUNT MAXIMUMALL] ( ) {MAXIMUMCOUNT}
drc.drcMaximumVertex.count.value	<p>Specifies an upper limit for the vertex count of any result polygon. The value must be an integer greater than or equal to 4.</p> <p>Used when drc.drcMaximumVertex.upperLimit.value = MAXIMUMCOUNT.</p> <p><b>Options</b> page, DRC Maximum Vertex, Maximum Vertex Count</p>	[ ] (integer) {4096}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.drcMaximumVertex.specified	Boolean controlling drc.drcMaximumVertex. <b>Options</b> page, DRC Maximum Vertex checkbox	[true false] (Boolean) {true}
drc.drcMaximumVertex.upperLimit.value	Specifies how the DRC MAXIMUM VERTEX command is used: <ul style="list-style-type: none"> <li>• <b>MAXIMUMCOUNT</b> — Provide a vertex count with drc.drcMaximumVertex.count.value.</li> <li>• <b>MAXIMUMALL</b> — Use the ALL keyword.</li> </ul> <b>Options</b> page, DRC Maximum Vertex, Upper Limit dropdown list	[MAXIMUMC OUNT MAXIMUMAL L] ( {MAXIMUMC OUNT}
drc.drcSideRdbs.parameters	A list of lists giving the parameters in the Side RDB table. The list has this format: [ [ use, type, filename] ... ] <ul style="list-style-type: none"> <li>• <i>use</i> — true or false, indicating whether the RDB is opened.</li> <li>• <i>type</i> — The type of RDB</li> <li>• <i>filename</i> — The filename.</li> </ul> <b>Outputs</b> page, “Side RDBs” checkbox	[] (list) { }
drc.drcSideRdbs.specified	Boolean controlling drc.drcSideRdbs. When “Side RDBs” is checked, the information in the Side RDB table determines which side RDBs are opened by Calibre RVE. <b>Outputs</b> page, “Side RDBs” checkbox	[false true] (Boolean) {false}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.drcdb.resultsCellName.value	<p>Specifies whether to output hierarchical DRC errors in their context cells or the top cell.</p> <ul style="list-style-type: none"> <li>• true — Use “DRC CELL NAME YES CELL SPACE XFORM ALL”</li> <li>• false — Use “DRC CELL NAME NO”</li> </ul> <p><b>Outputs</b> page, DRC Results Database area, “Output cell errors in cell space”</p>	[true false] (Boolean) {true}
drc.drcdb.resultsCellPrefix.specified	<p>Boolean controlling drc.drcdb.resultsCellPrefix</p> <p><b>Outputs</b> page, DRC Results Database area, “Cell prefix” checkbox</p>	[false true] (Boolean) {false}
drc.drcdb.resultsCellPrefix.value	<p>The string to use as a prefix for cell names. Used only when the DRC results database is GDSII or OASIS.</p> <p>PREFIX keyword for DRC Results Database.</p> <p><b>Outputs</b> page, DRC Results Database area, “Cell prefix” text box</p>	[] () {}
drc.drcdb.resultsCellSuffix.specified	<p>Boolean controlling drc.drcdb.resultsCellSuffix</p> <p><b>Outputs</b> page, DRC Results Database area, “Cell suffix” checkbox</p>	[false true] (Boolean) {false}
drc.drcdb.resultsCellSuffix.value	<p>The string to use as a suffix for cell names. Used only when the DRC results database is GDSII or OASIS.</p> <p>APPEND keyword for DRC Results Database.</p> <p><b>Outputs</b> page, DRC Results Database area, “Cell suffix” text box</p>	[] () {}
drc.drcdb.resultsCheckText.specified	<p>Boolean controlling drc.drcdb.resultsCheckText</p> <p><b>Outputs</b> page, DRC Results Database area, “DRC Check Text” check box</p>	[false true] (Boolean) {false}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.drcdb.resultsCheckText.value	Specifies the amount of rule check text to appear in the DRC results database. The options are keywords or keyword combinations to the DRC CHECK TEXT statement.  <b>Outputs</b> page, DRC Results Database area, “DRC Check Text” dropdown list	[“COMMENTS RFI” “COMMENTS” “ALL” “ALL RFI” “NONE”] ( {COMMENTS RFI}
drc.drcdb.resultsCheckTextFile.specified	Boolean controlling drc.drcdb.resultsCheckTextFile  Specifies whether to use the optional check text file output with the DRC Check Text statement.  <b>Outputs</b> page, DRC Results Database area, “Check Text File” check box	[false true] (Boolean) {false}
drc.drcdb.resultsCheckTextFile.value	The name of the check text file for the DRC Check Text statement.  <b>Outputs</b> page, DRC Results Database area, “Check Text File” text box	[] ( {}
drc.drcdb.resultsFile.value	The name of the DRC Results Database.  <b>Outputs</b> page, DRC Results Database area, “File” text box	[] ( {drc.results}
drc.drcdb.resultsFormat.value	The format of the DRC Results Database.  <b>Outputs</b> page, DRC Results Database area, Format dropdown list	[ASCII GDSII OASIS] ( {ASCII}
drc.drcdb.resultsOasis.specified	Boolean controlling drc.drcdb.resultsOasis  <b>Outputs</b> page, DRC Results Database area, “Compression” checkbox	[false true] (Boolean) {false}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.drcdb.resultsOasis.value	Specifies one or more keywords for the DRC Results Database statement, to control output in OASIS format. <b>Outputs</b> page, DRC Results Database area, “Compression” dropdown list	[CBLOCK STRICT AUTOMAP INDEX NOVIEW] ( {}
drc.drcdb.resultsPseudoCells.specified	Boolean controlling drc.drcdb.resultsPseudoCells <b>Outputs</b> page, DRC Results Database area, “Pseudocells results” checkbox	[false true] (Boolean) {false}
drc.drcdb.resultsPseudoCells.value	Specifies keywords for the handling of pseudocells in the DRC Results Database statement. <b>Outputs</b> page, DRC Results Database area, “Pseudocells results” dropdown list	["PSEUDO" "USER" "USER MERGE" "D" "COMBINE" "TOP"] ( {PSEUDO}
drc.envVars.setEnv(Name, Value)	Specifies an environment variable that is set in the runset. The format is: drc.envVars.setEnv("name", "value") See the <a href="#">envVars</a> configuration file command. <b>Environment</b> page	[] ( {}
drc.envVars.setEnvNamePattern(Pattern)	Specifies a filter for the environment variables displayed on the <b>Environment</b> page. The format is: drc.envVars.setEnvNamePattern("pattern")	[] ( {*}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.envVars.unsetEnv(Name)	<p>Specifies an environment variable that is unset in the runset. The format is: perc.envVars.unsetEnv("name")</p> <p>See the envVars configuration file command.</p> <p><b>Environment</b> page.</p>	[ () {}
drc.fxFormatOptions.addNewAndMissingShapesRules.value	<p>When true, adds checks to identify new and missing shapes in each layer using a pair of NOT operations.</p> <p>Adds the -enhanced option to the rule file generation command.</p> <p>Used when drc.createFXRuleFile.value = true.</p> <p><b>Fast XOR</b> page, Format Options, “Add New and Missing shapes rules”</p>	[false true] (Boolean) {false}
drc.fxFormatOptions.commonOnly.value	<p>When true, generates XOR rules only for layers common to both designs.</p> <p>Adds the -common_only option to the rule file generation command.</p> <p>Used when drc.createFXRuleFile.value = true.</p> <p><b>Fast XOR</b> page, Format Options, “Common Only”</p>	[false true] (Boolean) {false}
drc.fxFormatOptions.outputFilesPrefix.value	<p>The prefix for the name of summary file, results database file, and check map file in the generated rule file.</p> <p>The prefix option for the rule file generation command.</p> <p>Used when drc.createFXRuleFile.value = true.</p> <p><b>Fast XOR</b> page, Format Options, “Output Files Prefix”</p>	[ () {rules.fxor}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.fxFormatOptions.resultFormat.value	<p>Specifies one or more formats for the output results database.</p> <p>The -resultformat option for the rule file generation command.</p> <p>Used when drc.createFXRuleFile.value = true.</p> <p><b>Fast XOR</b> page, Format Options, “Result Format”</p>	[ASCII GDS OASIS] ( { }
drc.fxFormatOptions.rulesFileName.value	<p>The name for the generated rules file.</p> <p>Only used when cmn.disablePrefs.disableFXRuleCustomName.value = false.</p> <p>Used when drc.createFXRuleFile.value = true.</p> <p><b>Fast XOR</b> page, Format Options, “Rules File”</p>	[] ( {rules.fxor}
drc.fxMappingFiles.baseLayers.specified	<p>Boolean controlling drc.fxMappingFiles.baseLayers</p> <p>Used when drc.createFXRuleFile.value = true.</p> <p><b>Fast XOR</b> page, Mapping Files, “Use Base Layers” checkbox</p>	[false true] (Boolean) {false}
drc.fxMappingFiles.baseLayers.value	<p>The base layers for the -base_layers option for the FastXOR rule file generation command.</p> <p>Used when drc.createFXRuleFile.value = true.</p> <p><b>Fast XOR</b> page, Mapping Files, “Use Base Layers” text entry</p>	[] ( { }
drc.fxMappingFiles.excludeLayersFile.specified	<p>Boolean controlling drc.fxMappingFiles.excludeLayersFile</p> <p>Used when drc.createFXRuleFile.value = true.</p> <p><b>Fast XOR</b> page, Mapping Files, “Use Exclude Layers File” checkbox</p>	[false true] (Boolean) {false}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.fxMappingFiles.excludeLayersFile.value	The filename for the -exclude_layer option for the FastXOR rule file generation command.  Used when drc.createFXRuleFile.value = true.  <b>Fast XOR</b> page, Mapping Files, “Use Exclude Layers File” text entry	[] ( {}
drc.fxMappingFiles.includeLayersFile.specified	Boolean controlling drc.fxMappingFiles.includeLayersFile  Used when drc.createFXRuleFile.value = true.  <b>Fast XOR</b> page, Mapping Files, “Use Include Layers File” checkbox	[false true] (Boolean) {false}
drc.fxMappingFiles.includeLayersFile.value	The filename for the -include_layer option for the FastXOR rule file generation command.  Used when drc.createFXRuleFile.value = true.  <b>Fast XOR</b> page, Mapping Files, “Use Include Layers File” text entry	[] ( {}
drc.fxMappingFiles.includeRulesFile.specified	Boolean controlling drc.fxMappingFiles.includeRulesFile  Used when drc.createFXRuleFile.value = true.  <b>Fast XOR</b> page, Mapping Files, “Use Include Rules File” checkbox	[false true] (Boolean) {false}
drc.fxMappingFiles.includeRulesFile.value	The rule file for the -include_rules option for the FastXOR rule file generation command.  Used when drc.createFXRuleFile.value = true.  <b>Fast XOR</b> page, Mapping Files, “Use Include Rules File” text entry	[] ( {}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.fxMappingFiles.layerMapFile.sp ecified	Boolean controlling drc.fxMappingFiles.layerMapFile Used when drc.createFXRuleFile.value = true. <b>Fast XOR</b> page, Mapping Files, “Use Layer Map File” checkbox	[false true] (Boolean) {false}
drc.fxMappingFiles.layerMapFile.v alue	The layer mapping file for the -layermap option for the FastXOR rule file generation command.  Used when drc.createFXRuleFile.value = true. <b>Fast XOR</b> page, Mapping Files, “Use Layer Map File” text entry	[] ( {}
drc.fxMaxDiffCount.value	Optional positive integer specifying the maximum number of design differences before switching to traditional XOR. Adds the -maxdiff argument for DBdiff execution with FastXOR.  Only used when cmn.disablePrefs.disableFXMaxDiffP ref.value = false. <b>Fast XOR</b> page, “Max Diff”	[] (integer) {}
drc.fxOptions.compareAllPlacedCel ls.value	Checkbox that specifies whether DBdiff compares all placed cells in a FastXOR run. (-compareallplacedcells)  Only used when cmn.disablePrefs.disableFXOptions.v alue = false. <b>Fast XOR</b> page, Options, “Compare All Placed Cells” checkbox	[false true] (Boolean) {false}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.fxOptions.compareText.value	Checkbox that specifies whether DBdiff does text comparison in a FastXOR run. (-comparetext)  Only used when cmn.disablePrefs.disableFXOptions.value = false.  <b>Fast XOR</b> page, Options, “Compare Text” checkbox	[false true] (Boolean) {false}
drc.fxOptions.enablePassThroughLayers.value	Checkbox to enable pass-through layers for DBdiff in a FastXOR run. (Environment variable CALIBRE_FX_PASS_THROUGH_LAYER_ENABLE)  Only used when cmn.disablePrefs.disableFXOptions.value = false.  <b>Fast XOR</b> page, Options, “Enable Pass-Through Layers” checkbox	[false true] (Boolean) {false}
drc.fxOptions.filterSettings.value	Specifies filter settings for the Dbdiff RDB output file during a FastXOR run. Adds one or more selected options to the -rdb option.  Only used when cmn.disablePrefs.disableFXOptions.value = false.  <b>Fast XOR</b> page, Options, “RDB File Filter Settings” dropdown list	[FC FP FT -sortlayer] ( {}
drc.fxOptions.rdbFile.value	The name of the DBdiff RDB file for a FastXOR run. (-rdb)  Only used when cmn.disablePrefs.disableFXOptions.value = false.  <b>Fast XOR</b> page, Options, “RDB File”	[] ( {}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.fxOptions.reportFile.value	The name of the DBdiff report file for a FastXOR run. (-report)  Only used when cmn.disablePrefs.disableFXOptions.value = false.  <b>Fast XOR</b> page, Options, “Report File”	[] ( {}
drc.fxOptions.templateFile.value	The name of DBdiff template file for a Fast XOR run. (-template)  Only used when cmn.disablePrefs.disableFXOptions.value = false.  <b>Fast XOR</b> page, Options, “Template File”	[] ( {}
drc.htmlReport.configFile.value	The custom DRC HTML report configuration file. Used when drc.htmlReport.configTemplate.value = “Custom Template”.  <b>Outputs</b> page, Create HTML Report, “Custom Template”	[] ( {}
drc.htmlReport.configTemplate.value	Specifies a built-in template or a custom file for the DRC HTML report configuration file. The built-in templates specify how the results tree is grouped, where SC indicates the #SC shape class calculated property.  <b>Outputs</b> page, Create HTML Report, “Template”	[“Cell-Check” “Check-Cell-SC” “Check-Cell” “Check-SC” “Custom Template”] ( {Check-Cell-SC}
drc.htmlReport.reportDirectory.value	Path of the directory where the DRC HTML report is created.  <b>Outputs</b> page, Create HTML Report, “Output Directory”	[] ( {report}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.htmlReport.reportFormat.value	The format of the generated DRC HTML Report. (Not used when drc.htmlReport.configTemplate.value = “Custom Template”) <b>Outputs</b> page, Create HTML Report, “Format”	[HTML MHTML] () {HTML}
drc.htmlReport.specified	Checkbox for creating the DRC HTML report; enables drc.htmlReport options. <b>Outputs</b> page, “Create HTML Report” checkbox	[false true] (Boolean) {false}
drc.htmlReport.viewReport.value	Specifies whether to display the DRC HTML report after it is created. <b>Outputs</b> page, Create HTML Report, “View Report”	[true false] (Boolean) {true}
drc.includes.ruleFiles.value	The file names of the additional rules files. <b>Options</b> page, Include Rules Files, “Additional Rules Files”	[] () {}
drc.includes.specified	Boolean controlling drc.includes <b>Options</b> page, Include Rules Files checkbox	[false true] (Boolean) {false}
drc.layout.exportFromLayoutViewer.value	Boolean that controls the “Export from layout viewer” checkbox. <b>Inputs</b> page, Layout Path section	[] () {false}
drc.layout.extraGoldenLayoutFiles.specified	Boolean controlling drc.layout.extraGoldenLayoutFiles <b>Database</b> page, Library section	[false true] (Boolean) {false}
drc.layout.extraGoldenLayoutFiles.value	The file names of the additional golden layout files. <b>Database</b> page, Library section	[] () {}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.layout.extraLayoutFiles.specifie d	Boolean controlling drc.layout.extraLayoutFiles <b>Database</b> page, Library section	[false true] (Boolean) {false}
drc.layout.extraLayoutFiles.value	The file names of the additional layout files. <b>Database</b> page, Library section	[] () {}
drc.layout.format.value	The format of the layout database. <b>Inputs</b> page, Layout Path, “Layout Format”   <b>Note:</b> If the layout format is OPENACCESS, see cmn.layout.layoutLibraryPathFDI.value, cmn.layout.topCellLibraryFDI.value, and cmn.layout.topCellViewFDI.value. If the layout format is LEFDEF, see cmn.layout.lefFiles.value, cmn.layout.defFiles.value, and similarly named options.	[GDSII OASIS LEFDEF OPENACCESS] () {GDSII}
drc.layout.layoutFile.value	The path to the layout file for GDS or OASIS format layouts. <b>Inputs</b> page, Layout Path, “Layout File”	[] (environment variable valid) {}
drc.layout.topCell.value	The name of the layout top cell. Used for GDS, OASIS, and OPENACCESS input. <b>Inputs</b> page, Layout Path, “Top Cell”	[] () {}
drc.layout.topCellLibrary.value	The library name of the layout top cell to run Calibre on. Used only when the GUI is invoked from Cadence Virtuoso and “Export from layout viewer” is checked. <b>Inputs</b> page, Layout Path section	[] () {}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.layout.topCellView.value	The cell view name of the layout top cell to run Calibre on. Used only when the GUI is invoked from Cadence Virtuoso and “Export from layout viewer” is checked.  <b>Inputs</b> page, Layout Path section	[] ( {}
drc.layout2.defDirs.value	One or more directories containing DEF files.  Used for the second database in a FastXOR run, when drc.layout2.format.value = LEFDEF.  <b>Inputs</b> page, second Layout Path section, “DEF Directories”	[] ( {}
drc.layout2.defFiles.value	One or more DEF files.  Used for the second database in a FastXOR run, when drc.layout2.format.value = LEFDEF.  <b>Inputs</b> page, second Layout Path section, “DEF Files”	[] ( {}
drc.layout2.exportFromLayoutViewer.value	Boolean that controls the “Export from layout viewer” checkbox for the second database in a FastXOR run.  <b>Inputs</b> page, second Layout Path section	[] ( {false}
drc.layout2.format.value	The format of the second layout database in a FastXOR run.  <b>Inputs</b> page, second Layout Path section, “Layout Format”	[GDSII OASIS LEFDEF OPENACCESS] ( {GDSII}
drc.layout2.layoutFile.value	The name of the second layout file in a FastXOR run, for GDS or OASIS format layouts.  <b>Inputs</b> page, second Layout Path section, “Layout File2”	[] (environment variable valid) {}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.layout2.layoutLibraryPathFDI.value	The name of the OPENACCESS library definitions file for the second database in a FastXOR run. This could be the path to either the <i>cds.lib</i> or <i>lib.defs</i> library definitions file.  Used only when the layout format is OPENACCESS.  <b>Inputs</b> page, second Layout Path section, “Library Defs”	[] ( {}
drc.layout2.lefDirs.value	Specifies one or more directories containing LEF files.  Used for the second database in a FastXOR run, when drc.layout2.format.value = LEFDEF.  <b>Inputs</b> page, second Layout Path section, “DEF Directories”	[] ( {}
drc.layout2.lefFiles.value	Specifies one or more LEF files.  Used for the second database in a FastXOR run, when drc.layout2.format.value = LEFDEF.  <b>Inputs</b> page, second Layout Path section, “LEF Files”	[] ( {}
drc.layout2.lefTechFiles.value	Specifies one or more LEF technology files.  Used for the second database in a FastXOR run, when drc.layout2.format.value = LEFDEF.  <b>Inputs</b> page, second Layout Path section, “LEF Technology Files”	[] ( {}
drc.layout2.topCell.value	The name of the layout top cell for the second database in a FastXOR run. Used for GDS, OASIS, and OPENACCESS layout format.  <b>Inputs</b> page, second Layout Path section	[] ( {}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.layout2.topCellLibrary.value	The library name of the layout top cell for the second database in a FastXOR run. Used only when the GUI is invoked from Cadence Virtuoso and “Export from layout viewer” is checked.  <b>Inputs</b> page, second Layout Path section	[] ( {}
drc.layout2.topCellLibraryFDI.value	The name of layout library for the second database in a FastXOR run. Used only for OpenAccess databases.  <b>Inputs</b> page, second Layout Path section	[] ( {}
drc.layout2.topCellView.value	The view name for the second database in a FastXOR run. Used only when the GUI is invoked from Cadence Virtuoso and “Export from layout viewer” is checked.  <b>Inputs</b> page, second Layout Path section	[] ( {}
drc.layout2.topCellViewFDI.value	The view name for the second database in a FastXOR run. Used only for OpenAccess databases.  <b>Inputs</b> page, second Layout Path section	[] ( {}
drc.preserveCells.cellsList.specified	Boolean controlling drc.preserveCells.cellsList  <b>Options</b> page, Preserve Cells area, “Preserve cells from list” checkbox	[false true] (Boolean) {false}
drc.preserveCells.cellsList.value	A list of cells to preserve during the DRC run. Provide cells as a space-separated list.  Adds Layout Cell List and Layout Preserve Cell List statements to the control file.  <b>Options</b> page, Preserve Cells area, “Preserve cells from list”	[] ( {}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.preserveCells.waiverFiles.specified	Boolean controlling drc.preserveCells.waiverFiles <b>Options</b> page, Preserve Cells area, “Preserve cells from RVE waiver file(s)” checkbox	[false true] (Boolean) {false}
drc.preserveCells.waiverFiles.value	One or more .waived files used by Calibre RVE for DRC. All cells that have waived results are preserved from possible expansion into the parent cell during the Calibre run.  Adds Layout Cell List and Layout Preserve Cell List statements to the control file.  <b>Options</b> page, Preserve Cells area, “Preserve cells from RVE waiver file(s)”	[] (0) {}
drc.recipeEditor.setActiveRecipe(“ <i>recipe</i> ”)	Sets the active check recipe. The default recipe is “Checks selected in the rules file”. The function format is:  drc.recipeEditor.setActiveRecipe(“ <i>recipe</i> ”)  where <i>recipe</i> is the name of a built-in or user-defined recipe.  <b>Rules</b> page, Check Selection area, “Recipe” dropdown	[] (0) {}
drc.recipeEditor.setUserRecipes( <i>recipe_specification</i> )	Defines user check recipes using an internal specification format.	[] (0) {}
drc.recipePrefs.userRecipeFiles.specified	Boolean controlling drc.recipePrefs.userRecipeFiles <b>Preferences</b> page, Recipes, “User Recipe Files” checkbox	[false true] (Boolean) {false}
drc.recipePrefs.userRecipeFiles.value	A list of user check recipe files. Recipe files have a .rcp file extension.  <b>Preferences</b> page, Recipes, “User Recipe Files” entry	[] (0) {}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.reconMode.value	Specifies Calibre nmDRC Reconnaissance (Recon) mode: 0 — Do not use Calibre nmDRC Recon (default). 1 — Use Calibre nmDRC Recon (-recon command line option). 2 — Use Calibre nmDRC Recon inverse (“-recon inverse” command line option). <b>Inputs</b> page	[ 0 1 2 ] (integer) {0}
drc.reports.drcSummaryReport.specified	Boolean controlling drc.reports.drcSummaryReport	[true false] (Boolean) {true}
drc.reports.drcSummaryReport.value	The filename for the DRC SUMMARY REPORT statement. <b>Outputs</b> page, Reports section	[] () {drc.summary}
drc.reports.drcSummaryReportAccess.value	Specifies whether to overwrite or append to an existing DRC Summary Report file. <b>Outputs</b> page, Reports section	[REPLACE APPEND] () {REPLACE}
drc.reports.drcSummaryReportKeywords.specified	Boolean controlling drc.reports.drcSummaryReportKeywords	[true false] (Boolean) {true}
drc.reports.drcSummaryReportKeywords.value	Provides additional keywords to control the content of the DRC Summary Report. <b>Outputs</b> page, Reports section	[HIER EXECUTED ALL] () {}
drc.reports.drcSummaryReportViewAfterRun.value	Specifies whether to automatically open the DRC Summary Report after run is complete. <b>Outputs</b> page, Reports section	[false true] (Boolean) {false}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.rhdb.fileName.value	The path to a reusable hierarchical database (RHDB) file. <b>Outputs</b> page, Reusable Hierarchical Database section	[] (environment variable valid) { }
drc.rdhb.specfield	Specifies whether to use a reusable hierarchical database (RHDB) file. <b>Outputs</b> page	[false true] (Boolean) {false}
drc.rulesFile.value	The path to the rule file. <b>Rules</b> page	[] () {rules}
drc.runDir.value	The run directory. All relative pathnames are relative to the run directory. <b>Rules</b> page	[] () {. (current directory)}
drc.runType.value	The run type: <ul style="list-style-type: none"> <li>• HIER — A hierarchical run.</li> <li>• FLAT — A flat run.</li> <li>• FLATCB — A flat run with the Calibre CB license.</li> <li>• WAIVERGEN — A waiver shape generation run.</li> <li>• FASTXOR — A Calibre FastXOR layout comparison run.</li> </ul> <b>Inputs</b> page	[HIER FLAT FLATCB WAIVERGEN FASTXOR] () {HIER}
drc.runsetPrefs.runsetInfo.value	User-defined text specifying information about the runset. Whitespace is not allowed. <b>Preferences</b> page, Runset section	[] () { }
drc.runsetPrefs.runsetType.value	The runset type. Specify a built-in value or a user-defined text string. <b>Preferences</b> page, Runset section	[DRC LFD DFM <i>string</i> ] () { }

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.rve.autoStart.value	Specifies whether to automatically open the results database in Calibre RVE when the run is complete. <b>Run Control</b> page, RVE Options, “Show Results in RVE”	[true false] (Boolean) {true}
drc.rve.dontOpenEmptyMainRDB.value	When true, an empty DRC results database is not opened if side RDB files have been created.  This option is not available if drc.rve.dynamicRVE.value = true. <b>Run Control</b> page, RVE Options, “Do not open empty main results database in RVE”	[false true] (Boolean) {false}
drc.rve.dynamicRVE.value	Specifies whether to start Calibre RVE as soon as DRC results are available. <b>Run Control</b> page, RVE Options, “Start RVE as soon as results are available”	[true false] (Boolean) {true}
drc.rve.startRVEAfterBatch.value	Specifies whether to automatically start Calibre RVE after a batch run finishes. <b>Run Control</b> page, RVE Options, “Start RVE after batch run”	[false true] (Boolean) {false}
drc.selectChecksSettings.checksSortOrder.value	Specifies how to sort checks in the Recipe Editor display. <ul style="list-style-type: none"> <li>• SELECTION — By checked/unchecked status</li> <li>• ALPHA — Alphabetically</li> <li>• ORDER — By rule file order</li> <li>• PRIORITY — By priority</li> <li>• RES_COUNT — By result count</li> </ul> <b>Recipe Editor</b> page, Recipe checks	[SELECTION ALPHA ORDER PRIORITY RES_COUNT] ( {ORDER}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.selectChecksSettings.groupsShowHier.value	Specifies whether check groups in Recipe Editor are displayed hierarchically. <b>Recipe Editor</b> page, Groups	[false true] (Boolean) {false}
drc.selectChecksSettings.groupsShowTop.value	Specifies whether check groups in the Recipe Editor are displayed from the top down. <b>Recipe Editor</b> page, Groups	[false true] (Boolean) {false}
drc.selectChecksSettings.groupsSortOrder.value	Specifies how check groups in the Recipe Editor are sorted. <ul style="list-style-type: none"> <li>• SELECTION — By checked/unchecked status</li> <li>• ALPHA — Alphabetically</li> <li>• ORDER — By rule file order</li> </ul> <b>Recipe Editor</b> page, Groups	[SELECTION ALPHA ORDER] () {ORDER}
drc.svrfIncludes.specified	Checkbox for “Include Rule Statements”. The statements are specified with drc.svrfIncludes.svrfStatements.value or pex.svrfIncludes.tvfStatements.value. <b>Options</b> page, Include Rule Statements	[false true] (Boolean) {false}
drc.svrfIncludes.statementType.value	Available only when the main rule file is TVF.  Dropdown selection for “Type”, which specifies the language of the included rule file statements. SVRF statements are included within a VERBATIM block in the generated control file.  <b>Options</b> page, Include Rule Statements.	[none SVRF TVF] () {none}

**Table C-2. DRC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
drc.svrfIncludes.svrfStatements.value	SVRF statements to be included in the control file generated by Calibre Interactive.  Used when the main rule file is SVRF, or when the main rule file is TVF and drc.svrfIncludes.statementType.value = SVRF.  <b>Options</b> page, Include Rule Statements.	[] () {}
drc.svrfIncludes.tvfStatements.value	TVF statements to be included in the control file generated by Calibre Interactive.  Used when the main rule file is TVF and drc.svrfIncludes.statementType.value = TVF.  <b>Options</b> page, Include Rule Statements.	[] () {}

## LVS Runset Options

LVS Runset Options.

**Table C-3. LVS Runset Options**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.analysisStep.value	Defines what is analyzed in the run: <ul style="list-style-type: none"><li>• LVN — Layout versus netlist</li><li>• NVN — Netlist versus netlist</li><li>• HNE — Netlist extraction</li></ul> <b>Inputs</b> page, “Step” dropdown list	[LVN NVN HNE] () {LVN}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.configPrefs.configFiles.value	The file names of the Configuration Files. <b>Preferences</b> page, Configuration section, “Configuration Files”	[] () {}
lvs.configPrefs.specified	Boolean controlling lvs.configPrefs <b>Preferences</b> page, Configuration checkbox	[false true] (Boolean) {false}
lvs.envVars.setEnv(Name, Value)	Specifies an environment variable that is set in the runset. The format is:  lvs.envVars.setEnv(“name”, “value”) See the <a href="#">envVars</a> configuration file command.  <b>Environment</b> page	[] () {}
lvs.envVars.setEnvNamePattern(Pattern)	Specifies a filter for the environment variables displayed on the <b>Environment</b> page. The format is:  lvs.envVars.setEnvNamePattern(“pattern”) <b>Environment</b> page	[] () {*}
lvs.envVars.unsetEnv(Name)	Specifies an environment variable that is unset in the runset. The format is:  lvs.envVars.unsetEnv(“name”) See the envVars configuration file command.  <b>Environment</b> page	[] () {}
lvs.erc.execute.specified	Boolean controlling lvs.erc.execute <b>ERC</b> page, “LVS Execute ERC” checkbox	[true false] (Boolean) {true}
lvs.erc.execute.value	Specifies YES or NO for the LVS EXECUTE ERC statement.  <b>ERC</b> page, “LVS Execute ERC” dropdown	[YES NO] () {YES}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.ercMaximumResults.count.value	<p>Specifies the maximum result count for an individual rule check in ERC.</p> <p>Used when lvs.ercMaximumResults.upperLimit.value = MAXIMUMCOUNT.</p> <p><b>ERC page, ERC Maximum Results, “Count”</b></p>	[ ] (positive integer) {1000}
lvs.ercMaximumResults.specified	<p>Boolean controlling lvs.ercMaximumResults</p> <p><b>ERC page, “ERC Maximum Results” checkbox</b></p>	[false true] (Boolean) {false}
lvs.ercMaximumResults.upperLimit.value	<p>Specifies whether to use the ALL keyword or an integer count in the ERC MAXIMUM RESULTS statement.</p> <ul style="list-style-type: none"> <li>• <b>MAXIMUMCOUNT</b> — Specify a maximum result count with lvs.ercMaximumResults.count.value.</li> <li>• <b>MAXIMUMALL</b> — Use the ALL keyword.</li> </ul> <p><b>ERC page, ERC Maximum Results, “Upper Limit”</b></p>	[MAXIMUMCOUNT MAXIMUMALL] ( {MAXIMUMCOUNT}
lvs.ercMaximumVertex.count.value	<p>Specifies an upper limit for the vertex count of any result polygon in ERC.</p> <p>Used when lvs.ercMaximumVertex.upperLimit.value = MAXIMUMCOUNT.</p> <p><b>ERC page, ERC Maximum Vertex, “Count”</b></p>	[ ] (integer) {4096}
lvs.ercMaximumVertex.specified	<p>Boolean controlling lvs.ercMaximumVertex</p> <p><b>ERC page, “ERC Maximum Vertex” checkbox</b></p>	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.ercMaximumVertex.upperLimit.value	Specifies whether to use the ALL keyword or an integer count in the ERC MAXIMUM VERTEX statement. <ul style="list-style-type: none"><li>• <b>MAXIMUMCOUNT</b> — Specify a maximum vertex count for a result polygon with lvs.ercMaximumVertex.count.value.</li><li>• <b>MAXIMUMALL</b> — Use the ALL keyword. <b>ERC</b> page, ERC Maximum Vertex, “Upper Limit”</li></ul>	[MAXIMUMCOUNT MAXIMUMALL] ( {MAXIMUMCOUNT}
lvs.ercSummaryReport.report.specified	Boolean controlling lvs.ercSummaryReport.report <b>ERC</b> page, ERC Summary Report, “File” checkbox	[false true] (Boolean) {false}
lvs.ercSummaryReport.report.value	The filename for the ERC Summary Report statement. <b>ERC</b> page, ERC Summary Report, “File”	[] ( {erc.summary}
lvs.ercSummaryReport.reportAccess.specified	Boolean controlling lvs.ercSummaryReport.reportAccess <b>ERC</b> page, ERC Summary Report, “Previous Contents” checkbox	[false true] (Boolean) {false}
lvs.ercSummaryReport.reportAccess.value	Specifies whether to overwrite or append to previous contents of the summary file. <b>ERC</b> page, ERC Summary Report, “Previous Contents” dropdown	[REPLACE APPEND] ( {REPLACE}
lvs.ercSummaryReport.reportKeywords.specified	Boolean controlling lvs.ercSummaryReport.reportKeywords <b>ERC</b> page, ERC Summary Report, “Keywords” checkbox	[false true] (Boolean) {false}
lvs.ercSummaryReport.reportKeywords.value	Specifies to use the HIER keyword in the ERC Summary Report statement. <b>ERC</b> page, ERC Summary Report, “Keywords” dropdown	[HIER] ( {}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.ercSummaryReport.reportViewAfterRun.value	Specifies whether to automatically view the ERC Summary Report after the run is complete. <b>ERC</b> page, ERC Summary Report, “View ERC summary Report after run finishes” checkbox	[false true] (Boolean) {false}
lvs.ercdb.resultsCellName.value	Specifies whether to output hierarchical ERC errors in their context cells (true) or in the top cell (false). The true setting adds YES CELL SPACE XFORM ALL to the ERC Cell Name statement. <b>ERC</b> page, ERC Results Database, “Output cell errors in cell space” checkbox	[true false] (Boolean) {true}
lvs.ercdb.resultsFile.value	The name of the ERC Results Database. <b>ERC</b> page, ERC Results Database, “File”	[] () {erc.results}
lvs.ercdb.resultsPseudoCells.specified	Boolean controlling lvs.ercdb.resultsPseudoCells <b>ERC</b> page, ERC Results Database, “Psuedocells results” checkbox	[false true] (Boolean) {false}
lvs.ercdb.resultsPseudoCells.value	Specifies the handling of pseudocells in ERC results. The options are keywords in the ERC Results Database statement. <b>ERC</b> page, ERC Results Database, “Psuedocells results” dropdown	[PSEUDO TOP] () {PSEUDO}
lvs.hCells.automatch.value	Boolean that controls the “Match cells by name” checkbox. Adds -automatch to the Calibre command line. <b>H-Cells</b> page, H-Cells section	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.hCells.genhcells.value	Boolean that controls the “Automatically generate an H-Cells list” checkbox. Adds -genhcells to the Calibre command line.  This option is ignored if lvs.hCells.automatch.value = true.  <b>H-Cells</b> page, H-Cells section	[false true] (Boolean) {false}
lvs.hCells.genhcellselect.value	Boolean that controls the “Use default thresholds as hcell selection criteria” checkbox. Adds the “select” keyword to the -genhcells switch.  <b>H-Cells</b> page, H-Cells section	[false true] (Boolean) {false}
lvs.hCells.hCellsFile.specified	Boolean controlling lvs.hCells.hCellsFile  <b>H-Cells</b> page, H-Cells section, “H-Cells File” checkbox	[false true] (Boolean) {false}
lvs.hCells.hCellsFile.value	The hcell file. Adds “-hcell <hcell_file>” to the command line.  <b>H-Cells</b> page, H-Cells section, “H-Cells File”	[] () {hcells}
lvs.hCells.placementmatch.value	Checkbox that controls the “Match cells by number of placements” checkbox. Used only during Hcell Analysis.  <b>H-Cells</b> page, H-Cells section, “Match cells by number of placements”	[false true] (Boolean) {false}
lvs.hCells.qsTclFile.specified	Boolean controlling lvs.hCells.qsTclFile Available only when lvs.hCells.genhcells.value = true.  <b>H-Cells</b> page, H-Cells section, “Query Server Tcl script” checkbox	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.hCells.qsTclFile.value	The path to hcells Query Server Tcl script to be executed during the run. (-genhcells=<qs_tcl_file>)  Used only when lvs.hCells.genhcells.value = true and lvs.hCells.qsTclFile.specified = true.  <b>H-Cells</b> page, H-Cells section, “Query Server Tcl script”	[] () {qs_tcl_file}
lvs.includes.ruleFiles.value	The file names of the additional rules files.  <b>Options</b> page, Include Rules Files, “Additional Rules Files”	[] () {}
lvs.includes.specified	Boolean controlling lvs.includes  <b>Options</b> page, Include Rules Files checkbox	[false true] (Boolean) {false}
lvs.layersFile.specified	Boolean controlling lvs.layersFile  Used during signature generation.  <b>Signatures</b> page, “Layers File” checkbox	[true false] (Boolean) {true}
lvs.layersFile.value	The layers file for signature generation.  <b>Signatures</b> page, “Layers File”	[] () {}
lvs.layout.exportFromLayoutViewer.value	Boolean that controls the “Export from layout viewer” checkbox.  <b>Inputs</b> page, Layout Path section	[] () {false}
lvs.layout.extraGoldenLayoutFiles.specified	Boolean controlling lvs.layout.extraGoldenLayoutFiles  <b>Database</b> page, Library section	[false true] (Boolean) {false}
lvs.layout.extraGoldenLayoutFiles.value	The file names of the additional golden layout files.  <b>Database</b> page, Library section	[] () {}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.layout.extraLayoutFiles.sp ecified	Boolean controlling lvs.layout.extraLayoutFiles <b>Database</b> page, Library section	[false true] (Boolean) {false}
lvs.layout.extraLayoutFiles.va lue	The file names of the additional layout files. <b>Database</b> page, Library section	[] () {}
lvs.layout.format.value	The format of the layout database. <b>Inputs</b> page, Layout Path, “Layout Format”  <b>Note:</b> If the layout format is OPENACCESS, see cmn.layout.layoutLibraryFDI.value, cmn.layout.topCellLibraryFDI.value, and cmn.layout.topCellViewFDI.value. If the layout format is LEFDEF, see cmn.layout.leffFiles.value, cmn.layout.defFiles.value, and similarly named options.	[GDSII OASIS LEFDEF OPENACCESS SPICE] () {GDSII}
lvs.layout.layoutFile.value	The path to the layout file for GDS or OASIS format layouts. (Layout Path statement) <b>Inputs</b> page, Layout Path, “Layout File”	[] (environment variable valid) {}
lvs.layout.layoutNetlist.value	The file name of the layout SPICE file. <b>Inputs</b> page, Layout Path, “Layout Netlist”	[] (environment variable valid) {lay.net}
lvs.layout.topCell.value	The name of the layout top cell. Used for GDS, OASIS, and OPENACCESS input. <b>Inputs</b> page, Layout Path, “Top Cell”	[] () {}
lvs.layout.topCellLibrary.valu e	The library name of the layout top cell to run Calibre on. Used only when the GUI is invoked from Cadence Virtuoso and “Export from layout viewer” is checked. <b>Inputs</b> page, Layout Path section.	[] () {}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.layout.topCellView.value	The cell view name of the layout top cell to run Calibre on. Used only when the GUI is invoked from Cadence Virtuoso and “Export from layout viewer” is checked. <b>Inputs</b> page, Layout Path section	[] () {}
lvs.layoutCase.caseSensitive.s pecified	Boolean controlling lvs.layoutCase.caseSensitive <b>Options</b> page, “Layout Case” checkbox	[false true] (Boolean) {false}
lvs.layoutCase.caseSensitive.v alue	Specifies whether the layout netlist should be processed in a case-sensitive manner. (LAYOUT CASE) <b>Options</b> page, “Layout Case” dropdown	[NO YES] () {NO}
lvs.lvsBlackBoxPort.specified	Boolean controlling lvs.lvsBlackBoxPort Available when lvs.lvsBox.specified is true. <b>Options</b> page, “LVS Black Box Port” checkbox	[false true] (Boolean) {false}
lvs.lvsBlackBoxPort.paramete rs	A list of lists defining the contents of the table for the option “LVS Black Box Port”. This table defines parameters for the LVS Black Box Port statement.  Each sublist defines parameters for one statement. The sublist syntax is:  [ bool, “val1 val2 val3” ] <ul style="list-style-type: none"><li>• bool — true or false. Indicates whether the statement is used.</li><li>• val1 — The original layer that forms the port.</li><li>• val2 — The text layer.</li><li>• val3 — The interconnect layer.</li></ul> Both lvs.lvsBlackBox.specified and lvs.lvsBlackBoxPort.specified must be true for this option to be used. <b>Options</b> page, LVS Black Box Port section	[] (list) {}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.lvsBlackBoxPortDepth.specified	Boolean controlling lvs.lvsBlackBoxPortDepth Available when lvs.lvsBlackBoxPort.specified is true. <b>Options</b> page, “LVS Black Box Port Depth” checkbox	[false true] (Boolean) {false}
lvs.lvsBlackBoxPortDepth.hierarchyLevel.value	Setting for the “Hierarchy Level” option, which specifies the keyword used in the LVS Black Box Port Depth statement. <ul style="list-style-type: none"> <li>• DEPTHPRIMARY — PRIMARY keyword</li> <li>• DEPTHALL — ALL keyword</li> <li>• DEPTHLEVEL — Specify a level with lvsBlackBoxPortDepth.level.value</li> </ul> Used when lvs.lvsBlackBox.specified, lvs.lvsBlackBoxPort.specified, and lvs.lvsBlackBoxPortDepth.specified are all true. <b>Options</b> page, LVS Black Box Port section	[DEPTHPRIMARY DEPTHALL DEPTHLEVEL] ( {DEPTHPRIMARY} RY}
lvs.lvsBlackBoxPortDepth.level.value	Specifies the hierarchical level as a non-negative integer. 0 means top level. Used when lvs.lvsBlackBoxPortDepth.hierarchyLevel.value = DEPTHLEVEL. <b>Options</b> page, LVS Black Box Port section	[] (positive integer) {0}
lvs.lvsBox.specified	Boolean controlling lvs.lvsBox <b>Options</b> page, “LVS Box” checkbox.	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.lvsBox.parameters	<p>A list of lists defining the contents of the table for the option “LVS Box”. This table defines parameters for the LVS Box statement.</p> <p>Each sublist defines parameters for one statement. The sublist syntax is:</p> <pre>[ bool, "option_string" ]</pre> <ul style="list-style-type: none"> <li>• bool — true or false. Indicates whether the statement is used.</li> <li>• option_string — The parameters for the statement.</li> </ul> <p><b>Options</b> page, LVS Box section</p>	[] (list) {}
lvs.lvsDeviceSignatures.deviceSignatures.value	<p>A list of device signature strings. This is the contents of the Device Signatures table after a signatures file is loaded.</p> <p><b>Signatures</b> page, Device Signatures table</p>	[] () {}
lvs.lvsDeviceSignatures.includeDeviceSignatures.value	<p>Checkbox for “Include Device Signatures LVS run”. When set to true, the signatures in lvs.lvsDeviceSignatures.deviceSignatures.value are used in the Calibre nmLVS run.</p> <p><b>Signatures</b> page, “Include Device Signatures LVS run” checkbox</p>	[false true] (Boolean) {false}
lvs.lvsDeviceStatements.deviceStatements.value	<p>A list of device statements in table form, including the device statement used for signature generation.</p> <p><b>Signatures</b> page, Device Signatures table</p>	[] () {}
lvs.lvsFilterUnusedOption.specified	<p>Boolean controlling lvs.lvsFilterUnusedOption</p> <p><b>Options</b> page, “LVS Filter Unused Option” checkbox</p>	[false true] (Boolean) {false}
lvs.lvsFilterUnusedOption.parameters	<p>Controls the filtering of unused devices.</p> <p><b>Options</b> page, LVS Filter Unused Option section</p>	[] (list) {}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.lvsIsolateShorts.specified	Boolean that controls lvs.lvsIsolateShorts.. <b>Options</b> page, “Short Isolation” checkbox	[false true] (Boolean) {false}
lvs.lvsIsolateShorts.isolate.value	Specify whether to perform short isolation during extraction. <b>Options</b> page, Short Isolation section, “LVS Isolate Shorts” dropdown	[NO YES POWERSHORTS GROUNDSHORTS PWRGNDSHORTS IOSHORTS] ( {NO}
lvs.lvsIsolateShorts.accumulate.value	Specifies to identify a common path, if it exists, shared by multiple paths between shorted text objects. <b>Options</b> page, Short Isolation section, “Identify a common path” checkbox	[false true] (Boolean) {false}
lvs.lvsIsolateShorts.between.value	Specifies to isolate shorts between all names or specified names. Only available if lvs.lvsIsolateShortsoperand.specified is true. <b>Options</b> page, Short Isolation section, “Isolate between” dropdown	[ALLNAMES SPECIFIEDNAMES] ( {ALLNAMES}
lvs.lvsIsolateShorts.names.value	Specify a list of text names to isolate shorts between. Only available if lvs.lvsIsolateShortsoperand.specified is true and lvs.lvsIsolateShorts.between.value is SPECIFIEDNAMES. <b>Options</b> page, Short Isolation section, Specified names	[] (list) { }
lvs.lvsIsolateShorts.byCell.value	Specifies to report the cell of origin for every polygon written to the short isolation database. <b>Options</b> page, Short Isolation section, “Report shorts by cell” checkbox	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.lvsIsolateShorts.byLayer.value	Specifies to generate output in separate results per short and per layer in the short isolation database.  <b>Options</b> page, Short Isolation section, “Report shorts by layer” checkbox	[false true] (Boolean) {false}
lvs.lvsIsolateShorts.exclude.value	Specifies to exclude names specified from short isolation. Only available if lvs.lvsIsolateShortsoperand.specified is true.  <b>Options</b> page, Short Isolation section, “Exclude” checkbox	[false true] (Boolean) {false}
lvs.lvsIsolateShorts.inCell.value	Specifies what level of cell hierarchy short isolation should be performed.  <b>Options</b> page, Short Isolation section, “Isolate shorts in” dropdown	[CELL PRIMARY CELL ALL] () {CELL PRIMARY}
lvs.lvsIsolateShorts.noContact.s.value	Specifies to omit contact layers from the output.  <b>Options</b> page, Short Isolation section, “Exclude contact polygons” checkbox	[false true] (Boolean) {false}
lvs.lvsIsolateShortsoperand.specified	Boolean controlling lvs.lvsIsolateShortsoperand.  <b>Options</b> page, Short Isolation section, “Operand” checkbox	[false true] (Boolean) {false}
lvs.lvsIsolateShortsoperand.value	Specifies logical symbols to combine the CELL and NAME parameters.  <b>Options</b> page, Short Isolation section, “Operand” dropdown	[&&   ] () {&&}
lvs.lvsIsolateShorts.runFlat.value	Specifies to run short isolation flat even during hierarchical extraction.  <b>Options</b> page, Short Isolation section, “Run flat short isolation” checkbox	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.lvsIsolateShorts.unmerged.value	Specifies to not merge segmented polygons prior to output. <b>Options</b> page, Short Isolation section, “Do not merge polygons” checkbox	[false true] (Boolean) {false}
lvs.lvsMaximumShortResults.specified	Boolean controlling lvs.lvsMaximumShortResults. <b>Options</b> page, “LVS Maximum Short Results” checkbox	[false true] (Boolean) {false}
lvs.lvsMaximumShortResults.count.value	Specifies an upper limit on the number of isolated short paths that appear in the shorts database. Only available when lvs.lvsMaximumShortResults.upperLimit.value is MAXIMUMCOUNT. <b>Options</b> page, LVS Maximum Short Results section, Count	[] (integer) {50}
lvs.lvsMaximumShortResults.upperLimit.value	Specifies an upper limit on the number of isolated short paths that appear in the shorts database. <ul style="list-style-type: none"> <li>• MAXIMUMCOUNT — Define an upper limit.</li> <li>• MAXIMUMALL — All isolated short paths appear in the shorts database.</li> </ul> <b>Options</b> page, LVS Maximum Short section, “Upper Limit” dropdown	[MAXIMUMCOUNT MAXIMUMALL] ( {MAXIMUMCOUNT}
lvs.lvsRecognizeGates.specified	Boolean controlling lvs.lvsRecognizeGates When true, the LVS Recognize Gates statement is written to the control file. <b>Options</b> page, “LVS Recognize Gate” checkbox	[false true] (Boolean) {false}
lvs.lvsRecognizeGates.gatesType.value	Specifies the types of gates that the tool recognizes. The choices correspond to keywords for the LVS Recognize Gates statement. <b>Options</b> page, LVS Recognize Gates section, “Recognize” dropdown	[ALL SIMPLE NONE] ( {ALL}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.lvsRecognizeGates.mixSubTypes.value	Specifies whether to allow subtype mixing of transistors in series-parallel structures. Adds the MIX SUBTYPES keyword.  <b>Options</b> page, LVS Recognize Gate section, “Mix Subtypes” checkbox	[false true] (Boolean) {false}
lvs.lvsRecognizeGates.xAlso.value	Specifies whether to allow MOS transistors with X+ subtypes to be included. Adds the XALSO keyword.  <b>Options</b> page, LVS Recognize Gate section, “Include X+ Subtypes” checkbox	[false true] (Boolean) {false}
lvs.lvsReconSIMode	Controls the type of shorts to locate when running Calibre nmLVS Reconnaissance Short Isolation. <ul style="list-style-type: none"> <li>• <b>ALL</b> — Perform both Power Ground and IO short isolation</li> <li>• <b>PG</b> — Power Ground short isolation only</li> <li>• <b>OP</b> — IO short isolation only</li> </ul>	[ ALL PG OP ] () {ALL}
lvs.lvsReduceGates.mixTypes.value	Specifies that split gate structure contains transistors with different component types, pins, pin names or pin swappability. Adds the MIXTYPES keyword to LVS Reduce Split Gates YES.  Only available when lvs.lvsReduceGates.splitGates.value is YES and lvs.lvsReduceGates.splitGates.specified is true.  <b>Options</b> page, Gate Reduction section, “Split gates contain mixed types” checkbox.	[false true] (Boolean) {false}
lvs.lvsReduceGates.reduceParallelMOS.specified	Boolean controlling lvs.lvsReduceGates.reduceParallelMOS  <b>Options</b> page, Gate Reduction section, “LVS Reduce Parallel MOS” checkbox	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.lvsReduceGates.reduceParallelMOS.value	<p>Specify whether to reduce MOS transistors connected in parallel into single transistors.</p> <p>NO cannot be specified with lvs.lvsReduceGates.splitGates.value = YES.</p> <p><b>Options</b> page, Gate Reduction section, “LVS Reduce Parallel MOS” dropdown</p>	[YES NO] ( {YES}
lvs.lvsReduceGates.sameOrderSplitGates.value	<p>Specifies to reduce split gates only when the input order is the same in all transistor strings that form the split gate. Adds the keyword SAME ORDER to LVS Reduce Split Gates YES.</p> <p>Only available when lvs.lvsReduceGates.splitGates.value = YES and lvs.lvsReduceGates.splitGates.specified = true.</p> <p><b>Options</b> page, Gate Reduction section, “Split only when the input order is same” checkbox</p>	[false true] (Boolean) {false}
lvs.lvsReduceGates.semiSplitGates.value	<p>Specifies to reduce semi-split gate structures in addition to reducing fully split gates. Adds the keyword SEMI ALSO to LVS Reduce Split Gates YES.</p> <p>Only available when lvs.lvsReduceGates.splitGates.value = YES and lvs.lvsReduceGates.splitGates.specified = true.</p> <p><b>Options</b> page, Gate Reduction section, “Semi-split gates” checkbox</p>	[false true] (Boolean) {false}
lvs.lvsReduceGates.shortEquivalentNodes.specified	<p>Boolean controlling lvs.lvsReduceGates.shortEquivalentNodes</p> <p><b>Options</b> page, Gate Reduction section, “LVS Short Equivalent Nodes” checkbox</p>	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.lvsReduceGates.shortEquivalentNodes.value	Specifies whether to short together equivalent nodes in MOSFET split gate structure. Controls the primary keyword for the LVS Short Equivalent Nodes statement. PARALLEL and SPLIT keywords cannot be specified with lvs.lvsReduceGates.splitGates.value = YES. <b>Options</b> page, Gate Reduction section, “LVS Short Equivalent Nodes” dropdown	[NO PARALLEL SPLIT] () {NO}
lvs.lvsReduceGates.spAlso.value	Specifies series-parallel split gate reduction is performed in addition to regular split-gate reduction. Adds the keyword SP ALSO to LVS Reduce Split Gates YES.  Only available when lvs.lvsReduceGates.splitGates.value = YES and lvs.lvsReduceGates.splitGates.specified = true.  <b>Options</b> page, Gate Reduction section, “Split series-parallel gates” checkbox	[false true] (Boolean) {false}
lvs.lvsReduceGates.splitGates.specified	Boolean controlling lvs.lvsReduceGates.splitGates <b>Options</b> page, Gate Reduction section, “LVS Reduce Split Gates” checkbox	[false true] (Boolean) {false}
lvs.lvsReduceGates.splitGates.value	Specifies whether to reduce MOS split gates formed as serial-up or serial-down structures into single gate structures.  <b>Options</b> page, Gate Reduction section, “LVS Reduce Split Gates” checkbox	[YES NO] () {YES}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.lvsReduceGates.withinTolerance.value	<p>Specifies split-gate reduction should not occur in cases where the tolerance value in an LVS Split Gate Ratio statement would report a discrepancy. Adds the keyword WITHIN TOLERANCE to LVS Reduce Split Gates YES.</p> <p>Only available when lvs.lvsReduceGates.splitGates.value = YES and lvs.lvsReduceGates.splitGates.specified = true.</p> <p><b>Options</b> page, Gate Reduction section, “Do not split when tolerance reports discrepancy” checkbox</p>	[false true] (Boolean) {false}
lvs.lvsReportMaximum.specified	Boolean controlling lvs.lvsReportMaximum	[false true] (Boolean) {false}
lvs.lvsReportMaximum.upperLimit.value	<p>Specifies whether to place an upper limit on the number of items that appear in various sections of the LVS report.</p> <ul style="list-style-type: none"> <li>• MAXIMUMCOUNT — Adds the keyword “MAXIMUM number” to LVS REPORT. lvs.lvsReportMaximum.count.value specifies number.</li> <li>• MAXIMUMALL — Adds the keyword ALL to LVS REPORT.</li> </ul> <p><b>Options</b> page, LVS Report Maximum section, “Upper Limit” dropdown</p>	[MAXIMUMCOUNT MAXIMUMALL] ( {MAXIMUMCOUNT}
lvs.lvsReportMaximum.count.value	<p>Specifies the upper limit on the number of items printed in the LVS report.</p> <p>Only available when lvs.lvsReportMaximum.upperLimit.value is MAXIMUMCOUNT.</p> <p><b>Options</b> page, LVS Report Maximum section, Count</p>	[] (positive integer) {50}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.lvsReportOption.specified	Boolean controlling lvs.lvsReportOption <b>Options</b> page, “LVS Report Option” checkbox	[false true] (Boolean) {false}
lvs.lvsReportOption.optionKeywords.value	A list specifying the option keywords used for the LVS Report Option statement. Each option is quoted; for example:  <code>lvs.lvsReportOption.optionKeywords.value = [ "A:", "B" ]</code>  You can specify [“All”] to set all options; however, this greatly increases the report output.  <b>Options</b> page, LVS Report Option section, “Keywords” dropdown	[A B C D E EB EC EO ES F G H I N O P R S V W X BX AV BV CV E1 FX LPE MC NCA NE NOK NP RA RD SP SPE XR] ( {}
lvs.lvsSISelectConnects.siSelectBy.value	Specifies the Connect or Sconnect operations to participate in Calibre nmLVS Reconnaissance Short Isolation.  <b>Options</b> page, LVS SI Select Connects section, “Select By” dropdown	[ALL LAYER CONNECT OPERATIONS] ( {ALL})
lvs.lvsSISelectConnects.siSelectConnects.value	Specifies layer names that appear in Connect or Sconnect operations of the two-layer or triplet form when lvs.lvsSISelectConnects.siSelectBy.value = CONNECT.  <b>Options</b> page, LVS SI Select Connects section	[ ( {}
lvs.lvsSISelectConnects.siSelectLayers.value	Specifies layer names that appear in Connect or Sconnect operations in the rule file when lvs.lvsSISelectConnects.siSelectBy.value = LAYER.  <b>Options</b> page, LVS SI Select Connects section	[ ( {}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.lvsSISelectConnects.siSelectOperations.value	Specifies Layer, Connect, or Sconnect operations when lvs.lvsSISelectConnects.siSelectBy.value = OPERATIONS.  <b>Options</b> page, LVS SI Select Connects section	[] () {}
lvs.lvsSISelectConnects.specified	Boolean controlling lvs.lvsSISelectConnects  <b>Options</b> page, “LVS SI Select Connects” checkbox	[false true] (Boolean) {false}
lvs.recipeEditor.setActiveRecipe	Specifies the active check recipe. These recipes are used for ERC checks.  <b>Recipe Editor</b> page	[] () {}
lvs.recipeEditor.setUserRecipes	Specifies user check recipes using an internal specification format. These recipes are used for ERC checks.  <b>Recipe Editor</b> page	[] () {}
lvs.recipePrefs.userRecipeFiles.specified	Boolean controlling lvs.recipePrefs.userRecipeFiles  <b>Preferences</b> page, Recipes section, “User Recipe Files” checkbox	[false true] (Boolean) {false}
lvs.recipePrefs.userRecipeFiles.value	A list of user recipe files.  <b>Preferences</b> page, Recipes section	[] () {}
lvs.reports.lvsExtractionReportViewAfterRun.value	Boolean that controls whether the Extraction Report file is displayed after the run.  <b>Outputs</b> page, Reports section, “View Extraction Report after run finishes” checkbox	[false true] (Boolean) {false}
lvs.reports.lvsReport.value	Specifies the path to the LVS report file.  <b>Outputs</b> page, Reports section, LVS Report	[] () {lvs.report}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.reports.lvsReportViewAfterRun.value	Boolean that controls whether the LVS Report file is displayed after the run. <b>Outputs</b> page, Reports section, “View LVS Report after run finishes” checkbox	[false true] (Boolean) {false}
lvs.reports.lvsSummaryReport.specified	Boolean controlling lvs.reports.lvsSummaryReport <b>Outputs</b> page, Reports section, “LVS Summary Report” checkbox	[false true] (Boolean) {false}
lvs.reports.lvsSummaryReport.value	Specifies the path to a file containing the summary of an LVS run. <b>Outputs</b> page, Reports section, LVS Summary Report	[] () {lvs.summary}
lvs.reports.lvsSummaryReport.ViewAfterRun.value	Boolean that controls whether the LVS Summary Report is displayed after the run. Only available when lvs.reports.lvsSummaryReport.specifies is true. <b>Outputs</b> page, Reports section, “View LVS Summary Report after run finishes” checkbox	[false true] (Boolean) {false}
lvs.rulesFile.value	Specifies the file path of the rule file. <b>Rules</b> page, Rules File	[] () {rules}
lvs.runDir.value	Specifies the path of the run directory, from which the tool performs its actions and all relative pathnames are resolved. <b>Rules</b> page, Run Directory	[] () {. (current directory)}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.runType.value	<p>Specifies the run type.</p> <ul style="list-style-type: none"> <li>• HIER — Runs in hierarchical mode</li> <li>• FLAT — Runs in flat mode</li> <li>• RECONSI — Runs in Calibre nmLVS Reconnaissance Short Isolation mode</li> <li>• FLATCB — Runs with the Calibre CB (Cell/Block) flat license package.</li> <li>• HCELLANALYSIS — Performs hcell analysis.</li> <li>• SIGGEN — Generates device signatures.</li> <li>• WAIVERGEN — Runs auto-waiver generation.</li> </ul> <p><b>Inputs</b> page, “Run” dropdown</p>	<p>[HIER FLAT RECONSI FLATCB HCELLANALYI SIS SIGGEN WAIVERGEN] ( {HIER})</p>
lvs.runsetPrefs.runsetInfo.value	<p>One line of text for the runset info field.</p> <p><b>Preferences</b> page, Runset section, Info</p>	<p>[] ( {})</p>
lvs.runsetPrefs.runsetType.value	<p>One line of text for the runset type field.</p> <p><b>Preferences</b> page, Runset section, “Type” dropdown</p>	<p>[LVS ERC] ( {})</p>
lvs.rve.autoStart.value	<p>Boolean that controls whether to automatically display results in Calibre RVE when the run finishes.</p> <p><b>Run Control</b> page, RVE Options section, “Show Results in RVE” checkbox</p>	<p>[true false] (Boolean) {true}</p>
lvs.rve.startRVEAfterBatch.value	<p>Select this setting to automatically start RVE after batch run finishes.</p> <p><b>Run Control</b> page, RVE Options section, “Start RVE after batch run” checkbox</p>	<p>[false true] (Boolean) {false}</p>
lvs.seedPromotion.createReport.specified	<p>Boolean controlling lvs.seedPromotion.createReport</p> <p><b>Options</b> page, Seed Promotion section, “Create Seed Promotions Report” checkbox</p>	<p>[false true] (Boolean) {false}</p>

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.seedPromotion.createReport.value	Specifies whether to create a seed promotions report. <b>Options</b> page, Seed Promotion section, “Create Seed Promotions Report” dropdown	[NO YES] () {NO}
lvs.seedPromotion.maximum.specified	Boolean controlling lvs.seedPromotion.maximum <b>Options</b> page, Seed Promotion section, “Maximum polygons per seed promotion” checkbox	[false true] (Boolean) {false}
lvs.seedPromotion.maximum.value	Specifies the maximum number of seed promotions reported to the seed promotions report. <b>Options</b> page, Seed Promotion section, Maximum polygons per seed promotion	[] (positive integer) {50}
lvs.selectChecksSettings.checksSortOrder.value	Specifies the order in which to display the checks in the Recipe Editor. <b>ERC</b> page	[SELECTION ALPHA ORDER PRIORITY RES_COUNT] () {ORDER}
lvs.selectChecksSettings.groupsShowHier.value	Specifies whether to display the groups in the Recipe Editor hierarchically. <b>Recipe Editor</b> page, Advanced section, Groups, “Show groups hierarchy” checkbox	[false true] (Boolean) {false}
lvs.selectChecksSettings.groupsShowTop.value	Specifies whether to display the groups in the Recipe Editor from the top down. <b>Recipe Editor</b> page, Advanced section, Groups, “Show top level groups” checkbox	[false true] (Boolean) {false}
lvs.selectChecksSettings.groupsSortOrder.value	Specifies the order in which to display the groups in the Recipe Editor. <b>ERC</b> page	[SELECTION ALPHA ORDER] () {ORDER}

Table C-3. LVS Runset Options (cont.)

Name	Description	[Available Values] (Type restrictions) {default}
lvs.signaturesFile.value	Specifies the signatures file. <b>Signatures</b> page	[] () {}
lvs.source.exportFromSourceViewer.value	Specifies to have the source viewer export the netlist before running the flow. <b>Inputs</b> page, Source Path section, “Export from source viewer” checkbox	[] () {false}
lvs.source.extraSourceFiles.specified	Boolean controlling lvs.source.extraSourceFiles <b>Database</b> page, Library section, “Additional SPICE Files” checkbox	[false true] (Boolean) {false}
lvs.source.extraSourceFiles.value	File paths of extra input SPICE source netlist files. <b>Database</b> page, Library section, Additional SPICE Files	[] () {}
lvs.source.extraVerilogSourceFiles.specified	Boolean controlling lvs.source.extraVerilogSourceFiles <b>Database</b> page, Library section	[false true] (Boolean) {false}
lvs.source.extraVerilogSourceFiles.value	File paths of extra input VERILOG source netlist files. <b>Database</b> page, Library section, Additional VERILOG Files	[] () {}
lvs.source.format.value	The format of the source netlist input. <ul style="list-style-type: none"> <li>• SPICE — SPICE format</li> <li>• VERILOG — VERILOG format</li> <li>• MIXED — A combination of SPICE and VERILOG files.</li> </ul> For VERILOG and MIXED input, the netlist is converted to SPICE format before input to Calibre. <b>Inputs</b> page, Source Path section, Source Format	[SPICE VERILOG MIXED] () {SPICE}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.source.sourceFile.value	The path to the top level source netlist file. When lvs.source.format.value is SPICE or MIXED, this is the top level SPICE file. When lvs.source.format.value is VERILOG, this is the top level VERILOG file. (For MIXED input, the VERILOG file is specified by lvs.source.verilogSourceFile.value.) <b>Inputs</b> page, Source Path section	[] (environment variable valid) {}
lvs.source.topCell.value	The name of the primary (top) cell or subcircuit of the source netlist. This is the parameter for the Source Primary statement. <b>Inputs</b> page, Source Path section, Top Cell	[] () {}
lvs.source.topCellLibrary.value	The library name for the top cell of the source netlist. Used when exporting the source netlist from a schematic design tool. <b>Inputs</b> page, Source Path section	[] () {}
lvs.source.topCellView.value	The view name for the top cell of the source netlist. Used when exporting the source netlist from a schematic design tool. <b>Inputs</b> page, Source Path section	[] () {}
lvs.source.verilogSourceFile.value	The path to the VERILOG source netlist file when a combination of SPICE and VERILOG netlist input files is used (lvs.source.format.value = MIXED). <b>Inputs</b> page, Source Path section	[] () {}
lvs.source2.exportFromSourceViewer.value	Specifies to have the source viewer export the layout netlist before running the Netlist vs Netlist flow. <b>Inputs</b> page	[] () {false}
lvs.source2.exportFromLayoutViewer.value	Specifies to have the layout viewer export the netlist before running the flow. <b>Inputs</b> page, Layout Path section, “Export from layout viewer” checkbox	[] () {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.source2.sourceFile.value	The path to the top level layout netlist file when in Netlist vs Netlist mode. <b>Inputs</b> page, Layout Path section	[] (environment variable valid) {lay.net}
lvs.source2.topCell.value	The name of the primary (top) cell or subcircuit of the layout netlist. This is the parameter for the Layout Primary statement. <b>Inputs</b> page, Layout Path section, Top Cell	[] () {}
lvs.source2.topCellLibrary.value	The library name for the top cell of the layout netlist. Used when exporting the layout netlist from a schematic design tool. <b>Inputs</b> page, Layout Path section	[] () {}
lvs.source2.topCellView.value	The view name for the top cell of the layout netlist. Used when exporting the layout netlist from a schematic design tool. <b>Inputs</b> page, Layout Path section	[] () {}
lvs.sourceCase.caseSensitive.specified	Boolean controlling lvs.sourceCase.caseSensitive <b>Options</b> page, “Source Base” checkbox	[false true] (Boolean) {false}
lvs.sourceCase.caseSensitive.value	Specifies whether the source netlist is processed in a case-sensitive manner. <b>Options</b> page, “Source Base” dropdown	[NO YES] () {NO}
lvs.supply.groundNames.specified	Boolean controlling lvs.supply.groundNames <b>Options</b> page, Power Supply section, “Ground Nets” checkbox	[false true] (Boolean) {false}
lvs.supply.groundNames.value	A list of ground net names for the LVS Ground Name statement. <b>Options</b> page, Power Supply section, Ground Nets	[] (list) {}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.supply.lvsAbortOnSoftchk.specified	Boolean controlling lvs.supply.lvsAbortOnSoftchk <b>Options</b> page, Power Supply section, “LVS Abort On Softchk” checkbox	[false true] (Boolean) {false}
lvs.supply.lvsAbortOnSoftchk.value	Specifies whether to abort if soft connection errors are discovered in the extraction phase. <b>Options</b> page, Power Supply section	[NO YES] () {NO}
lvs.supply.lvsAbortOnSupplyError.specified	Boolean controlling lvs.supply.lvsAbortOnSupplyError <b>Options</b> page, Power Supply section, “LVS Abort On Supply Error” checkbox	[false true] (Boolean) {false}
lvs.supply.lvsAbortOnSupplyError.value	Specifies whether to abort if power/ground errors are discovered in the extraction phase. <b>Options</b> page, Power Supply section	[YES NO] () {YES}
lvs.supply.lvsIgnorePorts.specified	Boolean controlling lvs.supply.lvsIgnorePorts <b>Options</b> page, Power Supply section, “LVS Ignore Ports” checkbox	[false true] (Boolean) {false}
lvs.supply.lvsIgnorePorts.value	Specifies whether to ignore ports during LVS comparison. <b>Options</b> page, Power Supply section	[NO YES] () {NO}
lvs.supply.powerNames.specified	Boolean controlling lvs.supply.powerNames <b>Options</b> page, Power Supply section, “Power Nets” checkbox	[false true] (Boolean) {false}
lvs.supply.powerNames.value	A list of power net names for the LVS Power Name statement. <b>Options</b> page, Power Supply section, Power Nets	[] (list) {}
lvs.svdb.specified	Boolean controlling lvs.svdb <b>Outputs</b> page, “Mask SVDB” checkbox	[true false] (Boolean) {true}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.svdb.asciiixref.value	Boolean that controls whether to generate ASCII cross-reference files. Adds the IXF, NXF, and SLPH keywords to the Mask SVDB Directory statement.  <b>Outputs</b> page, Mask SVDB section, “Generate ASCII cross-reference files” checkbox	[false true] (Boolean) {false}
lvs.svdb.cci.value	Boolean that controls whether to generate Calibre Connectivity Interface data. Adds the CCI keywords to the Mask SVDB Directory statement.  <b>Outputs</b> page, Mask SVDB section, “Generate Calibre Connectivity Interface data” checkbox	[false true] (Boolean) {false}
lvs.svdb.dirPath.value	The directory name for the Mask SVDB Directory statement.  <b>Outputs</b> page, Mask SVDB section, SCVDB Directory	[] () {svdb}
lvs.svdb.pinloc.specified	Boolean controlling lvs.svdb.pinloc  <b>Outputs</b> page, Mask SVDB section, “Pin Location” checkbox	[false true] (Boolean) {false}
lvs.svdb.pinloc.value	Specifies pin location generation. <ul style="list-style-type: none"> <li>• PINLOC — Triggers generation of pin location information in the PHDB.</li> <li>• NOPINLOC — Suppress generation of pin location information in the PHDB if lvs.svdb.cci.value is true.</li> </ul> <b>Outputs</b> page, Mask SVDB section, Pin Location	[PINLOC NOPINLOC] () {PINLOC}
lvs.svdb.query.value	Boolean that controls whether to generate cross-reference data. Adds the QUERY keyword to the Mask SVDB Directory statement.  <b>Outputs</b> page, Mask SVDB section, “Generate cross-reference data for RVE” checkbox	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.svdb.xrc.value	Boolean that controls whether to generate data for Calibre xRC. Adds the XRC keyword to the Mask SVDB Directory statement.  <b>Outputs</b> page, Mask SVDB section, “Generate all necessary data for Calibre xRC”	[false true] (Boolean) {false}
lvs.svrfIncludes.specified	Boolean controlling lvs.svrfIncludes <b>Options</b> page, “Include Rule Statements” checkbox	[false true] (Boolean) {false}
lvs.svrfIncludes.statementType.value	Available only when the main rule file is TVF.  Specifies the language of the included rule file statements. SVRF statements will be included within a VERBATIM block in the generated control file.  <b>Options</b> page, Include Rule Statements	[none SVRF TVF] () {none}
lvs.svrfIncludes.svrfStatements.value	SVRF statements to be included in the control file generated by Calibre Interactive.  Used when the main rule file is SVRF, or when the main rule file is TVF and lvs.svrfIncludes.statementType.value is SVRF.  <b>Options</b> page, Include Rule Statements	[] () {}
lvs.svrfIncludes.tvfStatements.value	TVF statements to be included in the control file generated by Calibre Interactive.  Used when the main rule file is TVF and lvs.svrfIncludes.statementType.value is TVF.  <b>Options</b> page, Include Rule Statements	[] () {}
lvs.traceProperty.specified	Boolean controlling lvs.traceProperty <b>Options</b> page, “Trace Property” checkbox	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.traceProperty.parameters	List of Trace Property statement values. Controls which device properties are compared during LVS and how the comparison is performed.  <b>Options</b> page, Trace Property	[] (list) {}
lvs.v2LVSLibraries.sPICEIncFiles.specified	Boolean controlling lvs.v2LVSLibraries.sPICEIncFiles  <b>V2LVS</b> page, Libraries section, “Include SPICE Files” checkbox	[false true] (Boolean) {false}
lvs.v2LVSLibraries.sPICEIncFiles.value	List of SPICE files that are included in the v2lvs command line with the -s option.  <b>V2LVS</b> page, Libraries section, Include SPICE Files	[] () {}
lvs.v2LVSLibraries.sPICELibFiles.specified	Boolean controlling lvs.v2LVSLibraries.sPICELibFiles  <b>V2LVS</b> page, Libraries section, “SPICE Library Files” checkbox	[false true] (Boolean) {false}
lvs.v2LVSLibraries.sPICELibFiles.value	List of SPICE files that are included in the v2lvs command line with the -lst or -lsp option.  <b>V2LVS</b> page, Libraries section, SPICE Library Files	[] () {}
lvs.v2LVSLibraries.vLOGLibFiles.specified	Boolean controlling lvs.v2LVSLibraries.vLOGLibFiles  <b>V2LVS</b> page, Libraries section, “Verilog Library Files” checkbox	[false true] (Boolean) {false}
lvs.v2LVSLibraries.vLOGLibFiles.value	List of Verilog files that are included in the v2lvs command line with the -l option.  <b>V2LVS</b> page, Libraries section, Verilog Library Files	[] () {}
lvs.v2LVSOptions.addPinNames.specified	Boolean controlling lvs.v2LVSOptions.addPinNames  <b>V2LVS</b> page, Options section, “Add pins to each subckt” checkbox	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.v2LVSOptions.addPinNames.value	List of pin names that are added with the -addpin argument to the v2lvs command lines.  <b>V2LVS</b> page, Options section, Add pins to each subckt	[] (list) {}
lvs.v2LVSOptions.connectUPins.value	Boolean that controls the “Create numbered nets for unconnected pins” checkbox. Adds the -n argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
lvs.v2LVSOptions.dontConnectPower.value	Boolean that controls the “Don’t connect supply0/supply1 nets to global supply” checkbox. Adds the -sk argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
lvs.v2LVSOptions.generateTemplate.value	Boolean that controls the “Generate xRC source template file” checkbox. Adds the -t argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
lvs.v2LVSOptions.keepBackSlash.value	Boolean that controls the “Preserve backslashes in escaped identifiers” checkbox. Adds the -b argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
lvs.v2LVSOptions.positionalPins.value	Boolean that controls the “Use positional pin order (don’t use \$PINS)” checkbox. Adds the -i argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
lvs.v2LVSOptions.translateAlways.value	Boolean that controls the “Always translate before running Calibre” checkbox. If this setting is false, the translator does not run in the Verilog files are older than the translated SPICE file.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.v2LVSOptions.useCBLicense.value	Boolean that controls the “Prefer Calibre-CB license during license search” checkbox. Adds the -cb argument to the v2lvs command line.	[false true] (Boolean) {false}
lvs.v2LVSOptions.useRangeMode.value	Boolean that controls the “Use aggregate mode for bus bits” checkbox.  Specifies how SPICE library files are added on the v2lvs command lines: <ul style="list-style-type: none"><li>• true — Use the -lst argument, for range mode.</li><li>• false — Use the -lsp argument, for pin mode.</li></ul> <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
lvs.v2LVSOptions.warningLevel.value	Specifies the amount of warning level output for the v2lvs translator and defines the value for the -w argument. Each level includes all warnings from each previous level. <ul style="list-style-type: none"><li>• 0 — No warnings (-w)</li><li>• 1 — Warns about skipped module and multiple declarations of modules (-w1)</li><li>• 2 — Warns about cells to undeclared modules and pin array widths wider than ports (-w2)</li><li>• 3 — Warns about port array width mismatches and unsupported compiler directives. (-w3)</li><li>• 4 — Warns about ignored constructs. (-w4)</li></ul> <b>V2LVS</b> page, Options section, Warning Level	[0 1 2 3 4] ( {2}
lvs.v2LVSOptions.warningsAreErrors.value	Boolean that controls the “Treat warnings as errors” checkbox. Adds the -werror argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.v2LVSStrings.achars.specified	Boolean controlling lvs.v2LVSStrings.achars <b>V2LVS</b> page, Strings section, “Change array delimiters to” checkbox	[false true] (Boolean) {false}
lvs.v2LVSStrings.achars.value	The value for the -a option in the v2lvs command line, which changes the array delimiter from the default []. The first character replaces [, and the option second character replaces ]. <b>V2LVS</b> page, Strings section, Change array delimiters to	[] () {}
lvs.v2LVSStrings.cchars.specified	Boolean controlling lvs.v2LVSStrings.cchars <b>V2LVS</b> page, Strings section “Change illegal SPICE characters to” checkbox	[false true] (Boolean) {false}
lvs.v2LVSStrings.cchars.value	The value for the -c option in the v2lvs command line. Enter one or two characters enclosed in quotation marks. To include single quotes, as recommended, enclose the characters in single quotes, then outer double quotes. <b>V2LVS</b> page, Strings section, Change illegal SPICE characters to	[] () {}
lvs.v2LVSStrings.pprefix.specified	Boolean controlling lvs.v2LVSStrings.pprefix <b>V2LVS</b> page, Strings section, “Prefix for gate level primitives” checkbox	[false true] (Boolean) {false}
lvs.v2LVSStrings.pprefix.value	The value for the -p option in the v2lvs command line, which specifies a prefix to add to Verilog gate level primitive cells. <b>V2LVS</b> page, Strings section, Prefix for gate level primitives	[] () {}
lvs.v2LVSStrings.s0Net.specified	Boolean controlling lvs.v2LVSStrings.s0Net <b>V2LVS</b> page, Strings section “Connect pins with 0 value to net” checkbox	[false true] (Boolean) {false}

**Table C-3. LVS Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
lvs.v2LVSStrings.s0Net.value	The value for the -s0 option in the v2lvs command line, the default net name for mapping to pin_connections with a value of 0.  <b>V2LVS</b> page, Strings section, Connect pins with 0 value to net	[] 0 {}
lvs.v2LVSStrings.s1Net.specified	Boolean controlling lvs.v2LVSStrings.s1Net  <b>V2LVS</b> page, Strings section, “Connect pins with 1 value to net” checkbox	[false true] (Boolean) {false}
lvs.v2LVSStrings.s1Net.value	The value for the -s1 option in the v2lvs command line, the default net name for mapping to pin_connections with a value of 1.  <b>V2LVS</b> page, Strings section, Connect pins with 1 value to net	[] 0 {}
lvs.v2LVSStrings.uprefix.specified	Boolean controlling lvs.v2LVSStrings.uprefix  <b>V2LVS</b> page, Strings section, “Prefix for unnamed pins and gates” checkbox	[false true] (Boolean) {false}
lvs.v2LVSStrings.uprefix.value	The value for the -u option in the v2lvs command line, which specifies the prefix added to unnamed pin connections in module instantiations and to unnamed primitive and gate instantiations.  <b>V2LVS</b> page, Strings section, Prefix for unnamed pins and gates	[] 0 {}
lvs.v2LVSStrings.userOptions.specified	Boolean controlling lvs.v2LVSStrings.userOptions  <b>V2LVS</b> page, Strings section, “More v2lvs program options” checkbox	[false true] (Boolean) {false}
lvs.v2LVSStrings.userOptions.value	A list of options that are included without modification in the v2lvs command line.  <b>V2LVS</b> page, Strings section, More v2lvs program options	[] (list) {}

## PERC Runset Options

PERC Runset Options.

**Table C-4. PERC Runset Options**

Name	Description	[Available Values] (Type restrictions) {default}
perc.analysisStep.value	Defines what is analyzed in the run: <ul style="list-style-type: none"><li>• LAYOUT — Analyze a geometric layout.</li><li>• LAYOUTNETLIST - Analyze a layout netlist. Not available in a LDL run.</li><li>• SOURCENETLIST - Analyze a source netlist. Not available in a LDL run.</li><li>• SOURCEBASEDFLOW - Analyze a source netlist in a LDL run.</li></ul> <b>Inputs</b> page, “Step” dropdown list	[LAYOUT LAYOUTNETLIST SOURCENETLIST SOURCEBASEDFLOW] ( {LAYOUT})
perc.configPrefs.configFiles.value	The file names of the Configuration Files. <b>Preferences</b> page, Configuration section, “Configuration Files”	[] ( {})
perc.configPrefs.specified	Boolean controlling perc.configPrefs <b>Preferences</b> page, Configuration checkbox	[false true] (Boolean) {false}
perc.dfmdb.dirPath.value	The path to the DFM database directory that is created. Only available in LDL runs. <b>Outputs</b> page, DFM Database, “DFM Database Directory”	[] ( {dfmdb})

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.dfmdb.keywords.value	Specifies a list of one or more keywords that are used when generating the dfmdb. The values correspond to keywords in the DFM Database statement. Only available in LDL runs. <b>Outputs</b> page, DFM Database, “Keywords” dropdown list	[DEVICES PINLOC ANALYZE ORIGINAL MEASURE PROPERTY DENSITY] ( {} )
perc.dfmdb.specified	Boolean controlling perc.dfmdb Only available in LDL runs. <b>Outputs</b> page, DFM Database	[true false] (Boolean) { true }
perc.envVars.setEnv(Name, Value)	Specifies an environment variable that is set in the runset. The format is: perc.envVars.setEnv(“name”, “value”) See the <a href="#">envVars</a> configuration file command. <b>Environment</b> page	[] ( {})
perc.envVars.setEnvNamePattern(Pattern)	Specifies a filter for the environment variables displayed on the <b>Environment</b> page. The format is: perc.envVars.setEnvNamePattern(“pattern”) <b>Environment</b> page	[] ( {*})
perc.envVars.unsetEnv(Name)	Specifies an environment variable that is unset in the runset. The format is: perc.envVars.unsetEnv(“name”) See the <a href="#">envVars</a> configuration file command. <b>Environment</b> page	[] ( {})
perc.hCells.automatch.value	Boolean that controls the “Match cells by name” checkbox. Adds -automatch to the Calibre command line. <b>Inputs</b> page, H-Cells section	[false true] (Boolean) { false }

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.hCells.hCellsFile.specified	Boolean controlling perc.hCells.hCellsFile Adds “-hcell <hcell_file>” to the Calibre command line. <b>Inputs</b> page, H-Cells section	[false true] (Boolean) {false}
perc.hCells.hCellsFile.value	The hcells file. <b>Inputs</b> page, H-Cells section	[] () {hcells}
perc.includes.ruleFiles.value	The file names of the additional rules files. <b>Options</b> page, Include Rules Files, “Additional Rules Files”	[] () {}
perc.includes.specified	Boolean controlling perc.includes <b>Options</b> page, Include Rules Files checkbox	[false true] (Boolean) {false}
perc.layout.exportFromLayoutViewer.value	Boolean that controls the “Export from layout viewer” checkbox. <b>Inputs</b> page, Layout Path section	[] () {false}
perc.layout.extraGoldenLayoutFiles.specified	Boolean controlling perc.layout.extraGoldenLayoutFiles <b>Database</b> page, Library section	[false true] (Boolean) {false}
perc.layout.extraGoldenLayoutFiles.value	The file names of the additional golden layout files. <b>Database</b> page, Library section	[] () {}
perc.layout.extraLayoutFiles.specified	Boolean controlling perc.layout.extraLayoutFiles <b>Database</b> page, Library section	[false true] (Boolean) {false}
perc.layout.extraLayoutFiles.value	The file names of the additional layout files. <b>Database</b> page, Library section	[] () {}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.layout.format.value	The format of the layout database. <b>Inputs</b> page, Layout Path, “Layout Format”  Note: If the layout format is OPENACCESS, see cmn.layout.layoutLibraryFDI.value, cmn.layout.topCellLibraryFDI.value, and cmn.layout.topCellViewFDI.value. If the layout format is LEFDEF, see cmn.layout.leffFiles.value, cmn.layout.defFiles.value, and similarly named options.	[GDSII OASIS LEFDEF OPENACCESS SPICE] ( {GDSII}
perc.layout.layoutFile.value	The path to the layout file for GDS or OASIS format layouts. (Layout Path statement) <b>Inputs</b> page, Layout Path, “Layout File”	[] (environment variable valid) {}
perc.layout.layoutNetlist.value	The file name of the layout SPICE file. Used when analyzing the layout netlist (layout format SPICE). <b>Inputs</b> page, Layout Path, “Layout Netlist”	[] (environment variable valid) {lay.net}
perc.layout.topCell.value	The name of the layout top cell. Used for GDS, OASIS, and OPENACCESS input. <b>Inputs</b> page, Layout Path, “Top Cell”	[] ( {}
perc.layout.topCellLibrary.value	The library name of the layout top cell to run Calibre on. Used only when the GUI is invoked from Cadence Virtuoso and “Export from layout viewer” is checked. <b>Inputs</b> page, Layout Path section	[] ( {}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.layout.topCellView.value	The cell view name of the layout top cell to run Calibre on. Used only when the GUI is invoked from Cadence Virtuoso and “Export from layout viewer” is checked. <b>Inputs</b> page, Layout Path section	[] () {}
perc.lefDefDirectRead.cellInfoFile.specified	Checkbox for “Cell Info File”. The filename is specified with perc.lefDefDirectRead.cellInfoFile.value.  Available when perc.lefDefDirectRead.specified = true. <b>Inputs</b> page, Cell-Based flow, “Cell Info File” checkbox	[false true] (Boolean) {false}
perc.lefDefDirectRead.cellInfoFile.value	The name of the cell information file in the Calibre PERC LDL cell-based flow. Adds the “-soc -cell_info_file <filename>” command line options.  Used when perc.lefDefDirectRead.cellInfoFile.specified = true. <b>Inputs</b> page, Cell-Based flow, “Cell Info File”	[] () {}
perc.lefDefDirectRead.specified	Checkbox for “Cell-Based flow”. Only available for PERC LDL runs with LEF/DEF layout format. <b>Inputs</b> page, Cell-Based flow checkbox	[false true] (Boolean) {false}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.lvsBlackBoxPort.parameters	<p>A list of lists defining the contents of the table for the option “LVS Black Box Port”. This table defines parameters for the for the <a href="#">LVS Black Box Port</a> statement.</p> <p>Each sublist defines parameters for one statement. The sublist syntax is:</p> <pre>[ bool, "val1 val2 val3"]</pre> <ul style="list-style-type: none"> <li>• bool — true or false. Indicates whether the statement is used.</li> <li>• val1 — The original layer that forms the port.</li> <li>• val2 — The text layer.</li> <li>• val3 — The interconnect layer.</li> </ul> <p>Both perc.lvsBlackBoxPort.specified and perc.lvsBox.specified must be true for this option to be used.</p> <p><b>Options</b> page, LVS Black Box Port section</p>	[ ] (list) { }
perc.lvsBlackBoxPort.specified	<p>Boolean controlling perc.lvsBlackBoxPort</p> <p>Available when perc.lvsBox.specified = true.</p> <p><b>Options</b> page, LVS Black Box Port section</p>	[false true] (Boolean) {false}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.lvsBlackBoxPortDepth.hierarchyLevel.value	<p>Setting for the “Hierarchy Level” option, which specifies the keyword used in the <a href="#">LVS Black Box Port Depth</a> statement.</p> <ul style="list-style-type: none"> <li>• DEPTHPRIMARY — PRIMARY keyword.</li> <li>• DEPTHALL — ALL keyword.</li> <li>• DEPTHLEVEL — Specify a level with perc.lvsBlackBoxPortDepth.level.value.</li> </ul> <p>Used when perc.lvsBox.specified, perc.lvsBlackBoxPort.specified, and perc.lvsBlackBoxPortDepth.are all true.</p> <p><b>Options</b> page, LVS Black Box Port Depth section</p>	[DEPTHPRIMARY DEPTHALL DEPTHLEVEL] ( {DEPTHPRIMARY}
perc.lvsBlackBoxPortDepth.level.value	<p>Specifies the hierarchical level for the <a href="#">LVS Black Box Port Depth</a> statement. 0 means top level. Used when perc.lvsBlackBoxPortDepth.hierarchyLevel.value = DEPTHLEVEL.</p> <p><b>Options</b> page, LVS Black Box Port Depth section</p>	[] (positive integer) {0}
perc.lvsBlackBoxPortDepth.specified	<p>Boolean controlling perc.lvsBlackBoxPortDepth</p> <p>Available when perc.lvsBlackBoxPort.specified = true.</p> <p><b>Options</b> page, LVS Black Box Port Depth section</p>	[false true] (Boolean) {false}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.lvsBox.parameters	<p>A list of lists defining the contents of the table for the option “LVS Box”. This table defines parameters for the for the <b>LVS Box</b> statement.</p> <p>Each sublist defines parameters for one statement. The sublist syntax is:</p> <pre>[ bool, "option_string"]</pre> <ul style="list-style-type: none"> <li>• <b>bool</b> — true or false. Indicates whether the statement is used.</li> <li>• <b>option_string</b> — The parameters for the statement.</li> </ul> <p><b>Options</b> page, LVS Box section</p>	[ ] (list) { }
perc.lvsBox.specified	<p>Boolean controlling perc.lvsBox</p> <p><b>Options</b> page, LVS Box section</p>	[false true] (Boolean) {false}
perc.lvsFilterUnusedOption.parameters	<p>Controls the filtering of unused devices.</p> <p><b>Options</b> page, LVS Filter Unused Option section</p>	[ ] (list) { }
perc.lvsFilterUnusedOption.specified	<p>Boolean controlling perc.lvsFilterUnusedOption</p> <p><b>Options</b> page, LVS Filter Unused Option section</p>	[false true] (Boolean) {false}
perc.lvsRecognizeGates.gatesType.value	<p>Specify the types of gates that the tool recognizes. The choices correspond to keywords for the LVS Recognize Gates statement.</p> <p><b>Options</b> page, LVS Recognize Gates section</p>	[ALL SIMPLE NONE] () {ALL}
perc.lvsRecognizeGates.mixSubTypes.value	<p>Specifies whether to add the MIX SUBTYPES keyword to the LVS Recognize Gates statement.</p> <p><b>Options</b> page, LVS Recognize Gates section</p>	[false true] (Boolean) {false}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.lvsRecognizeGates.specified	Boolean controlling perc.lvsRecognizeGates <b>Options</b> page, LVS Recognize Gates checkbox	[false true] (Boolean) {false}
perc.lvsRecognizeGates.xAlso.value	Specifies whether to add the XALSO keyword to the LVS Recognize Gates statement. This includes MOS transistors with X+ subtypes in logic gate recognition.  <b>Options</b> page, LVS Recognize Gates section]	[false true] (Boolean) {false}
perc.lvsReduceGates.mixTypes.value	Specifies whether to add the keyword MIX TYPES to LVS Reduce Split Gates YES.  Enabled when perc.lvsReduceGates.splitGates.value = YES and perc.lvsReduceGates.splitGates.specified = true.  <b>Options</b> page, Gate Reduction section	[false true] (Boolean) {false}
perc.lvsReduceGates.reduceParallelMOS.specified	Boolean controlling perc.lvsReduceGates.reduceParallelMOS  <b>Options</b> page, Gate Reduction section	[false true] (Boolean) {false}
perc.lvsReduceGates.reduceParallelMOS.value	Specify whether to reduce MOS transistors connected in parallel into single transistors. (LVS Reduce Parallel MOS)  NO cannot be specified with perc.lvsReduceGates.splitGates.value = YES.  <b>Options</b> page, Gate Reduction section	[YES NO] () {YES}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.lvsReduceGates.sameOrderSplitGates.value	<p>Specifies whether to add the keyword SAME ORDER to LVS Reduce Split Gates YES. Enabled when perc.lvsReduceGates.splitGates.value = YES and perc.lvsReduceGates.splitGates.specified = true.</p> <p><b>Options</b> page, Gate Reduction section</p>	[false true] (Boolean) {false}
perc.lvsReduceGates.semiSplitGates.value	<p>Specifies whether to add the keyword SEMI ALSO to LVS Reduce Split Gates YES.</p> <p>Enabled when perc.lvsReduceGates.splitGates.value = YES and perc.lvsReduceGates.splitGates.specified = true.</p> <p><b>Options</b> page, Gate Reduction section</p>	[false true] (Boolean) {false}
perc.lvsReduceGates.shortEquivalentNodes.specified	<p>Boolean controlling perc.lvsReduceGates.shortEquivalentNodes</p> <p><b>Options</b> page, Gate Reduction section</p>	[false true] (Boolean) {false}
perc.lvsReduceGates.shortEquivalentNodes.value	<p>Specifies the primary keyword for the LVS Short Equivalent Nodes statement. PARALLEL and SPLIT cannot be specified with perc.lvsReduceGates.splitGates.value = YES.</p> <p><b>Options</b> page, Gate Reduction section</p>	[NO PARALLEL SPLIT] () {NO}
perc.lvsReduceGates.spAlso.value	<p>Specifies whether to add the keyword SP ALSO to LVS Reduce Split Gates YES.</p> <p>Enabled when perc.lvsReduceGates.splitGates.value = YES and perc.lvsReduceGates.splitGates.specified = true.</p> <p><b>Options</b> page, Gate Reduction section</p>	[false true] (Boolean) {false}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.lvsReduceGates.splitGates.specified	Boolean controlling perc.lvsReduceGates.splitGates <b>Options</b> page, Gate Reduction section	[false true] (Boolean) {false}
perc.lvsReduceGates.splitGates.value	Specifies the value for LVS Reduce Split Gates. Several sub-options are enabled when this is set to YES.  In addition, this setting affects the value of perc.lvsReduceGates.reduceParallelMOS.value and perc.lvsReduceGates.shortEquivalentNodes.value.  <b>Options</b> page, Gate Reduction section	[YES NO] () {YES}
perc.lvsReduceGates.withinTolerance.value	Specifies whether to add the keyword WITHIN TOLERANCE to LVS Reduce Split Gates YES  Enabled when perc.lvsReduceGates.splitGates.value = YES and perc.lvsReduceGates.splitGates.specified = true.  <b>Options</b> page, Gate Reduction section	[false true] (Boolean) {false}
perc.percLDLProbePath.probeFile.value	The pathname of a probe mapping file, which defines probe locations for a LDL CD or P2P run.  (PERC LDL PROBE PATH <file>)  <b>Inputs</b> page, PERC LDL Probe Path, “Probe Mapping File”	[] () {}
perc.percLDLProbePath.specified	Boolean controlling perc.percLDLProbePath  <b>Inputs</b> page, PERC LDL Probe Path checkbox	[false true] (Boolean) {false}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.percLDLWaiverPath.descriptionFile.value	The file that contains waiver descriptions to be used during Calibre PERC LDL run.  (PERC LDL WAIVER PATH <file>)  <b>Inputs</b> page, PERC LDL Waiver Path, “Waiver Description File”	[] () {PERC-LDL.waivers}
perc.percLDLWaiverPath.specified	Boolean controlling perc.percLDLWaiverPath  <b>Inputs</b> page, PERC LDL Waiver Path checkbox	[false true] (Boolean) {false}
perc.percReportMaximum.count.value	Specifies the maximum result count for the individual report sections in the PERC Report Maximum statement.  Used when perc.percReportMaximum.upperLimit.value = MAXIMUMCOUNT.  <b>Options</b> page, PERC Report Maximum, “Maximum Result Count”	[] (positive integer) {50}
perc.percReportMaximum.specified	Boolean controlling perc.percReportMaximum  <b>Options</b> page, PERC Report Maximum checkbox	[false true] (Boolean) {false}
perc.percReportMaximum.upperLimit.value	Specifies whether to use the ALL keyword or an integer count in the PERC Report Maximum statement. <ul style="list-style-type: none"> <li>• <b>MAXIMUMCOUNT</b> — Specify a maximum result count with perc.percReportMaximum.count.value.</li> <li>• <b>MAXIMUMALL</b> — Use the ALL keyword.</li> </ul> <b>Options</b> page, PERC Report Maximum, “Upper Limit”	[MAXIMUMCOUNT MAXIMUMALL] () {MAXIMUMCOUNT}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.percReportOption.keywords.value	<p>Specifies a list of one or more keywords that are used in the PERC Report Option statement.</p> <ul style="list-style-type: none"> <li>• ALL_NET_TYPE</li> <li>• NO_NET_TYPE</li> <li>• NO_DEVICE_PIN</li> <li>• MIN_MAX_VOLTAGE_ONLY</li> <li>• NO_NET_VOLTAGE</li> <li>• SKIP_PASSED_CELL</li> <li>• SKIP_RESULT_PROPERTY</li> <li>• SKIP_WAIVED_RESULT</li> <li>• PRINT_TVF_WARNING</li> <li>• TRACK_RESULT_SUBTITLE</li> </ul> <p><b>Options</b> page, PERC Report Option, “Keywords” dropdown</p>	[ALL_NET_TYPE NO_NET_TYPE NO_DEVICE_PIN MIN_MAX_VOLTAGE_ONLY NO_NET_VOLTAGE SKIP_PASSED_CELL SKIP_RESULT_PROPERTY SKIP_WAIVED_RESULT PRINT_TVF_WARNING TRACK_RESULT_SUBTITLE] ( { }
perc.percReportOption.specified	<p>Boolean controlling perc.percReportOption</p> <p><b>Options</b> page, PERC Report Option checkbox</p>	[false true] (Boolean) {false}
perc.percRuleGen.generatorFile.value	<p>The path to the PERC Rule Generator file.</p> <p><b>Options</b> page, Rule Generator, “PERC Rule Generator File”</p>	[] ( {percrulegen.prg}
perc.percWaiverPath.abortUnWaivable.value	<p>Specifies whether to include the ABORT UNWAIVABLE keyword in the PERC WAIVER PATH statement.</p> <p><b>Inputs</b> page, PERC Waiver Path, “Abort UnWaivable”</p>	[false true] (Boolean) {false}
perc.percWaiverPath.descriptionFile.value	<p>The waiver description file for the PERC WAIVER PATH statement.</p> <p><b>Inputs</b> page, PERC Waiver Path, “Waiver Description File”</p>	[] ( {PERC.waivers}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc_percWaiverPath.specified	Boolean controlling perc_percWaiverPath <b>Inputs</b> page, PERC Waiver Path checkbox	[false true] (Boolean) {false}
perc.reports.percReport.value	The filename for the PERC REPORT statement. <b>Outputs</b> page, Reports, “PERC Report”	[] () {perc.report}
perc.reports.percReportViewAfterRun.value	Specifies whether to automatically view the PERC Report after the run is complete. <b>Outputs</b> page, Reports, “View PERC Report after run finishes”	[false true] (Boolean) {false}
perc.reports.percSummaryReport.specified	Boolean controlling perc.reports.percSummaryReport <b>Outputs</b> page, Reports, “PERC Summary Report” checkbox	[false true] (Boolean) {false}
perc.reports.percSummaryReport.value	The filename for the PERC Summary Report statement. <b>Outputs</b> page, Reports, “PERC Summary Report”	[] () {perc.summary.re p}
perc.reports.percSummaryReportViewAfterRun.value	Specifies whether to automatically view the PERC Summary Report after the run is complete. <b>Outputs</b> page, Reports, “View PERC Summary Report after run finishes”	[false true] (Boolean) {false}
perc.rulesFile.value	The rule file for the run. <b>Rules</b> page, “Rules File”	[] () {rules}
perc.runDir.value	The run directory. <b>Rules</b> page, “Run Directory”	[] () {. (current directory)}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.runType.value	Specifies the run type. <b>Inputs</b> page, Run dropdown list	[HIER FLAT LDL] ( {HIER}
perc.runsetPrefs.runsetInfo.value	User-defined text specifying information about the runset. Whitespace is not allowed. <b>Preferences</b> page, Runset section	[] ( {}
perc.runsetPrefs.runsetType.value	The runset type. Specify a user-defined text string. <b>Preferences</b> page, Runset section	[string] ( {}
perc.rve.autoStart.value	Specifies whether to automatically start Calibre RVE with the results from Calibre PERC after the run finishes. <b>Run Control</b> page, RVE Options, “Show Results in RVE”	[true false] (Boolean) {true}
perc.rve.startRVEAfterBatch.value	Specifies whether to automatically start Calibre RVE after a batch run finishes. <b>Run Control</b> page, RVE Options, “Start RVE after batch run”	[false true] (Boolean) {false}
perc.seedPromotion.createReport.specified	Boolean controlling perc.seedPromotion.createReport Specifies whether to include the LVS SHOW SEED PROMOTIONS statement in the control file of Calibre Interactive. <b>Options</b> page, Seed Promotion, “Create Seed Promotions Report” checkbox	[false true] (Boolean) {false}
perc.seedPromotion.createReport.value	Specifies whether to create a seed promotions report named <layout_primary>.seed_promotion. <b>Options</b> page, Seed Promotion, “Create Seed Promotions Report” dropdown	[NO YES] ( {NO}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.seedPromotion.maximum.specified	Boolean controlling perc.seedPromotion.maximum <b>Options</b> page, Seed Promotion, “Maximum polygons per seed promotion” checkbox	[false true] (Boolean) {false}
perc.seedPromotion.maximum.value	Specifies the maximum number of polygons per seed promotion in the report.  LVS SHOW SEED PROMOTIONS MAXIMUM <number>  <b>Options</b> page, Seed Promotion, “Maximum polygons per seed promotion”	[] (positive integer) {50}
perc.source.exportFromSourceViewer.value	Checkbox for “Export from source viewer”.  <b>Inputs</b> page, Source Path section.	[] () {false}
perc.source.extraSourceFiles.specified	Boolean controlling perc.source.extraSourceFiles  <b>Database</b> page, Library section, “Additional SPICE Files” checkbox	[false true] (Boolean) {false}
perc.source.extraSourceFiles.value	A list of additional SPICE netlist files.  Used when perc.source.extraSourceFiles.specified = true.  <b>Database</b> page, Library sections, “Additional SPICE Files”	[] () {}
perc.source.extraVerilogSourceFiles.specified	Boolean controlling perc.source.extraVerilogSourceFiles  <b>Database</b> page, Library section, “Additional VERILOG Files” checkbox	[false true] (Boolean) {false}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.source.extraVerilogSourceFiles.value	A list of additional VERILOG netlist files.  Used when perc.source.extraVerilogSourceFiles.specified = true.  <b>Database</b> page, Library sections, “Additional VERILOG Files”	[] ( {}
perc.source.format.value	The format of the source netlist input files. <ul style="list-style-type: none"> <li>• SPICE — SPICE format</li> <li>• VERILOG — VERILOG format</li> <li>• MIXED — A combination of SPICE and VERILOG files.</li> </ul> <b>Inputs</b> page, Source Path section	[SPICE VERILOG MIXED] ( {SPICE}
perc.source.sourceFile.value	The path to the top level source netlist file.  When perc.source.format.value is SPICE or MIXED, this is the top level SPICE file. When perc.source.format.value is VERILOG, this is the top level VERILOG file. (For MIXED input, the VERILOG file is specified by perc.source.verilogSourceFile.value.)  <b>Inputs</b> page, Source Path section	[] (environment variable valid) {}
perc.source.topCell.value	The name of the primary (top) cell or subcircuit of the source netlist. This is the parameter for the Source Primary statement.  <b>Inputs</b> page, Source Path section	[] ( {}
perc.source.topCellLibrary.value	The library name for the top cell of the source netlist. Used when exporting the source netlist from a schematic design tool such as Cadence Composer.  <b>Inputs</b> page, Source Path section.	[] ( {}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.source.topCellView.value	The view name for the top cell of the source netlist. Used when exporting the source netlist from a schematic design tool such as Cadence Composer.  <b>Inputs</b> page, Source Path section.	[] () {}
perc.source.verilogSourceFile.value	The path to the VERILOG source netlist file when a combination of SPICE and VERILOG netlist input files is used (perc.source.format.value = MIXED)  <b>Inputs</b> page, Source Path section	[] () {}
perc.supply.groundNames.specified	Boolean controlling perc.supply.groundNames  <b>Options</b> page, Power Supply, “Ground Nets” checkbox	[false true] (Boolean) {false}
perc.supply.groundNames.value	A list of ground net names for the LVS Ground Name statement.  Used when perc.supply.groundNames.specified = true.  <b>Options</b> page, Power Supply, “Ground Nets”	[] (list) {}
perc.supply.lvsAbortOnSoftchk.specified	Boolean controlling perc.supply.lvsAbortOnSoftchk  <b>Options</b> page, Power Supply, “LVS Abort On SoftChk” checkbox	[false true] (Boolean) {false}
perc.supply.lvsAbortOnSoftchk.value	Specifies whether to abort processing if a LVS Softchk violation is found.  (LVS Abort On Softchk statement)  <b>Options</b> page, Power Supply, “LVS Abort On SoftChk” checkbox	[NO YES] () {NO}
perc.supply.lvsAbortOnSupplyError.specified	Boolean controlling perc.supply.lvsAbortOnSupplyError  <b>Options</b> page, Power Supply, “LVS Abort on Supply Error” checkbox	[false true] (Boolean) {false}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.supply.lvsAbortOnSupplyError.value	Specifies whether to abort processing when connectivity extraction detects a problem with the power or ground nets.  <b>(LVS Abort On Supply Error statement)</b>  <b>Options page, Power Supply, “LVS Abort On Supply Error”</b>	[YES NO] () {YES}
perc.supply.powerNames.specified	Boolean controlling perc.supply.powerNames  <b>Options page, Power Supply, “Power Nets” checkbox</b>	[false true] (Boolean) {false}
perc.supply.powerNames.value	A list of power net names for the LVS Power Name statement.  Used when perc.supply.powerNames.specified = true.  <b>Options page, Power Supply, “Power Nets”</b>	[] (list) {}
perc.svdb.asciixref.value	Checkbox for “Generate ASCII cross-reference files,” which adds the IXF, NXF, and SLPH keywords to the Mask SVDB Directory statement.  <b>Outputs page, Mask SVDB section</b>	[false true] (Boolean) {false}
perc.svdb.cci.value	Checkbox for “Generate Calibre Connectivity Interface data,” which adds the CCI keyword to the Mask SVDB Directory statement.  <b>Outputs page, Mask SVDB section</b>	[false true] (Boolean) {false}
perc.svdb.dirPath.value	The directory name for the Mask SVDB Directory statement.  <b>Outputs page, Mask SVDB section</b>	[] () {svdb}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.svdb.muwdb.value	Checkbox for “Allow multiple users to waive PERC results in RVE,” which adds the MUWDB keyword to the Mask SVDB Directory statement. <b>Outputs</b> page, Mask SVDB section	[false true] (Boolean) {false}
perc.svdb.pexIncrementalPDB.value	Specify to generate PDBs incrementally. If this setting is not selected, the PDB is recreated every time the PDB generation step is run.	[false true] (Boolean) {false}
perc.svdb.pinloc.specified	Boolean controlling perc.svdb.pinloc <b>Outputs</b> page, Mask SVDB section, “Pin Location” checkbox	[false true] (Boolean) {false}
perc.svdb.pinloc.value	Specifies the PINLOC or NOPINLOC keyword for the Mask SVDB Directory statement. Used when perc.svdb.pinloc.specified = true. <b>Outputs</b> page, Mask SVDB section	[PINLOC NOPINLOC] () {PINLOC}
perc.svdb.query.value	Checkbox for “Generate cross-reference data for RVE,” which adds the QUERY keyword to the Mask SVDB Directory statement. <b>Outputs</b> page, Mask SVDB section	[false true] (Boolean) {false}
perc.svdb.specified	Boolean controlling perc.svdb Specifies whether the to create the SVDB. The SVDB directory name is specified with perc.svdb.dirPath.value. <b>Outputs</b> page, “Mask SVDB”checkbox	[true false] (Boolean) {true}
perc.svdb.xrc.value	Checkbox for “Generate all necessary data for Calibre xRC,” which adds the XRC keyword to the Mask SVDB Directory statement. <b>Outputs</b> page, Mask SVDB section	[false true] (Boolean) {false}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.svrfIncludes.specified	Checkbox for “Include Rule Statements”. The statements are specified with perc.svrfIncludes.svrfStatements.value or perc.svrfIncludes.tvfStatements.value. <b>Options page, Include Rule Statements</b>	[false true] (Boolean) {false}
perc.svrfIncludes.statementType.value	Available only when the main rule file is TVF.  Dropdown selection for “Type”, which specifies the language of the included rule file statements. SVRF statements are included within a VERBATIM block in the generated control file.  <b>Options page, Include Rule Statements</b>	[none SVRF TVF] () {none}
perc.svrfIncludes.svrfStatements.value	SVRF statements to be included in the control file generated by Calibre Interactive.  Used when the main rule file is SVRF, or when the main rule file is TVF and perc.svrfIncludes.statementType.value = SVRF.  <b>Options page, Include Rule Statements</b>	[] () {}
perc.svrfIncludes.tvfStatements.value	TVF statements to be included in the control file generated by Calibre Interactive.  Used when the main rule file is TVF and perc.svrfIncludes.statementType.value = TVF.  <b>Options page, Include Rule Statements</b>	[] () {}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.traceProperty.parameters	<p>A list of lists defining the contents of the table for the option “Trace Property”. This table defines parameters for the for the <a href="#">Trace Property</a> statement.</p> <p>Each sublist defines parameters for one statement. The sublist syntax is:</p> <pre>[ bool, "params" ]   • bool — true or false. Indicates whether the statement is used.   • params — The parameters for the statement.</pre> <p>perc.traceProperty.specified must be true for this option to be used.</p> <p><b>Options</b> page, Trace Property section</p>	[ ] (list) { }
perc.traceProperty.specified	Boolean controlling perc.traceProperty <b>Options</b> page, Trace Property checkbox	[false true] (Boolean) {false}
perc.v2LVSLibraries.sPICEIncFiles.specified	Checkbox for “Include SPICE Files”. Files are specified with perc.v2LVSLibraries.sPICEIncFiles.value.	[false true] (Boolean) {false}
perc.v2LVSLibraries.sPICEIncFiles.value	<p>List of SPICE files that are included in the v2lvs command line with the -s option.</p> <p>Used when perc.v2LVSLibraries.sPICEIncFiles.specified = true.</p> <p><b>V2LVS</b> page, Libraries section</p>	[ ] ( ) { }
perc.v2LVSLibraries.sPICELibFiles.specified	Checkbox for “SPICE Library Files”. Files are specified with perc.v2LVSLibraries.sPICELibFiles.value.	[false true] (Boolean) {false}
	<b>V2LVS</b> page, Libraries section	

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.v2LVSLibraries.sPICELibFiles.value	List of SPICE files that are included in the v2lvs command line with the -lsr or -lsp option. (See perc.v2LVSOPTIONS.useRangeMode.value)  Used when perc.v2LVSLibraries.sPICELibFiles.specified = true.  <b>V2LVS</b> page, Libraries section	[] ( {}
perc.v2LVSLibraries.vLOGLibFiles.specified	Checkbox for “Verilog Library Files”. Files are specified with perc.v2LVSLibraries.vLOGLibFiles.value.  <b>V2LVS</b> page, Libraries section	[false true] (Boolean) {false}
perc.v2LVSLibraries.vLOGLibFiles.value	List of Verilog files that are included in the v2lvs command line with the -l option.  Used when perc.v2LVSLibraries.vLOGLibFiles.specified = true.  <b>V2LVS</b> page, Libraries section	[] ( {}
perc.v2LVSOPTIONS.addPinNames.specified	Boolean controlling perc.v2LVSOPTIONS.addPinNames  Checkbox for “Add pins to each subckt”. The pins are specified with perc.v2LVSOPTIONS.addPinNames.value.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
perc.v2LVSOPTIONS.addPinNames.value	List of pin names that are added with the -addpin argument to the v2lvs command line.  Used when perc.v2LVSOPTIONS.addPinNames.specified = true.  <b>V2LVS</b> page, Options section	[] (list) {}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.v2LVSOptions.connectUPin.s.value	Checkbox for “Create numbered nets for unconnected pins,” which adds the -n argument to the v2lvs command line. <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
perc.v2LVSOptions.dontConnectPower.value	Checkbox for “Don’t connect supply0/supply1 nets to global supply,” which adds the -sk argument to the v2lvs command line. <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
perc.v2LVSOptions.generateTemplate.value	Checkbox for “Generate xRC source template file,” which adds the -t argument to the v2lvs command line. <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
perc.v2LVSOptions.keepBackslash.value	Checkbox for “Preserve backslashes in escaped identifiers,” which adds the -b argument to the v2lvs command line. <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
perc.v2LVSOptions.positionalPins.value	Checkbox for “Use positional pin order (don’t use \$PINS)”, which adds the -i argument to the v2lvs command line. <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
perc.v2LVSOptions.translateAlways.value	Checkbox for “Always translate before running Calibre.” If this setting is false, the translator does not run if the Verilog files are older than the translated SPICE file. <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
perc.v2LVSOptions.useCBLicense.value	Checkbox for “Prefer Calibre-CB license during license search”, which adds the -cb argument to the v2lvs command line. <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.v2LVSOptions.useRangeMode.value	<p>Checkbox for “Use aggregate mode for bus bits”.</p> <p>Specifies how SPICE library files are added on the v2lvs command line:</p> <ul style="list-style-type: none"> <li>• true — Use the -lsr argument, for range mode.</li> <li>• false — Use the -lsp argument, for pin mode.</li> </ul> <p><b>V2LVS</b> page, Options section</p>	[false true] (Boolean) {false}
perc.v2LVSOptions.warningLevel.value	<p>Specifies the amount of warning level output for the v2lvs translator. This is the value for the -w argument.</p> <p><b>V2LVS</b> page, Options section</p>	[] (positive integer) {2}
perc.v2LVSOptions.warningsAreErrors.value	<p>Checkbox for “Treat warnings as errors”, which adds the -werror argument to the v2lvs command line.</p> <p><b>V2LVS</b> page, Options section</p>	[false true] (Boolean) {false}
perc.v2LVSStrings.achars.specified	<p>Checkbox for “Change array delimiters to.” The value is given by perc.v2LVSStrings.achars.value.</p> <p><b>V2LVS</b> page, Strings section</p>	[false true] (Boolean) {false}
perc.v2LVSStrings.achars.value	<p>The value for the -a option in the v2lvs command line, which changes the array delimiter from the default []. The first character replaces [, the optional second character replaces ].</p> <p>Used when perc.v2LVSStrings.achars.specified = true.</p> <p><b>V2LVS</b> page, Strings section</p>	[] () {}
perc.v2LVSStrings.cchars.specified	<p>Checkbox for “Change illegal SPICE characters to.” The value is given by perc.v2LVSStrings.cchars.value.</p> <p><b>V2LVS</b> page, Strings section</p>	[false true] (Boolean) {false}

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.v2LVSStrings.cchars.value	<p>The value for the -c option in the v2lvs command line. Enter one or two characters enclosed in quotes. To include single quotes in the command line, as recommended, enclose the characters in single quotes, then outer double quotes.</p> <p>Used when perc.v2LVSStrings.cchars.specified = true.</p> <p><b>V2LVS</b> page, Strings section</p>	[ ] ( ) { }
perc.v2LVSStrings.pprefix.specified	<p>Checkbox for “Prefix for gate level primitives” The value is given by perc.v2LVSStrings.pprefix.value.</p> <p><b>V2LVS</b> page, Strings section</p>	[false true] (Boolean) { false }
perc.v2LVSStrings.pprefix.value	<p>The value for the -p option in the v2lvs command line, which specifies a prefix to add to Verilog gate level primitive cells.</p> <p>Used when perc.v2LVSStrings.pprefix.specified = true.</p> <p><b>V2LVS</b> page, Strings section</p>	[ ] ( ) { }
perc.v2LVSStrings.s0Net.specified	<p>Checkbox for “Connect pins with 0 value to net.” The value is given by perc.v2LVSStrings.s0Net.value.</p> <p><b>V2LVS</b> page, Strings section</p>	[false true] (Boolean) { false }
perc.v2LVSStrings.s0Net.value	<p>The value for the -s0 option in the v2lvs command line.</p> <p>Used when perc.v2LVSStrings.s0Net.specified = true.</p> <p><b>V2LVS</b> page, Strings section</p>	[ ] ( ) { }

**Table C-4. PERC Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
perc.v2LVSStrings.s1Net.specified	Checkbox for “Connect pins with 1 value to net.” The value is given by perc.v2LVSStrings.s1Net.value. <b>V2LVS</b> page, Strings section	[false true] (Boolean) {false}
perc.v2LVSStrings.s1Net.value	The value for the -s1 option in the v2lvs command line.  Used when perc.v2LVSStrings.s1Net.specified = true.  <b>V2LVS</b> page, Strings section	[] () {}
perc.v2LVSStrings.uprefix.specified	Checkbox for “Prefix for unnamed pins and gates.” The value is given by perc.v2LVSStrings.uprefix.value. <b>V2LVS</b> page, Strings section	[false true] (Boolean) {false}
perc.v2LVSStrings.uprefix.value	The value for the “Prefix for unnamed pins and gates” option, which adds the -u argument to the v2lvs command line.  Used when perc.v2LVSStrings.uprefix.specified = true.  <b>V2LVS</b> page, Strings section	[] () {}
perc.v2LVSStrings.userOptions.specified	Checkbox for “More v2lvs program options.” The value is given by perc.v2LVSStrings.userOptions.value. <b>V2LVS</b> page, Strings section	[false true] (Boolean) {false}
perc.v2LVSStrings.userOptions.value	A list of options that are included without modification in the v2lvs command line.  Used when perc.v2LVSStrings.userOptions.specified = true.  <b>V2LVS</b> page, Strings section	[] (list) {}

# PEX Runset Options

The Calibre Interactive PEX-specific runset options have the prefix xrc.

**Table C-5. PEX Runset Options**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.configPrefs.configFiles.value	The file names of the Configuration Files.  <b>Preferences</b> page, Configuration section, “Configuration Files”	[] ( {}
xrc.configPrefs.specified	Boolean controlling xrc.configPrefs  <b>Preferences</b> page, Configuration section	[false true] (Boolean) {false}
xrc.envVars.setEnv(Name, Value)	Specifies an environment variable that is set in the runset. The format is:  xrc.envVars.setEnv(“name”, “value”)  See the <a href="#">envVars</a> configuration file command.  <b>Environment</b> page	[] ( {}
xrc.envVars.setEnvNamePattern(Pattern)	Specifies a filter for the environment variables displayed on the <b>Environment</b> page. The format is:  xrc.envVars.setEnvNamePattern(“pattern”)  <b>Environment</b> page	[] ( {*}
xrc.envVars.unsetEnv(Name)	Specifies an environment variable that is unset in the runset. The format is:  xrc.envVars.unsetEnv(“name”)  See the envVars configuration file command.  <b>Environment</b> page	[] ( {}
xrc.erc.execute.specified	Boolean controlling xrc.erc.execute  <b>LVS</b> page, “LVS Execute ERC” checkbox	[true false] (Boolean) {true}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.erc.execute.value	Specifies YES or NO for the LVS EXECUTE ERC statement. <b>LVS</b> page, “LVS Execute ERC” dropdown	[YES NO] () {YES}
xrc.ercMaximumResults.count.value	Specifies the maximum result count for an individual rule check in ERC. Used when xrc.ercMaximumResults.upperLimit.value = MAXIMUMCOUNT. <b>LVS</b> page, ERC Maximum Results, “Count”	[] (positive integer) {1000}
xrc.ercMaximumResults.specified	Boolean controlling xrc.ercMaximumResults <b>LVS</b> page, “ERC Maximum Results” checkbox	[false true] (Boolean) {false}
xrc.ercMaximumResults.upperLimit.value	Specifies whether to use the ALL keyword or an integer count in the ERC MAXIMUM RESULTS statement. <ul style="list-style-type: none"> <li>• <b>MAXIMUMCOUNT</b> — Specify a maximum result count with xrc.ercMaximumResults.count.value.</li> <li>• <b>MAXIMUMALL</b> — Use the ALL keyword.</li> </ul> <b>LVS</b> page, ERC Maximum Results, “Upper Limit” dropdown	[MAXIMUMCOUNT MAXIMUMALL] () {MAXIMUMCOUNT}
xrc.ercMaximumVertex.count.value	Specifies an upper limit for the vertex count of any result polygon in ERC. Used when xrc.ercMaximumVertex.upperLimit.value = MAXIMUMCOUNT. <b>LVS</b> page, ERC Maximum Vertex, Count	[] (integer) {4096}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.ercMaximumVertex.specified	Boolean controlling xrc.ercMaximumVertex <b>LVS</b> page, “ERC Maximum Vertex” checkbox	[false true] (Boolean) {false}
xrc.ercMaximumVertex.upperLimit.value	Specifies whether to use the ALL keyword or an integer count in the ERC MAXIMUM VERTEX statement. <ul style="list-style-type: none"> <li>• <b>MAXIMUMCOUNT</b> — Specify a maximum vertex count for a result polygon with xrc.ercMaximumVertex.count.value.</li> <li>• <b>MAXIMUMALL</b> — Use the ALL keyword.</li> </ul> <b>LVS</b> page, ERC Maximum Vertex, “Upper Limit”	[MAXIMUMC OUNT MAXIMUMAL L] ( {MAXIMUMC OUNT}
xrc.ercSummaryReport.report.specified	Boolean controlling xrc.ercSummaryReport.report <b>LVS</b> page, ERC Summary Report, “File” checkbox	[false true] (Boolean) {false}
xrc.ercSummaryReport.report.value	The filename for the ERC Summary Report statement. <b>LVS</b> page, ERC Summary Report, “File”	[] ( {erc.summary}
xrc.ercSummaryReport.reportAccess.specified	Boolean controlling xrc.ercSummaryReport.reportAccess <b>LVS</b> page, ERC Summary Report, “Previous Contents” checkbox	[false true] (Boolean) {false}
xrc.ercSummaryReport.reportAccess.value	Specifies whether to overwrite or append to previous contents of the summary file. <b>LVS</b> page, ERC Summary Report, “Previous Contents” dropdown	[REPLACE APPEND] ( {REPLACE}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.ercSummaryReport.reportKeywords.specified	Boolean controlling xrc.ercSummaryReport.reportKeywords  <b>LVS</b> page, ERC Summary Report, “Keywords” checkbox	[false true] (Boolean) {false}
xrc.ercSummaryReport.reportKeywords.value	Specifies to use the HIER keyword in the ERC Summary Report statement.  <b>LVS</b> page, ERC Summary Report, “Keywords” dropdown	[HIER] () {}
xrc.ercSummaryReport.reportViewAfterRun.value	Specifies whether to automatically view the ERC Summary Report after the run is complete.  <b>LVS</b> page, ERC Summary Report, “View ERC summary Report after run finishes” checkbox	[false true] (Boolean) {false}
xrc.ercdb.resultsCellName.value	Specifies whether to output hierarchical ERC errors in their context cells (true) or in the top cell (false).  The true setting adds YES CELL SPACE XFORM ALL to the ERC Cell Name statement.  <b>LVS</b> page, ERC Results Database, “Output cell errors in cell space” checkbox	[true false] (Boolean) {true}
xrc.ercdb.resultsFile.value	The name of the ERC Results Database.  <b>LVS</b> page, ERC Results Database, “File”	[] () {erc.results}
xrc.ercdb.resultsPseudoCells.specified	Boolean controlling xrc.ercdb.resultsPseudoCells  <b>LVS</b> page, ERC Results Database, “Psuedocells results” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.ercdb.resultsPseudoCells.value	Specifies the handling of pseudocells in ERC results. The options are keywords in the ERC Results Database statement.  <b>LVS</b> page, ERC Results Database, “Psuedocells results” dropdown	[PSEUDO TOP] ( {PSEUDO}
xrc(fmtMode.options(fmtAllowGroupCoupling).value	Boolean that specifies to ground all parasitic coupling capacitors.  <b>Outputs</b> page, Formatter section	[false true] (Boolean) {false}
xrc(fmtMode.options(fmtInfo).value	Boolean that specifies to display informational messages from the formatter.  <b>Outputs</b> page, Formatter section	[false true] (Boolean) {false}
xrc(fmtMode.options(fmtSelectedNets).value	Boolean that controls whether to specify the parasitics model per net.  <b>Outputs</b> page, Formatter section	[false true] (Boolean) {false}
xrc(fmtMode.options(fmtWarnings).value	Boolean that specifies to display warnings from the formatter.  <b>Outputs</b> page, Formatter section	[false true] (Boolean) {false}
xrc(fmtMode.rcMode.value	Specifies the type of parasitics to extract. <ul style="list-style-type: none"> <li>• r — Resistance</li> <li>• c — Capacitance and coupling capacitance</li> <li>• all — All parasitics</li> <li>• simple — No resistance or capacitance</li> <li>• adms — ADMS extraction run</li> </ul>	[r c all simple adms] ( {all}
xrc.hCells.automatch.value	Boolean that controls the “Match cells by name” checkbox. Adds -automatch to the Calibre LVS command line.  <b>Inputs</b> page, H-Cells section	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.hCells.hCellsFile.specified	Boolean controlling xrc.hCells.hCellsFile <b>Inputs</b> page, H-Cells section	[false true] (Boolean) {false}
xrc.hCells.hCellsFile.value	The hcells file. <b>Inputs</b> page, H-Cells section	[] () {hcells}
xrc.includes.specified	Boolean controlling xrc.includes <b>Options</b> page, Include Rules Files section	[false true] (Boolean) {false}
xrc.includes.ruleFiles.value	The file names of the additional rules files. <b>Options</b> page, Include Rules Files section	[] () {}
xrc.layout.exportFromLayoutViewer.value	Boolean that controls the “Export from layout viewer” checkbox. <b>Inputs</b> page, Layout Path section	[] () {false}
xrc.layout.extraGoldenLayoutFiles.specified	Boolean controlling xrc.layout.extraGoldenLayoutFiles <b>Database</b> page, Library section	[false true] (Boolean) {false}
xrc.layout.extraGoldenLayoutFiles.value	The file names of the additional golden layout files. <b>Database</b> page, Library section	[] () {}
xrc.layout.extraLayoutFiles.specified	Boolean controlling xrc.layout.extraLayoutFiles <b>Database</b> page, Library section	[false true] (Boolean) {false}
xrc.layout.extraLayoutFiles.value	The file names of the additional layout files. <b>Database</b> page, Library section	[] () {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.layout.format.value	The format of the layout database. <b>Inputs</b> page, Layout Path section  Note: If the layout format is OPENACCESS, see cmn.layout.layoutLibraryFDI.value, cmn.layout.topCellLibraryFDI.value, and cmn.layout.topCellViewFDI.value.	[GDSII OASIS OPENACCESS ] () {GDSII}
xrc.layout.layoutFile.value	The file name of the layout file for GDS or OASIS format layouts. <b>Inputs</b> page, Layout Path section	[] (environment variable valid) {}
xrc.layout.layoutNetlist.value	The name of the extracted netlist file. <b>Inputs</b> page, Layout Path section	[] (environment variable valid) {lay.net}
xrc.layout.topCell.value	The name of the layout top cell. <b>Inputs</b> page, Layout Path section	[] () {}
xrc.layout.topCellLibrary.value	The library name of the layout top cell to run Calibre on. Used only when the GUI is invoked from Cadence Virtuoso and “Export from layout viewer” is checked.  <b>Inputs</b> page, Layout Path section	[] () {}
xrc.layout.topCellView.value	The cell view name of the layout top cell to run Calibre on. Used only when the GUI is invoked from Cadence Virtuoso and “Export from layout viewer” is checked.  <b>Inputs</b> page, Layout Path section	[] () {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.layoutCase.caseSensitive.specified	Boolean controlling xrc.layoutCase.caseSensitive <b>Options</b> page, “Layout Case” checkbox	[false true] (Boolean) {false}
xrc.layoutCase.caseSensitive.value	Specifies whether the layout should be processed in a case-sensitive manner. <b>Options</b> page, “Layout Case” checkbox	[NO YES] () {NO}
xrc.licensing.runHierLVS.value	Boolean that controls whether to run the hierarchical version of LVS. This setting is ignored if xrc.licensing.useCBForLVS.value = true. <b>Run Control</b> page, Calibre Licensing section	[true false] (Boolean) {true}
xrc.licensing.useCBForLVS.value	Boolean that controls whether the Calibre CB license is used for the Calibre nmLVS run. <b>Run Control</b> page, Calibre Licensing section	[false true] (Boolean) {false}
xrc.licensing.useCBForPEX.value	Boolean that controls whether the Calibre CB license is used for the PEX run. <b>Run Control</b> page, Calibre Licensing section	[false true] (Boolean) {false}
xrc.lvsBlackBoxPort.specified	Boolean controlling xrc.lvsBlackBoxPort Available when xrc.lvsBox.specified is true. <b>Options</b> page, “LVS Black Box Port” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.lvsBlackBoxPort.parameters	<p>A list of lists defining the contents of the table for the option “LVS Black Box Port”. This table defines parameters for the LVS Black Box Port statement.</p> <p>Each sublist defines parameters for one statement. The sublist syntax is:</p> <pre>[ bool, "val1 val2 val3" ]</pre> <ul style="list-style-type: none"> <li>• bool — true or false. Indicates whether the statement is used.</li> <li>• val1 — The original layer that forms the port.</li> <li>• val2 — The text layer.</li> <li>• val3 — The interconnect layer.</li> </ul> <p>Both xrc.lvsBlacBox.specified and xrc.lvsBlackBoxPort.specifies must be true for this option to be used.</p> <p><b>Options</b> page, LVS Black Box Port section</p>	[ (list) {}]
xrc.lvsBlackBoxPortDepth.specified	<p>Boolean controlling xrc.lvsBlackBoxPortDepth Available when xrc.lvsBlackBoxPort.specified is true.</p> <p><b>Options</b> page, “LVS Black Box Port Depth” checkbox</p>	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.lvsBlackBoxPortDepth.hierarchyLevel.value	<p>Setting for the “Hierarchy Level” option, which specifies the keyword used in the LVS Black Box Port Depth statement.</p> <ul style="list-style-type: none"> <li>• DEPTHPRIMARY — PRIMARY keyword</li> <li>• DEPTHALL — ALL keyword</li> <li>• DEPTHLEVEL — Specify a level with xrc.BlackBoxPortDepth.level.value</li> </ul> <p>Used when xrc.lvsBlackBox.specified, xrc.lvsBlackBoxPort.specified, and xrc.lvsBlackBoxPortDepth.specified are all true.</p> <p><b>Options</b> page, LVS Black Box Port section</p>	[DEPTHPRIMARY DEPTHALL DEPTHLEVEL] ( {DEPTHPRIMARY}
xrc.lvsBlackBoxPortDepth.level.value	<p>Specifies the hierarchical level as a non-negative integer. 0 means top level. Used when xrc.lvsBlackBoxPortDepth.hierarchyLevel.value = DEPTHLEVEL.</p> <p><b>Options</b> page, LVS Black Box Port section</p>	[] (positive integer) {0}
xrc.lvsBox.specified	<p>Boolean controlling xrc.lvsBox</p> <p><b>Options</b> page, “LVS Box” checkbox</p>	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.lvsBox.parameters	<p>A list of lists defining the contents of the table for the option “LVS Box”. This table defines parameters for the LVS Box statement.</p> <p>Each sublist defines parameters for one statement. The sublist syntax is:</p> <pre>[ bool, "option_string" ] • bool — true or false. Indicates whether the statement is used. • option_string — The parameters for the statement.</pre> <p><b>Options</b> page, LVS Box section</p>	[ ] (list) { }
xrc.lvsFilterUnusedOption.specified	<p>Boolean controlling xrc.lvsFilterUnusedOption</p> <p><b>Options</b> page, “LVS Filter Unused Option” checkbox</p>	[false true] (Boolean) {false}
xrc.lvsFilterUnusedOption.parameters	<p>Controls the filtering of unused devices.</p> <p><b>Options</b> page, LVS Filter Unused Option section</p>	[ ] (list) { }
xrc.lvsRecognizeGates.specified	<p>Boolean controlling xrc.lvsRecognizeGates</p> <p>When true, the LVS Recognize Gates statement is written to the control file.</p> <p><b>Options</b> page, “LVS Recognize Gate” checkbox</p>	[false true] (Boolean) {false}
xrc.lvsRecognizeGates.gatesType.value	<p>Specifies the types of gates that the tool recognizes. The choices correspond to keywords for the LVS Recognize Gates statement.</p> <p><b>Options</b> page, LVS Recognize Gates section, “Recognize” dropdown</p>	[ALL SIMPLE NONE] ( ) {ALL}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.lvsRecognizeGates.mixSubType.s.value	<p>Specifies whether to allow subtype mixing of transistors in series-parallel structures. Adds the MIX SUBTYPES keyword.</p> <p><b>Options</b> page, LVS Recognize Gates section, “Mix Subtypes” checkbox</p>	[false true] (Boolean) {false}
xrc.lvsRecognizeGates.xAlso.value	<p>Specifies whether to allow MOS transistors with X+ subtypes to be included. Adds the XALSO keyword.</p> <p><b>Options</b> page, LVS Recognize Gates section, “Include X+ Subtypes” checkbox</p>	[false true] (Boolean) {false}
xrc.lvsReduceGates.mixTypes.value	<p>Specifies that split gate structure contains transistors with different component types, pins, pin names or pin swappability. Adds the MIXTYPES keyword to LVS Reduce Split Gates YES.</p> <p>Only available when xrc.lvsReduceGates.splitGates.value is YES and xrc.lvsReduceGates.splitGates.specified is true.</p> <p><b>Options</b> page, Gate Reduction section, “Split gates contain mixed types” checkbox.</p>	[false true] (Boolean) {false}
xrc.lvsReduceGates.reduceParallelMOS.specified	<p>Boolean controlling xrc.lvsReduceGates.reduceParallelMOS</p> <p><b>Options</b> page, Gate Reduction section, “LVS Reduce Parallel MOS” checkbox.</p>	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.lvsReduceGates.reduceParallelMOS.value	<p>Specify whether to reduce MOS transistors connected in parallel into single transistors.</p> <p>NO cannot be specified with xrc.lvsReduceGates.splitGates.value = YES.</p> <p><b>Options</b> page, Gate Reduction section, “LVS Reduce Parallel MOS” dropdown</p>	[YES NO] ( {YES}
xrc.lvsReduceGates.sameOrderSplitGates.value	<p>Specifies to reduce split gates only when the input order is the same in all transistor strings that form the split gate. Adds the keyword SAME ORDER to LVS Reduce Split Gates YES.</p> <p>Only available when xrc.lvsReduceGates.splitGates.value = YES and xrc.lvsReduceGates.splitGates.specified = true.</p> <p><b>Options</b> page, Gate Reduction section, “Split only when the input order is same” checkbox</p>	[false true] (Boolean) {false}
xrc.lvsReduceGates.semiSplitGates.value	<p>Specifies to reduce semi-split gate structures in addition to reducing fully split gates. Adds the keyword SEMI ALSO to LVS Reduce Split Gates YES.</p> <p>Only available when xrc.lvsReduceGates.splitGates.value = YES and xrc.lvsReduceGates.splitGates.specified = true.</p> <p><b>Options</b> page, Gate Reduction section, “Semi-split gates” checkbox</p>	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.lvsReduceGates.shortEquivalentNodes.specified	Boolean controlling xrc.lvsReduceGates.shortEquivalentNodes <b>Options</b> page, Gate Reduction section, “LVS Short Equivalent Nodes” dropdown	[false true] (Boolean) {false}
xrc.lvsReduceGates.shortEquivalentNodes.value	Specifies whether to short together equivalent nodes in MOSFET split gate structure. Controls the primary keyword for the LVS Short Equivalent Nodes statement.  PARALLEL and SPLIT keywords cannot be specified with xrc.lvsReduceGates.splitGates.value = YES.  <b>Options</b> page, Gate Reduction section, “LVS Short Equivalent Nodes” dropdown	[NO PARALLEL SPLIT] ( {NO}
xrc.lvsReduceGates.spAlso.value	Specifies series-parallel split gate reduction is performed in addition to regular split-gate reduction. Adds the keyword SP ALSO to LVS Reduce Split Gates YES.  Only available when xrc.lvsReduceGates.splitGates.value = YES and xrc.lvsReduceGates.splitGates.specified = true.  <b>Options</b> page, Gate Reduction section, “Split series-parallel gates” checkbox	[false true] (Boolean) {false}
xrc.lvsReduceGates.splitGates.specified	Boolean controlling xrc.lvsReduceGates.splitGates <b>Options</b> page, Gate Reduction section, “LVS Reduce Split Gates” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.lvsReduceGates.splitGates.value	<p>Specifies whether to reduce MOS split gates formed as serial-up or serial-down structures into single gate structures.</p> <p><b>Options</b> page, Gate Reduction section, “LVS Reduce Split Gates” checkbox</p>	[YES NO] ( {YES}
xrc.lvsReduceGates.withinTolerance.value	<p>Specifies split-gate reduction should not occur in cases where the tolerance value in an LVS Split Gate Ratio statement would report a discrepancy. Adds the keyword WITHIN TOLERANCE to LVS Reduce Split Gates YES.</p> <p>Only available when xrc.lvsReduceGates.splitGates.value = YES and xrc.lvsReduceGates.splitGates.specified = true.</p> <p><b>Options</b> page, Gate Reduction section, “Do not split when tolerance reports discrepancy” checkbox</p>	[false true] (Boolean) {false}
xrc.lvsReportOption.specified	<p>Boolean controlling xrc.lvsReportOption</p> <p><b>Options</b> page, “LVS Report Option” checkbox</p>	[false true] (Boolean) {false}
xrc.lvsReportOption.optionKeywords.value	<p>A list specifying the option keywords used for the LVS Report Option statement. Each option is quoted; for example:</p> <pre>xrc.lvsReportOption.optionKeywords.value = [ "A:", "B" ]</pre> <p>You can specify [“All”] to set all options; however, this greatly increases the report output.</p> <p><b>Options</b> page, LVS Report Option section, “Keywords” dropdown</p>	[A B C D E EB EC EO ES F G H I N O P R S V W X BX AV BV CV E1 FX LPE MC NCA NE NOK NP RA RD SP SPE XR] ( {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexBA.specified	Boolean controlling xrc.pexBA <b>Options</b> page, “PEX BA” checkbox	[false true] (Boolean) {false}
xrc.pexBA.calibreView.value	Checkbox for “Suppress LPE values from CalibreView netlist”. When enabled, this adds the CALIBREVIEW keyword to the PEX BA Mapfile statement. <b>Options</b> page, PEX BA section	[false true] (Boolean) {false}
xrc.pexBA.mapFile.value	Specifies the path to the back annotation map file for the PEX BA Mapfile statement. <b>Options</b> page, PEX BA section	[] () {}
xrc.pexCellMap.specified	Boolean controlling xrc.pexCellMap. Available when the output PEX netlist format is set to CALIBREVIEW. <b>Options</b> page, “CALIBREVIEW Cellmap Files” checkbox	[false true] (Boolean) {false}
xrc.pexCellMap.mapFiles.value	The list of cellmap files for Calibre View generation.. <b>Options</b> page, CALIBREVIEW Cellmap Files section	[] () {}
xrc.pexCornerCommand.corners.specified	Boolean controlling xrc.pexCornerCommand.corners <b>Outputs</b> page, “Corners” checkbox	[false true] (Boolean) {false}
xrc.pexCornerCommand.corners.value	The list of process corners. Adds the -corner option to the Calibre PEX command line. For example: [“typical”, “other”] <b>Outputs</b> page, Corners value	[] (list) {}
xrc.pexCreateDRTFile.specified	Boolean controlling xrc.pexCreateDRTFile <b>Options</b> page, “Generate Driver/Receiver File” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexCreateDRTFile.drtFile.value	Specifies the path to of the file containing driver/receiver information for RC extraction.  <b>Options</b> page, Generate Driver/Receiver File section	[] ( {driver.xls}
xrc.pexCreateDRTFile.filterTags.value	A list of values that specifies how to generate a driver/receiver file.  <b>Options</b> page, Generate Driver/Receiver File section	[] (list) {}
xrc.pexExcludeNets.layoutRecursiveNets.specified	Boolean controlling xrc.pexExcludeNets.layoutRecursiveNets  <b>Options</b> page, PEX Extract Exclude Nets section, “Exclude Recursive Layout Nets” checkbox	[false true] (Boolean) {false}
xrc.pexExcludeNets.layoutRecursiveNets.value	The list of nets for the statement PEX EXTRACT EXCLUDE LAYOUTNAMES RECURSIVE.  <b>Options</b> page, PEX Extract Exclude Nets section, Exclude Recursive Layout Nets	[] (list) {}
xrc.pexExcludeNets.layoutToplevelNets.specified	Boolean controlling xrc.pexExcludeNets.layoutToplevelNets  <b>Options</b> page, PEX Extract Exclude Nets section, “Exclude Toplevel Layout Nets” checkbox	[false true] (Boolean) {false}
xrc.pexExcludeNets.layoutToplevelNets.value	The list of nets for the statement PEX EXTRACT EXCLUDE LAYOUTNAMES TOLEVEL.  <b>Options</b> page, PEX Extract Exclude Nets section, Exclude Toplevel Layout Nets	[] (list) {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexExcludeNets.sourceRecursiveNets.specified	Boolean controlling xrc.pexExcludeNets.sourceRecursiveNets <b>Options</b> page, PEX Extract Exclude Nets section, “Exclude Recursive Source Nets” checkbox	[false true] (Boolean) {false}
xrc.pexExcludeNets.sourceRecursiveNets.value	The list of nets for the statement PEX EXTRACT EXCLUDE SOURCenames RECURSIVE. <b>Options</b> page, PEX Extract Exclude Nets section, Exclude Recursive Source Nets	[] (list) {}
xrc.pexExcludeNets.sourceToplevelNets.specified	Boolean controlling xrc.pexExcludeNets.sourceToplevelNets <b>Options</b> page, PEX Extract Exclude Nets section, “Exclude Recursive Toplevel Nets” checkbox	[false true] (Boolean) {false}
xrc.pexExcludeNets.sourceToplevelNets.value	The list of nets for the statement PEX EXTRACT EXCLUDE SOURCenames TOLEVEL. <b>Options</b> page, PEX Extract Exclude Nets section, Exclude Toplevel Source Nets	[] (list) {}
xrc.pexExtractFloatingNets.specified	Boolean controlling xrc.pexExtractFloatingNets <b>Options</b> page, “PEX Extract Floating Nets” checkbox	[false true] (Boolean) {false}
xrc.pexExtractFloatingNets.ignorePortText.value	Specifies to ignore port text placed on floating nets during extraction. Adds the IGNORE_PORT_TEXT keyword to the PEX Extract Floating Nets statement. <b>Options</b> page, PEX Extract Floating Nets section, “Ignore Port Text” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexExtractFloatingNets.netsType.value	Specifies how to treat floating nets. Adds the primary keyword to the PEX Extract Floating Nets statement. <b>Options</b> page, PEX Extract Floating Nets section	[GROUNDED ALL REDUCED “REDUCED DEPTH”] ( {GROUNDED})
xrc.pexExtractFloatingNets.scaleFactor.specified	Boolean controlling xrc.pexExtractFloatingNets.scaleFactor <b>Options</b> page, PEX Extract Floating Nets section, “Scale Factor” checkbox	[false true] (Boolean) {false}
xrc.pexExtractFloatingNets.scaleFactor.value	Specifies the scale of the capacitance values between floating and signal nets. The value must be greater than 0. Only used when xrc.pexExtractFloatingNets.netsType.value is GROUNDED and xrc.pexExtractFloatingNets.scaleFactor.specified is true. <b>Options</b> page, PEX Extract Floating Nets section, Scale Factor	[] ( {1.0})
xrc.pexExtractionType.inductanceMode.value	Specifies the type of parasitic Inductance to extract: <ul style="list-style-type: none"> <li>• “” — No inductance extracted</li> <li>• “I” — Self inductance</li> <li>• “Im” — Self and mutual inductance</li> </ul> <b>Outputs</b> page, Extraction Type section, “Inductance” dropdown	[“” “I” “Im”] ( {})

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexExtractionType.level.value	<p>Specifies the type of parasitic netlist to generate:</p> <ul style="list-style-type: none"> <li>• T — Transistor Level</li> <li>• G — Gate Level</li> <li>• H — Hierarchical</li> <li>• A — ADMS</li> </ul> <p>Generation of Gate Level, Hierarchical, and ADMS netlists requires a list of X-Cells in the X-Cells file field on the <b>Inputs</b> page.</p> <p><b>Outputs</b> page, Extraction Type section, “Level” dropdown</p>	[T G H A] ( {T}
xrc.pexExtractionType.options.pdbInfo.value	<p>Boolean that controls the “Display PDB THRESHOLDING information” checkbox. Enables the -pdb_info switch of the PDB stage.</p> <p><b>Outputs</b> page, Extraction Type section, Options</p>	[false true] (Boolean) {false}
xrc.pexExtractionType.options.selectNets.value	<p>Specifies to exclusively extract nets using the net names specified with the PEX Extract Include SVRF statement.</p> <p><b>Outputs</b> page, Extraction Type section, Options</p>	[SELECT CSELECT] ( {SELECT}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexExtractionType.rcMode.value	<p>Specifies the type of parasitics to extract:</p> <ul style="list-style-type: none"> <li>• R — Resistance</li> <li>• C — Capacitance + Coupling Capacitance</li> <li>• RC — Resistance + Capacitance</li> <li>• RCC — Resistance + Capacitance + Coupling Capacitance</li> <li>• simple — No Resistance or Capacitance</li> <li>• ADMS — ADMS extraction run</li> </ul> <p><b>Outputs</b> page, Extraction Type section, “Resistance/Capacitance” dropdown</p>	[r c rc rcc simple adms] () {rcc}
xrc.pexFSMExcludeNets.rec200LayoutNames.specified	<p>Boolean controlling xrc.pexFSMExcludeNets.rec200LayoutNames</p> <p><b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Recursive Layout net(s) 200” checkbox</p>	[false true] (Boolean) {false}
xrc.pexFSMExcludeNets.rec200LayoutNames.value	<p>A list of layout nets to be extracted with 200 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 200 EXCLUDE_NETS LAYOUTNAMES RECURSIVE.</p> <p><b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Recursive Layout net(s) 200</p>	[] (list) {}
xrc.pexFSMExcludeNets.rec200SourceNames.specified	<p>Boolean controlling xrc.pexFSMExcludeNets.rec200SourceNames</p> <p><b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Recursive Source net(s) 200” checkbox</p>	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexFSMExcludeNets.rec200SourceNames.value	A list of source nets to be extracted with 200 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 200 EXCLUDE_NETS SOURCENAMES RECURSIVE.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Recursive Source net(s) 200	[] (list) {}
xrc.pexFSMExcludeNets.rec600LayoutNames.specified	Boolean controlling xrc.pexFSMExcludeNets.rec600LayoutNames  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Recursive Layout net(s) 600” checkbox	[false true] (Boolean) {false}
xrc.pexFSMExcludeNets.rec600LayoutNames.value	A list of layout nets to be extracted with 600 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 600 EXCLUDE_NETS LAYOUTNAMES RECURSIVE.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Recursive Layout net(s) 600	[] (list) {}
xrc.pexFSMExcludeNets.rec600SourceNames.specified	Boolean controlling xrc.pexFSMExcludeNets.rec600SourceNames  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Recursive Source net(s) 600” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexFSMExcludeNets.rec600SourceNames.value	A list of source nets to be extracted with 600 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 600 EXCLUDE_NETS SOURCENAMES RECURSIVE.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Recursive Source net(s) 600	[] (list) {}
xrc.pexFSMExcludeNets.top200LayoutNames.specified	Boolean controlling xrc.pexFSMExcludeNets.top200LayoutNames  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Toplevel Layout net(s) 200” checkbox	[false true] (Boolean) {false}
xrc.pexFSMExcludeNets.top200LayoutNames.value	A list of layout nets to be extracted with 200 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 200 EXCLUDE_NETS LAYOUTNAMES TOLEVEL.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Toplevel Layout net(s) 200	[] (list) {}
xrc.pexFSMExcludeNets.top200SourceNames.specified	Boolean controlling xrc.pexFSMExcludeNets.top200SourceNames  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Toplevel Source net(s) 200” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexFSMExcludeNets.top200SourceNames.value	A list of source nets to be extracted with 200 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 200 EXCLUDE_NETS SOURCENAMES TOLEVEL.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Toplevel Source net(s) 200	[] (list) {}
xrc.pexFSMExcludeNets.top600LayoutNames.specified	Boolean controlling xrc.pexFSMExcludeNets.top600LayoutNames  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Toplevel Layout net(s) 600” checkbox	[false true] (Boolean) {false}
xrc.pexFSMExcludeNets.top600LayoutNames.value	A list of layout nets to be extracted with 600 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 600 EXCLUDE_NETS LAYOUTNAMES TOLEVEL.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Toplevel Layout net(s) 600	[] (list) {}
xrc.pexFSMExcludeNets.top600SourceNames.specified	Boolean controlling xrc.pexFSMExcludeNets.top600SourceNames  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Toplevel Source net(s) 600” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexFSMExcludeNets.top600SourceNames.value	A list of source nets to be extracted with 600 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 600 EXCLUDE_NETS SOURCENETS TOPLEVEL.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Toplevel Source net(s) 600	[] (list) {}
xrc.pexFSMLayers.layers200.specified	Boolean controlling xrc.pexFSMLayers.layers200  <b>FieldSolver</b> page, PEX Fieldsolver Mode Layers section, “Layers (200)” checkbox	[false true] (Boolean) {false}
xrc.pexFSMLayers.layers200.value	A list of layers to be extracted with 200 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 200 LAYERS.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Layers section, Layers (200)	[] (list) {}
xrc.pexFSMLayers.layers600.specified	Boolean controlling xrc.pexFSMLayers.layers600  <b>FieldSolver</b> page, PEX Fieldsolver Mode Layers section, “Layers (600)” checkbox	[false true] (Boolean) {false}
xrc.pexFSMLayers.layers600.value	A list of layers to be extracted with 600 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 600 LAYERS.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Layers section, Layers (600)	[] (list) {}
xrc.pexFSMNets.rec200LayoutNames.specified	Boolean controlling xrc.pexFSMNets.rec200LayoutNames  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Recursive Layout net(s) 200” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexFSMNets.rec200LayoutNames.value	A list of layout nets to be extracted with 200 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 200 EXCLUDE_NETS LAYOUTNAMES RECURSIVE. <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Recursive Layout net(s) 200	[] (list) {}
xrc.pexFSMNets.rec200SourceNames.specified	Boolean controlling xrc.pexFSMNets.rec200SourceNames <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Recursive Source net(s) 200” checkbox	[false true] (Boolean) {false}
xrc.pexFSMNets.rec200SourceNames.value	A list of source nets to be extracted with 200 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 200 EXCLUDE_NETS SOURCENAMES RECURSIVE. <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Recursive Source net(s) 200	[] (list) {}
xrc.pexFSMNets.rec600LayoutNames.specified	Boolean controlling xrc.pexFSMNets.rec600LayoutNames <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Recursive Layout net(s) 600” checkbox	[false true] (Boolean) {false}
xrc.pexFSMNets.rec600LayoutNames.value	A list of layout nets to be extracted with 600 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 600 EXCLUDE_NETS LAYOUTNAMES RECURSIVE. <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Recursive Layout net(s) 600	[] (list) {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexFSMNets.rec600SourceNames.specified	Boolean controlling xrc.pexFSMNets.rec600SourceNames <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Recursive Source net(s) 600” checkbox	[false true] (Boolean) {false}
xrc.pexFSMNets.rec600SourceNames.value	A list of source nets to be extracted with 600 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 600 EXCLUDE_NETS SOURCENAMES RECURSIVE. <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Recursive Source net(s) 600	[] (list) {}
xrc.pexFSMNets.top200LayoutNames.specified	Boolean controlling xrc.pexFSMNets.top200LayoutNames <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Toplevel Layout net(s) 200” checkbox	[false true] (Boolean) {false}
xrc.pexFSMNets.top200LayoutNames.value	A list of layout nets to be extracted with 200 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 200 EXCLUDE_NETS LAYOUTNAMES TOLEVEL. <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Toplevel Layout net(s) 200	[] (list) {}
xrc.pexFSMNets.top200SourceNames.specified	Boolean controlling xrc.pexFSMNets.top200SourceNames <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Toplevel Source net(s) 200” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexFSMNets.top200SourceNames.value	A list of source nets to be extracted with 200 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 200 EXCLUDE_NETS SOURCENAMES TOLEVEL.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Toplevel Source net(s) 200” checkbox	[] (list) {}
xrc.pexFSMNets.top600LayoutNames.specified	Boolean controlling xrc.pexFSMNets.top600LayoutNames  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Toplevel Layout net(s) 600” checkbox	[false true] (Boolean) {false}
xrc.pexFSMNets.top600LayoutNames.value	A list of layout nets to be extracted with 600 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 600 EXCLUDE_NETS LAYOUTNAMES TOLEVEL.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Toplevel Layout net(s) 600” checkbox	[] (list) {}
xrc.pexFSMNets.top600SourceNames.specified	Boolean controlling xrc.pexFSMNets.top600SourceNames  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, “Toplevel Source net(s) 600” checkbox	[false true] (Boolean) {false}
xrc.pexFSMNets.top600SourceNames.value	A list of source nets to be extracted with 600 accuracy mode. This list of nets is specified for PEX FIELDSOLVER MODE 600 EXCLUDE_NETS SOURCENAMES TOLEVEL.  <b>FieldSolver</b> page, PEX Fieldsolver Mode Nets section, Toplevel Source net(s) 600	[] (list) {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexFieldSolverMode.accuracy.value	Specifies the capacitance extraction accuracy mode used by Calibre xACT 3D in the PEX Fieldsolver Mode statement.  <b>Outputs</b> page, “Accuracy Mode” dropdown	[200 600] (0) {200}
xrc.pexGroundLayer.specified	Boolean controlling xrc.pexGroundLayer  <b>Options</b> page, “PEX Ground Layer” checkbox	[false true] (Boolean) {false}
xrc.pexGroundLayer.gndLayers.value	A list of ground layer names for the PEX Ground Layer statement.  <b>Options</b> page, PEX Ground Layer section	[] (list) {}
xrc.pexGroundLayer.stray.value	Specifies whether to eliminate all netlist output of intrinsic capacitance to substrate for the specified ground region. Adds the STRAY keyword to the PEX Ground Layer statement.  <b>Options</b> page, PEX Ground Layer section	[false true] (Boolean) {false}
xrc.pexIncludeNets.layoutRecursiveNets.specified	Boolean controlling xrc.pexIncludeNets.layoutRecursiveNets  <b>Options</b> page, PEX Extract Include Nets section, “Include Recursive Layout Nets” checkbox	[false true] (Boolean) {false}
xrc.pexIncludeNets.layoutRecursiveNets.value	The list of nets for the PEX Extract Include LAYOUTNAMES RECURSIVE statement.  <b>Options</b> page, PEX Extract Include Nets section, Include Recursive Layout Nets	[] (list) {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexIncludeNets.layoutToplevelNets.specified	Boolean controlling xrc.pexIncludeNets.layoutToplevelNets <b>Options</b> page, PEX Extract Include Nets section, “Include Toplevel Layout Nets” checkbox	[false true] (Boolean) {false}
xrc.pexIncludeNets.layoutToplevelNets.value	The list of nets for the PEX Extract Include LAYOUTNAMES TOLEVEL statement. <b>Options</b> page, PEX Extract Include Nets section, Include Toplevel Layout Nets	[] (list) {}
xrc.pexIncludeNets.sourceRecursiveNets.specified	Boolean controlling xrc.pexIncludeNets.sourceRecursiveNets <b>Options</b> page, PEX Extract Include Nets section, “Include Recursive Source Nets” checkbox	[false true] (Boolean) {false}
xrc.pexIncludeNets.sourceRecursiveNets.value	The list of nets for the PEX Extract Include SOURCENAMES RECURSIVE statement. <b>Options</b> page, PEX Extract Include Nets section, Include Recursive Source Nets	[] (list) {}
xrc.pexIncludeNets.sourceToplevelNets.specified	Boolean controlling xrc.pexIncludeNets.sourceToplevelNets <b>Options</b> page, PEX Extract Include Nets section, “Include Toplevel Source Nets” checkbox	[false true] (Boolean) {false}
xrc.pexIncludeNets.sourceToplevelNets.value	The list of nets for the PEX Extract Include SOURCENAMES TOLEVEL statement. <b>Options</b> page, PEX Extract Include Nets section, Include Toplevel Source Nets	[] (list) {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexInductanceFilter.specified	Boolean controlling xrc.pexInductanceFilter. Controls whether the PEX Inductance Filter statement is used.  <b>Inductance</b> page, “PEX Inductance Filter” checkbox	[false true] (Boolean) {false}
xrc.pexInductanceFilter.drtFile.value	The name of the driver/receiver file used for inductance extraction. The filename is specified with PEX Inductance Driver File. Used when xrc.pexInductanceFilter.filterType.value = FILE.  <b>Inductance</b> page, PEX Inductance Filter section, Driver/Receiver File	[] () {driver.xls}
xrc.pexInductanceFilter.filterTags.specified	Boolean controlling xrc.pexInductanceFilter.filterTags  <b>Inductance</b> page, PEX Inductance Filter section, “Driver/Receiver Tags” checkbox	[false true] (Boolean) {false}
xrc.pexInductanceFilter.filterTags.value	A list of values that specifies how to generate a driver/receiver file. Used for the PEX Inductance Filter statement.  <b>Inductance</b> page, PEX Inductance Filter section, Driver/Receiver Tags	[] (list) {}
xrc.pexInductanceFilter.filterType.value	Specifies the type of net filtering to apply. Used for the PEX Inductance Filter statement.  <b>Inductance</b> page, PEX Inductance Filter section, “Net Filter” dropdown	[APPROX OFF FILE EXCLUSIVE] () {APPROX}
xrc.pexInductanceFrequency.specified	Boolean controlling xrc.pexInductanceFrequency  <b>Inductance</b> page, “PEX Inductance Frequency” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexInductanceFrequency.broadBand.value	<p>Boolean controlling whether the broadband model is used in inductance extraction.</p> <p>Adds the MAXIMUM keyword to the PEX Inductance Frequency statement and enables the use of the PEX Inductance Switch Time statement.</p> <p><b>Inductance</b> page, PEX Inductance Frequency section, “Use the broadband model” checkbox</p>	[false true] (Boolean) {false}
xrc.pexInductanceFrequency.frequency.value	<p>Specifies the frequency in GHz for the PEX Inductance Frequency statement.</p> <p>The MAXIMUM keyword is added if the broadband model is used (xrc.pexInductanceFrequency.broadBand.value = true).</p> <p>Used when xrc.pexInductanceFrequency.gHzORSeconds.value = GHZ.</p> <p><b>Inductance</b> page, PEX Inductance Frequency section, Operating Frequency</p>	[] ( {10}
xrc.pexInductanceFrequency.gHzORSeconds.value	<p>Specifies whether to specify the operating frequency for inductance extraction as a frequency in GHz or as a switch time in seconds.</p> <p>To specify SECONDS, you must also use the broadband model (xrc.pexInductanceFrequency.broadBand.value = true).</p> <p><b>Inductance</b> page, PEX Inductance Frequency section, “Specify In” dropdown</p>	[GHZ SECONDS] ( {GHZ}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexInductanceFrequency.switchTime.value	Specifies the switch time in seconds for the PEX Inductance Switch Time statement.  Used with the broadband model when xrc.pexInductanceFrequency.gHzORSeconds.value = SECONDS.  <b>Inductance</b> page, PEX Inductance Frequency section, Switch Time	[] () {1e-10}
xrc.pexInductanceMinLength.minLength.specified	Boolean controlling xrc.pexInductanceMinLength.minLength  <b>Inductance</b> page, “PEX Inductance MinLength” checkbox	[false true] (Boolean) {false}
xrc.pexInductanceMinLength.minLength.value	Specifies the minimum path length in microns for the PEX Inductance Minlength statement.  <b>Inductance</b> page, PEX Inductance MinLength	[] () {100}
xrc.pexInductanceMode.extractMode.specified	Boolean controlling xrc.pexInductanceMode.extractMode  <b>Inductance</b> page, “PEX Inductance Mode” checkbox	[false true] (Boolean) {false}
xrc.pexInductanceMode.extractMode.value	Specifies the inductance extraction mode for the PEX Inductance Mode statement.  <b>Inductance</b> page, “PEX Inductance Mode” dropdown	[LOOP PEEC] () {LOOP}
xrc.pexInductanceRange.distance.specified	Boolean controlling xrc.pexInductanceRange.distance  <b>Inductance</b> page, “PEX Inductance Range” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexInductanceRange.distance.value	Specifies the maximum distance in microns from a given net segment to neighboring layout geometries when determining inductance. Defines the primary keyword for the PEX Inductance Range statement. <b>Inductance</b> page, PEX Inductance Range	[] ( {0}
xrc.pexInductanceReturnPath.netType.specified	Boolean controlling xrc.pexInductanceReturnPath.netType <b>Inductance</b> page, “PEX Inductance Returnpath” checkbox	[false true] (Boolean) {false}
xrc.pexInductanceReturnPath.netType.value	Specifies the type of nets that may be considered as candidates for return path in inductance extraction. Defines the primary keyword for the PEX Inductance Returnpath statement. <b>Inductance</b> page, “PEX Inductance Returnpath” dropdown	[GROUND POWER GANDP GANDPAVG NONE] ( {GROUND}
xrc.pexInductanceSameNetMutual.offOn.specified	Boolean controlling xrc.pexInductanceSameNetMutual.offOn <b>Inductance</b> page, “PEX Inductance Same Net Mutual” checkbox	[false true] (Boolean) {false}
xrc.pexInductanceSameNetMutual.offOn.value	Specifies whether to enable the extraction of the mutual inductance between wire segments within the same net. Defines the primary keyword used for the PEX Inductance Same Net Mutual statement. <b>Inductance</b> page, “PEX Inductance Same Net Mutual” dropdown	[OFF ON] ( {OFF}
xrc.pexInductanceVictimPath.victimFile.specified	Boolean controlling xrc.pexInductanceVictimPath.victimFile <b>Inductance</b> page, “Use PEX Inductance Victim File” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexInductanceVictimPath.victimFile.value	Specifies the path to a file containing victim path or victim information for inductance extraction.  This file is specified in the control file with an Include statement.  <b>Inductance</b> page, Use PEX Inductance Victim File	[] ( {}
xrc.pexLefExtractCellObstructions.yesNo.specified	Boolean controlling xrc.pexLefExtractCellObstructions.yesNo	[false true] (Boolean) {false}
xrc.pexLefExtractCellObstructions.yesNo.value		[NO YES] ( {NO}
xrc.pexNetListAdvanced.netListCreateSmashedDeviceNames.specified	Boolean controlling xrc.pexNetListAdvanced.netListCreateSmashedDeviceNames. Available when the netlist format is SPEF or DSPEF.  <b>Options</b> page, PEX Netlist Advanced Options section, “PEX Netlist Create Smashed Device Names” checkbox	[false true] (Boolean) {false}
xrc.pexNetListAdvanced.netListCreateSmashedDeviceNames.value	Specifies whether to output smashed device instance names in DSPEF and SPEF formats.  <b>Options</b> page, PEX Netlist Advanced Options section, “PEX Netlist Create Smashed Device Names” dropdown	[YES NO] ( {YES}
xrc.pexNetListAdvanced.netListDeviceResistanceModel.specified	Boolean controlling xrc.pexNetListAdvanced.netListDeviceResistanceModel  <b>Options</b> page, PEX Netlist Advanced Options section, “PEX Netlist Device Resistance Model” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetListAdvanced.netListDeviceResistanceModel.value	Specifies which model to use for parasitic resistors.  <b>Options</b> page, PEX Netlist Advanced Options section, “PEX Netlist Device Resistance Model” dropdown	[SPICE CALCULATE] () {SPICE}
xrc.pexNetListAdvanced.netListUpperCaseKeywords.specified	Boolean controlling xrc.pexNetListAdvanced.netListUpperCaseKeywords  <b>Options</b> page, PEX Netlist Advanced Options section, “PEX Netlist Uppercase”	[false true] (Boolean) {false}
xrc.pexNetListAdvanced.netListUpperCaseKeywords.value	Specifies whether to write element names, and HSPICE, DSPF, and SPECTRE keywords in uppercase text in the output netlist.  <b>Options</b> page, PEX Netlist Advanced Options section, “PEX Netlist Uppercase” dropdown	[NO YES] () {NO}
xrc.pexNetListAdvanced.netListVirtualConnect.specified	Boolean controlling xrc.pexNetListAdvanced.netListVirtualConnect  <b>Options</b> page, PEX Netlist Advanced Options section, “PEX Netlist Virtual Connect” checkbox	[false true] (Boolean) {false}
xrc.pexNetListAdvanced.netListVirtualConnect.value	Specifies whether to connect disjoint net model fragments created by Virtual Connect statements to the main net model using a resistor.  <b>Options</b> page, PEX Netlist Advanced Options section, “PEX Netlist Virtual Connect” dropdown	[NO YES] () {NO}
xrc.pexNetlist.busdelim.specified	Boolean controlling xrc.pexNetlist.busdelim. Available when the netlist format is SPEF or DSPF.  <b>Outputs</b> page, “PEX Netlist” section, BUSDELIM checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.busdelim.value	Specifies whether to set the BUS_DELIMITER variable in the DSPF or SPEF header. <b>Outputs</b> page, “PEX Netlist” section, BUSDELIM	[] ( {[]})
xrc.pexNetlist.calibrated.value	Specifies whether to report only calibrated layers in the LAYER_MAP selection of the netlist. Available when xrc.pexNetlist.layermap.value = true. <b>Outputs</b> page, “PEX Netlist” section, CALIBRATED checkbox	[false true] (Boolean) {false}
xrc.pexNetListConnectionSection.specified	Boolean controlling xrc.pexNetListConnectionSection. Available when the netlist format is SPEF or DSPF. <b>Options</b> page, “PEX Netlist Connection Section” checkbox	[false true] (Boolean) {false}
xrc.pexNetListConnectionSection.instLoc.value	Specifies whether to include instance location and layer information as comments in the * I section of the netlist. Adds the INST_LOC keyword to the PEX Netlist Connection Section statement. <b>Options</b> page, PEX Netlist Connection Section, “Include instance location” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetListConnectionSection.section.value	Specifies whether to include nonstandard SPICE statements in the connection section of the netlist. Also specifies the YES or NO keyword for the PEX Netlist Connection Section statement. YES includes nonstandard SPICE statements in the connection section of the netlist.  <b>Options</b> page, PEX Netlist Connection Section, “Include nonstandard SPICE statements” dropdown	[YES NO] () {YES}
xrc.pexNetListExportPorts.yesNo.specified	Boolean controlling xrc.pexNetListExportPorts.yesNo  <b>Options</b> page, “PEX Netlist Export Ports” checkbox	[false true] (Boolean) {false}
xrc.pexNetListExportPorts.yesNo.value	Specifies whether or not the formatter creates ports for internal nets. Specifies YES or NO for the PEX Netlist Export Ports statement.  <b>Options</b> page, “PEX Netlist Export Ports” dropdown	[NO YES] () {NO}
xrc.pexNetListGlobalNets.nets.specified	Boolean controlling xrc.pexNetListGlobalNets.nets  <b>Options</b> page, “PEX Netlist Global Nets” checkbox	[false true] (Boolean) {false}
xrc.pexNetListGlobalNets.nets.value	A list of nets to be removed from the subcircuit and placed it in a global name definition. Specifies the names for the PEX NETLIST GLOBAL NETS statement.  <b>Options</b> page, PEX Netlist Global Nets	[] (list) {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetListMutualResistance.yesNo.specified	Boolean controlling xrc.pexNetListMutualResistance.yesNo. <b>Inductance</b> page, “PEX Netlist Mutual Resistance” checkbox	[false true] (Boolean) {false}
xrc.pexNetListMutualResistance.yesNo.value	Specifies whether to use current-controlled current courses to model mutual resistance. <b>Inductance</b> page, PEX Netlist Mutual Resistance section	[YES NO] 0 {YES}
xrc.pexNetListNoXrefNames.yesNo.specified	Boolean controlling xrc.pexNetListNoXrefNames.yesNo. <b>Options</b> page, “PEX Netlist Noxref Net Names” checkbox	[false true] (Boolean) {false}
xrc.pexNetListNoXrefNames.yesNo.value	Specifies whether to keep parasitic net models with _noxref from netlists. Also specifies YES or NO for the PEX Netlist Noxref Net Names statement. <b>Options</b> page, “PEX Netlist Noxref Net Names” dropdown	[YES NO] 0 {YES}
xrc.pexNetListSelectFile.filePath.value	Specifies a file to control the extracted netlist during the formatting stage. <b>Options</b> page	[] 0 {}
xrc.pexNetListShortToplevelPorts.yesNo.specified	Boolean controlling xrc.pexNetListShortToplevelPorts.yesNo. Available when the netlist format is SPEF, DPSEF, HSPICE, or CALIBREVIEW. <b>Options</b> page, “PEX Netlist Short Toplevel Ports” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetListShortToplevelPorts.yesNo.value	Specifies whether to create multiple top-level ports in the extracted netlist. Specifies YES or NO for the PEX Netlist Short Toplevel Ports statement. Applies to SPEF, DSPEF, HSPICE, and Calibre View netlists. <b>Options</b> page, “PEX Netlist Short Toplevel Ports” dropdown	[YES NO] ( {YES}
xrc.pexNetlist.busdelim.specified	Boolean controlling xrc.pexNetlist.busdelim. Available when the netlist format is SPEF or DSPEF. <b>Outputs</b> page, PEX Netlist section, “BUSDELIM” checkbox	[false true] (Boolean) {false}
xrc.pexNetlist.busdelim.value	Specifies the string value for the BUSDELIM keyword in the PEX Netlist statement. The delimiter information is added to the SPEF or DSPEF netlist header. <b>Outputs</b> page, PEX Netlist section, BUSDELIM	[] ( {[]})
xrc.pexNetlist.clocation.value	The checkbox for CLOCATION, which adds the CLOCATION option to PEX Netlist. <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}
xrc.pexNetlist.coordscale.specified	Checkbox for COORDSCALE, which adds the COORDSCALE option to PEX Netlist. <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}
xrc.pexNetlist.coordscale.value	The scaling factor for coordinates, in meters. Specify 1.0e-06 to report coordinates in microns. This is the value for the COORDSCALE keyword in PEX Netlist. <b>Outputs</b> page, PEX Netlist section	[] ( {1})

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.eldo_kr.value	Adds R to L instances and KR to K instances. The f and v elements for mutual resistance are not added.	[false true] (Boolean) {false}
xrc.pexNetlist.ground.specified	Checkbox for GROUND, which adds the GROUND option to PEX Netlist. <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}
xrc.pexNetlist.ground.value	The name of the ground node for the GROUND keyword in PEX Netlist. <b>Outputs</b> page, PEX Netlist section	[] () {0}
xrc.pexNetlist.icellmap.value	Checkbox for ICELLMAP, which adds the ICELLMAP option to PEX Netlist. Specifies whether to output the device subtype name, if available. Only available when the netlist format is CALIBREVIEW. <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}
xrc.pexNetlist.layermap.value	Checkbox for LAYERMAP, which adds the LAYERMAP option to PEX Netlist. Specifies whether to report parasitic resistor layers and bounding boxes as comments for the EM/IR analysis flow. Used with resistance parameter keywords RWIDTH, RLENGTH, RVIACOUNT, RAREA, RLAYER, and RLOCATION. Only available when the netlist format is DSPF. <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.leadingx.value	<p>Checkbox for LEADINGX, which adds the LEADINGX keyword to PEX Netlist. Specifies whether to retain leading Xs on flattened instance names. Only available when the netlist format is DSPF or CALIBREVIEW and xrc.pexNetlist.noinstancex.value = true.</p> <p><b>Outputs</b> page, PEX Netlist section,</p>	[false true] (Boolean) {false}
xrc.pexNetlist.listprobes.value	<p>Checkbox for LISTPROBES, which adds the LISTPROBES option to PEX Netlist. Specifies whether to include probes in the .SUBCKT definitions. Only available when the netlist format is HSPICE, ELDO, or SPECTRE.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.location.value	<p>Checkbox for LOCATION, which adds the LOCATION option to PEX Netlist. Specifies whether to report device locations as comments. Intentional resistors and capacitors are also reported. The LOCATION keyword does not work with the SCHEMATICONLY and SOURCEBASED parameters.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.mapnames.value	<p>Checkbox for MAPNAMES, which adds the MAPNAMES option to PEX Netlist. Specifies whether to map instance names to numerical identifiers and use the identifiers in the netlist model sections. Only available when the netlist format is SPEF.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.multitxtport.value	Checkbox for MULTITXTPORT, which adds the MULTITXTPORT option to PEX Netlist. Specifies whether to output multiple net ports from multiple text ports. Only available when the netlist format is DSPF.  <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}
xrc.pexNetlist.multivalue.value	Checkbox for MULTIVALUES, which adds the MULTIVALUES option to PEX Netlist. Specifies whether to create a single file netlist with multiple corner values. Multiple corners must be specified.  MULTIVALUES creates a single netlist file with multiple corner values. Only available when the netlist format is SPEF.  <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}
xrc.pexNetlist.netgeometry.value	Checkbox for “Net Geometry”. Only available when the netlist format is CALIBREVIEW. When enabled, the LOCATION, RLAYER, RLENGTH, RLOCATION, and RWIDTH keywords are added to the PEX Netlist statement. In addition, SVDB output is enabled with the QUERY, XACT, and CCI keywords. The PEX XACT PDB statement is also enabled.  Specify this option to backannotate net geometries to the Calibre View.  <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}
xrc.pexNetlist.netlistFile.value	The filename of the generated parasitic netlist that is specified in the PEX Netlist statement.  <b>Outputs</b> page, PEX Netlist section	[] () {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.netlistFileViewAfterRun.value	Checkbox for “View Netlist after extraction run finishes”. Specifies whether to automatically view the Netlist file after extraction run is complete.  <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}
xrc.pexNetlist.netlistFormat.value	The format of the generated netlist. This is the keyword specified in the PEX Netlist statement.  <b>Outputs</b> page, PEX Netlist section	[HSPICE DSPF SPEF SPECTRE ELD0 CALIBREVIE W SPICE] ( {ELDO}
xrc.pexNetlist.netlistMode.specified	Boolean controlling xrc.pexNetlist.netlistMode  <b>Outputs</b> page, PEX Netlist section, “Netlist Compatibility” checkbox	[false true] (Boolean) {false}
xrc.pexNetlist.netlistMode.value	Adds the specified keyword to the PEX Netlist statement in order to output a netlist compatible with the specified tool. See the PEX Netlist statement to determine which keywords are supported for a particular netlist format and extraction mode.  <b>Outputs</b> page, PEX Netlist section, “Netlist Compatibility” dropdown	[ADITRAIL PRIMETIME HSIM ULTRASIM TOTEM QUICKCAP] ( {ADITRAIL}
xrc.pexNetlist.nodelocation.value	Checkbox for NODELOCATION, which adds the NODELOCATION keyword to PEX Netlist.  Specifies whether to report the coordinates of internal nodes in the netlist.  <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.noinstancesection.value	<p>Checkbox for NOINSTANCESECTION, which adds the NOINSTANCESECTION keyword to PEX Netlist. Specifies whether to output the instance section in the netlist. Only available when the netlist format is DSPF.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.noinstancex.value	<p>Checkbox for NOINSTANCEX , which adds the NOINSTANCEX keyword to PEX Netlist. Specifies whether to have the formatter filter Xs in the output.</p> <p>Only available when the netlist format is HSPICE, DSPF, SPEF, and SPECTRE.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.notopsubckt.value	<p>Checkbox for NOTOPSUBCKT, which adds the NOTOPSUBCKT keyword to PEX Netlist. Specifies whether to suppress the top-level subcircuit in the output netlist.</p> <p>Only available when the netlist format is DSPF.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.pindelim.value	<p>The value for the PINDELIM option in the PEX Netlist statement, which specifies the delimiter between instance names and pin names. The allowed values are a colon (:) or a period (.). A colon is the default. Only available when the netlist format is SPEF.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[: .] 0 {:}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.pinnamesep.specified	<p>Boolean controlling xrc.pexNetlist.pinnamesep. The checkbox for PINNAMESEP, which adds the PINNAMESEP keyword to PEX Netlist. The required value is specified with xrc.pexNetlist.pinnamesep.value.</p> <p>Only available when the netlist format is HSPICE.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.pinnamesep.value	<p>The separator character placed between device pins and parasitic model subcircuits when creating net names. Also the value for the PINNAMESEP option in PEX Netlist. Used only for the HSPICE netlist format and when xrc.pexNetlist.pinnamesep.specified = true.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[] () {_}
xrc.pexNetlist.prune.value	<p>Checkbox for PRUNE, which adds the PRUNE keyword to PEX Netlist. Specifies whether to remove all intentional devices not attached to the extracted nets.</p> <p>Only available when the netlist format is HSPICE, DSPF, or ELDO.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.ractual.value	<p>Checkbox for RACTUAL, which adds the RACTUAL keyword to PEX Netlist. Specifies whether to report the actual parasitic resistor width dimensions rather than the drawn width.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.rarea.value	<p>Checkbox for RAREA, which adds the RAREA keyword to PEX Netlist. Specifies whether to report the contact/via area as a comment.</p> <p>Only available when the netlist format is DSPF, HSPICE, and SPEF.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.rcnamed.value	<p>Checkbox for RCNAMED, which adds the RCNAMED keyword to PEX Netlist. Specifies whether to replace the R or C value of an intentional device with R=value or C=value.</p> <p>Only available when the netlist format is HSPICE, DSPF, or ELDO.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.rlayer.value	<p>Checkbox for RLAYER, which adds the RLAYER keyword to PEX Netlist. Specifies whether to report parasitic resistor layers as comments.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.rlength.value	<p>Checkbox for RLENGTH, which adds the RLENGTH keyword to PEX Netlist. Specifies whether to report parasitic resistor lengths as comments.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.rlocation.value	<p>Checkbox for RLOCATION, which adds the RLOCATION keyword to PEX Netlist. Specifies whether to report parasitic resistor locations as comments.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.rscale.specified	<p>Boolean controlling xrc.pexNetlist.rscale. The checkbox for RSCALE, which adds the RSCALE keyword to PEX Netlist.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.rscale.value	The value for the RSCALE option in PEX Netlist. The scaling factor for parasitic resistor properties. This setting does not scale device properties.  Used when xrc.pexNetlist.rscale.specified = true. <b>Outputs</b> page, PEX Netlist section	[] 0 {1}
xrc.pexNetlist.rthickness.value	Checkbox for RTHICKNESS, which adds the RTHICKNESS keyword to PEX Netlist. Specifies whether to report calculated parasitic resistor thicknesses as comments.  <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}
xrc.pexNetlist.rviacount.value	Checkbox for RVIACOUNT, which adds the RVIACOUNT keyword to PEX Netlist. Specifies whether to report the number of vias/contacts represented by the netlisted via resistor value as a comment. Not available for the CALIBREVIEW netlist format.  <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}
xrc.pexNetlist.rwidth.value	Checkbox for RWIDTH, which adds the RWIDTH keyword to PEX Netlist. Specifies whether to report calculated parasitic resistor widths as comments.  <b>Outputs</b> page, PEX Netlist section	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.rwidthMin.value	<p>Checkbox for MINIMUM, which adds the MINIMUM option for RWIDTH to PEX Netlist. Specifies whether to report the minimum parasitic resistor width for non-rectangular shapes.</p> <p>Only available when RWIDTH is checked (xrc.pexNetlist.rwidth.value = true).</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.scale.specified	<p>Boolean controlling xrc.pexNetlist.scale. Checkbox for “Scale”, which adds a scale to the PEX Netlist statement. The scale value is specified with xrc.pexNetlist.scale.value.</p> <p>Not available for the SPEF netlist format.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.scale.value	<p>The scale for the PEX Netlist statement. The scale specifies a size multiplier in meters for certain parameters of certain devices. Used only when xrc.pexNetlist.scale.specified = true.</p> <p>Not available for the SPEF netlist format.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[] () {1}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.schematicOnlyFile.specified	<p>Boolean controlling xrc.pexNetlist.schematicOnlyFile. Checkbox for “Device Info File” in the PEX Netlist section of the GUI. When enabled, the filename is given by xrc.pexNetlist.schematicOnlyFile.value.</p> <p>Only available when xrc.pexNetlist.system.value is SOURCENAMES or SCHEMATICONLY.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.schematicOnlyFile.value	<p>An optional input file that defines device parameters for the PEX Netlist statement. Used only with SOURCENAMES or SCHEMATICONLY, and when xrc.pexNetlist.schematicOnlyFile.specified = true.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[] ( {}
xrc.pexNetlist.separator.specified	<p>Boolean controlling xrc.pexNetlist.separator. Checkbox for SEPARATOR, which adds the SEPARATOR keyword to the PEX Netlist statement. The separator string is specified with xrc.pexNetlist.separator.value.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.separator.value	<p>The separator string specified for the SEPARATOR option in PEX Netlist. The default is a slash (/), except for the ELDO netlist, which has a default of a period (.). Used only when xrc.pexNetlist.separator.specified = true.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[/ .   ^ # _] ( )

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.shortpinnames.value	<p>Checkbox for SHORTPINNAMES, which adds the SHORTPINNAMES keyword to PEX Netlist. Specifies whether to replace pin names with an integer when creating SPICE net names.</p> <p>Only available for the HSPICE netlist format.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.singlefile.value	<p>Checkbox for SINGLEFILE, which adds the SINGLEFILE keyword to PEX Netlist. Specifies whether to create a single output file.</p> <p>Only available for the HSPICE, ELDO, and SPECTRE netlist formats.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.stringquotes.specified	<p>Boolean controlling xrc.pexNetlist.stringquotes. Checkbox for STRINGQUOTES, which adds the STRINGQUOTES keyword to PEX Netlist. The value is given by xrc.pexNetlist.stringquotes.value.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexNetlist.stringquotes.value	<p>The value for the STRINGQUOTES option for the PEX Netlist statement. Specifies whether to enclose string properties in quotes.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[YES NO] 0 {YES}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexNetlist.system.value	<p>Specifies how to derive names in the generated netlist. The value is a required keyword in the PEX Netlist statement.</p> <p><b>Outputs</b> page, PEX Netlist section, “Use Names From” dropdown</p>	[LAYOUTNAME ES SOURCENAME ES SOURCEBASE D SCHEMATICONLY] { } {SOURCENAME ES}
xrc.pexNetlist.tcoeff.value	<p>Checkbox for TCOEFF, which adds the TCOEFF keyword to PEX Netlist. Specifies whether to output temperature coefficients for parasitic resistors.</p> <p>Not available for the SPEF netlist format.</p> <p><b>Outputs</b> page, PEX Netlist section</p>	[false true] (Boolean) {false}
xrc.pexPinOrder.specified	<p>Boolean controlling xrc.pexPinOrder. Adds the PEX Pin Order statement.</p> <p><b>Options</b> page, “PEX Pin Order” checkbox</p>	[false true] (Boolean) {false}
xrc.pexPinOrder.allPins.value	<p>Adds the ALLPINS keyword to the PEX Pin Order statement. Specifies whether to include all pin names from the source netlist or netlist file in the extracted netlist.</p> <p>Used when xrc.pexPinOrder.specified = true and xrc.pexPinOrder.orderBy.value is set to SOURCE or FILE.</p> <p><b>Options</b> page, PEX Pin Order section, “All Pins” checkbox</p>	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexPinOrder.netlistFile.value	<p>The filename parameter for the PEX Pin Order statement when FILE is specified. Specifies the path to and name of the netlist file for ordering ports. The cell and port names in file must match the names in the layout.</p> <p>Used when xrc.pexPinOrder.specified = true and xrc.pexPinOrder.orderBy.value = FILE.</p> <p><b>Options</b> page, PEX Pin Order section, File</p>	[ () {}]
xrc.pexPinOrder.noXref.value	<p>Adds the NOXREF keyword to the PEX Pin Order statement. Specifies whether to allow pin ordering even when cross-reference database is available.</p> <p>Used when xrc.pexPinOrder.specified = true and xrc.pexPinOrder.orderBy.value = FILE.</p> <p><b>Options</b> page, PEX Pin Order section, “No Xref” checkbox</p>	[false true] (Boolean) {false}
xrc.pexPinOrder.orderBy.value	<p>The primary keyword for the PEX Pin Order statement. Specifies how to control the pin ordering of the netlist.</p> <p>Used when xrc.pexPinOrder.specified = true.</p> <p><b>Options</b> page, PEX Pin Order section, “Order By” dropdown</p>	[LAYOUT SOURCE FILE] () {LAYOUT}
xrc.pexReduceAnalog.specified	<p>Boolean controlling xrc.pexReduceAnalog</p> <p><b>Options</b> page, “PEX Reduce Analog” checkbox</p>	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReduceAnalog.delayError.specified	Boolean controlling xrc.pexReduceAnalog.delayError. Checkbox for “Timing Delay Threshold”, which adds the DELAY_ERROR keyword to the PEX Reduce Analog statement. The value is specified by xrc.pexReduceAnalog.delayError.value.  <b>Options</b> page, PEX Reduce Analog section	[false true] (Boolean) {false}
xrc.pexReduceAnalog.delayError.value	The timing delay threshold in seconds for the DELAY_ERROR keyword in PEX Reduce Analog statement. Used when xrc.pexReduceAnalog.delayError.specified = true  <b>Options</b> page, PEX Reduce Analog section, Timing Delay Threshold	[] 0 {4.999999999999999e-13}
xrc.pexReduceAnalog.noiseError.specified	Boolean controlling xrc.pexReduceAnalog.noiseError. Checkbox for “Noise Threshold”, which adds the NOISE_ERROR keyword to the PEX Reduce Analog statement.  The value is specified by xrc.pexReduceAnalog.noiseError.value.  <b>Options</b> page, PEX Reduce Analog section	[false true] (Boolean) {false}
xrc.pexReduceAnalog.noiseError.value	The noise ratio threshold for the NOISE_ERROR keyword in PEX Reduce Analog statement. Used when xrc.pexReduceAnalog.noiseError.specified = true.  <b>Options</b> page, PEX Reduce Analog section, Noise Threshold	[] 0 {0.01}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReduceAnalog.yesNo.value	The primary keyword for the PEX Reduce Analog statement. Specifies whether to apply reduction for analog designs.  Used when xrc.pexReduceAnalog.specified = true.  <b>Options</b> page, PEX Reduce Analog section, “Apply Reduction” dropdown	[NO YES] () {YES}
xrc.pexReduceCC.absoluteThreshold.specified	Boolean controlling xrc.pexReduceCC.absoluteThreshold. Checkbox for “Absolute Threshold”, which adds the ABSOLUTE keyword to PEX Reduce CC.  The threshold is given by xrc.pexReduceCC.absoluteThreshold.value.  <b>Options</b> page, PEX Reduce CC section	[false true] (Boolean) {false}
xrc.pexReduceCC.absoluteORratio.value	When both ABSOLUTE and RATIO are specified for PEX Reduce CC, specifies how combine the conditions. Used when xrc.pexReduceCC.absoluteThreshold.specified and xrc.pexReduceCC.ratioPercent.specified are both true.  <b>Options</b> page, PEX Reduce CC section, “Combine Absolute with Ratio” dropdown	[AND OR] () {AND}
xrc.pexReduceCC.absoluteThreshold.value	The “Absolute Threshold” value, in femtoFarads, for the PEX Reduce CC statement. Used when xrc.pexReduceCC.absoluteThreshold.specified = true.  <b>Options</b> page, PEX Reduce CC section	[] () {3}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReduceCC.ratioPercent.specified	<p>Boolean controlling xrc.pexReduceCC.ratioPercent. Checkbox for “Ratio Percent”, which adds the RATIO keyword to PEX Reduce CC.</p> <p>The percent value is given by xrc.pexReduceCC.ratioPercent.value.</p> <p><b>Options</b> page, PEX Reduce CC section</p>	[false true] (Boolean) {false}
xrc.pexReduceCC.ratioPercent.value	<p>The “Ratio Percent” value for the PEX Reduce CC statement. The value is a number between 0 and 1. Used when xrc.pexReduceCC.ratioPercent.specified = true.</p> <p><b>Options</b> page, PEX Reduce CC section</p>	[] 0 {1}
xrc.pexReduceCC.scale.specified	<p>Boolean controlling xrc.pexReduceCC.scale. Checkbox for “Scale”, which adds the SCALE keyword to PEX Reduce CC.</p> <p>The value is given by xrc.pexReduceCC.scale.value.</p> <p><b>Options</b> page, PEX Reduce CC section</p>	[false true] (Boolean) {false}
xrc.pexReduceCC.scale.value	<p>The “Scale” value for the PEX Reduce CC statement. Applied to the reduced capacitance before groundingUsed when xrc.pexReduceCC.scale.specified = true.</p> <p><b>Options</b> page, PEX Reduce CC section</p>	[] 0 {1}
xrc.pexReduceDigital.specified	<p>Boolean controlling xrc.pexReduceDigital. Adds the PEX Reduce Digital statement.</p> <p><b>Options</b> page, “PEX Reduce Digital” checkbox</p>	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReduceDigital.delayError.specified	<p>Checkbox for “Timing Delay Threshold”, which adds the DELAY_ERROR keyword to the PEX Reduce Digital statement.</p> <p>The value is specified by xrc.pexReduceDigital.delayError.value.</p> <p><b>Options</b> page, PEX Reduce Digital section</p>	[false true] (Boolean) {false}
xrc.pexReduceDigital.delayError.value	<p>The timing delay threshold in seconds for the DELAY_ERROR keyword in PEX Reduce Digital statement. Used when xrc.pexReduceDigital.delayError.specified = true.</p> <p><b>Options</b> page, PEX Reduce Digital section, Timing Delay threshold</p>	[] 0 {9.999999999999999e-13}
xrc.pexReduceDigital.noiseError.specified	<p>Checkbox for “Noise Threshold”, which adds the NOISE_ERROR keyword to the PEX Reduce Digital statement.</p> <p>The value is specified by xrc.pexReduceDigital.noiseError.value.</p> <p><b>Options</b> page, PEX Reduce Digital section</p>	[false true] (Boolean) {false}
xrc.pexReduceDigital.noiseError.value	<p>The noise error threshold for the NOISE_ERROR keyword in PEX Reduce Digital statement. Used when xrc.pexReduceDigital.noiseError.specified = true.</p> <p><b>Options</b> page, PEX Reduce Digital section, Noise Threshold</p>	[] 0 {0.02999999999999999}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReduceDigital.yesNo.value	The primary keyword for the PEX Reduce Digital statement. Specifies whether to apply reduction for digital designs.  <b>Options</b> page, PEX Reduce Digital section, “Apply Reduction” dropdown	[NO YES] ( {YES}
xrc.pexReduceMinCap.combine.specified	Boolean controlling xrc.pexReduceMinCap.combine. Checkbox for the COMBINE keyword in PEX Reduce Mincap.  <b>Options</b> page, PEX Reduce MinCap section	[false true] (Boolean) {false}
xrc.pexReduceMinCap.combine.value	The threshold value for the COMBINE keyword in PEX Reduce Mincap. Capacitors with a value below the specified threshold are combined and reported as a single parasitic capacitor.Used when xrc.pexReduceMinCap.combine.specified = true.  <b>Options</b> page, PEX Reduce MinCap section	[] ( {}
xrc.pexReduceMinCap.remove.specified	Boolean controlling xrc.pexReduceMinCap.remove. Checkbox for the REMOVE keyword in PEX Reduce Mincap.  <b>Options</b> page, PEX Reduce MinCap section	[false true] (Boolean) {false}
xrc.pexReduceMinCap.remove.value	The threshold value for the REMOVE keyword in PEX Reduce Mincap. Capacitors with a value below the specified threshold are removed from a net.  <b>Options</b> page, PEX Reduce MinCap section	[] ( {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReduceMinMutual.threshold.specified	Boolean controlling xrc.pexReduceMinMutual.threshold. The required value is given by xrc.pexReduceMinMutual.threshold.value. <b>Inductance</b> page, “PEX Reduce MinMutual” checkbox	[false true] (Boolean) {false}
xrc.pexReduceMinMutual.threshold.value	Specifies to remove parasitic mutual inductance below a threshold value to reduce netlist size. The threshold value must be greater than zero and less than one ( $0 < K < 1$ ). <b>Inductance</b> page, PEX Reduce MinMutual	[] ( {}
xrc.pexReduceMinRes.combine.specified	Boolean controlling xrc.pexReduceMinRes.combine <b>Options</b> page, PEX Reduce MinRes section	[false true] (Boolean) {false}
xrc.pexReduceMinRes.combine.value	The threshold for combine extracted resistors. Resistors with a value below the specified threshold will be added together and reported as a single parasitic resistor. <b>Options</b> page, PEX Reduce MinRes section	[] ( {}
xrc.pexReduceMinRes.layerwise.value	Boolean that controls whether to specify the reduction of parasitic resistors for each layer, and not across different layers. Available when xrc.pexReduceMinRes.combine.specified = true. <b>Options</b> page, PEX Reduce MinRes section	[false true] (Boolean) {false}
xrc.pexReduceMinRes.shortres.specified	Boolean controlling xrc.pexReduceMinRes.shortres <b>Options</b> page, PEX Reduce MinRes section	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReduceMinRes.shortres.value	Threshold for short extracted resistors. Resistors with a value below the specified threshold will be shorted. <b>Options</b> page, PEX Reduce MinRes section	[] (0) {}
xrc.pexReduceMinRes.yesNo.value	Specifies whether to turn on or off PEX REDUCE MINRES, which reduces the size of the netlist by decreasing the number of parasitic resistors. <b>Options</b> page, PEX Reduce MinRes section	[YES NO] (0) {YES}
xrc.pexReduceParallelMOS.specified	Boolean controlling xrc.pexReduceParallelMOS <b>Options</b> page, “PEX Reduce Parallel MOS” checkbox	[false true] (Boolean) {false}
xrc.pexReduceParallelMOS.mosTypes.value	Specifies the types of parallel mosfets to combine. Specify ALL to combine all mosfet devices or provide a list of mosfet device type names. <b>Options</b> page, PEX Reduce Parallel MOS section, Specified MOSFET Types	[] (list) {}
xrc.pexReduceParallelMOS.rThreshold.value	Specifies the resistance threshold between parallel device pins. <b>Options</b> page, PEX Reduce Parallel MOS section, Resistance threshold	[] (0) {0}
xrc.pexReduceParallelMOS.reduce.value	Specifies whether or not parallel mosfets that are the same model type and subtype are reduced to a single mosfet in the generated netlist. <b>Options</b> page, PEX Reduce Parallel MOS section, “Reduce MOSFETs” dropdown	[NO YES] (0) {NO}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReduceParallelMOS.sameWL.value	Specifies to merge only parallel devices of the same width and length. <b>Options</b> page, PEX Reduce Parallel MOS section, “SAMEWL” checkbox	[false true] (Boolean) {false}
xrc.pexReduceROnly.specified	Boolean controlling xrc.pexReduceROnly. Available for R-only extraction <b>Options</b> page, “PEX Reduce ROnly” checkbox	[false true] (Boolean) {false}
xrc.pexReduceROnly.reductionBy.specified	Boolean controlling xrc.pexReduceROnly.reductionBy. Checkbox for “Further Reduction by”, which enables an optional keyword for PEX Reduce ROnly. The keyword is specified by xrc.pexReduceROnly.reductionBy.value. Available when xrc.pexReduceROnly.specified = true. <b>Options</b> page, PEX Reduce ROnly section	[false true] (Boolean) {false}
xrc.pexReduceROnly.reductionBy.value	The value for the “Further Reduction by” setting, which specifies an optional secondary keyword for the PEX Reduce ROnly statement. Available when xrc.pexReduceROnly.reductionBy.specified = true. <b>Options</b> page, PEX Reduce ROnly section	[SPARSIFY AGGRESSIVE] ( {SPARSIFY}
xrc.pexReduceTicer.specified	Boolean controlling xrc.pexReduceTicer. Adds the PEX Reduce Ticer statement. <b>Options</b> page, “PEX Reduce Ticer” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReduceTicer.frequency.value	The frequency in Hertz for PEX Reduce Ticer, expressed using scientific notation.  <b>Options</b> page, PEX Reduce Ticer section, Frequency	[] ( {10000000000000}
xrc.pexReduceTicer.maxDegree.specified	Checkbox for “Max Degree”, which adds the MAXDEG keyword to the PEX Reduce Ticer statement. The value is given by xrc.pexReduceTicer.maxDegree.value.  <b>Options</b> page, PEX Reduce Ticer section	[false true] (Boolean) {false}
xrc.pexReduceTicer.maxDegree.value	The value for the MAXDEG keyword of PEX Reduce Ticer. An integer from 2 to 12 that represents the number of parasitic resistors connected to a node. The default is 2.  Used when xrc.pexReduceTicer.maxDegree.specified = true.  <b>Options</b> page, PEX Reduce Ticer section, Max Degree	[] (integer) {2}
xrc.pexReduceTicer.mergeRange.specified	Checkbox for “Port Merge Range”, which adds the PORTMERGE keyword to the PEX Reduce Ticer statement. The required range value is given by xrc.pexReduceTicer.mergeRange.value.  <b>Options</b> page, PEX Reduce Ticer section	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReduceTicer.mergeRange.value	The range value for the PORTMERGE keyword of PEX Reduce Ticer. Specifies a more aggressive form of TICER reduction. Parameter range is specified in a derived time unit equal to R units times C units.  Used when xrc.pexReduceTicer.mergeRange.specified = true.  <b>Options</b> page, PEX Reduce Ticer section, Port Merge Range	[] ( {0}
xrc.pexReduceViaResistance.specified	Boolean controlling xrc.pexReduceViaResistance  <b>Options</b> page, “PEX Reduce Via Resistance” checkbox	[false true] (Boolean) {false}
xrc.pexReduceViaResistance.parameters	List of PEX REDUCE VIA RESISTANCE Statement values. that specifies how to handle clusters of identically-connected vias when calculating resistance.  <b>Options</b> page, PEX Reduce Via Resistance	[] (list) {}
xrc.pexReportCC.specified	Boolean controlling xrc.pexReportCC  <b>Options</b> page, “PEX Report Coupling Capacitance” checkbox	[false true] (Boolean) {false}
xrc.pexReportCC.ccNumber.specified	Boolean controlling xrc.pexReportCC.ccNumber  <b>Options</b> page, PEX Report Coupling Capacitance section, “Number” checkbox	[false true] (Boolean) {false}
xrc.pexReportCC.ccNumber.value	Specifies the maximum number of coupled capacitors to include in the report.  <b>Options</b> page, PEX Report Coupling Capacitance section, Number	[] (positive integer) {1000}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReportCC.ccReportSystem.value	Specifies how to use names in the report.  <b>Options</b> page, PEX Report Coupling Capacitance section, “Use Names From” dropdown	[SOURCE LAYOUT] () {SOURCE}
xrc.pexReportCC.ccSplitNet.value	Boolean that controls whether to report the ratio of coupling capacitance for Net B on a separate line.  <b>Options</b> page, PEX Report Coupling Capacitance section, “SPLIT_NET” checkbox	[false true] (Boolean) {false}
xrc.pexReportCC.ccThreshold.specified	Boolean controlling xrc.pexReportCC.ccThreshold  <b>Options</b> page, PEX Report Coupling Capacitance section, “Threshold” checkbox	[false true] (Boolean) {false}
xrc.pexReportCC.ccThreshold.value	Specifies the threshold for coupling capacitance shown in the report. Capacitance values below this threshold are not shown.  <b>Options</b> page, PEX Report Coupling Capacitance section, Threshold	[] () {}
xrc.pexReportCC.pexReportCCFile.value	Specifies the name of the coupling capacitance report.  <b>Options</b> page, PEX Report Coupling Capacitance section, Coupling Capacitance Report	[] () {}
xrc.pexReportNetSummary.specified	Boolean controlling xrc.pexReportNetSummary  <b>Outputs</b> page, “PEX Report Netsummary” checkbox	[false true] (Boolean) {false}
xrc.pexReportNetSummary.nsCell.specified	Boolean controlling xrc.pexReportNetSummary.nsCell  <b>Outputs</b> page, PEX Report Netsummary section, “Cell” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReportNetSummary.nsCell.value	Specifies a cell name for PEX REPORT NETSUMMARY. Only nets within that cell are reported. <b>Outputs</b> page, PEX Report Netsummary section, Cell	[] ( {}
xrc.pexReportNetSummary.nsColumns.specified	Boolean controlling xrc.pexReportNetSummary.nsColumns <b>Outputs</b> page, PEX Report Netsummary section, “Columns” checkbox	[false true] (Boolean) {false}
xrc.pexReportNetSummary.nsColumns.value	Specifies keywords to control the report output. <b>Outputs</b> page, PEX Report Netsummary section, “Columns” dropdown	[“COLUMNS BASIC”] [“COLUMNS ADVANCED”] ( {“COLUMNS BASIC”}
xrc.pexReportNetSummary.nsLayoutNames.value	A list of layout net names to be reported in PEX REPORT NETSUMMARY. <b>Outputs</b> page, PEX Report Netsummary section	[] (list) {}
xrc.pexReportNetSummary.nsNetFile.value	A list of net file net names reported in PEX REPORT NETSUMMARY. <b>Outputs</b> page, PEX Report Netsummary section, Summary Report File	[] ( {}
xrc.pexReportNetSummary.nsQuery.specified	Boolean controlling xrc.pexReportNetSummary.nsQuery <b>Outputs</b> page, PEX Report Netsummary section, “Query” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReportNetSummary.nsQuery.value	Specifies whether the nets' capacitances are itemized by sections in the summary report or only reported for the whole net.  <b>Outputs</b> page, PEX Report Netsummary section, “Query” dropdown	[DETAIL SUMMARY] () {DETAIL}
xrc.pexReportNetSummary.nsReportSystem.value	Specifies how to provide the nets in the report. <ul style="list-style-type: none"> <li>• LAYOUT — Nets are listed with xrc.pexReportNetSummary.nsLayoutNames.value</li> <li>• SOURCE — Nets are listed with xrc.pexReportNetSummary.nsSrcNetNames.value</li> <li>• NETFILE — Nets are listed with xrc.pexReportNetSummary.nsNetFile.value</li> </ul> <b>Outputs</b> page, PEX Report Netsummary section, “Specify nets from” dropdown	[ALL LAYOUT SOURCE NETFILE] () {ALL}
xrc.pexReportNetSummary.nsScale.specified	Boolean controlling xrc.pexReportNetSummary.nsScale  <b>Outputs</b> page, PEX Report Netsummary section, “Scale” checkbox	[false true] (Boolean) {false}
xrc.pexReportNetSummary.nsScale.value	Specifies a floating point number to multiply the capacitive values reported.  <b>Outputs</b> page, PEX Report Netsummary section, Scale	[] () {1.0}
xrc.pexReportNetSummary.nsScope.specified	Boolean controlling xrc.pexReportNetSummary.nsScope  <b>Outputs</b> page, PEX Report Netsummary section, “Scope” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReportNetSummary.nsScope.value	Specifies the scope of the report. <b>Outputs</b> page, PEX Report Netsummary section, “Scope” dropdown	[FULL LOCAL ALL] () {FULL}
xrc.pexReportNetSummary.nsSrcNetNames.value	A list of source net names to be reported in PEX REPORT NETSUMMARY. <b>Outputs</b> page, PEX Report Netsummary section	[] (list) {}
xrc.pexReportNetSummary.summaryFile.value	Specifies the name of a file to contain a report of the total capacitances. <b>Outputs</b> page, PEX Report Netsummary section, Summary Report File	[] () {}
xrc.pexReportPoint2Point.specified	Boolean controlling xrc.pexReportPoint2Point <b>Outputs</b> page, “PEX Report Point2Point” checkbox	[false true] (Boolean) {false}
xrc.pexReportPoint2Point.p2pInputFile.value	Specifies the name of a file containing entries for a point-to-point resistance calculation. The entries take the following format:  RESISTANCE net_name location net_name location <b>Outputs</b> page, PEX Report Point2Point section, Input File	[] () {}
xrc.pexReportPoint2Point.p2pOutputFile.specified	Boolean controlling xrc.pexReportPoint2Point.p2pOutputFile <b>Outputs</b> page, PEX Report Point2Point section, “Output File” checkbox	[false true] (Boolean) {false}
xrc.pexReportPoint2Point.p2pOutputFile.value	Specifies the name of a file to contain calculated resistances. <b>Outputs</b> page, PEX Report Point2Point, Output File	[] () {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexReportPoint2Point.p2pSamePort.specified	Boolean controlling xrc.pexReportPoint2Point.p2pSamePort <b>Outputs</b> page, PEX Report Point2Point, “SAMEPORT” checkbox	[false true] (Boolean) {false}
xrc.pexReportPoint2Point.p2pSamePort.value	Specifies whether to short ports with the same net name. <b>Outputs</b> page, PEX Report Point2Point, “SAMEPORT” dropdown	["SAMEPORT SHORT" “SAMEPORT OPEN”] () {SAMEPORT SHORT}
xrc.pexReportPoint2Point.useDBU.value	Specifies whether to use database units for the COORD location parameter in the input file. <b>Outputs</b> page, PEX Report Point2Point, “Use DBU” checkbox	[false true] (Boolean) {false}
xrc.pexResistanceParameters.specified	Boolean controlling xrc.pexResistanceParameters <b>Options</b> page, “PEX Resistance Parameters” checkbox	[false true] (Boolean) {false}
xrc.pexResistanceParameters.parameters	Specifies layer-specific parameters for resistance calculations. <b>Options</b> page, PEX Resistance Parameters	[] (list) {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexSupply.pexGroundLayoutNames.specified	<p>Boolean controlling xrc.pexSupply.pexGroundLayoutNames. Checkbox for “PEX Ground Layout Nets”, which adds the PEX Ground statement with the LAYOUT keyword. Used in inductance extraction.</p> <p>The ground nets are specified with xrc.pexSupply.pexGroundLayoutNames.value.</p> <p><b>Inductance</b> page, PEX Power Supply Nets section</p>	[false true] (Boolean) {false}
xrc.pexSupply.pexGroundLayoutNames.value	<p>A list of ground net names for the PEX Ground statement with the LAYOUT keyword. Used when xrc.pexSupply.pexGroundLayoutNames.specified = true.</p> <p><b>Inductance</b> page, PEX Power Supply Nets section</p>	[] (list) {}
xrc.pexSupply.pexGroundSourceNames.specified	<p>Boolean controlling xrc.pexSupply.pexGroundSourceNames. Checkbox for “PEX Ground Source Nets”, which adds the PEX Ground statement with the SOURCE keyword. Used in inductance extraction. The ground nets are specified with xrc.pexSupply.pexGroundSourceNames.value.</p> <p><b>Inductance</b> page, PEX Power Supply Nets section</p>	[false true] (Boolean) {false}
xrc.pexSupply.pexGroundSourceNames.value	<p>A list of ground net names for the PEX Ground statement with the SOURCE keyword. Used when xrc.pexSupply.pexGroundSourceNames.specified = true.</p> <p><b>Inductance</b> page, PEX Power Supply Nets section</p>	[] (list) {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexSupply.pexPowerLayoutNames.specified	Boolean controlling xrc.pexSupply.pexPowerLayoutNames. Checkbox for “PEX Power Layout Nets”, which adds the PEX Power statement with the LAYOUT keyword. Used in inductance extraction. The power nets are specified with xrc.pexSupply.pexPowerLayoutNames.value.  <b>Inductance</b> page, PEX Power Supply Nets section	[false true] (Boolean) {false}
xrc.pexSupply.pexPowerLayoutNames.value	A list of power net names for the PEX Power statement with the LAYOUT keyword. Used when xrc.pexSupply.pexPowerLayoutNames.specified = true.  <b>Inductance</b> page, PEX Power Supply Nets section	[] (list) {}
xrc.pexSupply.pexPowerSourceNames.specified	Boolean controlling xrc.pexSupply.pexPowerSourceNames. Checkbox for “PEX Power Source Nets”, which adds the PEX Source statement with the SOURCE keyword. Used in inductance extraction. The power nets are specified with xrc.pexSupply.pexPowerSourceNames.value.  <b>Inductance</b> page, PEX Power Supply Nets section	[false true] (Boolean) {false}
xrc.pexSupply.pexPowerSourceNames.value	A list of power net names for the PEX POWER statement with the SOURCE keyword. Used when xrc.pexSupply.pexPowerSourceNames.specified = true.  <b>Inductance</b> page, PEX Power Supply Nets section	[] (list) {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.pexXcellPrecedence.specified	Boolean controlling xrc.pexXcellPrecedence <b>Options</b> page, “PEX Xcell Precedence” checkbox	[false true] (Boolean) {false}
xrc.pexXcellPrecedence.best.value	Specifies whether to add the BEST keyword to the PEX Xcell Precedence statement. Specifies whether only the xcell specifications with the most explicit name match are applied to the corresponding cell. Used when xrc.pexXcellPrecedence.specified = true. <b>Options</b> page, PEX Xcell Precedence section, “Best” checkbox	[false true] (Boolean) {false}
xrc.pexXcellPrecedence.precedence.value	Specifies the primary keyword for the PEX Xcell Precedence statement.Specifies which file, either SVRF rule file or xcell list file, takes precedence if conflicting xcell definitions are encountered. Used when xrc.pexXcellPrecedence.specified = true. <b>Options</b> page, PEX Xcell Precedence section, “Precedence” dropdown	[STATEMENT FILE] 0 {STATEMENT }
xrc.reports.lvsReport.value	The filename for the LVS Report statement. Specifies the path to a file containing the results of an LVS run. Available when Calibre nmLVS is run as part of the parasitic extraction run. <b>Outputs</b> page, Report section, LVS Report	[] () {lvs.report}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.reports.lvsReportViewAfterRun.value	Specifies whether to automatically view the LVS Report after the run is complete. <b>Outputs</b> page, Report section, “View LVS Report after run finishes” checkbox	[false true] (Boolean) {false}
xrc.reports.pexReport.specified	Boolean controlling xrc.reports.pexReport <b>Outputs</b> page, Reports section, “PEX Report” checkbox	[false true] (Boolean) {false}
xrc.reports.pexReport.value	The filename for the PEX Report statement. Used when xrc.reports.pexReport.specified = true. <b>Outputs</b> page, Reports section, PEX Report	[] () {}
xrc.reports.pexReportSystem.value	Specifies the required keyword LAYOUTNAMES or SOURCENAMES for the PEX Report statement. Specifies how to derive names in the generated report. Used when xrc.reports.pexReport.specified = true and xrc.pexNetlist.system.value is set to SOURCEBASED or SCHEMATICONLY. <b>Outputs</b> page, Reports section, “Use Names from” dropdown	[LAYOUTNAMES SOURCENAMES] ( {SOURCENAMES}
xrc.reports.pexReportViewAfterRun.value	Specifies whether to automatically view the PEX Report after run is complete. Used when xrc.reports.pexReport.specified = true. <b>Outputs</b> page, Reports section, “View PEX Report after run finishes” checkbox	[false true] (Boolean) {false}
xrc.rulesFile.value	The path to the rules file. <b>Rules</b> page	[] () {rules}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.runDir.value	The run directory from which the tool performs its actions and all relative pathnames are resolved. <b>Rules</b> page	[] ( {. (current directory)}
xrc.runPrefs.stopAfterLVSMismatch.value	Specifies whether to stop parasitic extraction if the LVS step indicates a mismatch between layout and source. <b>Preferences</b> page, Run section, “Stop parasitic extraction if LVS indicates mismatch” checkbox	[false true] (Boolean) {false}
xrc.runSteps.runFmtStep.value	Specifies whether to run the formatter step of Calibre xRC to create the parasitic netlist from the PDB. <b>Run Control</b> page, PEX Steps section, “Create the parasitic netlist” checkbox	[true false] (Boolean) {true}
xrc.runSteps.runLvsStep.value	Specifies whether to create the PHDB. The PHDB is created with Calibre xRC if layout names are used in PEX Netlist (LAYOUTNAMES). It is created with Calibre nmLVS if source name are used (SOURCENAMES, SOURCEBASED, or SCHEMATICONLY). <b>Run Control</b> page, PEX Steps section, “Create the PHDB” checkbox	[true false] (Boolean) {true}
xrc.runSteps.runXrcStep.value	Specifies to run the PDB step of Calibre xRC. <b>Run Control</b> page, PEX Steps section, “Create the PDB” checkbox	[true false] (Boolean) {true}
xrc.runsetPrefs.runsetInfo.value	A string providing runset information. This information is displayed in the Load Runset dialog box. <b>Preferences</b> page, Runset section, Info	[] ( {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.runsetPrefs.runsetType.value	The user-defined runset type. <b>Preferences</b> page, Runset section, “Type” dropdown	[DRC LFD DFM LVS ERC CMP ] ( {}
xrc.rve.autoStart.value	Specifies whether to automatically show the results in Calibre RVE when the run finishes. <b>Run Control</b> page, RVE Options section, “Show Results in RVE” checkbox	[true false] (Boolean) {true}
xrc.rve.startRVEAfterBatch.value	Specifies whether to automatically show the results in Calibre RVE after a batch run finishes. <b>Run Control</b> page, RVE Options section, “Start RVE after batch run” checkbox	[false true] (Boolean) {false}
xrc.source.exportFromSourceViewer.value	Specifies whether to have the source viewer export the netlist before running the flow. <b>Inputs</b> page, Source Path section, “Export from source viewer” checkbox	[] ( {false}
xrc.source.extraSourceFiles.specified	Boolean controlling xrc.source.extraSourceFiles. The additional SPICE netlist files are specified with xrc.source.extraSourceFiles.value. <b>Database</b> page, Library section, “Additional SPICE Files” checkbox	[false true] (Boolean) {false}
xrc.source.extraSourceFiles.value	A list of additional SPICE netlist files. Used when xrc.source.extraSourceFiles.specified = true. <b>Database</b> page, Library section, Additional SPICE Files	[] ( {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.source.extraVerilogSourceFiles.specified	Boolean controlling xrc.source.extraVerilogSourceFiles. The additional VERILOG netlist files are specified with xrc.source.extraVerilogSourceFiles.value.  <b>Database</b> page, Library section, “Additional VERILOG Files” checkbox	[false true] (Boolean) {false}
xrc.source.extraVerilogSourceFiles.value	A list of additional VERILOG netlist files. Used when xrc.source.extraVerilogSourceFiles.specified = true.  <b>Database</b> page, Library section, Additional VERILOG Files	[] () {}
xrc.source.format.value	The format of the source netlist input. For VERILOG and MIXED input, the netlist is converted to SPICE format. <ul style="list-style-type: none"> <li>• SPICE — SPICE format</li> <li>• VERILOG — VERILOG format</li> <li>• MIXED — A combination of SPICE and VERILOG files.</li> </ul> <b>Inputs</b> page, Source Path section, “Source Format” dropdown	[SPICE VERILOG MIXED] () {SPICE}
xrc.source.sourceFile.value	The path to the top level source netlist file. When xrc.source.format.value is SPICE or MIXED, this is the top level SPICE file. When xrc.source.format.value is VERILOG, this is the top level VERILOG file. (For MIXED input, the VERILOG file is specified by xrc.source.verilogSourceFile.value.)  <b>Inputs</b> page, Source Path section	[] (environment variable valid) {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.source.topCell.value	The name of the primary (top) cell or subcircuit of the source netlist. This is the parameter for the Source Primary statement.  <b>Inputs</b> page, Source Path section, Top Cell	[] ( {}
xrc.source.topCellLibrary.value	The library name for the top cell of the source netlist. Used when exporting the source netlist from a schematic design tool such as Cadence Composer.  <b>Inputs</b> page, Source Path section	[] ( {}
xrc.source.topCellView.value	The view name for the top cell of the source netlist. Used when exporting the source netlist from a schematic design tool such as Cadence Composer.  <b>Inputs</b> page, Source Path section	[] ( {}
xrc.source.verilogSourceFile.value	The path to the VERILOG source netlist file when a combination of SPICE and VERILOG netlist input files is used (xrc.source.format.value = MIXED).  <b>Inputs</b> page, Source Path section	[] ( {}
xrc.sourceCase.caseSensitive.specified	Boolean controlling xrc.sourceCase.caseSensitive. The value for the Source Case statement is given by xrc.sourceCase.caseSensitive.value.  <b>Options</b> page, “Source Case” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.sourceCase.caseSensitive.value	Primary keyword for the Source Case statement, which specifies whether the source netlist is processed in a case-sensitive manner. Used when xrc.sourceCase.caseSensitive.specified = true.  <b>Options</b> page, “Source Case” dropdown	[NO YES] ( {NO}
xrc.supply.groundNames.specified	Boolean controlling xrc.supply.groundNames. The ground net names are specified by xrc.supply.groundNames.value.  <b>LVS</b> page, Power Supply section, “Ground Nets” checkbox	[false true] (Boolean) {false}
xrc.supply.groundNames.value	A list of ground net names for the LVS Ground Name statement. Used when xrc.supply.groundNames.specified = true.  <b>LVS</b> page, Power Supply section, Ground Nets	[] (list) {}
xrc.supply.powerNames.specified	Boolean controlling xrc.supply.powerNames. The ground net names are specified by xrc.supply.powerNames.value.  <b>LVS</b> page, Power Supply section, “Power Nets” checkbox	[false true] (Boolean) {false}
xrc.supply.powerNames.value	A list of power net names for the LVS Power Name statement. Used when xrc.supply.powerNames.specified = true.  <b>LVS</b> page, Power Supply section, Power Nets	[] (list) {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.svdb.specified	Boolean controlling xrc.svdb. Adds the Mask SVDB Directory statement.  This option enables the other options that begin with xrc.svdb  <b>Outputs</b> page, “Mask SVDB” checkbox	[true false] (Boolean) {true}
xrc.svdb.asciiixref.value	Checkbox for “Generate ASCII cross-reference files,” which adds the IXF, NXF, and SLPH keywords to the Mask SVDB Directory statement.  <b>Outputs</b> page, Mask SVDB section	[false true] (Boolean) {false}
xrc.svdb.cci.value	Checkbox for “Generate Calibre Connectivity Interface data,” which adds the CCI keyword to the Mask SVDB Directory statement.  <b>Outputs</b> page, Mask SVDB section	[false true] (Boolean) {false}
xrc.svdb.dirPath.value	The directory name for the Mask SVDB Directory statement.  <b>Outputs</b> page, Mask SVDB section	[] () {svdb}
xrc.svdb.pexIncrementalPDB.value	Checkbox for “Generate PDB incrementally”. If this setting is not selected, the PDB is recreated every time the PDB generation step is run.  <b>Outputs</b> page, Mask SVDB section	[false true] (Boolean) {false}
xrc.svdb.pinloc.specified	Checkbox for “Pin Location,” which includes the PINLOC or NOPINLOC keyword for the Mask SVDB Directory statement.  The parameter value is specified with xrc.svdb.pinloc.value.  <b>Outputs</b> page, Mask SVDB section	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.svdb.pinloc.value	Specifies the PINLOC or NOPINLOC keyword for the Mask SVDB Directory statement.  Used when xrc.svdb.pinloc.specified = true.  <b>Outputs</b> page, Mask SVDB section	[PINLOC NOPINLOC] ( {PINLOC}
xrc.svdb.query.value	Checkbox for “Generate cross-reference data for RVE,” which adds the QUERY keyword to the Mask SVDB Directory statement.  <b>Outputs</b> page, Mask SVDB section	[false true] (Boolean) {false}
xrc.svrfIncludes.specified	Boolean controlling xrc.svrfIncludes. The statements are specified with xrc.svrfIncludes.svrfStatements.value or xrc.svrfIncludes.tvfStatements.value.  <b>Options</b> page, “Include Rule Statements” checkbox	[false true] (Boolean) {false}
xrc.svrfIncludes.statementType.value	Available only when the main rule file is TVF.  Dropdown selection for “Type”, which specifies the language of the included rule file statements. SVRF statements are included within a VERBATIM block in the generated control file.  <b>Options</b> page, Include Rule Statements section	[none SVRF TVF] ( {none}
xrc.svrfIncludes.svrfStatements.value	SVRF statements to be included in the control file generated by Calibre Interactive.  Used when the main rule file is SVRF, or when the main rule file is TVF and xrc.svrfIncludes.statementType.value = SVRF.  <b>Options</b> page, Include Rule Statements section	[] ( {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.svrfIncludes.tvfStatements.value	TVF statements to be included in the control file generated by Calibre Interactive. Used when the main rule file is TVF and xrc.svrfIncludes.statementType.value = TVF.  <b>Options</b> page, Include Rule Statements section	[] ( {})
xrc.traceProperty.parameters	List of parameters that specify which device properties are compared during LVS and how the comparison is performed.  <b>LVS</b> page, Trace Property section	[] (list) {}
xrc.traceProperty.specified	Boolean controlling xrc.traceProperty <b>LVS</b> page, “Trace Property” checkbox	[false true] (Boolean) {false}
xrc.v2LVSLibraries.sPICEIncFiles.s pecified	Boolean controlling xrc.v2LVSLibraries.sPICEIncFiles. Files are specified with xrc.v2LVSLibraries.sPICEIncFiles.va lue.  <b>V2LVS</b> page, Libraries section, “Include SPICE Files” checkbox	[false true] (Boolean) {false}
xrc.v2LVSLibraries.sPICEIncFiles. value	List of SPICE files that are included in the v2lvs command line with the -s option.  Used when xrc.v2LVSLibraries.sPICEIncFiles.sp ecified = true.  <b>V2LVS</b> page, Libraries section, Include SPICE Files	[] ( {})

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.v2LVSLibraries.sPICELibFiles.specified	Boolean controlling xrc.v2LVSLibraries.sPICELibFiles. Files are specified with xrc.v2LVSLibraries.sPICELibFiles.value.  <b>V2LVS</b> page, Libraries section, “SPICE Library Files” checkbox	[false true] (Boolean) {false}
xrc.v2LVSLibraries.sPICELibFiles.value	List of SPICE files that are included in the v2lvs command line with the -lsr or -lsp option. (See xrc.v2LVSOPTIONS.useRangeMode.value)  Used when xrc.v2LVSLibraries.sPICELibFiles.specified = true.  <b>V2LVS</b> page, Libraries section, SPICE Library Files	[] () {}
xrc.v2LVSLibraries.vLOGLibFiles.specified	Boolean controlling xrc.v2LVSLibraries.vLOGLibFiles. Files are specified with xrc.v2LVSLibraries.vLOGLibFiles.value.  <b>V2LVS</b> page, Libraries section, “Verilog Library Files” checkbox	[false true] (Boolean) {false}
xrc.v2LVSLibraries.vLOGLibFiles.value	List of Verilog files that are included in the v2lvs command line with the -l option.  Used when xrc.v2LVSLibraries.vLOGLibFiles.specified = true.  <b>V2LVS</b> page, Libraries section, Verilog Library Files	[] () {}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.v2LVSOptions.addPinNames.specified	Boolean controlling xrc.v2LVSOptions.addPinNames. The pins are specified with xrc.v2LVSOptions.addPinNames.value.  <b>V2LVS</b> page, Options section, “Add pins to each subckt” checkbox	[false true] (Boolean) {false}
xrc.v2LVSOptions.addPinNames.value	List of pin names that are added with the -addpin argument to the v2lvs command line.  Used when xrc.v2LVSOptions.addPinNames.specified = true.  <b>V2LVS</b> page, Options section, Add pins to each subckt	[] (list) {}
xrc.v2LVSOptions.connectUPins.value	Checkbox for “Create numbered nets for unconnected pins,” which adds the -n argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
xrc.v2LVSOptions.dontConnectPower.value	Checkbox for “Don’t connect supply0/supply1 nets to global supply,” which adds the -sk argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
xrc.v2LVSOptions.generateTemplate.value	Checkbox for “Generate xRC source template file”, which adds the -t argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
xrc.v2LVSOptions.keepBackSlash.value	Checkbox for “Preserve backslashes in escaped identifiers,” which adds the -b argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.v2LVSOptions.positionalPins.value	Checkbox for “Use positional pin order (don’t use \$PINS)”, which adds the -i argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
xrc.v2LVSOptions.translateAlways.value	Checkbox for “Always translate before running Calibre.” If this setting is false, the translator does not run if the Verilog files are older than the translated SPICE file.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
xrc.v2LVSOptions.useCBLicense.value	Checkbox for “Prefer Calibre-CB license during license search”, which adds the -cb argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
xrc.v2LVSOptions.useRangeMode.value	Checkbox for “Use aggregate mode for bus bits”.  Specifies how SPICE library files are added on the v2lvs command line: <ul style="list-style-type: none"><li>• true — Use the -lsr argument, for range mode.</li><li>• false — Use the -lsp argument, for pin mode.</li></ul> <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}
xrc.v2LVSOptions.warningLevel.value	Specifies the amount of warning level output for the v2lvs translator. This is the value for the -w argument.  <b>V2LVS</b> page, Options section, Warning Level	[0 1 2 3 4] (positive integer) {2}
xrc.v2LVSOptions.warningsAreErrors.value	Checkbox for “Treat warnings as errors”, which adds the -werror argument to the v2lvs command line.  <b>V2LVS</b> page, Options section	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.v2LVSStrings.achars.specified	Checkbox for “Change array delimiters to”. The value is given by xrc.v2LVSStrings.achars.value. <b>V2LVS</b> page, Strings section	[false true] (Boolean) {false}
xrc.v2LVSStrings.achars.value	The value for the -a option in the v2lvs command line, which changes the array delimiter from the default []. The first character replaces [, the optional second character replaces ]. Used when xrc.v2LVSStrings.achars.specified = true. <b>V2LVS</b> page, Strings section. Change array delimiters to	[] () {}
xrc.v2LVSStrings.cchars.specified	Checkbox for “Change illegal SPICE characters to”. The value is given by xrc.v2LVSStrings.cchars.value. <b>V2LVS</b> page, Strings section	[false true] (Boolean) {false}
xrc.v2LVSStrings.cchars.value	The value for the -c option in the v2lvs command line. Enter one or two characters enclosed in quotes. To include single quotes in the command line, as recommended, enclose the characters in single quotes, then outer double quotes. Used when xrc.v2LVSStrings.cchars.specified = true. <b>V2LVS</b> page, Strings section, Change illegal SPICE characters to	[] () {}
xrc.v2LVSStrings.pprefix.specified	Checkbox for “Prefix for gate level primitives”. The value is given by xrc.v2LVSStrings.pprefix.value. <b>V2LVS</b> page, Strings section	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.v2LVSStrings.pprefix.value	The value for the -p option in the v2lvs command line, which specifies a prefix to add to Verilog gate level primitive cells.  Used when xrc.v2LVSStrings.pprefix.specified = true.  <b>V2LVS</b> page, Strings section, Prefix for gate level primitives	[] ( {}
xrc.v2LVSStrings.s0Net.specified	Checkbox for “Connect pins with 0 value to net”. The value is given by xrc.v2LVSStrings.s0Net.value.  <b>V2LVS</b> page, Strings section	[false true] (Boolean) {false}
xrc.v2LVSStrings.s0Net.value	The value for the -s0 option in the v2lvs command line.  Used when xrc.v2LVSStrings.s0Net.specified = true.  <b>V2LVS</b> page, Strings section, Connect pins with 0 value to net	[] ( {}
xrc.v2LVSStrings.s1Net.specified	Checkbox for “Connect pins with 1 value to net”. The value is given by xrc.v2LVSStrings.s1Net.value.  <b>V2LVS</b> page, Strings section	[false true] (Boolean) {false}
xrc.v2LVSStrings.s1Net.value	The value for the -s1 option in the v2lvs command line. Specifies the default net name for mapping to pin connections with a value of 1.  <b>V2LVS</b> page, Strings section, Connect pins with 1 value to net	[] ( {}
xrc.v2LVSStrings.uprefix.specified	Checkbox for “Prefix for unnamed pins and gates”. The value is given by xrc.v2LVSStrings.uprefix.value.  <b>V2LVS</b> page, Strings section	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.v2LVSStrings.uprefix.value	The value for the “Prefix for unnamed pins and gates” option, which adds the -u argument to the v2lvs command line.  <b>V2LVS</b> page, Strings section	[] ( {}
xrc.v2LVSStrings.userOptions.specified	Checkbox for “More v2lvs program options”. The value is given by xrc.v2LVSStrings.userOptions.value.  <b>V2LVS</b> page, Strings section	[false true] (Boolean) {false}
xrc.v2LVSStrings.userOptions.value	A list of options that are included without modification in the v2lvs command line.  Used when xrc.v2LVSStrings.userOptions.specified = true.  <b>V2LVS</b> page, Strings section	[] (list) {}
xrc.xCells.xCellsFile.value	Specifies the path to the file containing a list of cells to be preserved during extraction. Available when xrc.pexExtractionType.level.value = H.  <b>Inputs</b> page, xCells section	[] ( {}
xrc.xrcExtractTemp.circuit.value	The circuit temperature in degrees Celsius.  <b>Options</b> page, PEX Extract Temperature section, Circuit Temperature	[] ( {25}
xrc.xrcExtractTemp.range.specified	Boolean controlling xrc.xrcExtractTemp.range  <b>Options</b> page, PEX Extract Temperature section, “Range” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.xrcExtractTemp.range.value	The three temperatures, in degrees Celcius, used in calculating TC1 and TC2. <b>Options</b> page, PEX Extract Temperature section, Range	[-55 27 155] ( {}
xrc.xrcExtractTemp.specified	Boolean controlling xrc.xrcExtractTemp <b>Options</b> page, “PEX Extract Temperature” checkbox	[false true] (Boolean) {false}
xrc.xrcMode.mode.value	Specifies the run mode for xRC. <b>Outputs</b> page, “xRC Mode” dropdown	[“xRC” “xACT 3D” “xACT 3D Hybrid”] ( {xRC}
xrc.xrcProbeSpecification.probeSpecification.value	A list of probe point column lists. Each column list contains settings for rows in the probe table.  {SELECTED {0/1 ...}} {NAME {probe_name ...}} {CELL {cell_name ...}} {LOCATION {{x y} ...}} {LAYER {layer_name ...}} <b>Probes</b> page, Probe Specification section	[] ( {}
xrc.xrcProbeSpecification.protectProbes.value	Specifies whether to prevent probe points from being removed when a probe point is at the same node as a pin or port.  <b>Probes</b> page, Probe Specification section, “Protect Probes” checkbox	[false true] (Boolean) {false}

**Table C-5. PEX Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
xrc.xrcProbeSpecification.replaceProbesWithPorts.value	Specifies whether to replace probes on the net represented as subnodes (* S) with ports (* P) when extracting a DSPF netlist.  Probes page, Probe Specification section, “Replace Probs with Ports” checkbox	[false true] (Boolean) {false}

## Trigger Functions in the Calibre Interactive Runset

Trigger definitions for Calibre Interactive are saved to the runset as object definitions and function calls.

### Trigger Definitions: add...Trigger Functions

The commands that define a trigger function in the runset are the same as those that define a trigger function in the configuration file. However, in the runset, the string “tool” in the trigger function name is replaced with drc, lvs, perc, xrc, or pex for the DRC, LVS, PERC, PEX, and xACT applications.

This is an example of a trigger definition in the runset. The trigger runs before Calibre xACT execution.

```
var preTrigger_2_runXact_trig = pex.addProcessPreTrigger("runXact",
                                                       "preTrigger_2")
preTrigger_2_runXact_trig.command = "echo \"Running on cell %L\""
```

The trigger object is created with the addProcessPreTrigger function, and the trigger command is defined with the command property. See the [add ... Trigger](#) command for details.

### Options for Trigger Execution

You can set several options for post-execution triggers. These options are set with the following functions, where <app> indicates the application and is one of drc, lvs, perc, xrc, or pex.

Runset Entry	Definition
<app>.runPostTriggerInBackground("run_mode", true)	Run the post-execution trigger selected by <i>run_mode</i> in the background.

Runset Entry	Definition
<code>&lt;app&gt;.runPostTriggerOnTerm("run_mode", true)</code>	Run the post-execution trigger selected by <i>run_mode</i> even if the Calibre run is terminated.
<code>&lt;app&gt;.remotePostTrigger("run_mode", true)</code>	Run the post-execution trigger selected by <i>run_mode</i> on the remote host when Calibre is run on a remote host.

The *run\_mode* parameter indicates which execution step the function applies to. The *run\_mode* parameter can have these values, depending on the application:

drc: runDRC

lvs: runLVS

perc: runPERC

xrc: runPreExec, runLvs, runXrc, runFmt, runPostExec

pex: runLvs, runXact

See the [add ... Trigger](#) function and “[Internal Trigger Execution in Calibre Interactive](#)” on page 415 for more explanation of the trigger run modes for the parasitic extraction applications.

## Example

This trigger is for Calibre Interactive xACT. The trigger runs after Calibre LVS execution and in the shell environment. The function process\_LVSreport is defined in the shell environment and takes an argument of the LVS report name, given by the %n replaceable parameter. The trigger is run in the background.

```
var postTrigger_1_runLvs_trig = pex.addProcessPostTrigger ("runLvs",
                                                       "postTrigger_1")
postTrigger_1_runLvs_trig.command = "process_LVSreport %n"
pex.runPostTriggerInBackground ("runLvs", true)
```

# Template Definitions

File-naming templates for the Calibre Interactive GUI are saved to the runset as function calls.

## Function Syntax

```
<app>.setNameTemplate("option", "template_value")
```

where *<app>* is one of drc, lvs, perc, xrc, or pex (for xACT).

## Parameters

- *option*

Specifies which template is being defined. The *option* can be one of the values listed in the following table. The corresponding runset option is also given—this is the runset option that the resolved template value is saved to.

<i>option</i>	Definition (Corresponding Runset Option)	Default
<b>General Template Settings</b>		
nt_controlFile	The control file template. (cmn.rulesPrefs.controlFile.value)	No default
nt_rules	The rule file template. (<app>.rulesFile.value)	No default
nt_runDir	The run directory template. (<app>.runDir.value)	No default
nt_transcriptFile	The transcript file template (cmn.transcriptPrefs.transcriptEchoFile. value)	No default
<b>Input and Design Database Template Settings</b>		
nt_defFile (DRC, LVS, PERC)	The template for the DEF file for LEF/ DEF databases. (cmn.layout.defFiles.value)	%l.def
nt_efLayVwr	The template for the “Export from layout viewer setting”. The <b>template_value</b> is an integer: <ul style="list-style-type: none"> <li>• 1 — Yes</li> <li>• 0 — No</li> <li>• -1 — Runset</li> </ul> (<app>.layout.exportFromLayoutView er.value)	1
nt_efSrcVwr (not DRC)	The template for the “Export from source viewer setting”. The <b>template_value</b> is an integer: <ul style="list-style-type: none"> <li>• 1 — Yes.</li> <li>• 0 — No</li> <li>• -1 — Runset</li> </ul> (<app>.source.exportFromSourceView er.value)	-1

<b><i>option</i></b>	<b>Definition (Corresponding Runset Option)</b>	<b>Default</b>
nt_layFile	The template for the layout filename. (<app>.layout.layoutFile.value)	%l.calibre.db
nt_layLib	The template for the layout library. (<app>.layout.topCellLibrary.value)	%L
nt_layView	The template for the layout view. (<app>.layout.topCellView.value)	%V
nt_srcFile (not DRC)	The template for the source filename. (<app>.source.sourceFile.value)	%s.src.net
nt_srcLib (not DRC)	The template for the source library. (<app>.source.topCellLibrary.value)	%S
nt_srcView (not DRC)	The template for the source view. (<app>.source.topCellView.value)	%W
<b>DRC-Specific Template Settings</b>		
nt_dfmDefaults (DRC)	The template for the default filename in the DFM Defaults statement. (drc.dfmDefaults.resultsFile.value)	%l.dfmDefaults.db
nt_drcResults	The template for the DRC results database filename. (drc.drcdb.resultsFile.value)	%l.drc.results
nt_drcSummaryReport	The template for the DRC Summary Report filename. (drc.reports.drcSummaryReport.value)	%l.drc.summary
nt_rhdb	The template for the reusable hierarchical database (RHDB) filename. (drc.rhdb.fileName.value)	%l.rhdb
<b>LVS-Specific Template Settings</b>		
nt_ercResults	The template for the ERC Results Database filename. (lvs.ercdb.resultsFile.value)	%l.erc.results
nt_ercSummaryReport	The template for the ERC Summary Report filename. (lvs.ercSummaryReport.report.value)	%l.erc.summary

<i>option</i>	<b>Definition (Corresponding Runset Option)</b>	<b>Default</b>
nt_hCellFile	The template for the hcell file. (lvs.hCells.hCellsFile.value)	No default
nt_layNetlist	The template for the layout netlist. (lvs.layout.layoutNetlist.value)	%l.sp
nt_lvsReport	The template for the LVS Report filename. (lvs.reports.lvsReport.value)	%l.lvs.report
nt_maskDir	The template for the Mask SVDB Directory name. Relative paths are relative to the run directory. (lvs.svdb.dirPath.value)	svdb

#### PERC-Specific Template Settings

nt_dfmdb	The template for DFM Database directory path. (perc.dfmdb.dirPath.value)	dfmdb
nt_hCellFile	The template for the hcell file. (perc.hCells.hCellsFile.value)	No default
nt_layNetlist	The template for the layout netlist (perc.layout.layoutNetlist.value)	%l.sp
nt_maskDir	The template for the Mask SVDB Directory name. Relative paths are relative to the run directory. (perc.svdb.dirPath.value)	svdb
nt_percReport	The template for the PERC Report filename. (perc.reports.percReport.value)	%l.perc.report

#### PEX-Specific Template Settings

nt_calibreView	Template for Calibre View	calibre
nt_hCellFile	The template for the hcell file. (xrc.hCells.hCellsFile.value)	No default
nt_lvsReport	The template for the LVS Report filename. (xrc.reports.lvsReport.value)	%l.lvs.report

<b><i>option</i></b>	<b>Definition (Corresponding Runset Option)</b>	<b>Default</b>
nt_maskDir	The template for the SVDB directory name.  (xrc.svdb.dirPath.value)	svdb
nt_netFile	The template for the filename of the extracted parasitic netlist.  (xrc.pexNetlist.netlistFile.value)	%l.pex.netlist
nt_pexReport	The template for the PEX Report filename.  (xrc.reports.pexReport.value)	%l.pex.report
nt_xCellFile	The template for the xcell file.  (xrc.xCells.xCellsFile.value)	No default

#### xACT-Specific Template Settings

nt_calibreView	Template for Calibre View	calibre
nt_hCellFile	The template for the hcell file.  (pex.hCells.hCellsFile.value)	No default
nt_lvsReport	The template for the LVS Report filename.  (pex.reports.lvsReport.value)	%l.lvs.report
nt_maskDir	The template for the SVDB directory name.  (pex.svdb.dirPath.value)	svdb
nt_netFile	The template for the filename of the extracted parasitic netlist.  (pex.pexNetlist.netlistFile.value)	%l.pex.netlist
nt_pexReport	The template for the PEX Report filename.  (pex.reports.pexReport.value)	%l.pex.report
nt_xCellFile	The template for the xcell file.  (pex.xCells.xCellsFile.value)	No default

- *template\_value*

The value of the template, which is a string for most template options. The string value can include the replaceable parameters given in the following table. The value can also

include an environment variable if the relevant GUI option supports environment variables.

Unlike the other template options, the *template\_value* for export settings (nt\_efLayVwr and nt\_efSrcVwr) is an integer, not a string. The possible values are described in the preceding table of option values.

**Table C-6. Replaceable Parameters in Template Values for Calibre Interactive**

Parameter	Definition
%l	The primary cell in the layout database.
%s	The primary cell in the source netlist.
%L	The name of the layout library for OpenAccess databases.
%V	The name of the layout view for OpenAccess databases.
%S	The name of the source library for OpenAccess databases.
%W	The name of the source view for OpenAccess databases.
<b>Replaceable parameters for the output parasitic netlist filename</b>	
%X	The extraction type. %X is replaced by these values: <ul style="list-style-type: none"> <li>• rcc — R+C+CC extraction</li> <li>• rc — R+C extraction</li> <li>• r — R only extraction</li> <li>• cc — C+CC extraction</li> <li>• norc — No R/C extraction</li> </ul>
%x	The parasitic netlist format. %x is replaced by these values: <ul style="list-style-type: none"> <li>• cv — Calibre View netlist</li> <li>• dspf — DSPF format</li> <li>• spi — Eldo format</li> <li>• sp — HSPICE format</li> <li>• scs — SPECTRE format</li> <li>• spef — SPEF format</li> <li>• spice — SPICE format</li> </ul>

**Table C-6. Replaceable Parameters in Template Values for Calibre Interactive (cont.)**

Parameter	Definition
%o	The netlist format option. %o is replaced by these values: <ul style="list-style-type: none"> <li>• ar — ADiT Rail netlist compatibility</li> <li>• pt — PrimeTime netlist compatibility</li> <li>• hsim — HSIM netlist compatibility</li> <li>• us — ULTRASIM netlist compatibility</li> <li>• totem — TOTEM netlist compatibility</li> <li>• qc — QUICKCAP netlist compatibility</li> <li>• ng — Net Geometry</li> </ul>

### Example 1

```
pex.setNameTemplate("nt_layFile", "%l.exported.gds")
```

This sets the template for the layout file in Calibre Interactive xACT. The template is applied when Calibre Interactive xACT is invoked from a design tool. The replaceable parameter %l is replaced with the name of the open cell in the design tool.

If the open cell is opamp, the layout filename on the **Inputs** page is *opamp.exported.gds*. When the runset is saved, this is saved for the corresponding runset option:

```
pex.layout.layoutFile.value = "opamp.exported.gds"
```

### Example 2

For parasitic extraction runs, the %X, %x, and %o replaceable parameters can be used to indicate the extraction type and netlist format. This sets the template for the parasitic netlist in Calibre Interactive PEX:

```
xrc.setNameTemplate("nt_netFile", "%l.pex.%X.%x")
```

For an open cell named opamp, RCC extraction, and a SPEF netlist format, the parasitic netlist file name is *opamp.pex.rcc.spef*. When the runset is saved, this is saved for the corresponding runset option:

```
xrc.pexNetlist.netlistFile.value = "opamp.pex.rcc.spef"
```

## Waiver Setup Runset Options

Most waiver setup runset options start with cmn.waiver. The waiver options are used in Calibre Interactive nmDRC and in Calibre Interactive nmLVS with ERC enabled.

**Table C-7. Waiver Setup Runset Options**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.miscPrefs.waiverSetupFileName.specified	Boolean controlling cmn.miscPrefs.waiverSetupFileName Preferences page, “Specify waiver setup file name” checkbox	[false true] (Boolean) {false}
cmn.miscPrefs.waiverSetupFileName.value	The user-defined name of the waiver setup file created by Calibre Interactive when creating or using waivers.  Preferences page, “Specify waiver setup file name”	[] () {}
cmn.waiverCreation.addWaiverHierarchy.value	Specifies whether to place waiver cells in an intermediate cell. (ADD_WAIVER_HIERARCHY)  Waivers page, Waiver Creation, “Place waiver cells in intermediate cells”	[false true] (Boolean) {false}
cmn.waiverCreation.magnifyTextSpacing.value	Specifies whether to apply a magnification factor for spacing between text. (MAGNIFY_TEXT_SPACING)  Waivers page, Waiver Creation, “Magnify Text Spacing”	[false true] (Boolean) {false}
cmn.waiverCreation.textMagnification.value	Specifies a magnification factor when creating a GDS waiver database. (TEXT_MAG)  Waivers page, Waiver Creation, “Text Magnification”	[] () .005}
cmn.waiverCreation.waiverCellFiles.value	The filenames for the waiver cell files. (WAIVER_CELLS)  Waivers page, Waiver Creation, “Waiver Cell Files”	[] () {}
cmn.waiverCreation.waiverSummaryFile.value	The name of the waiver summary report file created during waiver creation. (WAIVER_SUMMARY)  Waivers page, Waiver Creation, “Summary File”	[] () {waiver.summary}

**Table C-7. Waiver Setup Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.waiverDescriptionFiles.criteriaFile.value	The filenames of the waiver criteria file(s). (WAIVER_CRITERIA) <b>Waivers</b> page, Description Files, “Waiver Criteria Files”	[] ( {}
cmn.waiverDescriptionFiles.waiverTmpDir.value	Path of the directory in which temporary files will be created during waiver operations. (TEMP_DIR) <b>Waivers</b> page, Description Files, “Temp Files Directory”	[] ( {}
cmn.waiverInputs.useWaivers.value	Specifies whether to run Calibre with Calibre Auto-Waivers applied. (The -waiver command line option.) <b>Inputs</b> page, Waivers, “Use Waivers when running <app>”	[false true] (Boolean) {false}
cmn.waiverLayerOperations.waiverLayerOperationDFMAnalyze.value	Specifies whether to output waiver geometry output from DFM Analyze operations during waiver generation. <ul style="list-style-type: none"><li>• true — Output. (NOT_IGNORE)</li><li>• false — Do not output. (IGNORE)</li></ul> <b>Waivers</b> page, Layer Operation, “DFM Analyze”	[true false] (Boolean) {true}
cmn.waiverLayerOperations.waiverLayerOperationDFMCopy.value	Specifies whether to output waiver geometry output from DFM Copy operations during waiver generation. <ul style="list-style-type: none"><li>• true — Output. (NOT_IGNORE)</li><li>• false — Do not output. (IGNORE)</li></ul> <b>Waivers</b> page, Layer Operation, “DFM Copy”	[true false] (Boolean) {true}
cmn.waiverLayerOperations.waiverLayerOperationDFMProperty.value	Specifies whether to output waiver geometry output from DFM Property operations during waiver generation. <ul style="list-style-type: none"><li>• true — Output. (NOT_IGNORE)</li><li>• false — Do not output. (IGNORE)</li></ul> <b>Waivers</b> page, Layer Operation, “DFM Property”	[true false] (Boolean) {true}

**Table C-7. Waiver Setup Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.waiverLayerOperations.waiverLayerOperationDFMRDB.value	<p>Specifies whether to output waiver geometry output from DFM RDB operations during waiver generation.</p> <ul style="list-style-type: none"> <li>• true — Output. (NOT_IGNORE)</li> <li>• false — Do not output. (IGNORE)</li> </ul> <p><b>Waivers page, Layer Operation, “DFM RDB”</b></p>	[true false] (Boolean) {true}
cmn.waiverLayerOperations.waiverLayerOperationDFMSpace.value	<p>Specifies whether to output waiver geometry output from DFM Space operations during waiver generation.</p> <ul style="list-style-type: none"> <li>• true — Output. (NOT_IGNORE)</li> <li>• false — Do not output. (IGNORE)</li> </ul> <p><b>Waivers page, Layer Operation, “DFM Space”</b></p>	[true false] (Boolean) {true}
cmn.waiverLayerOperations.waiverLayerOperationNetArea.value	<p>Specifies whether to output waiver geometry output from Net Area operations during waiver generation.</p> <ul style="list-style-type: none"> <li>• true — Output. (NOT_IGNORE)</li> <li>• false — Do not output. (IGNORE)</li> </ul> <p><b>Waivers page, Layer Operation, “Net Area”</b></p>	[false true] (Boolean) {false}
cmn.waiverLayerOperations.waiverLayerOperationNetAreaRatio.value	<p>Specifies whether to output waiver geometry output from Net Area Ratio operations during waiver generation.</p> <ul style="list-style-type: none"> <li>• true — Output. (NOT_IGNORE)</li> <li>• false — Do not output. (IGNORE)</li> </ul> <p><b>Waivers page, Layer Operation, “Net Area Ratio”</b></p>	[false true] (Boolean) {false}
cmn.waiverLayerOperations.waiverLayerOperationNetAreaRatioAccumulate.value	<p>Specifies whether to output waiver geometry output from Net Area Ratio Accumulate operations during waiver generation.</p> <ul style="list-style-type: none"> <li>• true — Output. (NOT_IGNORE)</li> <li>• false — Do not output. (IGNORE)</li> </ul> <p><b>Waivers page, Layer Operation, “Net Area Ratio Accumulate”</b></p>	[false true] (Boolean) {false}

**Table C-7. Waiver Setup Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.waiverLayerOperations.waiverLayerOperationTVF.value	<p>Specifies whether to output waiver geometry output from TVF layer operations during waiver generation.</p> <ul style="list-style-type: none"> <li>• true — Output. (NOT_IGNORE)</li> <li>• false — Do not output. (IGNORE)</li> </ul> <p><b>Waivers</b> page, Layer Operations, “TVF”</p>	[false true] (Boolean) {false}
cmn.waiverLayersAndText.layersAndTexts.calibreVersionTextLayer.value	<p>Layer number for Calibre version objects. (CALIBRE_VERSION_LAYER)</p> <p><b>Waivers</b> page, Layers and Text, “Calibre Version Layer”</p>	[] (positive integer) {1234}
cmn.waiverLayersAndText.layersAndTexts.calibreVersionTextType.value	<p>Texttype number for Calibre version objects. (CALIBRE_VERSION_TEXTTYPE)</p> <p><b>Waivers</b> page, Layers and Text, “Calibre Version Layer”</p>	[] (positive integer) {7}
cmn.waiverLayersAndText.layersAndTexts.calibreWaiverNumber.specified	<p>Boolean controlling cmn.waiverLayersAndText.layersAndTexts.calibreWaiverNumber</p> <p>Not available for waiver generation.</p> <p><b>Waivers</b> page, Layers and Text, “Specify Internal Waiver Layer” checkbox</p>	[true false] (Boolean) {true}
cmn.waiverLayersAndText.layersAndTexts.calibreWaiverNumber.value	<p>Layer number for internal Calibre internal use during waiver processing. (CALIBRE_WAIVER_NUMBER)</p> <p>Not available for waiver generation.</p> <p><b>Waivers</b> page, Layers and Text, “Specify Internal Waiver Layer”</p>	[] (positive integer) {2000}
cmn.waiverLayersAndText.layersAndTexts.checksumTextLayer.value	<p>Layer Number for Checksum objects. (CHECKSUM_TEXT_LAYER)</p> <p><b>Waivers</b> page, Layers and Text, “Checksum Layer”</p>	[] (positive integer) {1234}

**Table C-7. Waiver Setup Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.waiverLayersAndText.layersAndTexts.checksumTextType.value	Texttype number for Checksum objects. (CHECKSUM_TEXTTYPE) <b>Waivers</b> page, Layers and Text, “Checksum Layer”	[] (positive integer) {1}
cmn.waiverLayersAndText.layersAndTexts.commentTextLayer.value	Layer number for Comment objects. (COMMENT_TEXT_LAYER) Not available for waiver generation. <b>Waivers</b> page, Layers and Text, “Comment Layer”	[] (positive integer) {1234}
cmn.waiverLayersAndText.layersAndTexts.commentTextType.value	Texttype number for Comment objects. (COMMENT_TEXTTYPE) Not available for waiver generation. <b>Waivers</b> page, Layers and Text, “Comment Layer”	[] (positive integer) {5}
cmn.waiverLayersAndText.layersAndTexts.criteriaTextLayer.value	Layer number for Criteria objects. (CRITERIA_TEXT_LAYER) <b>Waivers</b> page, Layers and Text, “Criteria Layer”	[] (positive integer) {1234}
cmn.waiverLayersAndText.layersAndTexts.criteriaTextType.value	Texttype number for Criteria objects. (CRITERIA_TEXTTYPE) <b>Waivers</b> page, Layers and Text, “Criteria Layer”	[] (positive integer) {2}
cmn.waiverLayersAndText.layersAndTexts.datatypeNumber.value	Datatype number for Waiver Shape objects. (DATATYPE_NUMBER) <b>Waivers</b> page, Layers and Text, “Waiver Shape Layer”	[] (positive integer) {5678}
cmn.waiverLayersAndText.layersAndTexts.dateTextLayer.value	Layer number for Date objects. (DATE_TEXT_LAYER) Not available for waiver generation. <b>Waivers</b> page, Layers and Text, “Date Layer”	[] (positive integer) {1234}

**Table C-7. Waiver Setup Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.waiverLayersAndText.layersAndTexts.dateTextType.value	Texttype number for Date objects. (DATE_TEXTTYPE)  Not available for waiver generation. <b>Waivers</b> page, Layers and Text, “Date Layer”	[] (positive integer) {4}
cmn.waiverLayersAndText.layersAndTexts.ipmatchTextType.value	Texttype number for IP Match objects. These objects use the same layer number as Waiver Shapes. (IPMATCH_TEXTTYPE)  <b>Waivers</b> page, Layers and Text, “IP Match Texttype”	[] (positive integer) {6}
cmn.waiverLayersAndText.layersAndTexts.ruleChecksumTextLayer.value	Layer number for Rule checksum objects. (RULE_CHECKSUM_TEXT_LAYER)  <b>Waivers</b> page, Layers and Text, “Rule checksum Layer”	[] (positive integer) {1234}
cmn.waiverLayersAndText.layersAndTexts.ruleChecksumTextType.value	Texttype number for Rule checksum objects. (RULE_CHECKSUM_TEXTTYPE) <b>Waivers</b> page, Layers and Text, “Rule checksum Layer”	[] (positive integer) {9}
cmn.waiverLayersAndText.layersAndTexts.usernameTextLayer.value	Layer number for User name objects. (USERNAME_TEXT_LAYER)  Not available for waiver generation. <b>Waivers</b> page, Layers and Text, “User Name Layer”	[] (positive integer) {1234}
cmn.waiverLayersAndText.layersAndTexts.usernameTextType.value	Texttype number for User name objects. (USERNAME_TEXTTYPE)  Not available for waiver generation. <b>Waivers</b> page, Layers and Text, “User Name Layer”	[] (positive integer) {3}

**Table C-7. Waiver Setup Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.waiverLayersAndText.layersAndTexts.waiverShapeLayerNumber.value	Layer number for Waiver Shape objects. (LAYER_NUMBER) <b>Waivers</b> page, Layers and Text, “Waiver Shape Layer”	[] (positive integer) {1234}
cmn.waiverRdbFiles.densityRDBOption.value	Specifies whether the <i>density_waived.rdb</i> is generated during a waiver run. (DENSITY_WAIVER_RDB) <b>Waivers</b> page, Waiver RDB Files, “Density waiver RDB Control Option”	[YES NO] () {YES}
cmn.waiverRdbFiles.enableCellReporting.value	Specifies whether to write a property named WAIVER_CELL to the waived results database files. (ENABLE_WAIVER_CELL_REPO RTING) <b>Waivers</b> page, Waiver RDB Files, “Enable waiver cell reporting” checkbox	[false true] (Boolean) {false}
cmn.waiverRdbFiles.ignoreStats.value	Specifies whether to exclude statistics from the summary file for waived results and used and unused waivers RDBs. This option only affects the databases that are <i>not</i> generated, and is only available in such cases. (NONE specified for the waived, used, or unused waiver RDBs.) (OPTIMIZE_NONE_WAIVER_STA TISTICS) <b>Waivers</b> page, Waiver RDB Files, “Ignore waiver statistics in summary file”	[false true] (Boolean) {false}
cmn.waiverRdbFiles.unusedRDBOption.value	Specifies options for the <i>unused_waiver.rdb</i> results database. (UNUSED_WAIVER_RDB) <b>Waivers</b> page, Waiver RDB Files, “Unused waiver RDB Control Option”	[ALL PROP USER GEOM NONE] () {ALL}

**Table C-7. Waiver Setup Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.waiverRdbFiles.usedRDBOption.value	Specifies options for the <i>used_waiver.rdb</i> results database. (USED_WAIVER_RDB)  <b>Waivers</b> page, Waiver RDB Files, “Used waiver RDB Control Option”	[ALL PROP USER GEOM NONE] ( {ALL}
cmn.waiverRdbFiles.waiverRDBOption.value	Specifies options for the <i>waived.rdb</i> results database. (WAIVER_RDB)  <b>Waivers</b> page, Waiver RDB Files, “Waived RDB Control Option”	[ALL PROP USER GEOM NONE] ( {ALL}
cmn.waiverRun.enableDynamicResultsReporting.value	Specifies whether to enable dynamic results reporting in Calibre RVE. (ENABLE_DYNAMIC_RESULTS_REPORTING)  <b>Waivers</b> page, DRC run, ‘Enable dynamic results reporting’	[false true] (Boolean) {false}
cmn.waiverRun.ignoreWaivingDensity.value	Specifies whether to ignore waiving rule checks with density operations. (IGNORE_DENSITY)  <b>Waivers</b> page, DRC Run, “Ignore waiving density operations”	[false true] (Boolean) {false}
cmn.waiverRun.retainResultType.value	Specifies whether to retain unwaived edge and error results, rather than converting them to polygons. Ignored in waiver creation.  RETAIN_RESULT_TYPE YES NO  <b>Waivers</b> page, DRC Run, “Retain edge and error result types”	[false true] (Boolean) {false}
cmn.waiverRun.runCalibreVersionChecksum.value	Specifies whether to verify Calibre version in waiver cells. Ignored in waiver creation.  (RUN_CALIBRE_VERSION)  <b>Waivers</b> page, DRC Run, “Verify Calibre version”	[false true] (Boolean) {false}

**Table C-7. Waiver Setup Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.waiverRun.runIPMatchChecks um.value	Specifies whether to verify IP_MATCH checksum text object in the waiver cells. Ignored in waiver creation. (RUN_IP_MATCH)  <b>Waivers</b> page, DRC Run, “Verify IP_MATCH checksums”	[true false] (Boolean) {true}
cmn.waiverRun.runLayoutChecksums m.value	Specifies whether to verify checksum text objects in waiver cells. Ignored in waiver creation. (RUN_LAYOUT_CHECKSUM)  <b>Waivers</b> page, DRC Run, “Verify waiver cell checksums”	[true false] (Boolean) {true}
cmn.waiverRun.runRuleChecksum. value	Specifies whether to verify Calibre rule checksum text objects in waiver cells. Ignored in waiver creation. (RUN_RULE_CHECKSUM)  <b>Waivers</b> page, DRC Run, “Verify rule checksums”	[false true] (Boolean) {false}
cmn.waiverRun.useCriteriaFromCriteriaFile.value	Specifies whether to use criteria from the waiver criteria file(s). (WAIVER_CRITERIA <file>)  <b>Waivers</b> page, DRC Run, “Use criteria from the waiver criteria files”	[true false] (Boolean) {true}
cmn.waiverRun.useCriteriaFromShapesFile.value	Specifies whether to use criteria from the waiver criteria shapes file(s). (WAIVER_CRITERIA EXTRACT <file>)  If both useCriteriaFromCriteriaFile and useCriteriaFromShapesFile are set to true, APPEND is used instead of EXTRACT.  <b>Waivers</b> page, DRC Run, “Use criteria from the waiver shape files”	[false true] (Boolean) {false}

**Table C-7. Waiver Setup Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.waiverShapes.mergeWaivers.value	<p>Specifies the source of waivers for a waiver run. The option is the keyword used for the MERGE statement in the waiver setup file.</p> <ul style="list-style-type: none"> <li>• NO — Use waivers both from the waiver database and from the input layout.</li> <li>• YES — Use only waivers embedded in the input layout.</li> <li>• IGNORE_DESIGN_WAIVERS — Use waivers only from the waiver database.</li> </ul> <p><b>Waivers page, Waiver Shapes, “Waiver Database”</b></p>	[NO YES IGNORE_DESIGN_WAIVERS] ( {NO}
cmn.waiverShapes.precisionConversion.value	<p>Specifies the YES (true) or NO (false) keyword for the PRECISION_CONVERSION statement in the waiver setup file.</p> <p><b>Waivers page, Waiver Shapes, “Use rule file precision”</b></p>	[true false] (Boolean) {true}
cmn.waiverShapes.waiverShapeFile.value	<p>The name of the waiver shape database created in a waiver generation run. (WAIVER_DATABASE)</p> <p><b>Waivers page, Waiver Shapes, “Waiver Shapes File”</b></p>	[] ( {waived.gds}
cmn.waiverShapes.waiverShapeFiles.specified	<p>Boolean controlling cmn.waiverShapes.waiverShapeFiles</p> <p>The checkbox controlling whether waiver shape files are used in a Calibre Auto-Waivers run.</p> <p><b>Waivers page, Waiver Shapes, “Waiver Shapes Files” checkbox</b></p>	[false true] (Boolean) {false}

**Table C-7. Waiver Setup Runset Options (cont.)**

Name	Description	[Available Values] (Type restrictions) {default}
cmn.waiverShapes.waiverShapeFile.s.value	<p>One or more waiver shape files that are used in a Calibre Auto-Waivers run.</p> <p>The waiver shapes files are not used if cmn.waiverShapes.mergeWaivers.value is set to YES (for MERGE YES).</p> <p>Waivers page, Waiver Shapes, “Waiver Shapes Files”</p>	[ () {}]

# Glossary

---

## Control File

A system generated text file that controls the Calibre applications that you invoke from Calibre Interactive. Refer to “[About Control Files](#)” on page 44 for more information on this file.

## Intermediate Files

Files created by Calibre Interactive at runtime to make it possible to run Calibre using the settings in the Calibre Interactive with an existing rule file. If a layout viewer is open, Calibre Interactive also creates an intermediate file to make it possible to process the data in the layout viewer.

## Name Generation Template

Template for system generated names for the intermediate files that Calibre Interactive creates.

## Preference Files

System generated text files created by Calibre Interactive to save your preferences as defined in Setup Preferences dialog box. Calibre Interactive creates a different preference file for each type of run.

## Rule File

The user supplied primary SVRF rule file for the Calibre run invoked from Calibre Interactive. Certain settings in this file may be overridden or ignored by Calibre Interactive.

## Run Directory

The directory location from which you run the Calibre application and from which all relative pathnames are resolved. By default, Calibre Interactive sets the run directory to your current directory. Changing the run directory may require that you update the paths to input and output files.

## Runset

A *user created* text file created from within Calibre Interactive to store the settings you specify in Calibre Interactive. For more information on runsets, refer to “[About Calibre Interactive Runsets](#)” on page 34 for more information on this file.

## Template

see “Name Generation Template”

## Working Directory

The directory from which you invoke Calibre Interactive.



## **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.

