

Virtuoso Custom Design Migrate User Guide

**Product Version IC23.1
November 2023**

© 2023 Cadence Design Systems, Inc. All rights reserved.

Printed in the United States of America.

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

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

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

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

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

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

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

Contents

1

<u>Introduction to Custom Design Migration</u>	3
--	---

2

<u>Virtuoso Custom Schematic Design Migration</u>	5
---	---

<u>Schematic Migration Overview</u>	5
-------------------------------------	---

<u>Virtuoso Custom Schematic Design Migration Prerequisites</u>	5
---	---

<u>Setting up Schematic Migration</u>	6
---------------------------------------	---

<u>Tool Setup</u>	6
-------------------	---

<u>Dependencies</u>	6
---------------------	---

<u>Mapping File Syntax</u>	7
----------------------------	---

<u>Preparing Source Schematic Data</u>	7
--	---

<u>Reviewing the Source Data</u>	7
----------------------------------	---

<u>Using Source Design and Technology</u>	7
---	---

<u>Creating a Technology Map</u>	8
----------------------------------	---

<u>Creating two cds.lib files</u>	11
-----------------------------------	----

<u>Running SKILL pre-trigger Function Per Instance or Cell</u>	11
--	----

<u>Migration Part 1: Save Source Schematic Data</u>	11
---	----

<u>Handling Instances and Parameters</u>	16
--	----

<u>Migration Part 2: Creating Target Schematic</u>	17
--	----

<u>Running SKILL Post-trigger Function Per Instance or Cell</u>	19
---	----

<u>Migration Part 3: Viewing Results Browser</u>	20
--	----

<u>Performing Multiple Cell Schematic Migration</u>	22
---	----

A

<u>Schematic Mapping Options</u>	39
----------------------------------	----

<u>Schematic Mapping Configuration File Options</u>	39
---	----

<u>Other Environment Options</u>	46
----------------------------------	----

B

<u>Schematic Utilities</u>	49
<u>Inspect Schematic Instance Properties</u>	49
<u>Schematic Design Inventory</u>	50
<u>Print Nondefault Pcell CDFs</u>	53
<u>Hierarchical Check and Save</u>	54
<u>Library Check and Save</u>	54
<u>Source-side Migration Map Generator</u>	54

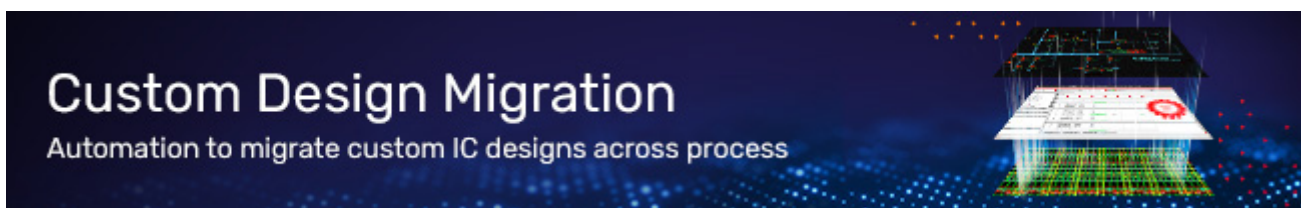
C

<u>Mapping File Syntax for Schematic Migration</u>	59
<u>Header Row</u>	59
<u>Creating a Mapping Spreadsheet - Examples</u>	60
<u>Conditional Equation Syntax</u>	63
<u>Other Parameters</u>	64
<u>Other Parameter Examples</u>	67

D

<u>Virtuoso Custom Design Migration Forms</u>	69
<u>Select Libraries that have Cells to Map Form</u>	70
<u>Select a Cell to Inspect Form</u>	71
<u>Schematic Design Inventory Form</u>	72
<u>Schematic Mapping Settings Form</u>	73
<u>Virtuoso Schematic Mapping Form (Part 1)</u>	74
<u>Select the Migration to Finish Form (Part 2)</u>	76
<u>Schematic Migration Results Browser Form</u>	77

Introduction to Custom Design Migration



Technology advancements have been continuously striving to offer benefits, such as higher circuit performance and lower power consumption. As chip sizes are reducing, there is an increased need to migrate existing designs across foundries and technologies. For example, you might want to migrate a design from 5nm to 3nm within the same foundry or a 3nm design from one foundry to another.

Migration of analog circuits is often a cumbersome and designer-intensive process. When migrating a block that has already been designed, verified, and tested to a new process, it is often desirable to maintain the same architecture and, in many cases, the same or similar circuit performance of the original source design.

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

Related Topics

[Virtuoso Custom Schematic Design Migration](#)

Virtuoso Custom Design Migrate User Guide

Introduction to Custom Design Migration

Virtuoso Custom Schematic Design Migration

Schematic Migration Overview

Custom schematic design migration uses a CSV file to migrate your schematic design from one technology to another. It consists of the following steps:

1. The first part gathers information from each source schematic, guided by the mapping file. It produces a skeleton target schematic that is ready to receive instances from the target technology, and saves mapping details to intermediate files. For more information, refer to [Migration Part 1: Save Source Schematic Data](#).
2. The second part completes each skeleton target schematic with instances from the target technology, transforming and rewiring these instances according to the data gathered in the first part. For more information, refer to [Migration Part 2: Creating Target Schematic](#).
3. Finally, you can view the migration results in the Results Browser. For more information, refer to [Migration Part 3: Viewing Results Browser](#).

Note: The mapping process is split into separate steps so that you can potentially run each part in separate Virtuoso sessions. This might be needed if it is not possible to load both the source and target technologies in the same session.

Virtuoso Custom Schematic Design Migration Prerequisites

The prerequisites for running Virtuoso Custom Schematic Design Migration are:

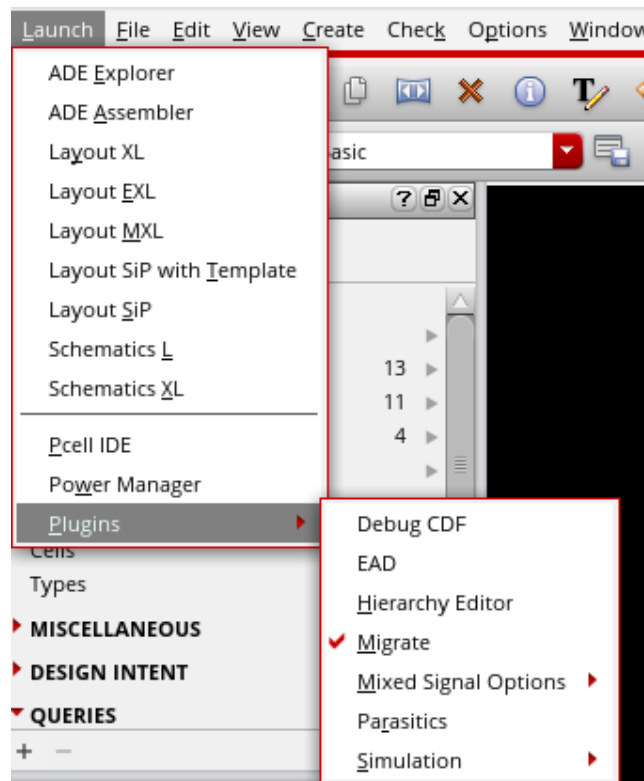
- **Virtuoso Studio version:** IC23.1
- **License:** Virtuoso Schematic Editor XL (95115) and Virtuoso_Custom_Design_Migration (95130)

Setting up Schematic Migration

Schematic Migration is a Virtuoso Schematic Editor XL feature.

Tool Setup

You can enable the tool in Virtuoso Schematic Editor XL under *Launch – Plugins – Migrate*.



Dependencies

To use schematic migration, you need the following:

- A source and target PDK (can be the same PDK for re-mapping)
- Schematic data
- A migration map file (comma separated values file)

Mapping File Syntax

For more details, refer to [Mapping File Syntax for Schematic Migration](#).

Preparing Source Schematic Data

Reviewing the Source Data

The source schematic data should be checked and saved without errors. If errors or missing data exists, fix the source schematic data before running Schematic Migration.

The source schematic library and hierarchy should not have any missing cells or libraries. However, simple missing instance references that have a mapping defined in the map file (not Parameterized Cells) will rebind correctly. However, connectivity checking and advanced symbol change features will not function if the source instance is missing. You can find the utility to perform a Hierarchical or Library Check and Save on the pull-down menu:

Migrate – Schematic – Utilities – Hierarchical Check and Save

or

Migrate – Schematic – Utilities – Library Check and Save

Using Source Design and Technology

Become familiar with the source design and source technology; also become familiar with the target technology. You can find the utility to query and display the contents of the source schematic data:

Migrate – Schematic – Utilities – Schematic Design Inventory

The *Schematic Design Inventory* utility provides multiple, different type of information summaries of the source schematic data.

In addition to the Schematic Design Inventory feature, a partial migration map file can be automatically created with

Migrate – Schematic – Utilities – Create Source-side Migration Map

However, this automatically generated migration map file is less efficient than a well-planned migration map.

Creating a Technology Map

This section explains how to create a technology map. As an example, Cadence provides a Generic Process Development Kits (GPDs) on Cadence Online Support (COS), at <http://support.cadence.com>. Figure 1 shows a portion of a schematic from the 45nm GPDK (library gpdk045) on the left and the same portion of a schematic from the 10nm GPDK (library cds_ff_mpt) on the right. Both use parameterized cells (Pcells) from their respective PDKs.

Note: You can create the mapping file in any spreadsheet editor, however, you must save it in CSV format for use in the Schematic Mapping tool.

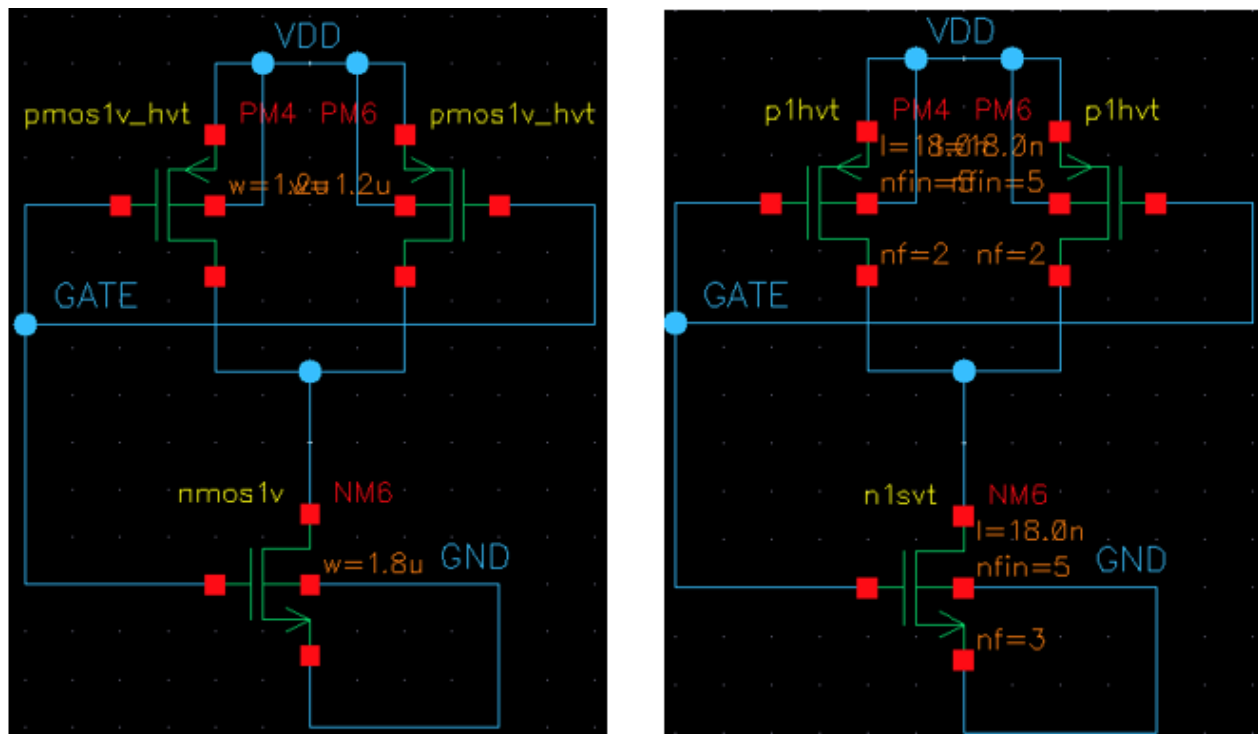


Figure 2-1 Example Source and Target Schematic

First, notice the n-type device name (in yellow, near the bottom-center), changes from nmos1v to n1svt. The spreadsheet syntax to accomplish this device map is shown below.

Keyword	Source Library or Source parameter name	Source Cell or Source Parameter Description	Source Parameter Value	Display	Target Library or Target Parameter Name	Target Cell or Target Parameter Description	Target Parameter Value	Notes	Other Parameters
device	gpdk045	nmos1v			cds_ff_mpt	n1svt			
match parameter									

Figure 2-2 Syntax to Accomplish Device Map

Note:

- The mapping spreadsheet or map file must have a keyword as the first field of the first line. The first line must have 10 fields. and the first line must have 9 commas. Special characters in the first line should be avoided.
- The source parameter $w = 1.8\mu m$ changes to target parameters $nfin = 5$, and $nf = 3$ (ignoring the $l = 18.0n$ as it does not relate to the source width). These parameters are *Component Description Format* (CDF) parameters of the PDK Pcells. The figure below shows the spreadsheet syntax to accomplish this CDF parameter mapping.

Keyword	Source Library or Source parameter name	Source Cell or Source Parameter Description	Source Parameter Value	Display	Target Library or Target Parameter Name	Target Cell or Target Parameter Description	Target Parameter Value	Notes	Other Parameters
device	gpd045	nmos1v			cds_ff_mpt	n1svt			
match parameter	w	Width	1.8u		nfin	Number of Fins	5		
match parameter					nf	Number of Fingers	3		

Figure 2-3 CDF Parameter Mapping

The above figure shows a hard-coded mapping of the Pcell CDF parameters. Only `nmos1v` Pcells with a width of `1.8u` will map to `n1svt` Pcells. Unmatched instances of `nmos1v` or other 45nm PDK Pcells will not be migrated (they will be removed from the target schematic, unless the configuration file variable:

```
unmappedPassThru = t
```

See [Schematic Mapping Configuration File Options](#).

If the `w`, `Width`, `1.8u` match parameter fields are removed, then all `gpd045 nmos1v` Pcells would map to `cds_ff_mpt n1svt` Pcells with `nfin = 5` and `nf = 3`.

Schematic Migration enables further flexibility. For example, the target parameters `nfin` and `nf` could be set based on an equation - an equation that may use source parameters and SKILL functions. Given the following relationship of `nfin` and `nf` to `w`, the figure below shows how to create the mapping spreadsheet.

```
nfin = round(1e6*w/0.36)
```

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

`nf = round(1e6*w/0.6)`

Keyword	Source Library or Source parameter name	Source Cell or Source Parameter Description	Source Parameter Value	Display	Target Library or Target Parameter Name	Target Cell or Target Parameter Description	Target Parameter Value	Notes	Other Parameters
device	gpd045	nmos1v			cds_ff_mpt	n1svt			
match parameter	w	Width	1.8u		nfin	Number of Fins	<code>round(1e6*cc_w/0.36)</code>		
match parameter					nf	Number of Fingers	<code>round(1e6*cc_w/0.6)</code>		

Figure 2-4 Mapping Spreadsheet

The `cc_` prefix, which appears before the `w` parameter, indicates that the parameter is a copy of the source parameter.

Finally, the process of building a mapping file involves creating a map for all the instances. The below figure completes the mapping for the example schematic. More information on the conditional, `calc`, syntax below is in the [Mapping File Syntax for Schematic Migration](#).

Keyword	Source Library or Source parameter name	Source Cell or Source Parameter Description	Source Parameter Value	Display	Target Library or Target Parameter Name	Target Cell or Target Parameter Description	Target Parameter Value	Notes	Other Parameters
device	gpd045	nmos1v			cds_ff_mpt	n1svt			
match parameter	w	Width	1.8u		nfin	Number of Fins	<code>round(1e6*cc_w/0.36)</code>		
match parameter					nf	Number of Fingers	<code>round(1e6*cc_w/0.6)</code>		
device	gpd045	pmos1v_hvt			cds_ff_mpt	p1hvt			
match parameter					nfin	Number of Fins Per Finger	5		
match parameter					nf	Number of Fingers	<code>calc [eq]3 [cc_w]> 1.6u]</code>		
match parameter					nf	Number of Fingers	<code>calc [eq]2 [cc_w]<= 1.6u]</code>		

Figure 2-5 Schematic Example

Note: An optional tenth column may be added after the *Notes* column for instance specific, *Other Parameters*. The available *Other Parameters* are:

- `symbolOffsetX`, `symbolOffsetY`, `keepPinLoc`, `keepCenter`: Positions the target symbol.
- `pinRename`: Describes the association between source and target instance pins that have the same function but different names.
- `connectPins`: Connects the target pins to other target pins
- `symbolName`: Renames the target symbol

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

- `symbolSize`: Resizes a target symbol when a `symbol_xform` view type is used.
- `useStubs`, `stubFromPinNet`: Connects pins by wire stub.
- `moveAndStub`: Places a symbol outside the current schematic boundary and using wire stubs to connect pins.
- `useInstPreCheck`: Is an instance specific option, related to the global option, for checking whether an instance will create a short before it is placed.
- `mfactor2vector`, `vector2mfactor`: map iterated instances: (That is <0:3>) to `mfactor` (`m=4`) or vice versa.
- `postMapTrigger`: Applies an instance trigger function after mapping.
- `defSymOrient`: Rotates a target symbol to match its source symbol orientation.

For more details about other Parameters, see [Mapping File Syntax for Schematic Migration](#).

Creating two `cds.lib` files

Since two PDKs are involved, two `cds.lib` files may be created, one for each, unless the two PDKs may co-exist without overlapping parameters or callback function names. Consider naming these as `cds.lib.<source technology>` and `cds.lib.<target technology>`. In the example above, the two files would be `cds.lib.gpdk045` and `cds.lib.cds_ff_mpt`. Each `cds.lib` file should only contain technology libraries from one PDK. These files will be used in the Schematic Migration flow.

Running SKILL pre-trigger Function Per Instance or Cell

A user-defined pre-trigger function can be run before the migration on the source data. [Schematic Mapping Options](#), Configuration File, Configuration Options describes the setting `preMigInstTrigger` that applies a pre-trigger function on source instances before migration. Similarly, the setting, `preMigCellTrigger`, will run the pre-trigger function on the source cells. The function will run per instance or per cell before the migration, such that each function receives the instance, or cellview object as an argument.

Migration Part 1: Save Source Schematic Data

Select *Migrate – Schematic – Mapping – Save Source Schematic Data (Part 1)*.

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

In the form that appears after selecting *Save Source Schematic Data (Part 1)*, select the target library or select *Create New...* in the target library and follow the flow through the creation of a new target library.

Note: When creating a new target library, select *Attach to an existing technology library* when creating a new library, and attach the library to the source PDK library.

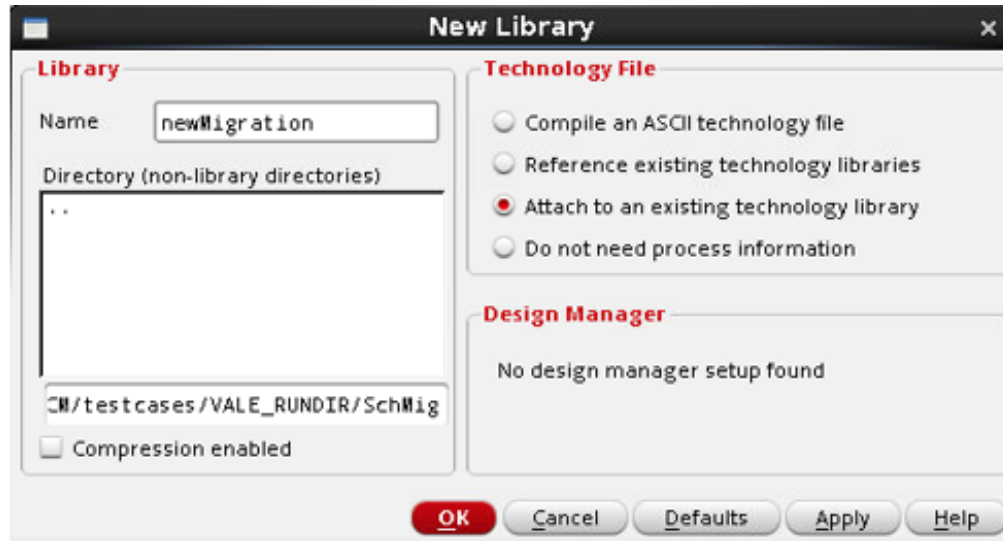


Figure 2-6 The New Library Form

Figure 2-7 Save Source Schematic Data (Part 1) Form

The figure above shows more options than are necessary to complete Part 1 for the single cell migration. No *Process Hierarchy*, *Process all referenced libraries*, *Migration Mapping File* or *Design Library Mapping* entries are needed for a single cell migration. However, the more options are there to enable flexibility for more complex migrations.

First, if the source schematic is a hierarchical schematic, then the *Process Hierarchy* check box under the *Source* group box will traverse the schematic hierarchy and migrate all the cells in the source library hierarchy, filtered by the options *View Name*, *Process all views of each cell*, *Exclude View List*, and *Exclude Library List*. Libraries other than the source library hierarchy (and not listed in the mapping file) will not be migrated by default. If there are outside design library references that should be migrated, select the *Process all referenced libraries*, and either enter the source and target library map as a space separated pair, and separate each pair by a comma; or use the corresponding *Browse* button to specify a comma separated values file. For example, the *Design Library Mapping* comma separate values file is shown below:

```
# Source library, Target Library
```

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

```
Framework_Examples_45_RefLib1, MigLib1
Framework_Examples_45_RefLib2, MigLib1
#Framework_Examples_45_RefLib3, MigLib1
Framework_Examples_45_RefLib3, MigLib2
```

Note: The source design libraries may be mapped to different target design libraries. If the Target Library specified does not exist, Schematic Migration will create it.

At the completion of Part 1, Schematic Migration will print the library mapping summary:

INFO: PART 1 INPUT SUMMARY

```
> The migration map libraries:
    Framework_Examples_45_RebindLib1
    gpdk045

> The source design libraries:
    Framework_Examples_45

> The exclude libraries:
    behavioral_lib
    basic
    analogLib
    US_8ths
    borders
```

The following cells have no mapping, No_Map:

```
gpdk045 nmoslv_lvt symbol, instance NM3 in example_SchMig_allDevs
gpdk045 nmoslv symbol, instance NM0 in example_SchMig_allDevs
gpdk045 nmoslv symbol, instance NM0 in QATest2c
```

The following cells will carry-over to the new schematic, As_Is, because the library name is not listed in the "Design Library Mapping" field in the form, or in the Migration Map File:

```
PHLib1 SchMig_Cell2 symbol, instance I1 in QATest2c
```

Instance references outside the source design library fall in one of three categories:

- **Map Library:** the map file contains a reference to this library
 - ☐ If there is a map for this cell
 - Instance is mapped
 - ☐ If there is NO map for this cell

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

- If `unmappedPassThru` is unset, or set to `nil`, the instance is not mapped, it is deleted in the target schematic
 - If `unmappedPassThru` is set to `t`
 - Instance is passed-through unmodified to the target schematic
- **Exclude Library:** the library is listed in the input environment variable `excludedLibs` in the `schMapDefaults.il` (configuration file)
 - Instance is passed-through unmodified to the target schematic
- **Unknown Library:** Not a map library, design (migrate) library, or exclude library
 - Instance is passed-through unmodified to the target schematic

Second, the *Process Library* check box will run all the schematics of a library as single cells, filtered by the options *View Name*, *Process all views of each cell*, *Exclude View List*, and *Exclude Library List*. The hierarchy of each schematic will not be traversed unless *Process all referenced libraries* is selected.

Third, the *View Name* source field limits the cell view types to traverse. If all view types should be traversed, then select the *Process all views of each cell* check box. Specific view types may be excluded using the input environment variable, `excludedViews`, in the `schMapDefaults.il` (configuration file).

Fourth, Schematic Migration includes a number of options for renaming migrated cell views. In the target field, the *Cell Name* may be set to *Use name options below*. This enables the target field *Name Options* with its qualifiers *Prefix*, *Replace*, and *Suffix*. These options apply for all the cells in *Process Hierarchy* and *Process entire Library*, and for all migrated view types.

For example, if the source *Cell Name* is `Aug20_doc1`. The following examples will use this setting to demonstrate how to rename migrated cell views.

- **Prefix**
 - Set target field *Cell Name* to *Use name options below*
 - Set *Name Options* field to `new_` (no quotes)
 - Select the *Prefix* qualifier
 - When migration completes, the target cell name will be `new_Aug20_doc1`
- **Replace**
 - Set target field *Cell Name* to *Use name options below*

- ☐ Set Name Options field to `Aug20 Dec23` (space separated, no quotes)
- ☐ Select the Replace qualifier
- ☐ When migration completes, the target cell name will be `Dec23_doc1`

■ **Suffix**

- ☐ Set target field Cell Name to Use name options below
- ☐ Set Name Options field to `_mig` (no quotes)
- ☐ Select the Suffix qualifier
- ☐ When migration completes, the target cell name will be `Aug20_doc1_mig`

Finally, when all the options are set, click *OK*. Part 1 will run, and display Schematic Migration: Part 1 when it completes successfully. Check the CIW for error messages to see if there is anything to correct before continuing to Part 2.

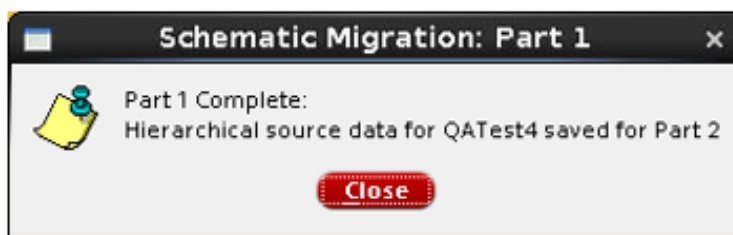


Figure 2-8 Schematic Migration: Part 1

If the above window does not appear, or Part 1 does not display the completion message in the CIW, check the CIW for error messages.

Otherwise, close Virtuoso, if performing a migration with two different PDKs that overlap parameters or callback function names, and continue to Part 2.

Handling Instances and Parameters

This section describes how schematic migration handles instances and parameters:

1. `unmappedPassThru = t`: Typically a same PDK Flow (intra-PDK migration)
 - ☐ Change Pcell: The map file should map one Pcell to another, `nmos1v` to `nmos1v_hvt` for example, using all the parameter mapping capability to meet the migration needs (that is `width = cc_width`). All Pcells used in the design hierarchy or library should have a map in the mapping file.

- ❑ Change Pcell parameter only: The map file should specify a mapping of one Pcell to itself, using all the parameter mapping capability to meet the migration needs (i.e. `width = cc_width*2`)
- ❑ No Pcell change needed: Leave the Pcell out of the map file, it will appear in the target cell unmodified.

2. `unmappedPassThru = nil`: Typically a different PDK Flow (inter-PDK migration)

- ❑ Change Pcell: The map file should map one Pcell to another, `nmos1v` to `n1hvt` for example, using all the parameter mapping capability to meet the migration needs (that is, `width = cc_width/1.5`). All Pcells used in the design hierarchy or library should have a map in the mapping file.
- ❑ Change Pcell parameter only: Since this is a different PDK migration, this situation is not supported.
- ❑ No Pcell change needed: This is uncommon for a different PDK migration since the source PDK will be removed, all Pcells must be mapped or mapped to the keyword cell name, `placeholder`. If any Pcells are missing a mapping in the map file, then they will be missing from the target schematic (deleted and not re-created).

3. For Both Flows:

- ❑ Non-Pcell Instances inside the source design library: Will re-bind to the target library (if *Process Hierarchy* or *Process entire Library* is selected)
- ❑ Non-Pcell Instances outside the source design library:
 - If a member of `excludedLibs`, pass through unmodified
 - If listed as a map in the mapping file, will rebind (Note that this cell, hierarchy, or library must be migrated separately, first, so that it is available for a rebind)
 - If none of the above, pass through unmodified

Migration Part 2: Creating Target Schematic

Before you create target data, ensure to complete the steps shared in the section, [Migration Part 1: Save Source Schematic Data](#).

If performing a two-session migration (generally an inter-PDK migration), and once *Migration Part 1: Saving Source Data* is complete, copy the `cds.lib.<target technology>` file to `cds.lib` and restart Virtuoso. Otherwise, for a one-session migration (generally an intra-PDK migration), you can skip this step.

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

The difference between the one-session migration and the two-session migration is that the two-session migration requires a Virtuoso restart while pointing to the target PDK, and not pointing to the source PDK. To ignore whether the source PDK is removed from the current session or not, the following configuration file option is set, as shown below:

```
bypassLibCheck = t
```

Note: A two-sessions migration is necessary when CDF Parameter callbacks have namespace overlaps, or when the source and target design environments are on two different file systems.

Open the new schematic, created by Part 1, with the target library selected or created in Part 1, and the target name used in Part 1.

Note: See the below figure, the schematic will contain only wires and unmodified instances. This is because the instances are re-created instead of remastered, unless an instance passes through unmodified.

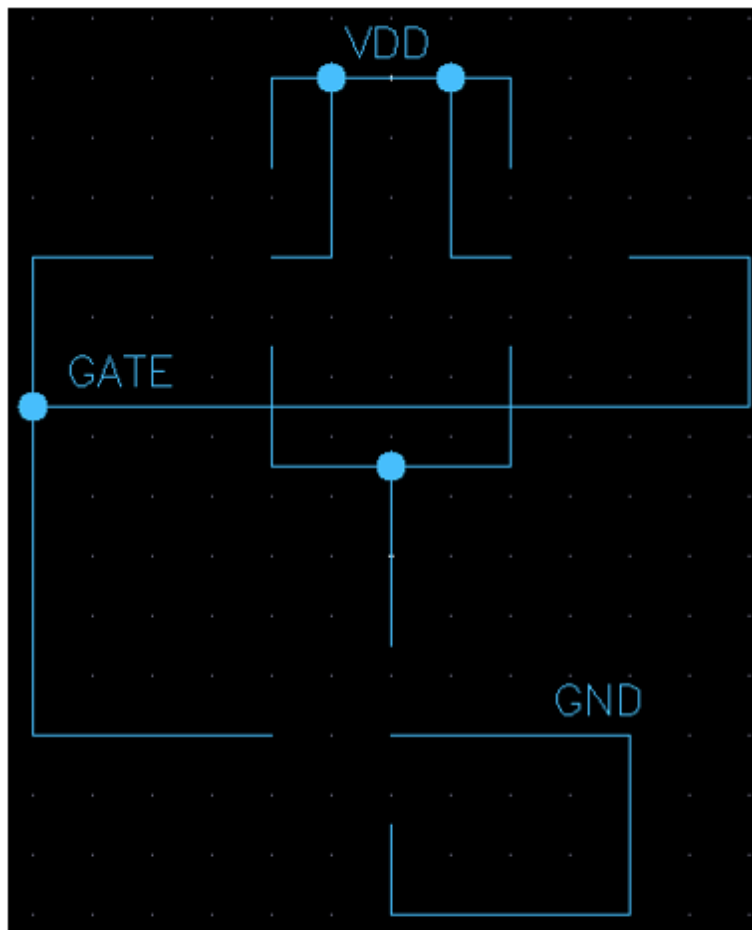


Figure 2-9 Schematic with Wires and Unmodified Instances

Running SKILL Post-trigger Function Per Instance or Cell

There are many options to control how instances are recreated.

A user-defined post-trigger function can be run after creating the target data. [Schematic Mapping Options](#), Configuration File, Configuration Options describes the settings `postMigInstTrigger` and `postMigCellTrigger`. These environment variables apply post-trigger functions to the instance or cell after recreating instances, the function receives the instance or cellview object as an argument.

Because the tool setup has already been completed, the *Migrate* menu appears on the *Virtuoso Schematic Editor* GUI.

Select *Migrate – Schematic – Mapping – Create Target Schematic (Part 2)*.

Schematic Migration Part 2 uses the results from Part 1. The *Select the migration to finish* form (figure below) is displayed and points Part 2 to the correct source data set. In this way, it is possible to migrate multiple cellviews through Part 1, close Virtuoso, then finish them one after another in Part 2.

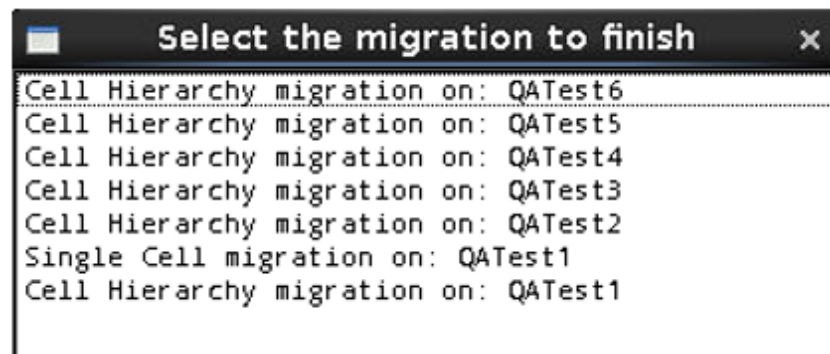


Figure 2-10 The Source Data Selection Form

When Schematic Migration Part 2 finishes successfully, it will open a dialog box with two buttons (see [Successful Completion of Schematic Migration Part 2](#)).

- The *OK* button will simply close the dialog box.
- The *Open Results Browser* button will open the Migration Results Browser

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

Any one of these buttons completes Schematic Migration, Part 2.

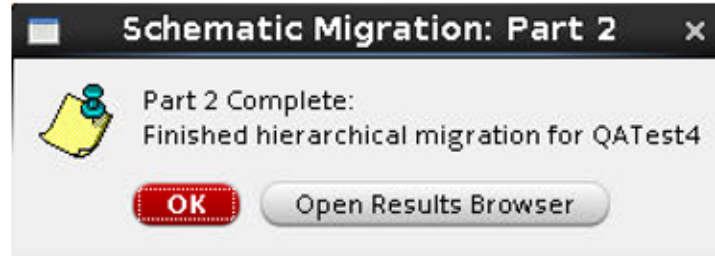


Figure 2-11 Successful Completion of Schematic Migration Part 2

Migration Part 3: Viewing Results Browser

After creating the target data, the *Migration Results Browser* is displayed at the end of Part 2, with the button in [Successful Completion of Schematic Migration Part 2](#), or by using the *Migrate* menu, Migration Results Browser (see [Opening the Migration Results Browser](#)). The configuration file describes a setting `migBrowserStatusFilters` that applies a default filter. When the Migration Results Browser is opened from the Part 2 message dialog, *Open Results Browser*, the default filter does not apply. The filter may be dynamically modified to change what is displayed, see [Schematic Mapping Options](#).

Note: A run directory must be present and a results file must be present in it for the *Migration Results Browser* to work.

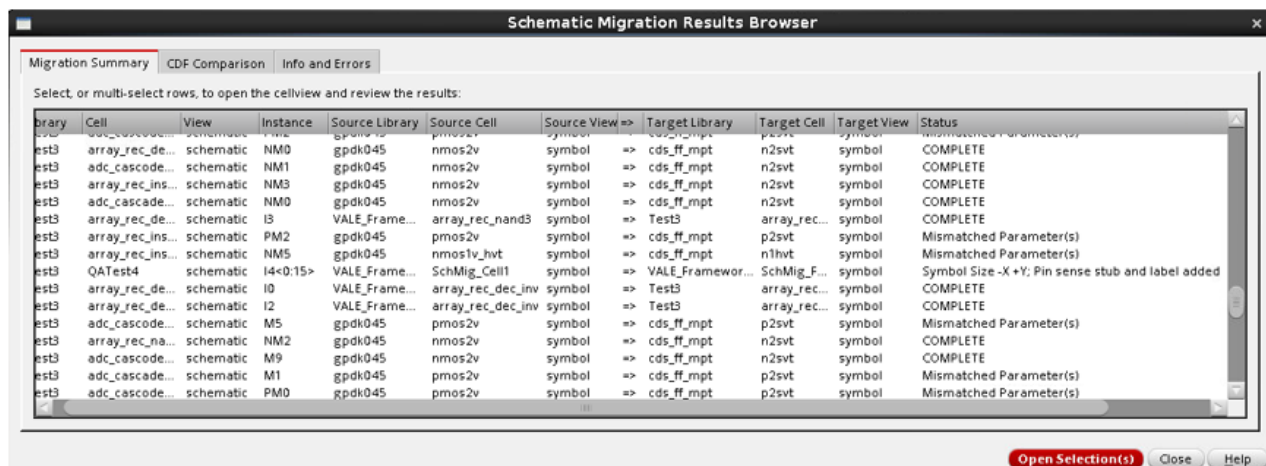


Figure 2-12 Opening the Migration Results Browser

The results browser can be opened from the cells in the hierarchy, based on the results file for the top cell that must be present in the run directory. When migration results browser is opened from a lower level cell, the tool looks for any results files present in the run directory and if one or more files contain the results for the current cell, Virtuoso provides an option to select a results file, as shown below:



Figure 2-13 Cell Example

This gives a more flexible entry point from any cell in the hierarchy and display its hierarchy in the Results Browser. When there are no results in run directory that contain the results for current cell, then a dialog box will display as follows:

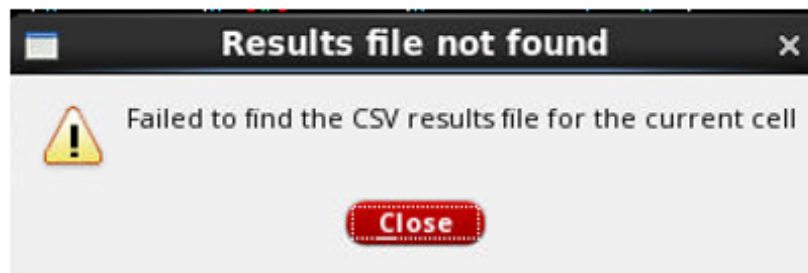


Figure 2-14 The Results File Not Found Form

In the Migration Results Browser, each column is sizable and sortable. The *Status* column in the Migration Summary is the most important column. It is a guide for the design to review the results. See [Schematic Mapping Options](#) on filtering the Status column as desired.

When one or more row is selected, and the *Open Selection(s)* button is pressed, the browser will open the referenced cellviews and create markers for the indicated results. By default, the maximum number of cells it will open is 5, but that may be changed with the configuration file option:

`maxCellsToOpen = <integer>`

Performing Multiple Cell Schematic Migration

Multiple cell schematic migration is a group of tools that allows users to migrate a list of selected design top cells from source libraries to target libraries by providing a text file as input. The multiple cell schematic migration commands are available in the Virtuoso CIW Menu.

To run multiple cell schematic migration:

1. Build a file that contains a list of top cells and the related options. See section, [Top Cell List and Preset Files](#)
2. Set the environment settings in `schMapDefaults.il`. See section, [Environment Settings in schMapDefaults.il](#)
3. For running multiple cells, the process is divided into a three step process to prevent Virtuoso from destroying data from child cells that maybe shared between top cells and does not impact memory. For details, see section, [Using Virtuoso Menu](#).

Top Cell List and Preset Files

A cell list file has eight columns: index, source library, source cell, source view, target library, target view, and preset. Each row defines one top cell migration. The text file is in a SKILL data list form.

- Column 1: index, an integer starting from 1.
- Column 2: source library name string.
- Column 3: a source top cell name string from the source library.
- Column 4: source top cell view string. always “schematic”.
- Column 5: target library name string to migrate to.
- Column 6: target top cell name string to migrate to.
- Column 7: target top cell view string. always “schematic”.
- Column 8: preset file name. Usage of preset file will be explained later. If no special request, it can be just empty string.

Both cell list file and preset files need to be placed in the multiple cell schematic migration data input folder. This data folder will be mentioned in the part of working environment settings:

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

An example of cell list file, `celllist.txt`, is as follows:

```
Index   srcLib           srcCell   view   tgtLib   tgtCell   view   preset
(
(1 "Framework_Examples_45" "QATest3" "schematic" "MIG3" "MYCell1" "schematic" "")
(2 "Framework_Examples_45" "QATest3_01" "schematic" "MIG3" "MYCell2" "schematic" "")
(3 "Framework_Examples_45" "QATest3_02" "schematic" "MIG4" "MYCell3" "schematic" "")
(4 "Framework_Examples_45" "QATest3_03" "schematic" "MIG4" "MYCell4" "schematic" "")
)
```

Note:

- Source and target view columns are always schematic.
- Preset column allows to specify a preset file. Preset file is in the same format of `form89.preset` used in interactive schematic migration for configuration of the migration Part1. In interactive mode, this file is generated by Part1 input settings form and saved in migration working directory. But, in multiple cells processing mode, since the settings are mainly for a list of top cells migration, most of the fields are with default values from `schMapDefaults.il` file. Only limited fields need to be modified if the values are not same as the defaults. An example preset file with design reference library migration is given below.

The default preset file is `preset.default`. If preset field is empty string "", the default preset file will be used. If no `preset.default` file exists in the cell list folder, one file will be generated internally, which is basically for the top cell migration with hierarchy option. If a preset file is provided, it is then used accordingly.

An example of a preset file with design reference library migration is as follows:

```
form89DoHier = t
form89DoLib = nil
form89DoAllViews = nil
form89DoAllRefLibs = t
form89TgtLibSel = "MIG0"
form89TgtCellSel = "Use source name"
form89TgtView = "schematic symbol"
form89MigMap = "/path/...../map_file.csv"
form89TgtNameOpt = ""
form89TgtNmOpts = "Prefix"
form89SrcView = "schematic symbol"
form89TermMap = nil
form89ExclLibs = "basic analogLib US_8ths borders"
form89ExclViews = "text adex1 constraint physConfig maestro layout abstract"
form89DesLibMaps = "Framework_Examples_45_RefLib2 MigLib1"
```

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

Note: A preset file is used in schematic migration interactive mode for storing the GUI settings. In a multi-top cell process, besides cell list files, which is used to specify the source cells and target cells migration mapping, it keeps assisting some other settings, which are not given in the `celllist.txt` file. Only limited fields are needed to be provided with values.

1. `form89DoHier = t`

Is fixed for top cell with hierarchy.

2. `form89TgtLibSel = ""`

Target library name. Set or no set, the final value will be from `celllist.txt` target library name.

3. `form89TgtCellSel = "Use source name"`

Target cell name. Set or not set, the final value will be from `celllist.txt` target cell name.

4. `form89TgtView = "schematic symbol"`

`form89SrcView = "schematic symbol"`

Fixed to "schematic symbol".

5. `form89MigMap = "/path/...../map_file_name.csv"`

Migration map file with path.

6. `form89DoAllRefLibs = t`

`form89DesLibMaps = "Framework_Examples_45_RefLib2 MigLib1"`

Enable and specify reference lib migration. `form89DoAllRefLibs` set to `t`, and `form89DesLibMaps` set to mapping string, or map file as in interactive GUI.

7. `form89ExclLibs = "basic analogLib US_8ths borders"`

`form89ExclViews = "text adexl constraint physConfig maestro layout abstract"`

Exclude libraries and views. Set same with `schMapDefaults.il`.

8. `form89DoLib = nil`

Not used. Set to `nil`.

9. `form89DoAllViews = nil`

Not used. Set to `nil`.

10. `form89TgtNameOpt = ""`

`form89TgtNmOpts = "Prefix"`

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

Not used.

11. `form89TermMap = nil`

Not used. Set to nil.

Another example of top cells list file, `celllist.txt` and the preset files:

celllist.txt

```
((1 "Framework_Examples_45" "QATest1" "schematic" "Mig1" "Test1" "schematic" "preset1.txt")
(2 "Framework_Examples_45" "QATest2" "schematic" "Mig2" "Test2" "schematic" "preset2.txt")
(3 "Framework_Examples_45" "QATest3" "schematic" "Mig3" "Test3" "schematic" "preset3.txt")
(4 "Framework_Examples_45" "QATest4" "schematic" "Mig4" "Test4" "schematic" "preset4.txt")
(5 "Framework_Examples_45" "QATest5" "schematic" "Mig5" "Test5" "schematic" "preset5.txt")
(6 "Framework_Examples_45" "QATest6" "schematic" "Mig6" "Test6" "schematic" "preset6.txt")
(7 "Framework_Examples_45" "QATest7" "schematic" "Mig7" "Test7" "schematic" "preset7.txt"))
```

preset1.txt

```
form89DoHier = nil
form89DoLib = nil
form89DoAllViews = nil
form89DoAllRefLibs = nil
form89TgtLibSel = "Mig1"
form89TgtCellSel = "Use source name"
form89TgtView = "schematic symbol"
form89MigMap = "../SchMig_QA_Test1.csv"
form89TgtNameOpt = ""
form89TgtNmOpts = "Prefix"
form89SrcView = "schematic symbol"
form89TermMap = nil
form89ExclLibs = "basic analogLib US_8ths borders"
form89ExclViews = "text adexl constraint physConfig maestro layout abstract"
form89DesLibMaps = ""
```

preset2.txt

```
form89DoHier = t
form89DoLib = nil
form89DoAllViews = nil
form89DoAllRefLibs = nil
form89TgtLibSel = "Mig2"
form89TgtCellSel = "Use source name"
form89TgtView = "schematic symbol"
form89MigMap = "../SchMig_QA_Test2.csv"
```

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

```
form89TgtNameOpt = ""
form89TgtNmOpts = "Prefix"
form89SrcView = "schematic symbol"
form89TermMap = nil
form89ExclLibs = "basic analogLib US_8ths borders"
form89ExclViews = "text adexl constraint physConfig maestro layout abstract"
form89DesLibMaps = ""
```

preset3.txt

```
form89DoHier = t
form89DoLib = nil
form89DoAllViews = nil
form89DoAllRefLibs = nil
form89TgtLibSel = "Mig3"
form89TgtCellSel = "Use source name"
form89TgtView = "schematic symbol"
form89MigMap = "../SchMig_QA_Test3.csv"
form89TgtNameOpt = ""
form89TgtNmOpts = "Prefix"
form89SrcView = "schematic symbol"
form89TermMap = nil
form89ExclLibs = "basic analogLib US_8ths borders"
form89ExclViews = "text adexl constraint physConfig maestro layout abstract"
form89DesLibMaps = ""
```

preset4.txt

```
form89DoHier = t
form89DoLib = nil
form89DoAllViews = nil
form89DoAllRefLibs = t
form89TgtLibSel = "Mig4"
form89TgtCellSel = "Use source name"
form89TgtView = "schematic symbol"
form89MigMap = "../SchMig_QA_Test4.csv"
form89TgtNameOpt = ""
form89TgtNmOpts = "Prefix"
form89SrcView = "schematic symbol"
form89TermMap = nil
form89ExclLibs = "basic analogLib US_8ths borders"
form89ExclViews = "text adexl constraint physConfig maestro layout abstract"
form89DesLibMaps = "../Include/SchMig_DesignLibraryMapping.csv"
```

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

preset5.txt

```
form89DoHier = t
form89DoLib = nil
form89DoAllViews = nil
form89DoAllRefLibs = nil
form89TgtLibSel = "Mig5"
form89TgtCellSel = "Use source name"
form89TgtView = "schematic symbol"
form89MigMap = "../SchMig_QA_Test5.csv"
form89TgtNameOpt = ""
form89TgtNmOpts = "Prefix"
form89SrcView = "schematic symbol"
form89TermMap = nil
form89ExclLibs = "basic analogLib US_8ths borders"
form89ExclViews = "text adexl constraint physConfig maestro layout abstract"
form89DesLibMaps = ""
```

preset6.txt

```
form89DoHier = t
form89DoLib = nil
form89DoAllViews = nil
form89DoAllRefLibs = nil
form89TgtLibSel = "Mig6"
form89TgtCellSel = "Use source name"
form89TgtView = "schematic symbol"
form89MigMap = "../SchMig_QA_Test6.csv"
form89TgtNameOpt = ""
form89TgtNmOpts = "Prefix"
form89SrcView = "schematic symbol"
form89TermMap = nil
form89ExclLibs = "basic analogLib US_8ths borders"
form89ExclViews = "text adexl constraint physConfig maestro layout abstract"
form89DesLibMaps = ""
```

preset7.txt

```
form89DoHier = t
form89DoLib = nil
form89DoAllViews = nil
form89DoAllRefLibs = nil
form89TgtLibSel = "Mig7"
form89TgtCellSel = "Use source name"
```

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

```
form89TgtView = "schematic symbol"
form89MigMap = "../SchMig_QA_Test7.csv"
form89TgtNameOpt = ""
form89TgtNmOpts = "Prefix"
form89SrcView = "schematic symbol"
form89TermMap = nil
form89ExclLibs = "basic analogLib US_8ths borders"
form89ExclViews = "text adexl constraint physConfig maestro layout abstract"
form89DesLibMaps = ""
```

In this example, since each top cell migration target library and migration map file is different, each top cell requires a preset file. Usually, the migration target library can be same, or can be grouped into a few, and migration map files can be shared by a group of top cells. Then, the `celllist.txt` can be rearranged, and the needed preset files can be reduced.

For example, if all the top cells are migrated into the same target lib Mig, and share a combined map file, `SchMig_Test.csv`, cell Test1 does not require hierarchy, cells Test2, Test3, Test4, Test5, Test6 and Test7 need hierarchy, and cell Test4 need reference design library migration, the `celllist.txt` and preset files are presented below:

celllist.txt

```
((1 "Examples" "Test1" "schematic" "Mig" "Test1" "schematic" "preset1.txt")
(2 "Examples" "Test2" "schematic" "Mig" "Test2" "schematic" "preset2.txt")
(3 "Examples" "Test3" "schematic" "Mig" "Test3" "schematic" "preset2.txt")
(5 "Examples" "Test5" "schematic" "Mig" "Test5" "schematic" "preset2.txt")
(6 "Examples" "Test6" "schematic" "Mig" "Test6" "schematic" "preset2.txt")
(7 "Examples" "Test7" "schematic" "Mig" "Test7" "schematic" "preset2.txt")
(4 "Examples" "Test4" "schematic" "Mig" "Test4" "schematic" "preset3.txt"))
```

preset1.txt

```
form89DoHier = nil
form89DoLib = nil
form89DoAllViews = nil
form89DoAllRefLibs = nil
form89TgtLibSel = "Mig"
form89TgtCellSel = "Use source name"
form89TgtView = "schematic symbol"
form89MigMap = "../SchMig_Test.csv"
form89TgtNameOpt = ""
form89TgtNmOpts = "Prefix"
form89SrcView = "schematic symbol"
form89TermMap = nil
```

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

```
form89ExclLibs = "basic analogLib US_8ths borders"
form89ExclViews = "text adexl constraint physConfig maestro layout abstract"
form89DesLibMaps = ""
```

preset2.txt

```
form89DoHier = t
form89DoLib = nil
form89DoAllViews = nil
form89DoAllRefLibs = nil
form89TgtLibSel = "Mig"
form89TgtCellSel = "Use source name"
form89TgtView = "schematic symbol"
form89MigMap = "../SchMig_Test.csv"
form89TgtNameOpt = ""
form89TgtNmOpts = "Prefix"
form89SrcView = "schematic symbol"
form89TermMap = nil
form89ExclLibs = "basic analogLib US_8ths borders"
form89ExclViews = "text adexl constraint physConfig maestro layout abstract"
form89DesLibMaps = ""
```

preset3.txt

```
form89DoHier = t
form89DoLib = nil
form89DoAllViews = nil
form89DoAllRefLibs = t
form89TgtLibSel = "Mig"
form89TgtCellSel = "Use source name"
form89TgtView = "schematic symbol"
form89MigMap = "../SchMig_Test.csv"
form89TgtNameOpt = ""
form89TgtNmOpts = "Prefix"
form89SrcView = "schematic symbol"
form89TermMap = nil
form89ExclLibs = "basic analogLib US_8ths borders"
form89ExclViews = "text adexl constraint physConfig maestro layout abstract"
form89DesLibMaps = "../Include/SchMig_DesignLibraryMapping.csv"
```

Environment Settings in schMapDefaults.il

Most of the settings of single migration for hierarchy option are still valid for multiple cells migration process. Here is a list of shared settings for multiple cells process.

The working folder setting will be used for multiple cells process working folder also.

```
runDir = "./SCHMAP_RUNDIR" Run directory.  
mapFile = "./SCHMAP_CONFIG/mapping.csv" Mapping file.
```

Source Technology:

```
sourceTech = "gpdk045" Source Technology.  
targetTech = "cds_ff_mpt" Target Technology.  
excludedLibs = "basic analogLib US_8ths borders" Default excluded libraries.  
excludedViews = "text adexl constraint physConfig maestro layout abstract" Default  
excluded views.  
  
.....
```

The new introduced settings for multiple cells process:

- Data input directory-This folder includes a cell list file and preset file.

```
mcRunDir = "./SCHMAP_RUNDIR/MCRUNDIR"
```

- Cell list file. For example,

```
cellsFile = "./SCHMAP_RUNDIR/MCRUNDIR/celllist.txt"
```

- Target libs location.

```
tmpTgtLibLoc = "./SCHMAP_RUNDIR/tgtLibs"
```

- The temporary target library name prefix. For example,

```
tmpTgtLibPrefix = "TmpTgt_"
```

In the migration, a temporary target library folder will be created for each top cell migration. The temporary library name is Target_Library_Name + _prefix + index, for example, TopCell_TmpTgt_1.

- Target library define flag. For example,

```
defineLibs = t
```

This is optional one. If part 2 migration is running on different Virtuoso and at different location with Part1, the `cds.lib` file may not have the target migration libraries defined in Part1. If this option is enabled, multiple cells migration Part2 will create the definitions for the target libraries in the `cds.lib` file. However, if Part2 uses `cds.lib` with the target libraries defined in Part1, this option should not be enabled.

Using Virtuoso Menu

- ➔ To use the multiple cell schematic migration feature in Virtuoso, click the below submenus:

In the CIW, select *Tools – Migrate – Multiple Schematics Mapping*. The *Multiple Schematics Mapping* submenu includes the following items:

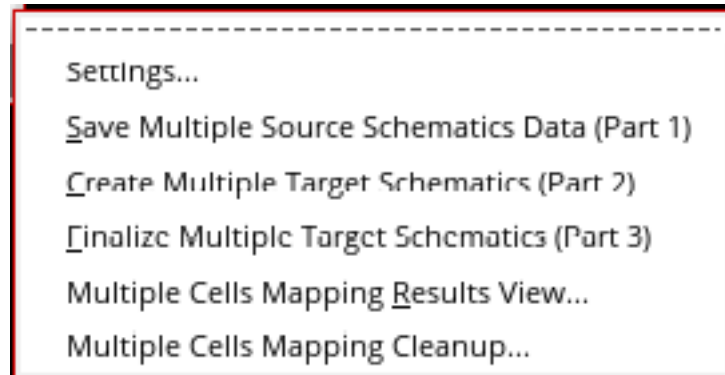
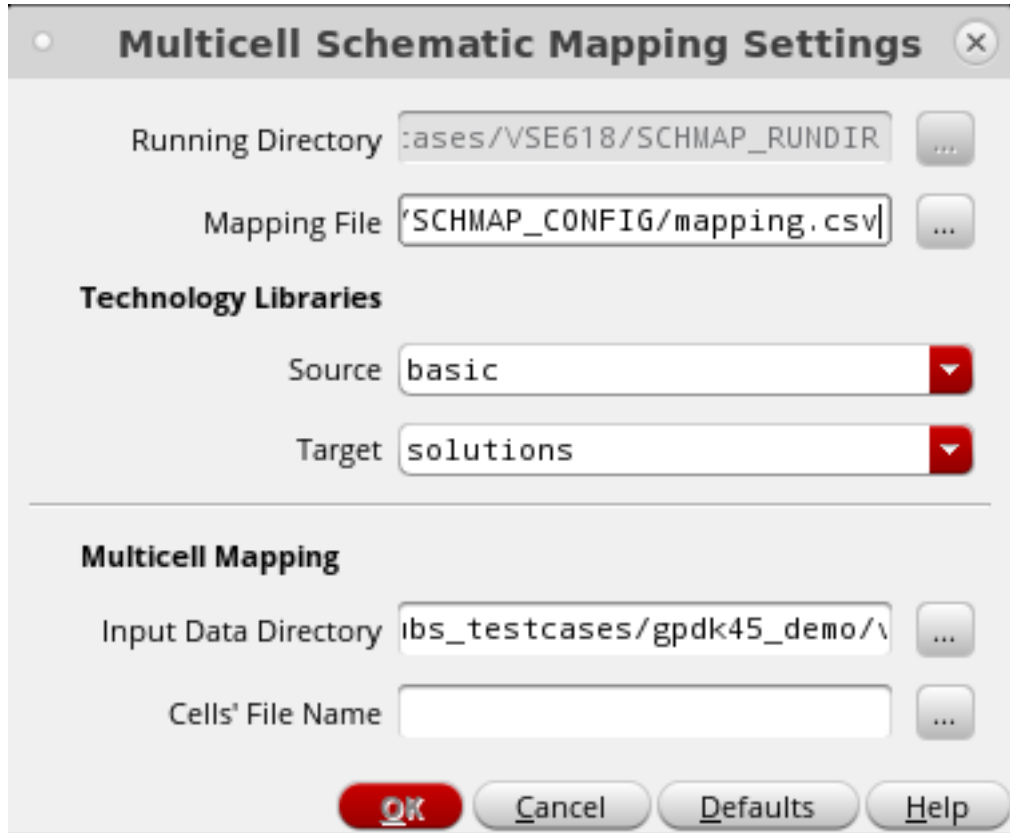


Figure 2-15 Multiple Schematics Mapping Menu

Multicell Schematic Mapping Settings

You must use the *Multicell Schematic Mapping Settings* form to provide all necessary information before starting the multiple cell schematic migration.



The image shows a dialog box titled "Multicell Schematic Mapping Settings". It contains several input fields and buttons. The "Running Directory" field is set to "cases/VSE618/SCHMAP_RUNDIR". The "Mapping File" field is set to "SCHMAP_CONFIG/mapping.csv". Under the "Technology Libraries" section, the "Source" dropdown is set to "basic" and the "Target" dropdown is set to "solutions". Under the "Multicell Mapping" section, the "Input Data Directory" field is set to "ibs_testcases/gpdk45_demo/" and the "Cells' File Name" field is empty. At the bottom, there are four buttons: "OK", "Cancel", "Defaults", and "Help".

Field	Value
Running Directory	cases/VSE618/SCHMAP_RUNDIR
Mapping File	SCHMAP_CONFIG/mapping.csv
Source (Technology Libraries)	basic
Target (Technology Libraries)	solutions
Input Data Directory (Multicell Mapping)	ibs_testcases/gpdk45_demo/
Cells' File Name (Multicell Mapping)	

Figure 2-16 The Multicell Schematic Mapping Settings Form

Save Multiple Source Schematic Data (Part 1)

This is equivalent to interactive migration Part 1, instead it processes a list of top cells defined in a cell list file. It displays the Part 1 confirmation form.

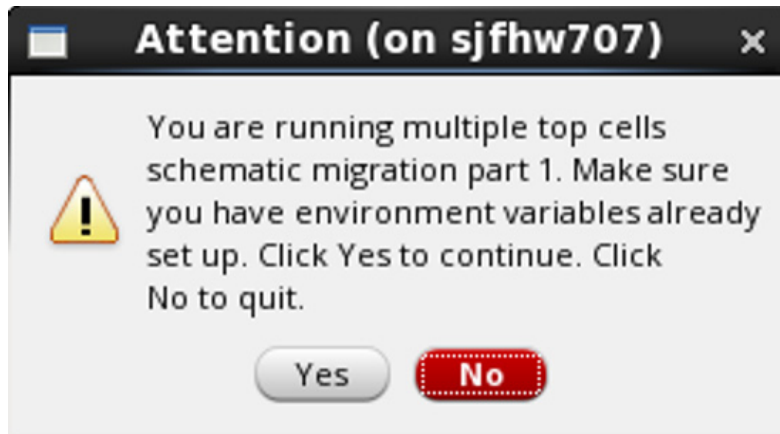


Figure 2-17 The Part 1 Confirmation Form

Create Multiple Target Schematic (Part 2)

This is equivalent to interactive migration Part 2. It displays the Part 2 confirmation form.

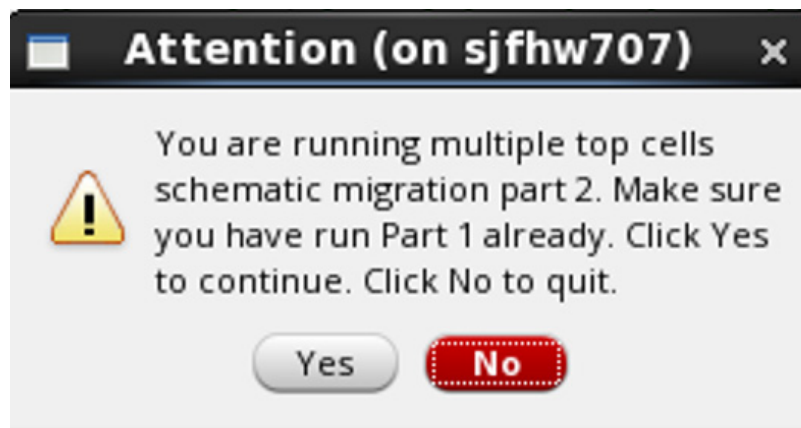


Figure 2-18 The Part 2 Confirmation Form

Finalize Multiple Target Schematic (Part 3).

It displays the Part 3 confirmation form.

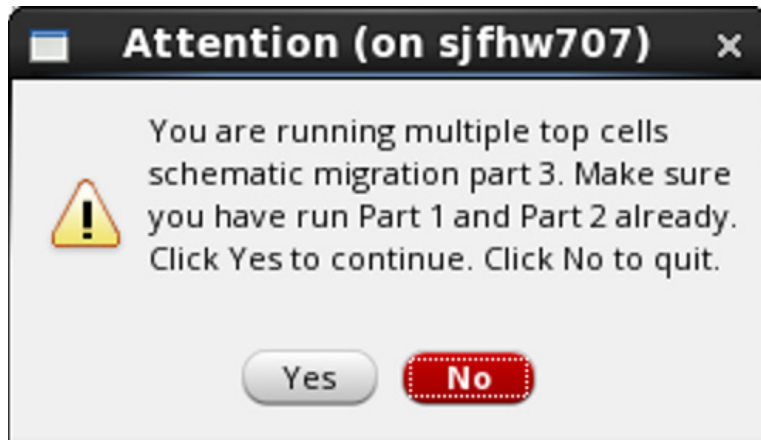


Figure 2-19 The Part 3 Confirmation Form

View Logs Report and Results

A report form displays processed list of cells. Columns 1 to 8 are cells list information. The last three columns are three parts processing status. Right-click on a row to display the

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

context menu, or click *Report Viewer* or the *Result Browser* button to view logs, or to launch the Result Browser for the selected cell.



Figure 2-20 The Results Viewer Form

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

1. The *Report Viewer* form displays the reports of the 3 part3.

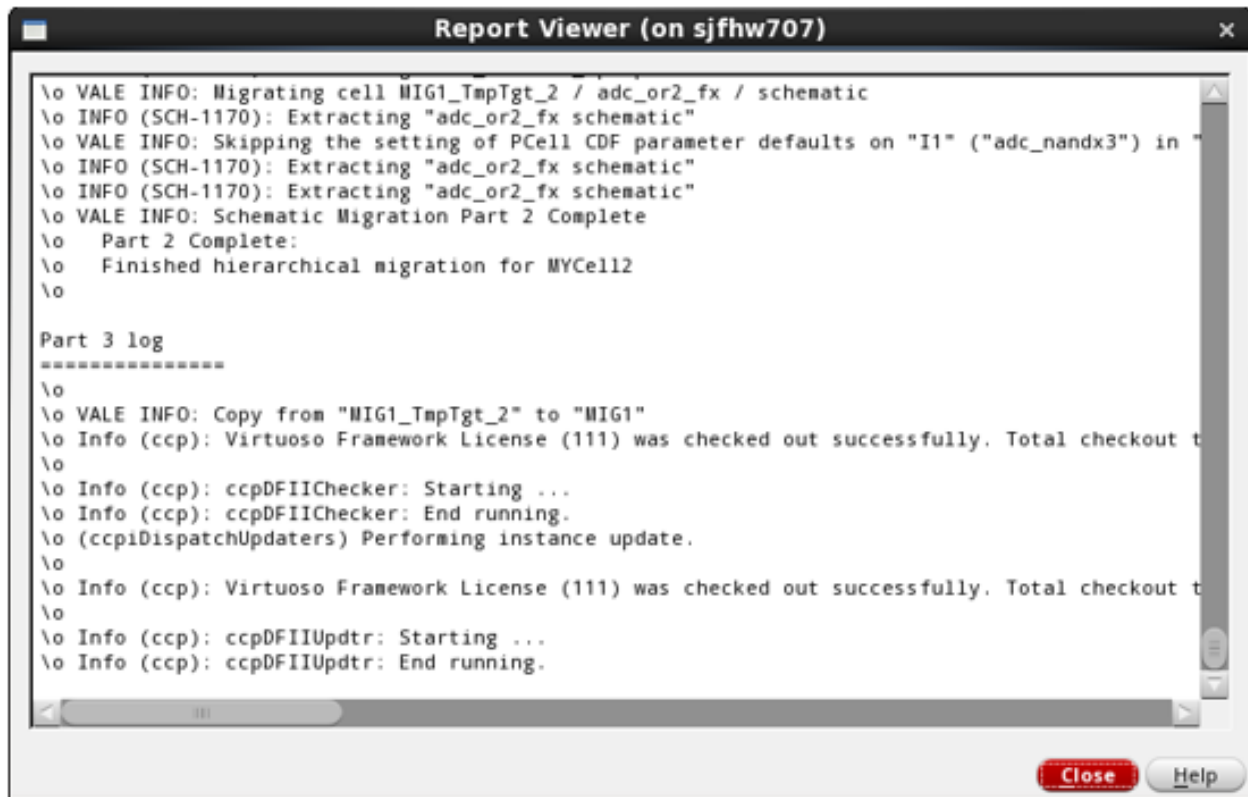


Figure 2-21 The Report Viewer Form

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

2. Result Browser.

Views the processing results as interactive migration.

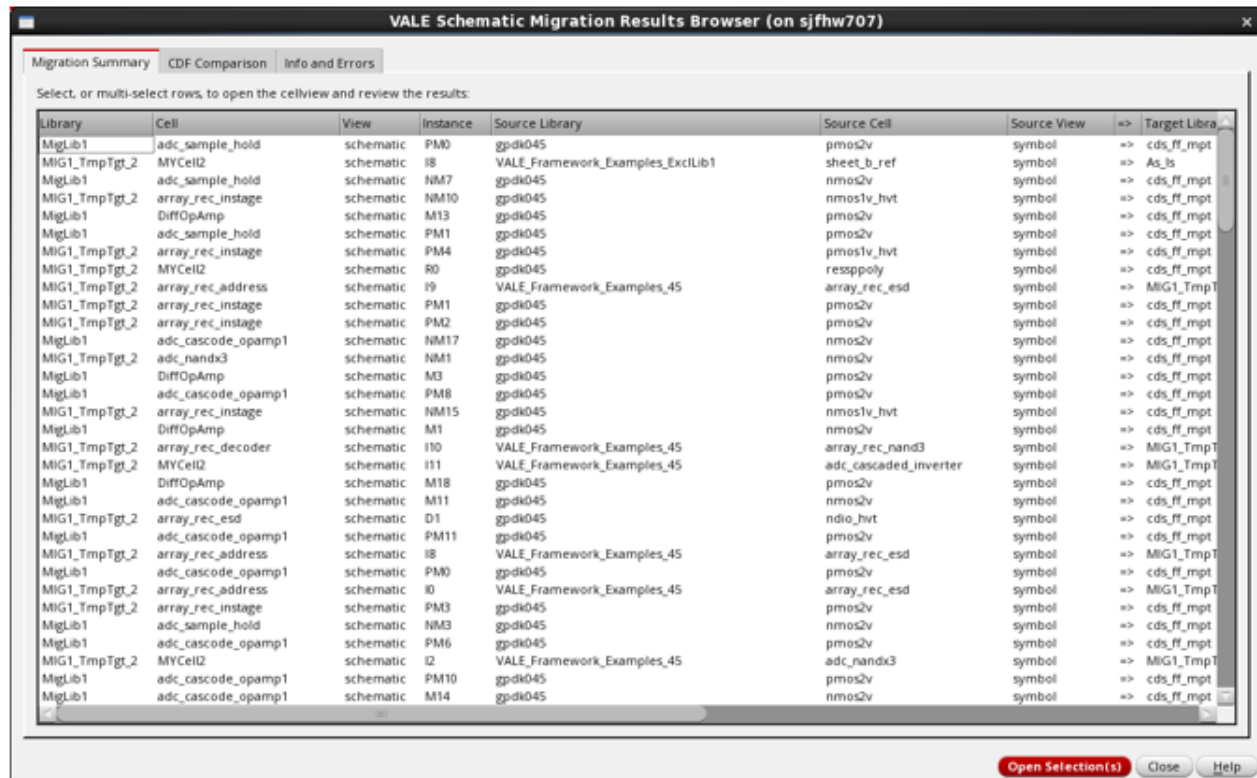


Figure 2-22 The Schematic Migration Results Browser Form

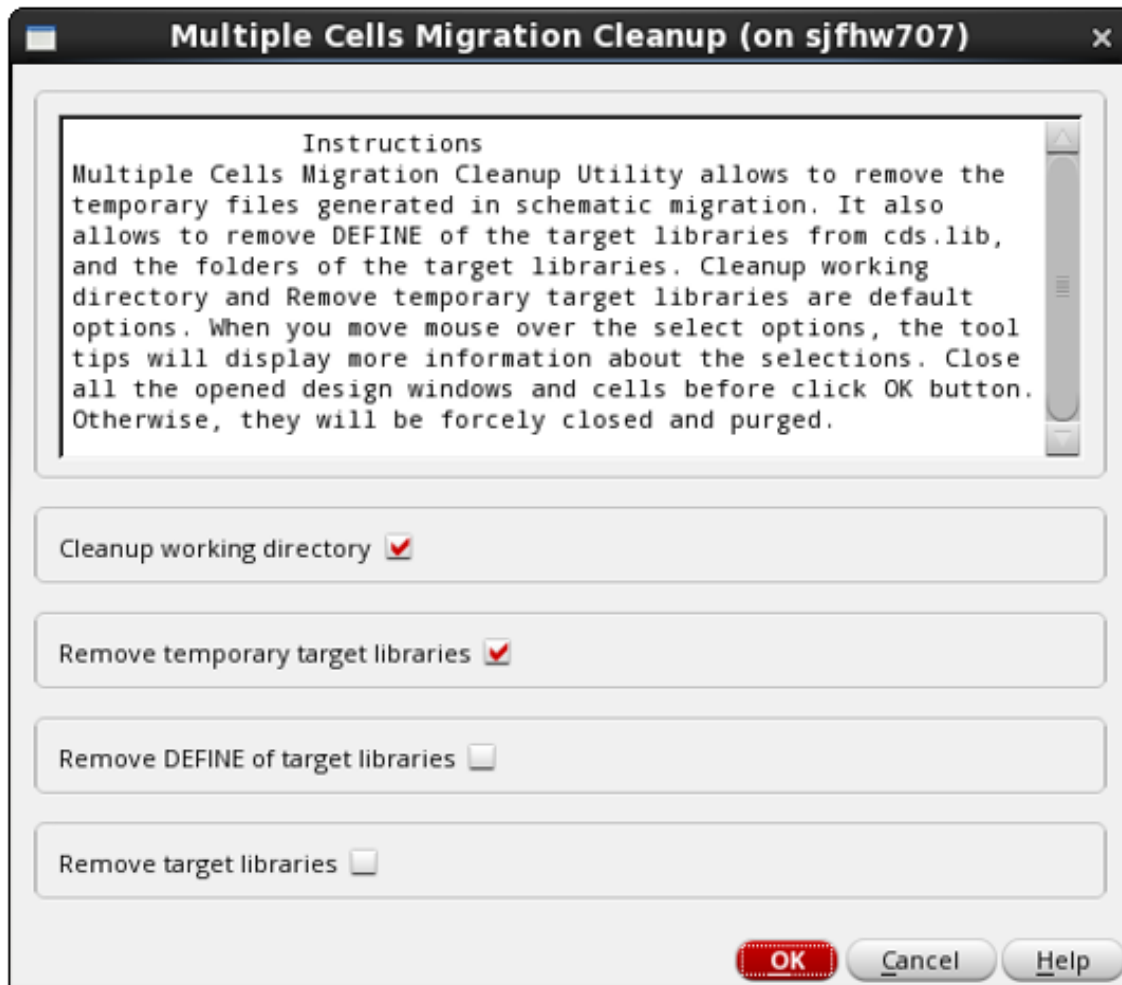
3. Clean up utility.

The software allows you to clean the working folder, temporary target libraries, or all the libraries. Note that the cleaning utility is applied to the same process with the similar settings. For example, if migration with a selected cell list has been setup, and has run

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Schematic Design Migration

part 1, part 2, and part 3, then you can run the utility to clean up the created files. If the settings have been changed, the files created in previous processes will not be removed.



The Multiple Cells Migration Cleanup Form

Schematic Mapping Options

You can specify the spreadsheet mapping file using the following methods. The first one is the most common method to map a schematic file:

- Import a CSV file:

```
mapFile = "./SCHMAP_CONFIG/mapping.csv"
```

- On the Schematic Migration Part 1 Form, under the "More Options" section, use the *Browse* button of the *Migration Mapping File* to locate and run a CSV mapping file.

Schematic Mapping Configuration File Options

Below is the list of schematic mapping configuration file options:

useInFlowMapFileChecker

If set to `t`, the mapping file is checked by the internal map file checker located in this directory:

```
<cds_install>/share/cdssetup/dfII/schMap/schMigMapChecker.py
```

Note: `schMigMapChecker.py` requires Python 3.

If not specified, or specified as `nil`, no map file checking is performed. However, the map file can be checked before running migration with the following command:

```
<cds_install>/share/cdssetup/dfII/schMap/schMigMapChecker.py /path/to/mapfile.csv  
cc_7path/for/the/output.csv
```

where parameter prefix is `cc_` by default.

unmappedPassThru

If set to `t`, any instance with a library name defined in the mapping file for another cell (a known library), and no mapping in the map file for this cell, passes through migration unmodified. If unset, or set to `nil`, the instance is deleted from the target schematic and wires are left open-circuit.

enableTgtNameBySrcPar

If set to `t`, the target cell name in the map file may be a reference to a source cell parameter. For example, if the source cell parameter *cellName* has the target cell name, then placing `cc_cellName` in the target cell name field of the map file greatly reduces the number of lines in the map file.

bypassPreTriggers

If set to `t`, Schematic Design Migration bypasses the pre-trigger functions in Part 1 of the migration.

preMigInstTrigger

If set to a user defined pre-trigger function name, then the function must receive an instance object as an argument, to be run per instance loop before saving the source data in Part1 migration. The function name can be set as string or symbol value, for example:

`preInstUserTrig()` or `preInstUserTrig`.

preMigCellTrigger

If set to a user defined pre-trigger function name, then the function must receive a cell object as an argument, to be run per cell loop before saving the source data in Part1 migration. The function name can be set as string or symbol value, for example: `preCellUserTrig()` or `preCellUserTrig`.

deleteSrcSimHistory

By default, any maestro or adexl history data will be copied to the target libraries. To delete this data on the source side (this modifies the source library) before copy, set this option to `t`.

deleteTgtSimHistory

By default, any maestro or adexl history data will be copied from the source libraries. To delete this data on the target side after copy, set this option to `t`.

bypassPostTriggers

If set to `t`, Schematic Design Migration bypasses the post-trigger functions in Part 2 of the migration.

postMigInstTrigger

If set to a user defined post-trigger function name, then the function must receive an instance object as an argument, to be run per instance after re-creating the target data in Part2 migration. The function name can be set as string or symbol value, for example:

`postInstUserTrig()` or `postInstUserTrig`.

Note: A per cell post migration trigger will override post instance trigger defined by `postMigInstTrigger`

postMigCellTrigger

If set to a user defined post-trigger function name, then the function must receive a cell object as an argument, to be run per cell after re-creating the target data in Part2 migration. The function name can be set as string or symbol value, for example: `postCellUserTrig()` or `postCellUserTrig`.

bypassLibCheck

If set to `t`, Schematic Migration will bypass the default check to ensure that the source PDK has been removed from the `cds.lib` file for Part 2 of the migration.

bypassRewiring

If set to `t`, Schematic Migration will bypass the automatic rewiring in Part 2 of the migration.

checkPCellToMap

If set to `t`, Schematic Migration compares the post-migration Pcell CDF Parameter values to the specification of CDF Parameters in migration mapping file. The results are displayed in the *CDF Parameters* tab of the Migration Results Browser. The default is `t`.

skipPCellDefaults

If set to `t`, Part 2 will create an instance of a Pcell but not set all the Pcell defaults. If the Pcells were defined to be created with default parameters all of the time, this step is not necessary. The default is `t`.

setTargetParameterTopDown

If set to `t`, the mapping file, *match parameters*, will set from just below the "device" map line to the last match parameter for that device. The default order is to set the last match parameter first, and build up to the "device" map line. Some PDKs require CDF parameters to be set in a particular order.

keepDataFile

If set to `t`, Part 2 will not remove temporary migration data; all Part1 data files will be saved. This is especially helpful during testing of a new map file, where an undo will allow multiple Part 2 runs on the same Part 1 schematic. This is typically used for debug with the `doNotSaveResults` option.

doNotSaveResults

If set to `t`, Part 2 will not save the migration result, all Part2 cellviews will be modified but not saved. This is especially helpful during testing of a new map file, where an undo will allow multiple Part 2 runs on the same Part 1 schematic. This is typically used for debug with the `keepDataFile` option.

removeDanglingNets

If set to `t`, any "line" or "path" object type in the schematic that does not connect to anything either are removed. Floating, unlabeled wires are removed all the way to the original instance pin, or first labeled wire (including unnecessary solder dots). This parameter may be entered at any time in the CIW.

postMigrationNetCheck

If set to `t`, a before and after comparison of net names will output any new nets that have been created in the "Status" column of the Results Browser Cell Mappings tab. In addition, each instance terminal will be checked before and after, and any change to the net name reported in the same location of the Results Browser.

maxCellsToOpen

If set, it will limit the Migration Results Browser to opening that integer number of cellviews at once. The default value is 5. This parameter may be entered at any time in the CIW.

migBrowserStatusFilters

If set, when *Migration Results Browser* is selected in the Migration menu, this field, a list, will filter what gets displayed in the Cell Mappings tab of the Results Browser. It is a list of regular expression strings. Each line of the table will be filtered through each regular expression string before being displayed on the Results Browser. This parameter may be entered at any time in the CIW. For example, if the following two regular expression strings are specified as follows:

```
list("^Mismatched Parameter\\(s\\)$" "COMPLETE")
```

then any line item in the table with “Mismatched Parameter(s)” or “COMPLETE” will be filtered out and not displayed.

Note: By default, when Schematic Migration Part 2 completes, there is no filter when “Open Results Browser” is selected. It is only set to the `migBrowserStatusFilters` value when run from the menu.

migBrowserResultsFile

If set, the tool uses this as the default results file when you do not provide any results file for a lower level cell. This can be used after running a top-level cell, and if multiple results files are present in the run directory which contains the results for current cell. You can click *Cancel* in the UI form that appears for file selection, and the tool automatically checks if the results are present in default results file set using:

```
migBrowserResultsFile = "/path/to/specific/runDir/Top_CellMappings.csv"
```

symbolChangeOptions

This option provides control of symbol and pin optimization methods.

The default value is `list (t t nil nil t)`. If each value in the list represents a position:

- Position 0: Check symbol for size changes
- Position 1: Perform optimization for size changes
- Position 2: Check symbol for rotation (applies to map file libraries)
- Position 3: Perform optimization for rotation
- Position 4: Enable pin optimization methods

copyProperties

This option controls the copying of design properties (not to be confused with CDF Parameters on a Pcell). The default value is `nil`.

- If set to `t`, all cell and instance properties are copied from source to target.
- If set to `nil`, no cell or instance properties are copied from source to target.
- If set to `list("copyProp")`, only the map file `#copyProp` information is used.
- If set to `list("prop1Name" "prop2Name")`, any cell view or instance properties with `prop1Name` or `prop2Name` are copied from source to target.

instConnChangesByCDF

Used to provide a list of cell names that change their connectivity based on a parameter setting. If Schematic Migration encounters one of the listed cell names, no pre-placement check, according to `useInstPreCheck` will occur.

usePlacementEffort

If non-`nil`, and if `useInstPreCheck` is not `nil`, an alternate method for instance placement is attempted as follows:

- If `#keepPinLoc` is specified, then `#keepCenter` is attempted for shorts avoidance.
- If `#keepCenter` is specified, then source instance origin is used.
- If `#symbolOffsetX` and/or `#symbolOffsetY` is specified, then source instance origin is used.

useInstPreCheck

Enables a pre-placement check of the target instance pins to see if they land on an existing wire that would create a short circuit. If `usePlacementEffort` is not `nil`, an alternate method for placement may be used, otherwise any shorts from the current placement method are reported in the Migration Results Browser.

mfactorAndVectorTrans

Enables migrating from m-factored devices to vectored devices, or from vectored devices to m-factored devices, using the Other Parameters specification `#mfactor2vector` or `#vector2mfactor`.

For example, given a source instance named `PM0` with parameter `m=4`:

- `#mfactor2vector=m(0)` produces target instance `PM0<0:3>`
- `#mfactor2vector=m(2)` produces target instance `PM0<2:5>`
- `#mfactor2vector=m(4)` produces target instance `PM0<4:7>`

In `#mfactor2vector=m(4)`, 4 is the starting bit index of the target instance, while `m` is the name of the multiplicity parameter in the source. The specification changes with the name of the multiplicity parameter, that is, if the parameter is `f` instead of `m`, the specification in this example becomes `#mfactor2vector=f(4)`.

`#vector2mfactor` works the opposite way and has a different syntax. Given a source instance named `PM0<0:3>`:

- `#vector2mfactor=m|eq|(cc_icnt*2)` produces target instance `PM0` with `m=8`
- `#vector2mfactor=m|eq|(cc_icnt*4)` produces target instance `PM0` with `m=16`

In the expression `m|eq|(cc_icnt*2)`:

- `m` is the name of the m-factor parameter of the target instance
- `eq` is a keyword that marks the following text as a mathematical expression
- `cc_icnt` is a variable that holds the instance count, which is the number of bits, for a mapped instance

You can use this variable in an expression whose result is the value of the target m-factor parameter.

postMigrationCb

Adds a user defined callback after clicking OK on the Select the Migration to Finish (Part 2) form. If defined, it returns the following arguments:

```
postMigrationCb = 'myCb
myCb(libOrTopCellName migMode runDir)
libOrTopCellName "cellA"
```

Virtuoso Custom Design Migrate User Guide

Schematic Mapping Options

In case of a full library migration, the name of the library is returned.

```
migMode "Single Cell migration"
```

Other options are "Cell Hierarchy migration" and "Full library migration".

```
runDir "./SCHMAP_RUNDIR"
```

cdfgFormEmulation

If `cdfgForm` is used in the callbacks of parameters, the below value will emulate *cdfgForm* to *cdfgData*:

```
cdfgFormEmulation = 'cdfgData'
```

designInventoryPath

Used to define the path of the schematic design inventory file. The default value is `nil`. The default path of the schematic design inventory file is `"./SCHMAP_CONFIG/SchDesignInventory.il"`.

shortAvoidance

If set to `t`, the tool attempts to fix shorts that are created as a result of the migration. Certain instances may be placed to the side of the schematic, reconnected using wire stubs. The default is `nil`.

Note: The tool can cause flightlines to appear on the schematic.

Other Environment Options

excludedLibs: Any instance from one of the ignore libraries will pass through to the target schematic unmodified.

```
excludedLibs = "basic US_8ths borders analogLib"
```

sourceTech: The source technology library.

```
sourceTech = "gpdk045"
```

targetTech: The target technology library.

```
targetTech = "cds_ff_mpt"
```

excludedViews: Specify views to skip during migration.

```
excludedViews = "layout abstract"
```


Virtuoso Custom Design Migrate User Guide

Schematic Mapping Options

runDir: Optionally set the migration run directory. The default is to use the current run directory and `runDir/SchMig`. For example:

```
runDir = "/path/for/run/directory"
```

Virtuoso Custom Design Migrate User Guide

Schematic Mapping Options

Schematic Utilities

Inspect Schematic Instance Properties

Enables the review and optional deletion of instance properties often carried over by copies of cellviews. Instance properties are not Pcell CDF Parameters.

To use this utility, open one or more schematic windows. For this example, Figure 1's target schematic is used.

Select *Migrate – Schematic – Utilities – Inspect Schematic Instance Properties*, and a form will open with the currently opened schematics. Select the cell of interest. If there are no schematic instance properties, the following message will be displayed:

VALE INFO: No instance properties without a corresponding CDF Parameter were found in the current cellview, Aug20_doc1 to inspect and possibly remove.

If there are schematic instance properties, the following form will appear, listing the cell name, instance name, and property name found on the instance.



Figure B-1 Figure 19 Select one or more for Deletion

Choose *Cancel* to only observe and exit. Choose one or more line item (click and shift-click) and select *OK* to remove the instance property.

Schematic Design Inventory

Displays information about the current design hierarchy.

Select *Migrate – Schematic – Utilities – Schematic Design Inventory*, and it will display the following four sections in a new text window.

Virtuoso Custom Design Migrate User Guide

Schematic Utilities

Section 1:

1. Design Tree for Framework_Examples_ff_mpt / example_SchDevInv_1 / schematic

```
;cds_ff_mpt p2svt symbol (1)
;cds_ff_mpt n2svt symbol (1)
;Framework_Examples_ff_mpt_REF SchMig_FF_Cell1 symbol (1)
;      ;cds_ff_mpt pllvt symbol (1)
;      ;cds_ff_mpt n1lvt symbol (1)
;      ;cds_ff_mpt p2svt symbol (1)
...
```

Section 2:

2. Library References and Cell Usage

Reference Lib = cds_ff_mpt (Lib Contents: 51 cells):

n1hvt	=	3
n1lvt	=	3
n1svt	=	5
n2svt	=	3
p1hvt	=	3
p1lvt	=	3
p1svt	=	4
p2svt	=	4

Sum : 28 instances (from 8 cells)

...

Section 3:

The specific parameters listed are defined in the design inventory file. You can define the path of the file using the `designInventoryPath` argument.

Virtuoso Custom Design Migrate User Guide

Schematic Utilities

3. Table of PDK Devices grouped by similar CDF Parameters

finpitch	Library	Cell	Count	nfin	nf	l	m	polypitch
	ns	Cell Hierarchy	Path					
=====	=====	=====	=====	=====	=====	=====	=====	=====
48n	cds_ff_mpt	n1svt	3	5	3	18.0n	1	poly86
	1	example_SchDevInv_1/.../n1svt	(NM1)					
		example_SchDevInv_1/n1svt	(NM1)					
		example_SchDevInv_1/.../n1svt	(NM1)					
48n	cds_ff_mpt	p1svt	3	2	1	18n	1	poly86
	1	example_SchDevInv_1/.../p1svt	(PM0)					
		example_SchDevInv_1/p1svt	(PM0)					
		example_SchDevInv_1/.../p1svt	(PM0)					
48n	cds_ff_mpt	p1hvt	3	6	6	18.0n	1	poly86
	1	example_SchDevInv_1/.../p1hvt	(PM1)					
		example_SchDevInv_1/p1hvt	(PM1)					
		example_SchDevInv_1/.../p1hvt	(PM1)					

Section 4:

A listing of the PDK Pcells per cell, and specific parameters of each PDK Pcell according to a user specification, in the design inventory file. You can define the path of the file using the `designInventoryPath` argument.

4. CDF Parameter List per Instance

```
Framework_Examples_ff_mpt / example_SchDevInv_1 / schematic
  p2svt (M0) in Framework_Examples_ff_mpt / example_SchDevInv_1 / schematic
    Parameter nfin = "6"
    Parameter nf = "1"
    Parameter l = "18.0n"
```

Virtuoso Custom Design Migrate User Guide

Schematic Utilities

```
Parameter m = "1"
Parameter polypitch = "poly86"
Parameter finpitch = "48n"
Parameter ns = "1"
n2svt (M1) in Framework_Examples_ff_mpt / example_SchDevInv_1 / schematic
Parameter nfin = "5"
Parameter nf = "5"
Parameter l = "18.0n"
Parameter m = "1"
Parameter polypitch = "poly86"
Parameter finpitch = "48n"
Parameter ns = "1"
```

The specification of section 3 and 4 CDF parameters for Pcells is a file. The cell name and parameters can be used explicitly (as in the `n1lvt` example below), or a regular expression syntax can be used for a number of similar devices (as in the `n1svt`, `p1svt`, `n1hvt`, `p1hvt` examples below).

```
rtDpl->parameterList = list(
  ;; cds_ff_mpt
  ;; n1lvt
  list(nil 'lib "cds_ff_mpt" 'cell "n1lvt" 'type "FET"
    'parameters list(
      "nfin" "nf" "l" "m" "polypitch" "finpitch"
    ))
  ;; n1svt p1svt n1hvt p1hvt
  list(nil 'lib "cds_ff_mpt" 'regexp "[np]1[hs]vt" 'type "FET"
    'parameters list(
      "nfin" "nf" "l" "m" "polypitch" "finpitch"
    ))
)
```

Print Nondefault Pcell CDFs

Select *Migrate – Schematic – Utilities – Print Nondefault Pcell CDF* to display the Pcell CDF parameters for all the Pcells in the current cellview that are changed from the default. It opens a text view file with the following information.

```
Device Name: "PM6"      Cell Name: "p1hvt"      Library: "cds_ff_mpt"
"PM6" has the following differences:
  "nfin" has been changed from  "2" to "5"
  "nf" has been changed from  "1" to "2"
  "fw" has been changed from  "62n" to "206.00n"
```

Virtuoso Custom Design Migrate User Guide

Schematic Utilities

"asej" has been changed from "6.528e-15" to "3.264e-14"

"adej" has been changed from "6.528e-15" to "1.632e-14"

"psej" has been changed from "2.32e-07" to "7.52e-07"

"pdej" has been changed from "2.32e-07" to "3.76e-07"

"lrtd" has been changed from "18n" to "68.0n"

Device Name: "PM4" Cell Name: "plhvt" Library: "cds_ff_mpt"

"PM4" has the following differences:

"nfin" has been changed from "2" to "5"

"nf" has been changed from "1" to "2"

"fw" has been changed from "62n" to "206.00n"

"asej" has been changed from "6.528e-15" to "3.264e-14"

"adej" has been changed from "6.528e-15" to "1.632e-14"

"psej" has been changed from "2.32e-07" to "7.52e-07"

"pdej" has been changed from "2.32e-07" to "3.76e-07"

"lrtd" has been changed from "18n" to "68.0n"

Device Name: "NM6" Cell Name: "nlsvt" Library: "cds_ff_mpt"

"NM6" has the following differences:

"nfin" has been changed from "2" to "5"

"nf" has been changed from "1" to "3"

"fw" has been changed from "62n" to "206.00n"

"asej" has been changed from "6.528e-15" to "3.264e-14"

"adej" has been changed from "6.528e-15" to "3.264e-14"

"psej" has been changed from "2.32e-07" to "7.52e-07"

"pdej" has been changed from "2.32e-07" to "7.52e-07"

"lrtd" has been changed from "18n" to "68.0n"

Scan Complete

Hierarchical Check and Save

Run Check and Save hierarchically on the current cellview.

Library Check and Save

Run Check and Save on the library referenced by the current cellview.

Source-side Migration Map Generator

This feature traverses the hierarchy of the currently open cellview and creates a Comma Separated Values file for all the source devices that need to be mapped.

Virtuoso Custom Design Migrate User Guide

Schematic Utilities

To run, select:

Select *Migrate – Schematic – Utilities – Create Source-side Migration Map*

A form opens to allow the user to select which libraries contain cells that need to be mapped. In this case, gpdk045 is the source PDK, and VALE_Framework_Examples_45_RebindLib1 is a design library that should not be traversed, but simply have its instances referenced.

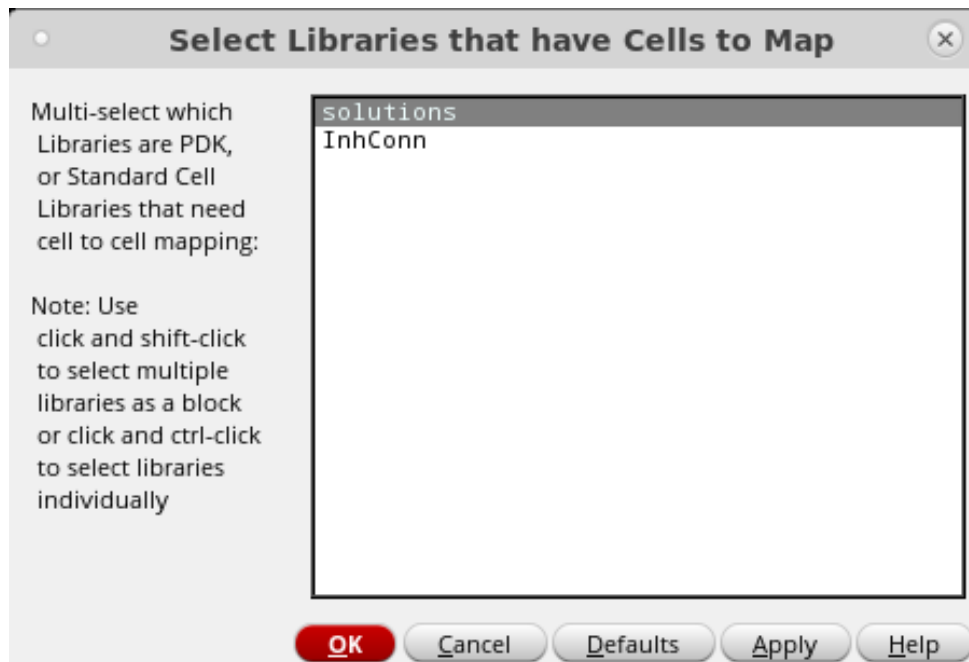


Figure B-2 Select Libraries that Have Cells to Map

Another form opens after the utility has queried the cells to map for CDF Parameters that are *changed from the default*. Select only the relevant CDF Parameter names for mapping. Some of the CDF Parameters are likely derived parameters and are therefore redundant.

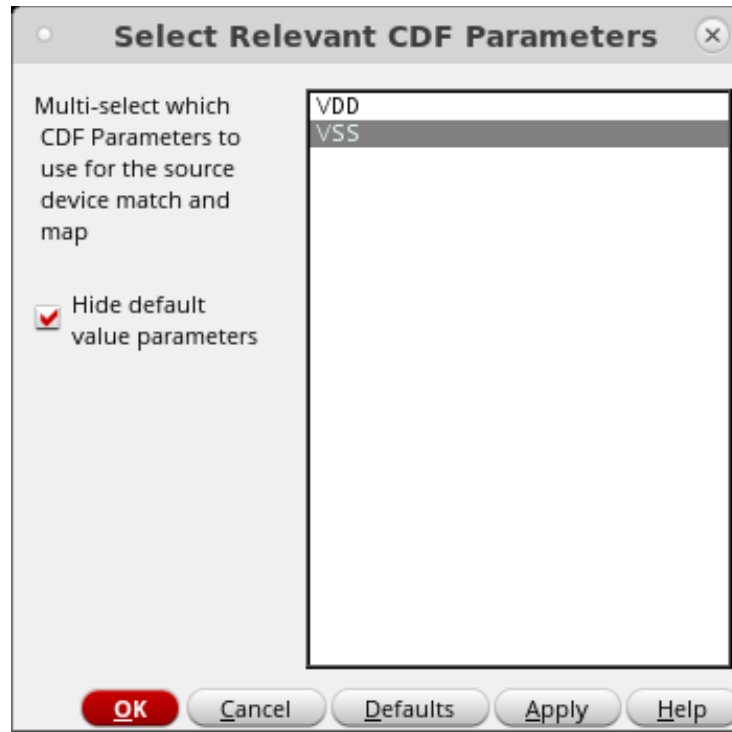


Figure B-3 Select Relevant CDF Parameters

Note: Only CDF parameters that are changed from the default are included because using CDF parameters that are the same as the default only adds unnecessary information to the migration map.

Virtuoso Custom Design Migrate User Guide

Schematic Utilities

The source-side migration data will be generated according to the selections above, and will generate a file in /RUNDIR/SchMig/srcMigMap.csv

```
device,gpdk045,pmos2v,,,,,,,,
match parameter,m,,50,,,,,,,,
match parameter,l,,150n,,,,,,,,
match parameter,w,,5u,,,,,,,,
match parameter,fw,,5u,,,,,,,,
match parameter,fingers,,1,,,,,,,,
,,,,,,,,
device,gpdk045,pmos2v,,,,,,,,
match parameter,m,,12,,,,,,,,
match parameter,l,,350n,,,,,,,,
match parameter,w,,5u,,,,,,,,
match parameter,fw,,5u,,,,,,,,
match parameter,fingers,,1,,,,,,,,
,,,,,,,,
device,gpdk045,nmos2v,,,,,,,,
match parameter,m,,1,,,,,,,,
match parameter,l,,280n,,,,,,,,
match parameter,w,,1.2u,,,,,,,,
match parameter,fw,,1.2u,,,,,,,,
match parameter,fingers,,1,,,,,,,,
,,,,,,,,
```

Figure B-4 Source-side Map File

Virtuoso Custom Design Migrate User Guide

Schematic Utilities

Mapping File Syntax for Schematic Migration

Header Row

Keyword	Source Library or Source parameter name	Source Cell or Source Parameter Description	Source Parameter Value	Display	Target Library or Target Parameter Name	Target Cell or Target Parameter Description	Target Parameter Value	Notes	Other Parameters
---------	---	---	------------------------	---------	---	---	------------------------	-------	------------------

Figure C-1 Mapping File Header

- The keyword can be “device”, “match parameter”, “display parameter” or “Map Library”. It must have 10 fields, and therefore 9 commas. The header row should not contain special characters. Only alphanumeric characters and spaces can be used.
- The Source Library or Source Parameter Name is the name of the Pcell CDF parameter with the prefix "cc_".
- Source Cell or Source Parameter Description field: this field is optional, for descriptive purposes only. The field accepts wildcard characters (*), which stand for zero or more characters. For example, *mos* and *_ch_* can stand for either nmos4 and n_ch_4, or pmos4 and p_ch_4, respectively.
- Source Parameter Value:
 - ❑ If it is a value (a number or string), then the parameter will be used as a requirement for the binding from the source to the target.
 - ❑ If it is a conditional (determine by the presence of the delimiter character "|"), then the conditional expressions will be evaluated as a requirement for the binding (a logical and function)
- The Display field, if set to "parameter&value", prints text regarding the source parameter name and value to the canvas as a note when keyword is "display parameter".
- The Target Library or Target Parameter Name is the name of the Pcell CDF parameter with the prefix "cc_".

Virtuoso Custom Design Migrate User Guide

Mapping File Syntax for Schematic Migration

- The Target Cell or Target Parameter Description field: this field is optional, for descriptive purposes only. The field accepts wildcard characters (*), which are filled in with the characters matched by the corresponding source.
- The Target Parameter Value field will set the value of mapped cell parameters.
 - ❑ If the Target Parameter Value field references a value, it will use the value to set the target parameter.
 - ❑ If the Target Parameter Value field references a string that begins with "calc", it is the start of a conditional evaluation to set the target parameter value.

Creating a Mapping Spreadsheet - Examples

The following examples explain how to create a mapping spreadsheet:

Map a device without any CDF parameter mapping

Note: Each map does not need the header row, it is only there for reference in each example.

Keyword	Source Library or Source parameter name	Source Cell or Source Parameter Description	Source Parameter Value	Display	Target Library or Target Parameter Name	Target Cell or Target Parameter Description	Target Parameter Value	Notes	Other Parameters
device	gpd045	pmos1v			cds_ff_mpt	p1svt			
match parameter									

Figure C-2

This example maps the PDK library, "gpd045" cell "pmos1v" to PDK Library "cds_ff_mpt" cell "p1svt". The empty match parameter line is not necessary.

Note: The *Notes* column and *Other Parameters* column headings are required in the spreadsheet header to generate a proper CSV file with 10 columns, but the columns can be empty. If the spreadsheet does not generate a 10 commas per line, the CSV will not import properly.

Map a device only when a set of source parameters and values match

Keyword	Source Library or Source parameter name	Source Cell or Source Parameter Description	Source Parameter Value	Display	Target Library or Target Parameter Name	Target Cell or Target Parameter Description	Target Parameter Value	Notes	Other Parameters
device	gpd045	pmos1v_mt			cds_ff_mpt	p1mt			
match parameter	fw	Finger Width	180n						

Figure C-3 Set of Source Parameters and Values Match

If a parameter exists on the source side, it is a required match parameter. The mapping will only happen if all the parameters listed here match the values on the source device. In this example, if PDK Library "gpdk045", cell "pmos1v" has a CDF Parameter named "fw" and it is equal to 180n, then this device will migrate to a PDK Library "cds_ff_mpt" cell "p1lv" device; otherwise, pmos1v_hvt will not map to p1lv.

Ensure that any required match parameter does not leave other devices of the same type unmatched. For example, to compliment the fw = 180n match above, create a fw != 180n below. When a source parameter is used in the Source Parameter Value field, the source parameter prefix "cc_" is required.

Keyword	Source Library or Source parameter name	Source Cell or Source Parameter Description	Source Parameter Value	Display	Target Library or Target Parameter Name	Target Cell or Target Parameter Description	Target Parameter Value	Notes	Other Parameters
device	gpdk045	pmos1v_mt			cds_ff_mpt	p1lv			
match parameter	fw	Finger Width	cc_fw!=180n						

Figure C-4 Set of Source Parameters and Values Match

The *Source Parameter Value* field will allow for literals (numbers, strings), special string handling with single quotes, equations with other source parameters, and conditional "and" equations with source parameters. These options are for controlling the map of a source device to a target device.

Keyword	Source Library or Source parameter name	Source Cell or Source Parameter Description	Source Parameter Value	Display	Target Library or Target Parameter Name	Target Cell or Target Parameter Description	Target Parameter Value	Notes	Other Parameters
device	gpdk045	nmos2v			cds_ff_mpt	n2svt			
match parameter	par1		2.0					float	
match parameter	par2		3.00E-06					scientific notation	
match parameter	par3		thickOx					string	
match parameter	par4		'this-2/that-3'					special string in single quote	
match parameter	par5		cc_fingers*2					equation	
match parameter	par6		cc_fingers<4 cc_m =2					conditional and	

Figure C-5 Further map match filtering options

In checking source CDF parameter values, the Pcell may store a float such as 2.0 as a string. It may require iteration to see what is required for a particular Pcell and a particular PDK. In some cases, special equations or functions may need to be used to make the match effective. For example, it may be necessary to wrap a floating point value such as 2.0 in a function such as pcExprToString(2.0) to achieve the desired result.

Virtuoso Custom Design Migrate User Guide

Mapping File Syntax for Schematic Migration

If you include any math operators, the mapping system will assume you want your text to be evaluated as an expression. To prevent this, enclose your text in single quotes. For example, $a-b$ is interpreted as the expression a minus b , while `'a-b'` is interpreted as literally the text inside the single quotes. Quoting is useful if you need to refer to a CDF parameter whose name contains embedded math operators.

You can also use double quotes around text to prevent evaluation. This can be clearer to read if the text inside the quotes contains single quotes. Below are some examples that show how to preserve the literal text of an expression, rather than evaluating it during mapping:

```
'pPar("p1") '  
'pPar("n1")*2 '  
"f('w) "  
"n-ch"  
'p-ch'
```

In the above examples, `"f('w) "` calls the function `f()` with a symbol argument, while `"n-ch"` means literally n -dash- ch , and not n minus ch .

Multiple conditional statements are separated by a space for source parameters. Each conditional has the following structure:

`leftSide|operator|rightSide`

where `leftSide` and `rightSide` may both be a literal number or string, or an equation, and the operator can be `=`, `!=`, `<`, `>`, `>=`, or `<=`.

Setting Target Device Parameters

Keyword	Source Library or Source parameter name	Source Cell or Source Parameter Description	Source Parameter Value	Display	Target Library or Target Parameter Name	Target Cell or Target Parameter Description	Target Parameter Value	Notes	Other Parameters
device	gpd045	nmos1v_hvt			cds_ff_mpt	n1hvt			
match parameter					m	Multiplicity	4		
match parameter					nf	Number of Fingers	cc_fingers		
match parameter					nfin	Number of Fins	round(cc_fw/64n)		

Figure C-6 Target Device Parameters

Like source parameters, target parameters may be literal numbers or strings, source parameters references, SKILL equations or conditional equations. If a string literal contains mathematical characters such as `-`, `+`, `/` or `*`, single quotes must be used around the string, but do not wrap equations in single quotes.

Virtuoso Custom Design Migrate User Guide

Mapping File Syntax for Schematic Migration

In this example, PDK Library "gpdk045" cell "nmos1v_hvt" always maps to PDK Library "cds_ff_mpt" cell "n1hvt", without source match conditions as in examples 1 and 2. However, for each map, the target instance has three CDF Parameters set in different ways.

The first match parameter line sets the CDF Parameter "m" to a literal integer value 4. Depending on how the CDF Parameter is coded, special attention to value type might be necessary. For example, if the value 1.0 is used, and the CDF Parameter expects an integer, it might issue an error and not map this parameter. Check the CIW output to see if there are any errors or warnings coming directly from the Pcell itself; if so, adjust the Target Parameter Value column.

The second match parameter line sets the CDF Parameter "nf" to a reference source parameter, "fingers". The prefix "cc_" tells Schematic Migration that this is a referenced parameter.

The third match parameter line sets the CDF Parameter "nfin" to a SKILL expression, including constants and referenced source parameters. In this case, "fw" is 120n, which, when divided by 64n yields 1.875, which is rounded to 2, so "nfin" is set to 2. A divisor, in this case the 64n, must be a floating point number or SKILL division will assume integer only division. Special processing might be needed in the target parameter value field to handle certain expressions. For example, `cdfParseFloatString("30n")` convert "30n" to 3e-8.

Finally, the order of parameters is important in some Pcells. The parameters are set in a bottom-up order. However, if the option `setTargetParameterTopDown = t`, then the parameters are set in a top-down order.

Conditional Equation Syntax

In general, the conditional expression syntax is:

```
calc [eq|Result] [condition1] [condition2] .... [conditionN]
```

The keywords above are fixed placeholders and should not be modified. `Result` may be a literal number or string, or an equation. `Result` may also be a single quoted string. Each condition has the format:

```
[leftSide|operator|rightSide]
```

Virtuoso Custom Design Migrate User Guide

Mapping File Syntax for Schematic Migration

where `leftSide` and `rightSide` may both be a literal number or string, or an equation, and operator can be `=`, `!=`, `<`, `>`, `>=`, or `<=`.

Target Library or Target Parameter Name	Target Cell or Target Parameter Description	Target Parameter Value	Notes	Other Parameters		
nfin	Number of Fins	calc [eq 2] [cc_fw = 320n] [cc_fingers < 4]				
nfin	Number of Fins	calc [eq 4] [cc_fw = 320n] [cc_fingers < 6] [cc_fingers >= 4]				
nfin	Number of Fins	calc [eq 6] [cc_fw = 320n] [cc_fingers < 10] [cc_fingers >= 6]				

Figure C-7 Condition Equation Syntax

The analysis of the condition expression begins at the top match parameter line (for one parameter like “nfin”, even if other parameters are evaluated bottom-up). If one of the conditions fails, then the analysis moves to the next match parameter line. If all of the conditions match on any one line in this order, then “nfin” is set to the result.

In the conditional expression example above, the top match parameter line has:

```
calc [eq|2] [cc_fw|=|320n] [cc_fingers|<|4]
```

If source parameter “fw” is 320n, and source parameter “fingers” is less than 4, then target parameter “nfin” is set to 2.

Otherwise, if source parameter “fw” is 320n, and source parameter “fingers” is less than 6 but greater than or equal to 4, then target parameter “nfin” is set to 4.

Otherwise, if source parameter “fw” is 320n, and source parameter “fingers” is less than 10 but greater than or equal to 6, then target parameter “nfin” is set to 6.

Otherwise, the target parameter “nfin” is set to the default Pcell value.

Note: In the `[eq|result]` portion of the conditional expression, the result may be a SKILL equation and reference other source parameters. For example: `[eq|cc_fingers*4]`

Other Parameters

The *Other Parameters* enables special, map specific options. For example, symbol offsets may be specified from the source symbol to the target symbol. Multiple “Other Parameters” may be specified for the same device map. The options must be separated by a space. Each separate option begins with a # character. Each option applies to that cell map (all instances of the target cell).

■ Instance Symbol Offsets

- ❑ `#symbolOffsetX=<floating point value>`
- ❑ `#symbolOffsetY=<floating point value>`

■ Pin rename from source to target

- ❑ `#pinRename=<source pin name 1>|<target pin name 1> [<source pin name 2>|<target pin name 2>]`

■ Symbol Rename

- ❑ `#symbolName=<target symbol name>`

■ Symbol Size Transform

- ❑ `#symbolSize=<magnification>`

■ Align Symbol by Target Pin Name

- ❑ `#keepPinLoc=<target pin name>`

■ Align Target Symbol center point to Source Symbol center point

- ❑ `#keepCenter=yes`

■ Shorts Avoidance Effort

- ❑ `#usePlacementEffort=yes`

■ Add wire stub and net name label

Adds a wire stub and net name label to unmapped pins on the target symbol. For example, if the target symbol has an extra `VPP` pin which should be connected to net `VDD`, `#useStub=VPP|VDD` can be used to add a wire joining instance pins `VPP` and `VDD`. The specified net can be an existing or non-existing net.

The created wire stub orients itself according to the position of the pin on the symbol and also according to the orientation of the instance. For example, horizontal wire stubs are created if the pins are on left and right sides of a symbol, and vertical stubs are created if the pins are on the top and bottom of the symbol.

Note: The pin should not be mapped using `#pinRename`.

Example:

```
#useStub=<pin name>[|<net/label name>][<pin name 2>[|<net/label name 2>]]
```

■ Move the Symbol Outside and Add Wire Stubs and Net Name Labels

- ❑ `#moveAndStub=<t|nil>`

Virtuoso Custom Design Migrate User Guide

Mapping File Syntax for Schematic Migration

■ Connect Pin to Pin

Connects additional and unmapped pins on the target symbol. It can use horizontal, vertical or diagonal wires to connect the pins depending on their position.

Note: The pin should not be mapped using `#pinRename`.

Example:

```
#connectPins=<target pin 1>|<existing pin 1>[;<target pin 2>|<existing pin 2>]
```

■ Connect a Target Pin to the Same Net of the Specified Source Pin

Syntax

```
#stubFromPinNet=<target pin 1>|<source pin 1>[;<target pin 2>|<source pin 2>]
```

Example 1

```
#pinRename=INP|IN2 #stubFromPinNet=SNS|SIDDQ
```

Target pin is SNS, source pin is SIDDQ

Notice that SIDDQ and SNS do not appear in `#pinRename`

Copy Cell Properties

Per cell map in the spreadsheet:

```
#copyProp=<t|nil>
```

If set to `t`, all properties on the source instance of this cell will be copied.

Requires `copyProperties = list("copyProp")`, see section, [Schematic Mapping Options](#).

Per cell map in the spreadsheet:

```
#copyProp=<source property name 1>|<target property name 1>|<display>[;<source property name 2>|<target property name 2>|<display>]
```

```
#copyProp=Grid|nil|t;w
```

```
#copyProp=Grid|rpGrid|t;l|length
```

Requires `copyProperties = list("copyProp")`

The source property name is required

The target property name is optional, if nil or unspecified, migration will use the source property name for the target property name

Virtuoso Custom Design Migrate User Guide

Mapping File Syntax for Schematic Migration

The display option is optional, if nil or unspecified, the migration will not display the property on the finished schematic. If set to t, this property will be display on-canvas

Per Cell Post Migration Trigger Function

```
#postMapTrigger=MYcustomFunction(<current instance object>)
```

The user may define a custom procedure that takes one argument. The argument is the current target instance object: for example:

```
procedure(MYcustomFunction(instObj)...
```

The custom procedure must be placed in the \$CDS_MVS_IMF_CM/SKILL directory on Virtuoso start-up.

Note: A per cell post migration trigger overrides post instance triggers defined by postMigInstTrigger.

Default Symbol Orientation Difference

```
#defSymOrient=<orientation>
```

Valid orientations are R90 R180 R270 MX MY MXR90 MYR90

Note: Square brackets, [], indicate an optional addition. Corner brackets, < >, indicate a parameter value.

Other Parameter Examples

The below example shows CSV saved from the Excel spreadsheet:

```
device,Framework_Examples_45_RebindLib1,SchMig_Cell1,,,←source library, source
cell, source view
Framework_Examples_ff_RebindLib1,SchMig_FF_Cell3,,, ←target library, target cell,
target view
#symbolOffsetY=2.5      ←offset to target instance placement
#pinRename=VDD|VDDQ;GATES|NMG ←terminal renaming
#symbolName=sym2       ←symbol renaming
#keepPinLoc=NMG,       ←move target instance to pin

device,VALE_Framework_Examples_45_RebindLib1,SchMig_Cell1,,,
VALE_Framework_Examples_ff_RebindLib1,SchMig_FF_Cell1_AddPin,,,
#moveAndStub=t         ←move target instance outside and stub and label it
#useStub=sense|SEN,    ←add short wire and label to added pin
```

Virtuoso Custom Design Migrate User Guide

Mapping File Syntax for Schematic Migration

Virtuoso Custom Design Migration Forms

[Select Libraries that have Cells to Map Form](#)

[Select a Cell to Inspect Form](#)

[Schematic Design Inventory Form](#)

[Schematic Mapping Settings Form](#)

[Virtuoso Schematic Mapping Form \(Part 1\)](#)

[Select the Migration to Finish Form \(Part 2\)](#)

[Schematic Migration Results Browser Form](#)

Select Libraries that have Cells to Map Form

Use this form to select the preferred library and extract the CDF parameters you need in order to create a CSV file for all the source devices that need to be mapped. It is accessed using the *Migrate – Schematic – Utilities – Create Source-side of Migration Map* command.

Field	Description
List of Libraries	<p>Lists the libraries that are available in the current design. You can select one or more libraries that contain the CDF parameters to be captured into the reuse file.</p> <p>Use <code>Shift</code> + click to select a range of (consecutive) libraries.</p> <p>Use <code>Ctrl</code> + click to select multiple libraries individually.</p>

Related Topics

[Source-side Migration Map Generator](#)

Select a Cell to Inspect Form

Use this form to review and delete instance properties carried over by copies of cellviews. It is accessed using the *Migrate – Schematic – Utilities – Inspect Schematic Instance Properties* command.

Field	Description
List of instance properties	Lists all instance properties in the current design. You can select an instance property and click OK to open the corresponding cellview.

Related Topics

[Inspect Schematic Instance Properties](#)

Schematic Design Inventory Form

Use this form to display information about the current design hierarchy, for example, a summary of the instances used and the parameters these instances have. It is accessed using the *Migrate – Schematic – Utilities – Schematic Design Inventory* command.

Related Topics

[Schematic Design Inventory](#)

Schematic Mapping Settings Form

Use this form to define the basic settings for schematic migration. This form can be accessed using the *Migrate – Schematic – Mapping – Settings* command.

Field	Description
<i>Configuration File</i>	Shows the path to the configuration file with the settings used for the migration.
<i>Running Directory</i>	Specifies the running directory (pre-selected).
<i>Mapping File</i>	Selects the mapping file to be used for the migration from a browser window.
<i>Technology Libraries</i>	This section lets you specify the libraries for the migration process.
<i>Source</i>	Defines the library from which the source schematic data must be captured.
<i>Target</i>	Defines the library to which the captured source schematic data is to be applied.

Virtuoso Schematic Mapping Form (Part 1)

Use this form to save data from the source schematic. It is accessed using the *Migrate – Schematic – Mapping – Save Source Schematic Data (Part 1)* command.

Field	Description
Source	This section lets you specify the source library, cell, and view for the migration.
<i>Library</i>	Lets you select the source library.
<i>Cell Name</i>	Lets you select the source cell.
<i>View Name</i>	Defines the source view name.
	<ul style="list-style-type: none"> ■ <i>Process Hierarchy</i>: Migrates the selected cellview and the underlying hierarchy. ■ <i>Process Library</i>: Migrates all cells in the source library. ■ <i>Process all views of each cell</i>: Migrates all views of each cell. ■ <i>Process all reference libraries</i>: Migrates all reference libraries.
Target	This section lets you specify the target library and cell for the migration.
<i>Library</i>	Lets you select the target library or create a new library. To create a new library, select Create New from the drop-down list. The <u>New Library</u> form opens.
<i>Cell Name</i>	Lets you select the target cell.
<i>View Name</i>	Lets you select the target view name (pre-selected from source).
<i>Name Options</i>	<p>A string field that can be used when <i>Use name options below</i> is selected for cell name.</p> <ul style="list-style-type: none"> ■ <i>Prefix</i>: The name in the <i>Name Options</i> field will be a prefix of the cell name. ■ <i>Replace</i>: The name in the <i>Name Options</i> field will replace the cell name. ■ <i>Suffix</i>: The name in the <i>Name Options</i> will be a suffix of the cell name.

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Design Migration Forms

Field	Description
<i>More Options</i>	This section lets you specify additional options for the migration.
<i>Mapping File</i>	Specifies the mapping file.
<i>Design Library Mapping</i>	Specifies the path to the file that contains mappings of layers, devices, and parameters between the source and the target.
<i>Exclude Library List</i>	Specifies a list of libraries to be ignored during migration.
<i>Exclude View List</i>	Specifies a list of views to be ignored during migration.

Related Topics

[Migration Part 1: Save Source Schematic Data](#)

Select the Migration to Finish Form (Part 2)

Use this form to create the target schematic based on the Save Source Schematic Data (Part 1) form. It is accessed using the *Migrate – Schematic – Mapping – Create Target Schematic (Part 2)* command.

Related Topics

[Migration Part 2: Creating Target Schematic](#)

Schematic Migration Results Browser Form

Use this form to open the Schematic Mapping Results Browser, which displays filterable summary and error information on the completed migration. Click *Open Selection(s)* to display the results in the Annotation Browser. The form is accessed using the *Migrate – Schematic – Mapping – Migration Results Browser* command.

Related Topics

[Migration Part 3: Viewing Results Browser](#)

Virtuoso Custom Design Migrate User Guide

Virtuoso Custom Design Migration Forms
