Command-Line IP Selector (CLIPS) User Guide

Product Version IC23.1
June 2023

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

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

Cadence is committed to using respectful language in our code and communications. We are also active in the removal and replacement of inappropriate language from existing content. This product documentation may however contain material that is no longer considered appropriate but still reflects long-standing industry terminology. Such content will be addressed at a time when the related software can be updated without end-user impact.

Contents

1

Introduction to Command-Line IP Selector (CLIPS)

In an advanced mixed-signal design flow, a mixed-signal block is commonly used as a part of the System on Chip (SoC) simulation with the Digital on Top (DoT) methodology. Spectre AMS Designer, which is a powerful tool used to simulate mixed-signal designs, uses a config or schematic view from the Virtuoso database to manage a complex binding configuration in a mixed-signal block. However, when this mixed-signal block or mixed-signal IP is reused in the DoT flow, manual export and reconfiguration of these blocks become challenging for a designer. Command-Line IP Selector (CLIPS) is a utility that simplifies the Virtuoso mixed-signal IP reuse.

CLIPS provides a bridge between Virtuoso, which is a UI-based, analog, and mixed-signal design environment, and other command-line, digital, and mixed-signal simulation tools and flows that use text-based tests. The design verification team can use CLIPS to verify the digital representation of an IP with their corresponding analog representation. CLIPS has several benefits for mixed-signal designers:

- Leverages an existing testbench setup
- Provides a powerful digital verification mechanism
- Eliminates the need to import large and complex digital designs in Virtuoso
- Provides an integrated design management system

Licensing Requirements

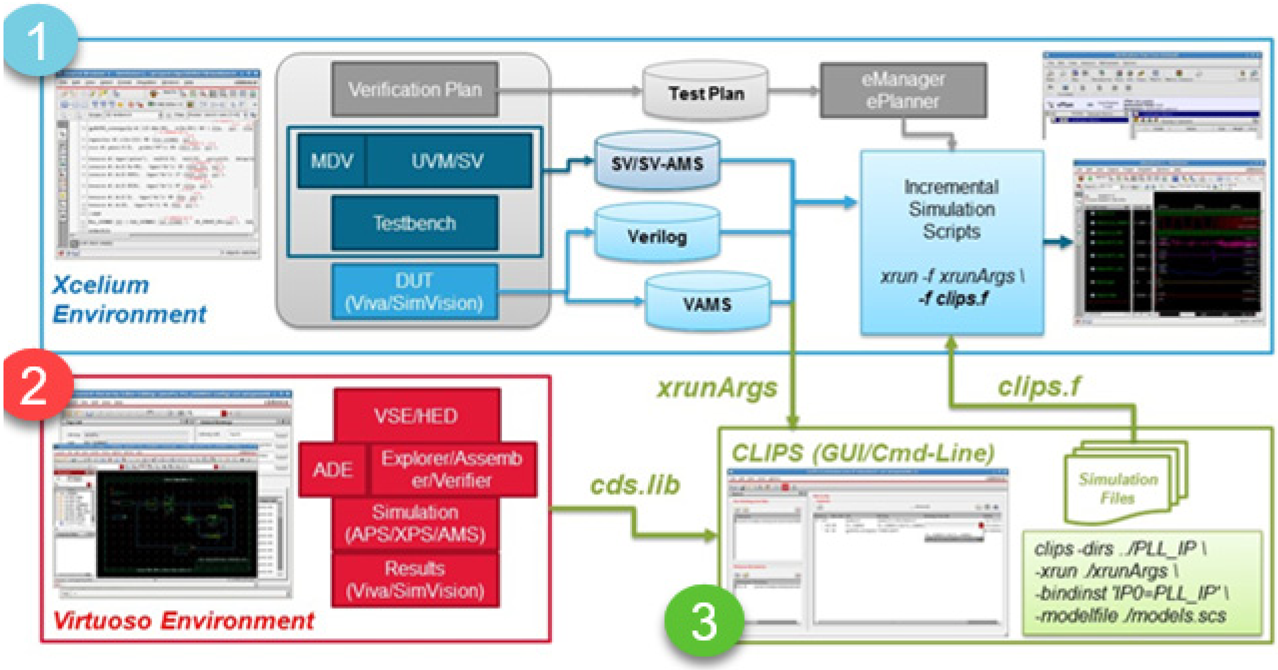
You require the 70000, the AMS_environment license, to run CLIPS.

Note: If you are using an ADE state for netlisting, you would also require a license of the ADE product in which that state was saved.

For more details, refer to the *Virtuoso Software Licensing and Configuration User Guide*

CLIPS Flow

The illustration given shows how CLIPS works with Virtuoso and Xcelium™ Logic Simulator:



```
xrun -f xrunArgs \
-f clips.f
```

```
clips -dirs ../PLL_IP \
-xrun ./xrunArgs \
-bindinst 'IP0=PLL_IP' \
-modelfile ./models.scs
```

1. In Xcelium, a high-level simulation and verification is run after the top-down flow sets up the SoC-level verification methodology by using Metric-Driven Verification (MDV) with Universal Verification Methodology (UVM) or SystemVerilog (SV).

2. In Virtuoso Analog Design Environment (ADE), a bottom-up mixed-signal IP is designed and verified.

3. CLIPS bridges the gap between the top-down flow of the Xcelium environment and the bottom-up flow of the Virtuoso environment. When you import the Xcelium simulation setup and the Virtuoso AMS IP configuration in CLIPS with the help of the `xrunArgs` files and the `cds.lib` files respectively, it automatically generates netlist and packages the IP config into an independent directory, and generates an incremental file, `clips.f`, on top of the existing `xrunArgs` file. When both these files, `xrunArgs` and `clips.f`, are added to the `xrun` command, the AMS IP automatically replaces its digital counterpart in the SoC simulation setup.

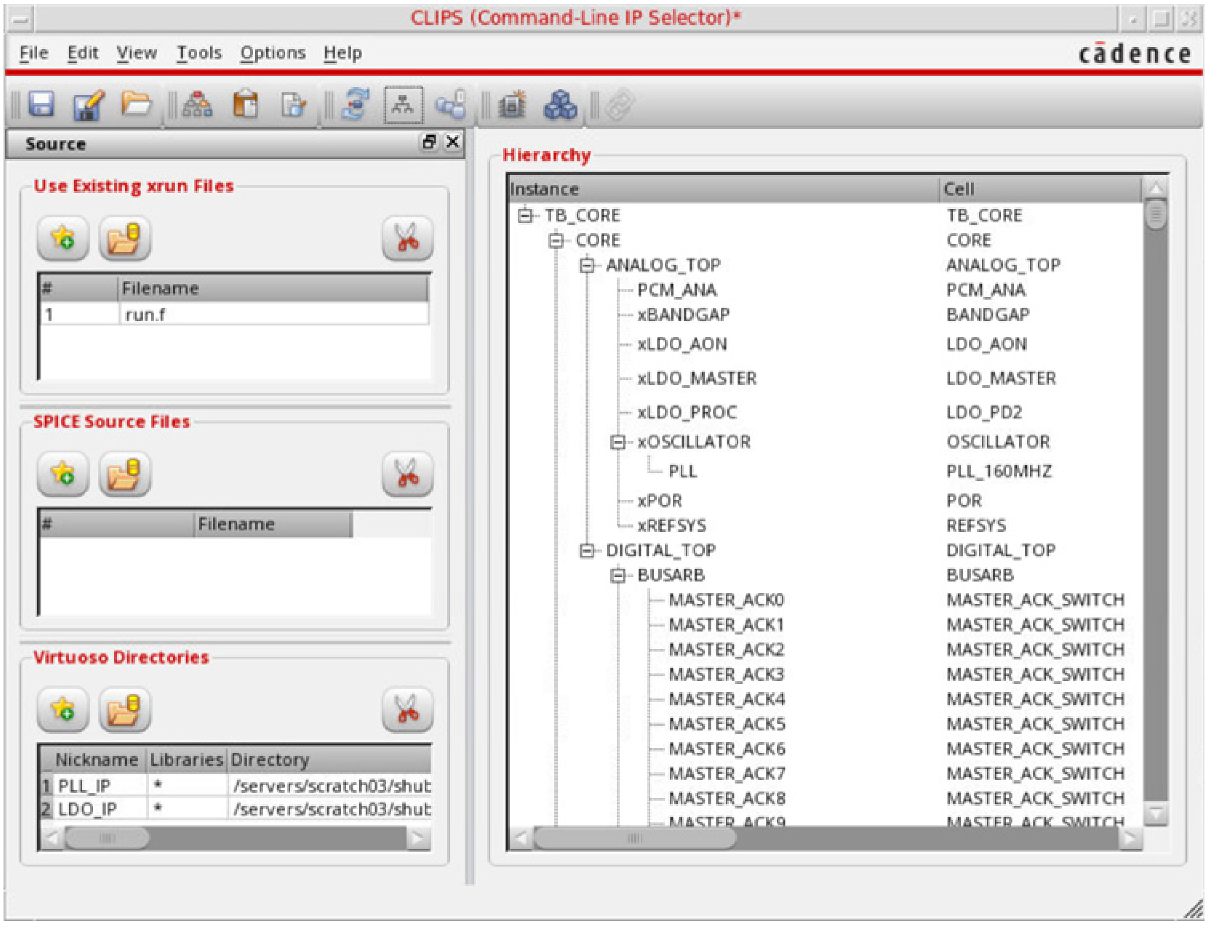
Modes for Launching CLIPS

A CLIPS session can be launched in the following modes:

- GUI mode: You can use the command `clips &` to launch CLIPS in GUI mode.

  Note: Ensure that the path for Xcelium 17.10 or above is set and added to the PATH environment variable.

  The CLIPS window is displayed.


CLIPS (Command-Line IP Selector)*
File Edit View Tools Options Help
cādence
Source
Hierarchy

Use Existing xrun Files

#	Filename
1	run.f

SPICE Source Files

#	Filename

Virtuoso Directories

	Nickname	Libraries	Directory
1	PLL_IP	*	/servers/scratch03/shub
2	LDO_IP	*	/servers/scratch03/shub

Instance	Cell
TB_CORE	TB_CORE
CORE	CORE
ANALOG_TOP	ANALOG_TOP
PCM_ANA	PCM_ANA
xBANDGAP	BANDGAP
xLDO_AON	LDO_AON
xLDO_MASTER	LDO_MASTER
xLDO_PROC	LDO_PD2
xOSCILLATOR	OSCILLATOR
PLL	PLL_160MHZ
xPOR	POR
xREFSYS	REFSYS
DIGITAL_TOP	DIGITAL_TOP
BUSARB	BUSARB
MASTER_ACK0	MASTER_ACK_SWITCH
MASTER_ACK1	MASTER_ACK_SWITCH
MASTER_ACK2	MASTER_ACK_SWITCH
MASTER_ACK3	MASTER_ACK_SWITCH
MASTER_ACK4	MASTER_ACK_SWITCH
MASTER_ACK5	MASTER_ACK_SWITCH
MASTER_ACK6	MASTER_ACK_SWITCH
MASTER_ACK7	MASTER_ACK_SWITCH
MASTER_ACK8	MASTER_ACK_SWITCH
MASTER_ACK9	MASTER_ACK_SWITCH

- Command-line mode: You can use the command `clips <optional command-line arguments>` to launch CLIPS in command-line mode.

  A few examples are given below.

```
clips -load myTextPll.clips
```

```
clips -dirs ../PLL_IP -xrun xrunArgs &
```

```
clips -load myTextPll.clips -batch -log myBatchRun.log
```

```
clips -dirs ../PLL_IP -xrun xrunArgs -switch "inst=testbench.IP0.I11.I14
config=PLL_IP::amsPLL/dffnr_2x_hv/config ade=PLL_IP::amsPLL/pll_top/
maestro:pll_top:1" -switch "inst=testbench.IP0.I11.I13 config=PLL_IP::amsPLL/
nor2_2x_nv/config ade=PLL_IP::amsPLL/pll_top/ams_state" -batch
```

```
clips -export "config=amsPLL/dffnr_2x_hv/config ade=PLL_IP::amsPLL/pll_top/
ams_state outdir=clips_export_dffnr cdsdir=../test/IP_dir" -batch
```

The command-line arguments can be saved in a text file and used while launching CLIPS with the help of the following command:

```
clips -f <command-line arguments file>
```

Related Topics:

CLIPS GUI

CLIPS Command-line Arguments

CLIPS GUI

The CLIPS GUI comprises the following components.

- Menu bar
- Toolbar

Menu bar

Menu	Description
File	
New	Creates a new CLIPS setup.
Save/Save As	Saves the current setup.
Load	Loads an existing CLIPS setup.
Save Command Line Options	Saves the currently set command-line arguments in a file.
Exit	Exits the tool.
Edit	
Clear Log	Clears the status log.
Netlist Output Directory	Specifies the directory to be used as the netlist output directory.
View	

Menu	Description
Show / Hide Source	Sets your preferred general options to show or hide xrun files and Virtuoso setup directories. You can also show or hide the settings and options pane, the status log pane, and the advanced hierarchy tree.
Show / Hide Settings	Shows the output directory where the netlisted Virtuoso cellviews and the related xrun files created or exported by CLIPS are saved.
Show / Hide Logs	Shows the netlisted Virtuoso cellviews.
Tools	
Netlist	Creates a netlist for Virtuoso hierarchy bindings in the output directory.
Check Bindings	Checks the binding overrides by elaborating the original xrun files, the netlisted Virtuoso cellview bindings, and the CLIPS-generated xrun files.
Add xrun -f files	Adds xrun files.
Update xrun -f files	Updates xrun files.
Remove xrun -f files	Removes xrun files.
Add Virtuoso	Adds a Virtuoso session by specifying a directory containing a `cds.lib` file.
Setup Model Files	Opens the Setup Model Files form where you can specify the model files to be used with the CLIPS setup.
Remove All Binding Overrides	Removes the manually added binding overrides from all the instances in the hierarchy.
Refresh Hierarchy	Refreshes the hierarchy to elaborate any updated xrun files.
Show Switchable Binding Overrides	Expands the hierarchy tree to show the instances that have switchable binding overrides.
Options	
Copy Model Files	Controls whether the model files are copied or linked to the netlist output directory.
Cache View Names	Caches the available views in the library cellviews.
Hide Hierarchy Instance	Hides the *Hierarchy instance* column in the hierarchy tree.

Menu	Description
Hide Binding Override	Hides the *Binding Override* column in the hierarchy tree.
Hide Notes	Hides the *Notes* column in the hierarchy tree.
Show Details in Log	Shows the details of file parsing and hierarchy processing in the log file.
Show Settings Groups	Enables the group setting panel. If unchecked, only global setting panel is visible.
Check Bindings After Netlisting	Automatically runs the elaboration or binding check after creating the netlist.
Enable Searching Mismatched DUT Name	Finds all the Virtuoso config or schematic views whose schematic DUT name matches the module name of digital text file.

Toolbar

Icon	Command name	Description
	Save	Saves the current CLIPS setup at the current location and with the existing name.
	Save As	Saves the current CLIPS setup to a file with suffix `.clips`, with a specific name and at a specific location. This file can be loaded later.
	Load	Loads the `.clips` file, which contains a state or settings.
	Show/hide source xrun, text files/ directories and Virtuoso setup	Shows or hides the source section for including the xrun files and Virtuoso directories.
	Show/hide Settings and Options	Shows or hides the Settings and Options section.
	Show/hide Status log	Shows or hides the Status log section.

Icon	Command name	Description
	--	Elaborates the xrun files. For any IP switch and binding override changes, it updates and descends through complete instance hierarchy and bindings.
	--	Expands the hierarchy tree to show which instances have switchable binding overrides. Trees with no available binding overrides are collapsed.
	--	Shows the advanced hierarchy tree control panel on top of the hierarchy window. You can search instance or cell or binding in the hierarchy and control expand level.
	--	Creates the netlist of Virtuoso hierarchy bindings in the output directory.
	--	Opens the default or user-defined output directory to view the netlisted Virtuoso cellviews and the related xrun files created/exported by CLIPS.
	--	Checks the binding overrides by elaborating the original xrun files, the netlisted Virtuoso cellview bindings, and the CLIPS-generated xrun files.

CLIPS Command-line Arguments

Depending on your requirements, you can use one or more of the following optional command-line arguments:

Argument	Description
`-batch`	Runs (netlists) a loaded CLIPS file and exits with status (`0`==pass, `1`==fail).
`–cdsPre` *`value`*	Sets the full path to a UNIX script file to set up Virtuoso-specific environment variables. This C Shell script is sourced before Virtuoso sessions are started. Enter `setenv CLIPS_VIRTUOSO_BOURNE_SHELL` to use the Bourne shell format. **Note:** This is the command-line equivalent of the field *Virtuoso Setup Script*.

Argument	Description
`–checkbind` *`value`*	Automatically runs elaboration/binding check after netlisting, when CLIPS is run in batch mode. Default values are `yes` or `no`.
`clips -help`	Displays the help information for CLIPS.
`–continue`	Continues after load/setup error, if not in batch. Default behavior is to exit, which does not allow any changes.
`–depth` *`depth`*	Sets the depth of hierarchy to show. `0` means all levels.
`–details` *`value`*	Shows parsing and hierarchy processing in the log area / file. Default values are `yes` or `no`.
`–dirs` *`dirs`*	Loads Virtuoso directories: `<dir>[:<nickname>][,<dir>[:<nickname>]]...`

`-export`	Exports an IP block from a remote Virtuoso config or schematic view, without top-level SoC design access to a package directory. To use this argument for batch mode, specify the following commands (for IP providers).
	`clips -batch`
	`-export "config=<lib/cell/view>`
	`ade=<stateLib/stateCell/stateView[/testName]>`
	`outdir=<ip_export_dir>`
	`[cdsdir=<dir where cds.lib exists>]"`

`–export` *`value`*	Exports an IP remotely to assemble with a test bench in AMS UNL. It supports both, ADE and Maestro states. For multiple instances, add an option and value for each instance.
`–hideCell` *`value`*	Shows the list of cell names for which instances will be hidden in the hierarchy. Its default value is the value of the environment variable $`CLIPS_HIDE_CELLS`.
`–load` *`file`*	Loads the setup from previously saved CLIPS file.
`-local`	Automatically connects to Virtuoso using a local `cds.lib` when started without loading.
`–log` *`file`*	Sets the status log file. The default value is `CLIPS.log`. Set as `no` to disable log write.
`–modelfile` *`value`*	Used for netlisting configs. Its syntax: `<model1[:model2(sectionName)]>`. Its default value is the value in the environment variable `$CLIPS_MODELFILE`.
`-netlist`	Runs (netlists) a loaded CLIPS file on startup before interactive use.

Argument	Description
`–output` *`value`*	Netlists and exports the hierarchy to a new `CLIPSOUTPUT` (or -subDir <name>) under this directory (this directory must exist). It defaults to './'
`–postrun`	Enables running of any batch, post netlisting run commands.
`–postRunFail` *`value`*	Specifies a UNIX command to run after running netlisting, when in batch mode. It is used if netlisting has failed. Example: `sendFailMail projectLeaders <logfile>`
`–postRunPass` *`value`*	Specifies a UNIX command to run after running netlisting when in batch mode, if -`postrun` is set. It is used only if netlisting has passed. Example: `xrun -f xrunArgs -f ./CLIPS_OUTPUT/clips.f`
`–postRun` *`value`*	Specifies a UNIX command to always run (pass or fail) after running netlisting, when in batch mode, if -`postrun` is set. Example: `mv ./CLIPS_OUTPUT ~/simFiles`
`–preRun` *`value`*	Specifies a UNIX command to run before running netlisting, when in batch mode, if -`postrun` is set. If it returns a non-zero value, then netlisting does not run.
`–savebatch` *`file`*	When in batch mode, saves the setup to `<file>` before running netlisting. It includes all command-line changes.
`–subDir` *`value`*	Creates subdirectory under `-output <directory>` during netlisting. We can include <date> or `$envvarnam`, which will be processed when the directory is created.

`-switch`	Imports multiple packages into an SoC top-level design to assemble them together.To use this argument for batch mode, specify the following commands (for IP users).
	`clips -batch`
	`-xrun <top soc xrunArgs file>\`
	`-output <output directory after import>\`
	`-switch "inst=<inst hierarchy in top design> import=<ip1_export_dir>"\`
	`-switch "inst=<inst hierarchy in top design> import=<ip2_export_dir>"`

`–switch` *`value`*	Switches the settings of specified instances to an HED configuration. For multiple instances, add an option and value for each instance.

Argument	Description
`–xrun` *`file,...`*	Populates the hierarchy from one or more existing `xrun -f` files. Runs and analyzes the created `.pak` file.

Saving a CLIPS Setup

You can save the current CLIPS setup when using the GUI mode. This setup can then be loaded

1. Select *File – Save As*.

   The CLIPS Save Filename form appears.

2. Specify a name for the CLIPS setup and click *OK*.

   The setup is saved in a `.clips` file.

Loading a CLIPS Setup

To load an existing CLIPS setup in the GUI mode:

1. Select *File – Load*.

   The CLIPS Load Filename form appears.

2. Select the `.clips` file from which you want to load the setup, and click *OK*.

   Alternatively, to load an existing CLIPS setup in the command-line mode, run the following command:

   ```
   clips -load <xxx.clips file>
   ```

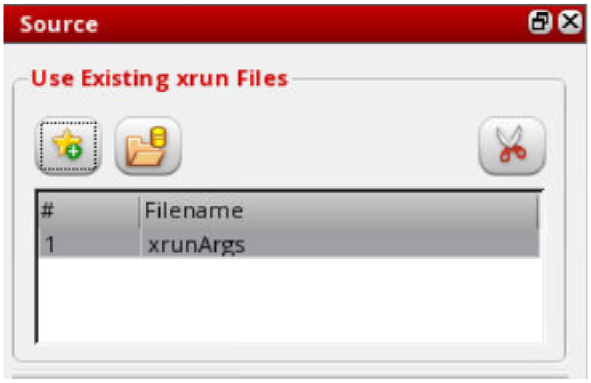
2

Working with CLIPS

This section includes the following topics that will help you get started with CLIPS:

xrun Files Management

CLIPS internally executes the `xrun` command to run simulations using Xcelium. You can specify the design files, input files, and the command-line options in the argument files that are used by the `xrun` command. You can manage these xrun argument files in the *Use Existing xrun Files* section of the Source assistant window.



#	Filename
1	xrunArgs

When you add or remove an `xrun` script file, CLIPS prompts you to update the hierarchy. Click *Update* on the toolbar to update the hierarchy. An elaboration runs in the background, if needed, to generate the latest hierarchy structure of the SoC simulation setup. The *Update* command also checks the time stamp of the existing `xrunArgs` files and re-runs the elaboration.

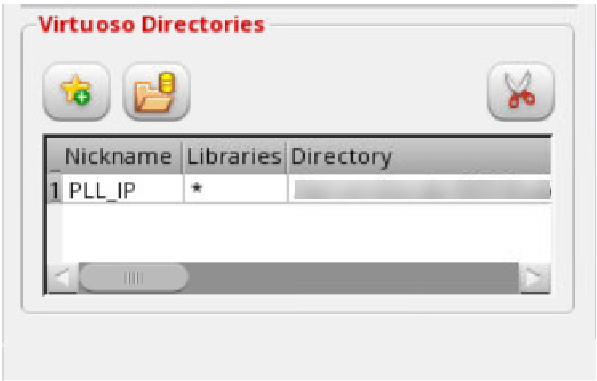
Related Topic

Source Assistant

Virtuoso Directories Management

If you have used Virtuoso to define analog or mixed-signal IP blocks corresponding to the digital blocks in your design, you can override the digital blocks with the blocks from Virtuoso. These overridden blocks are then used to created netlists that contain a mixed-signal representation of the complete hierarchy.

The *Virtuoso Directories* section in the CLIPS window is used to specify the location of the directories that contain the Virtuoso library files, `cds.lib`. These `cds.lib` files contain the paths to the libraries where design files are saved. When you specify a Virtuoso directory, CLIPS uses a nickname in the *`<Virtuoso directory nickname>:<library>/<cell>/<view>`* format to identify that directory.



	Nickname	Libraries	Directory
1	PLL_IP	*	

Related Topic

Source Assistant

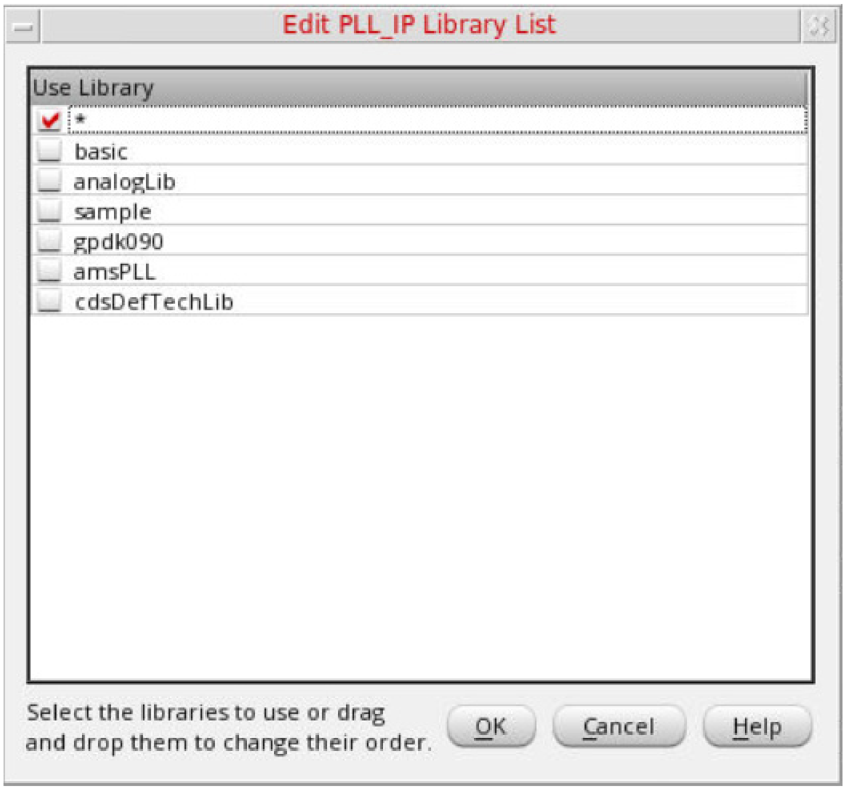
Searching a Library

CLIPS searches the specified Virtuoso directories to look for the Virtuoso config files that can override the corresponding digital block in your design. By default, all the directories specified in the *Virtuoso Directories* section can be searched for config files. This is indicated by the `*` symbol shown in the *Libraries* column of this section. However, you can choose the libraries to be searched while finding the schematic or config files that can be overridden, that is, switchable IP configurations. Other Virtuoso directories are excluded from the search. This helps improve the searching and matching performance.

To select the libraries to be searched, perform the following steps:

1. Double-click the *Libraries* column in the Virtuoso Directories section.

The Edit <Library_name> Library List form appears.



Edit PLL_IP Library List

Use Library

- ☑ *
- □ basic
- □ analogLib
- □ sample
- □ gpdk090
- □ amsPLL
- □ cdsDefTechLib

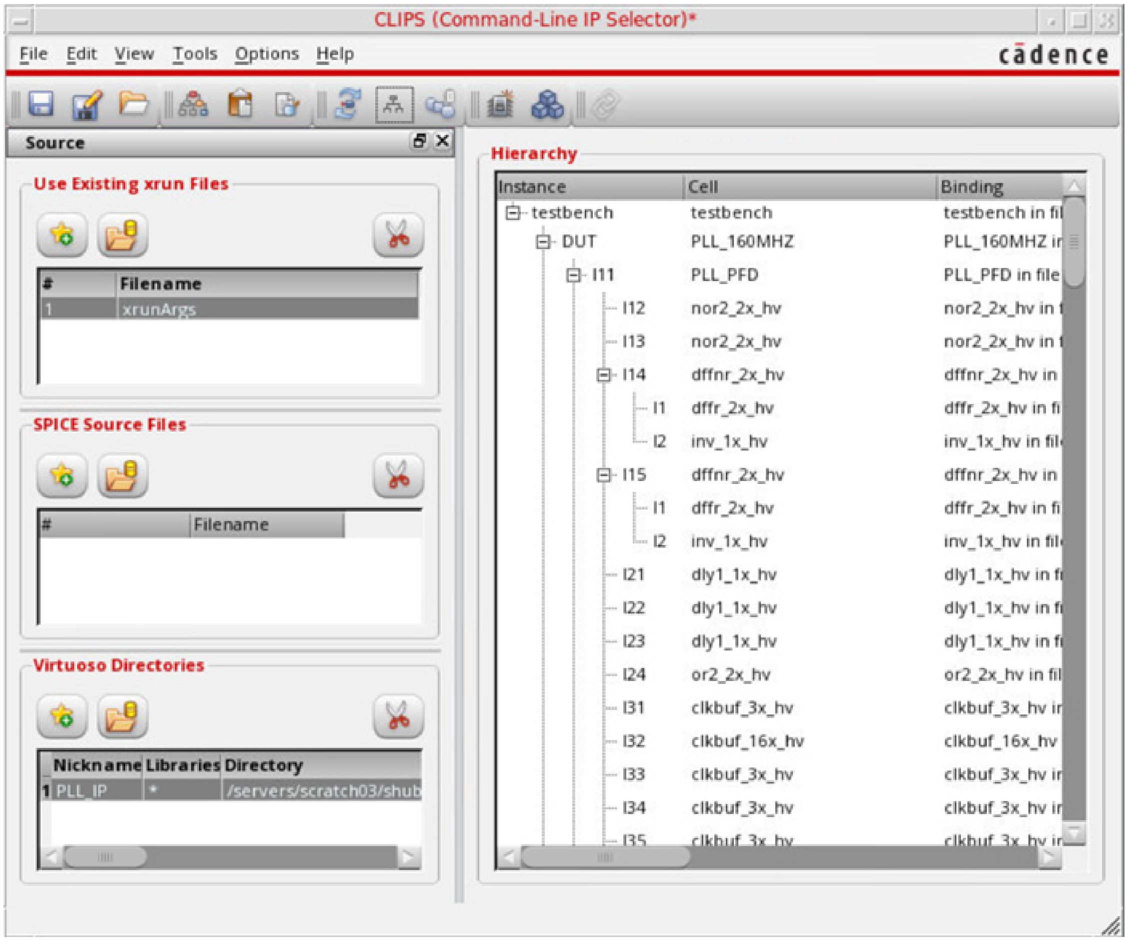
Select the libraries to use or drag and drop them to change their order.

2. Select the check box next to the libraries you want to use in search.

   Alternatively, you can drag and drop the library names in this list to specify the search order.

3. Click *OK* to close the form.

The `xrunArgs` files are elaborated and the digital hierarchy is represented in a tree in the *Hierarchy* section.



#	Filename
1	xrunArgs

#	Filename

	Nickname	Libraries	Directory
1	PLL_IP	*	/servers/scratch03/shub

Instance	Cell	Binding
testbench	testbench	testbench in fil
DUT	PLL_160MHZ	PLL_160MHZ ir
I11	PLL_PFD	PLL_PFD in file
I12	nor2_2x_hv	nor2_2x_hv in f
I13	nor2_2x_hv	nor2_2x_hv in f
I14	dffnr_2x_hv	dffnr_2x_hv in
I1	dffr_2x_hv	dffr_2x_hv in fi
I2	inv_1x_hv	inv_1x_hv in fil
I15	dffnr_2x_hv	dffnr_2x_hv in
I1	dffr_2x_hv	dffr_2x_hv in fi
I2	inv_1x_hv	inv_1x_hv in fil
I21	dly1_1x_hv	dly1_1x_hv in fi
I22	dly1_1x_hv	dly1_1x_hv in fi
I23	dly1_1x_hv	dly1_1x_hv in fi
I24	or2_2x_hv	or2_2x_hv in fil
I31	clkbuf_3x_hv	clkbuf_3x_hv ir
I32	clkbuf_16x_hv	clkbuf_16x_hv
I33	clkbuf_3x_hv	clkbuf_3x_hv ir
I34	clkbuf_3x_hv	clkbuf_3x_hv ir
I35	clkbuf_3x_hv	clkbuf_3x_hv ir

Hierarchy Management

In the *Hierarchy* section, you can switch or override the cellview bindings for selected blocks or cells to use config or schematic views from Virtuoso instead of the text cellviews (current bindings). If you switch the binding for a cellview to use a config or schematic view, you also need to either associate that cell with an ADE state, which provides the information required for netlisting, or directly add model files.

Note: For netlisting, it is not necessary to have model files for schematic view. It is generally required while simulating.

Related Topics

Hierarchy Controls

Overriding Instance Bindings

Associating an Instance with an ADE State

Hierarchy Controls

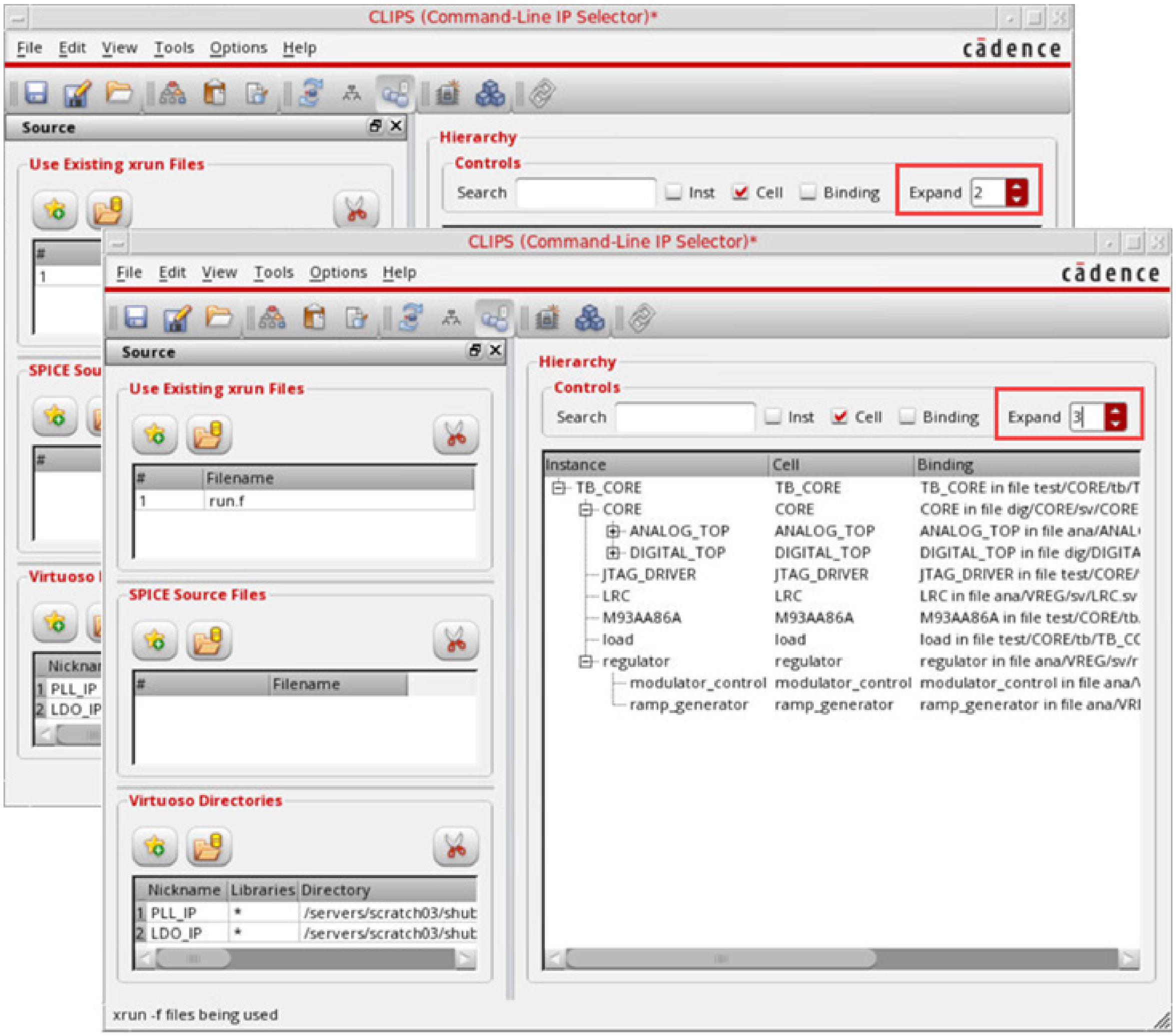
For a large design hierarchy, you might need to scroll to look for a specific cell or view. Instead, you can use the controls provided in the *Hierarchy* section to view the design components of interest.

To access the controls, select *View – Hierarchy Controls*.

The *Controls* group box is displayed in the *Hierarchy* section. You can use the controls in this box to do the following:

- Search for a specific instance, or cells or bindings. For this, you can type in the search box.
- Limit the view to display only instance or cells or bindings. For this, you can the select the corresponding check box.

- Expand the instances in the hierarchy to higher levels to get a more detailed tree structure. For this you can select the value from the drop-down list box, as shown in the figure below.



Expand 2

Expand 3

#	Filename
1	run.f

Nickname	Libraries	Directory
1 PLL_IP	*	/servers/scratch03/shub
2 LDO_IP	*	/servers/scratch03/shub

Instance	Cell	Binding
TB_CORE	TB_CORE	TB_CORE in file test/CORE/tb/T
CORE	CORE	CORE in file dig/CORE/sv/CORE
ANALOG_TOP	ANALOG_TOP	ANALOG_TOP in file ana/ANAL
DIGITAL_TOP	DIGITAL_TOP	DIGITAL_TOP in file dig/DIGITA
JTAG_DRIVER	JTAG_DRIVER	JTAG_DRIVER in file test/CORE/
LRC	LRC	LRC in file ana/VREG/sv/LRC.sv
M93AA86A	M93AA86A	M93AA86A in file test/CORE/tb.
load	load	load in file test/CORE/tb/TB_CC
regulator	regulator	regulator in file ana/VREG/sv/r
modulator_control	modulator_control	modulator_control in file ana/\
ramp_generator	ramp_generator	ramp_generator in file ana/VRI

xrun -f files being used

Overriding Instance Bindings

You can override a cellview binding to use a Virtuoso config or schematic view instead of a text cellview. The drop-down list in the *Binding Override* column shows the name of the corresponding library, cell, and config or schematic view.

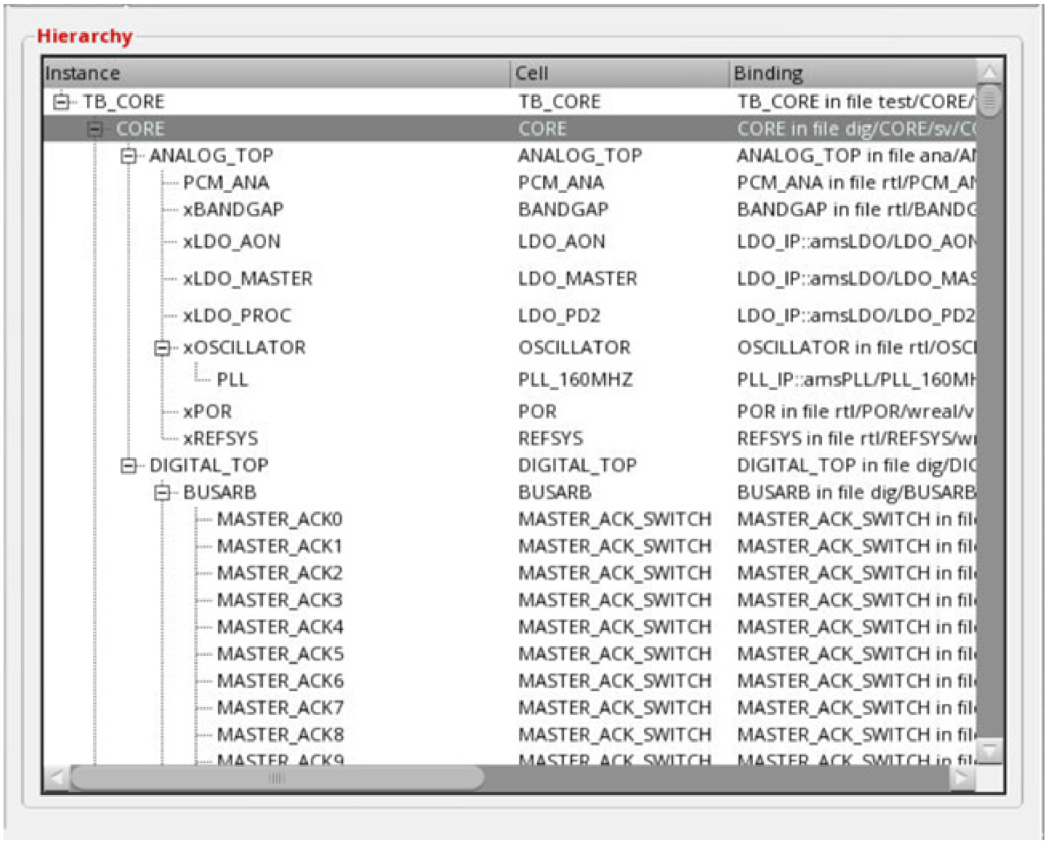
The Virtuoso cellviews for which either of the following conditions is satisfied are displayed in the *Binding Override* column:

- Aconfig view is available and the design sub-top (schematic or text) cell name is the same as the cell or block name
- Aschematic view is available and the cell name is the same as the cell or block name

To switch instance bindings:

1. Add xrun files in the *Use Existing xrun Files* section.
2. Add Virtuoso directories in the *Virtuoso Directories* section.
3. Click *Update* on the toolbar to update the hierarchy after adding all the `xrun` scripts and Virtuoso directories.

CLIPS elaborates the design to identify the Virtuoso cellviews available for the cells listed in the *Cell* column.

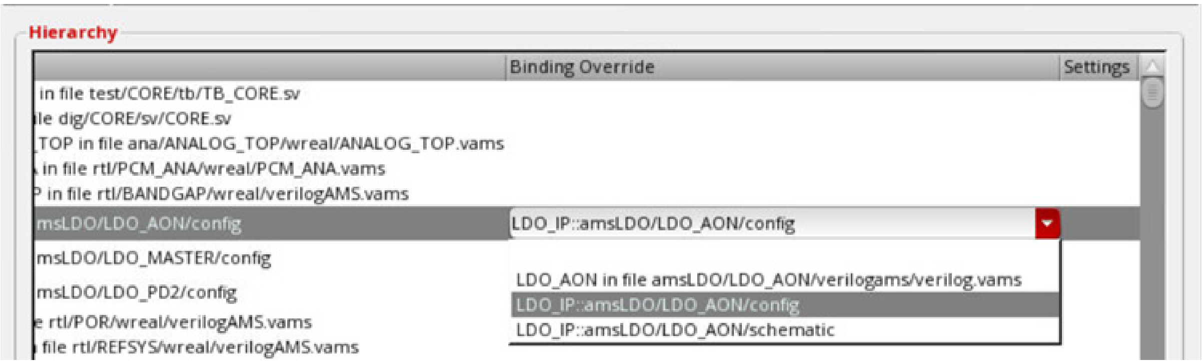


Hierarchy

Instance	Cell	Binding
TB_CORE	TB_CORE	TB_CORE in file test/CORE/
CORE	CORE	CORE in file dig/CORE/sv/C(
ANALOG_TOP	ANALOG_TOP	ANALOG_TOP in file ana/Al
PCM_ANA	PCM_ANA	PCM_ANA in file rtl/PCM_AI
xBANDGAP	BANDGAP	BANDGAP in file rtl/BANDC
xLDO_AON	LDO_AON	LDO_IP::amsLDO/LDO_AON
xLDO_MASTER	LDO_MASTER	LDO_IP::amsLDO/LDO_MAS
xLDO_PROC	LDO_PD2	LDO_IP::amsLDO/LDO_PD2
xOSCILLATOR	OSCILLATOR	OSCILLATOR in file rtl/OSCI
PLL	PLL_160MHZ	PLL_IP::amsPLL/PLL_160MI
xPOR	POR	POR in file rtl/POR/wreal/v
xREFSYS	REFSYS	REFSYS in file rtl/REFSYS/wı
DIGITAL_TOP	DIGITAL_TOP	DIGITAL_TOP in file dig/DIC
BUSARB	BUSARB	BUSARB in file dig/BUSARB
MASTER_ACK0	MASTER_ACK_SWITCH	MASTER_ACK_SWITCH in fil
MASTER_ACK1	MASTER_ACK_SWITCH	MASTER_ACK_SWITCH in fil
MASTER_ACK2	MASTER_ACK_SWITCH	MASTER_ACK_SWITCH in fil
MASTER_ACK3	MASTER_ACK_SWITCH	MASTER_ACK_SWITCH in fil
MASTER_ACK4	MASTER_ACK_SWITCH	MASTER_ACK_SWITCH in fil
MASTER_ACK5	MASTER_ACK_SWITCH	MASTER_ACK_SWITCH in fil
MASTER_ACK6	MASTER_ACK_SWITCH	MASTER_ACK_SWITCH in fil
MASTER_ACK7	MASTER_ACK_SWITCH	MASTER_ACK_SWITCH in fil
MASTER_ACK8	MASTER_ACK_SWITCH	MASTER_ACK_SWITCH in fil
MASTER_ACK9	MASTER_ACK_SWITCH	MASTER_ACK_SWITCH in fil

Note: You can find all the Virtuoso config or schematic views whose schematic DUT name matches the module name of digital text file by selecting *Enable Searching Mismatched DUT Name* in the *Options* menu.

4. For each cell for which you need to override the cellview binding, select the config or schematic cellview name from the drop-down list in the *Binding Override* column.



Hierarchy

	Binding Override	Settings
in file test/CORE/tb/TB_CORE.sv		
ile dig/CORE/sv/CORE.sv		
TOP in file ana/ANALOG_TOP/wreal/ANALOG_TOP.vams		
in file rtl/PCM_ANA/wreal/PCM_ANA.vams		
P in file rtl/BANDGAP/wreal/verilogAMS.vams		
msLDO/LDO_AON/config	LDO_IP::amsLDO/LDO_AON/config	
msLDO/LDO_MASTER/config		
msLDO/LDO_PD2/config		
e rtl/POR/wreal/verilogAMS.vams		
file rtl/REFSYS/wreal/verilogAMS.vams		

LDO_AON in file amsLDO/LDO_AON/verilogams/verilog.vams
LDO_IP::amsLDO/LDO_AON/config
LDO_IP::amsLDO/LDO_AON/schematic

Note: You can specify a testbench to display the design hierarchy. From the hierarchy, you can switch any instance with available binding override choices to an HED configuration. Then, AMS-UNL is called to export the HED configuration as an IP and assemble the IP together with the testbench.

Related Topics

Associating an Instance with an ADE State.

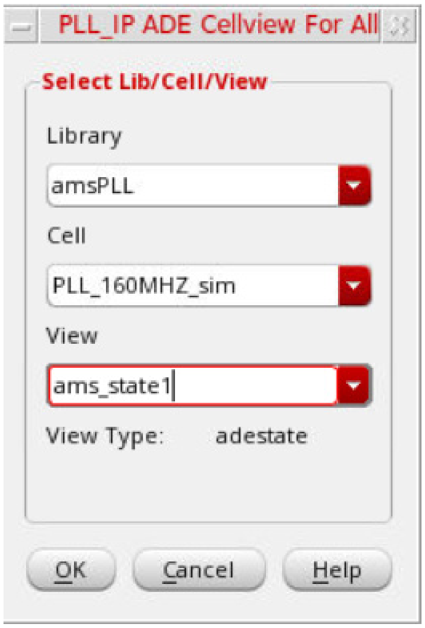
Adding a Model File.

Associating an Instance with an ADE State

If you switch the original binding of an instance with a Virtuoso config or schematic view, you need to associate the instance with an ADE state for providing the information required for the netlisting. However, in case the ADE state provides only model files for netlisting and simulation, you can directly add the model files in CLIPS.

To associate an instance with an ADE state:

1. Right-click the row of the instance for which you have overridden the binding and choose *Add/Edit ADE Cellview.*



PLL_IP ADE Cellview For All

Select Lib/Cell/View

Library: amsPLL

Cell: PLL_160MHZ_sim

View: ams_state1

View Type: adestate

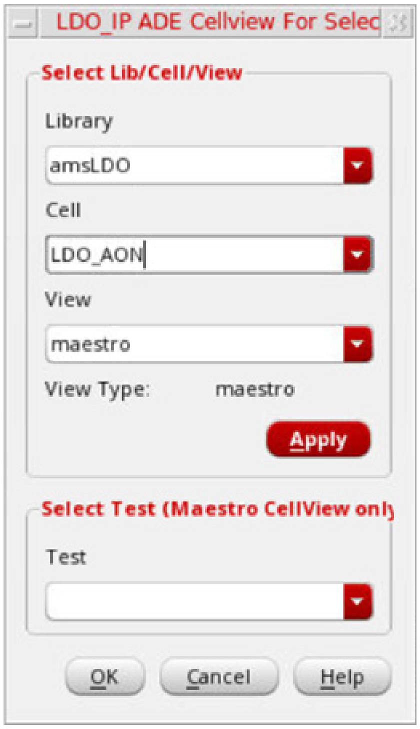
Note: When an ADE state is added to an instance in the digital hierarchy, a Virtuoso nickname is added as a prefix to the ADE cellview lib/cell/view in the settings column for the instances with which it is associated.

2. In the *View* drop-down list, select the name of the ADE state to be associated with the instance and click *OK*.
3. Click the *Update* command on the toolbar to update the hierarchy.

Associating an Instance with a maestro Cellview

You can also attach the maestro cellview state in CLIPS. To do this,

1. Right-click an instance in the *Hierarchy* section and select *Add/Edit Maestro Cellview*.



LDO_IP ADE Cellview For Selec

Select Lib/Cell/View

Library: amsLDO

Cell: LDO_AON

View: maestro

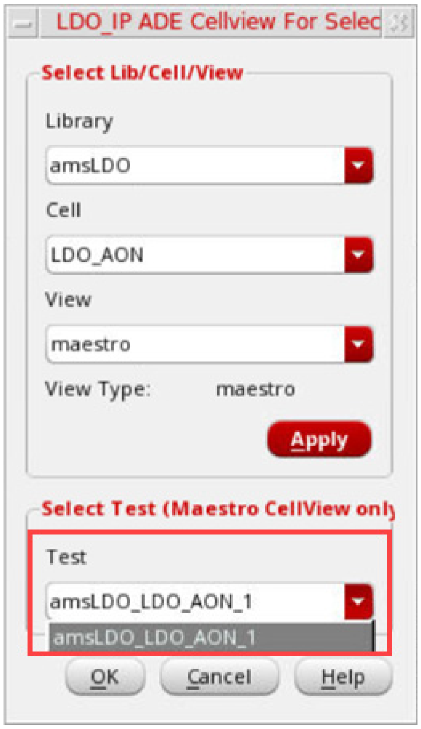
View Type: maestro

Select Test (Maestro CellView only

Test:

2. In the *View* drop-down list, select the name of the maestro state to be associated with the instance and click *Apply*.

3. In the *Test* drop-down list select the name of the test associated with the specified maestro state and click *OK*.


LDO_IP ADE Cellview For Selec
Select Lib/Cell/View
Apply
Select Test (Maestro CellView only
OK
Cancel
Help

Library: amsLDO

Cell: LDO_AON

View: maestro

View Type: maestro

Test: amsLDO_LDO_AON_1

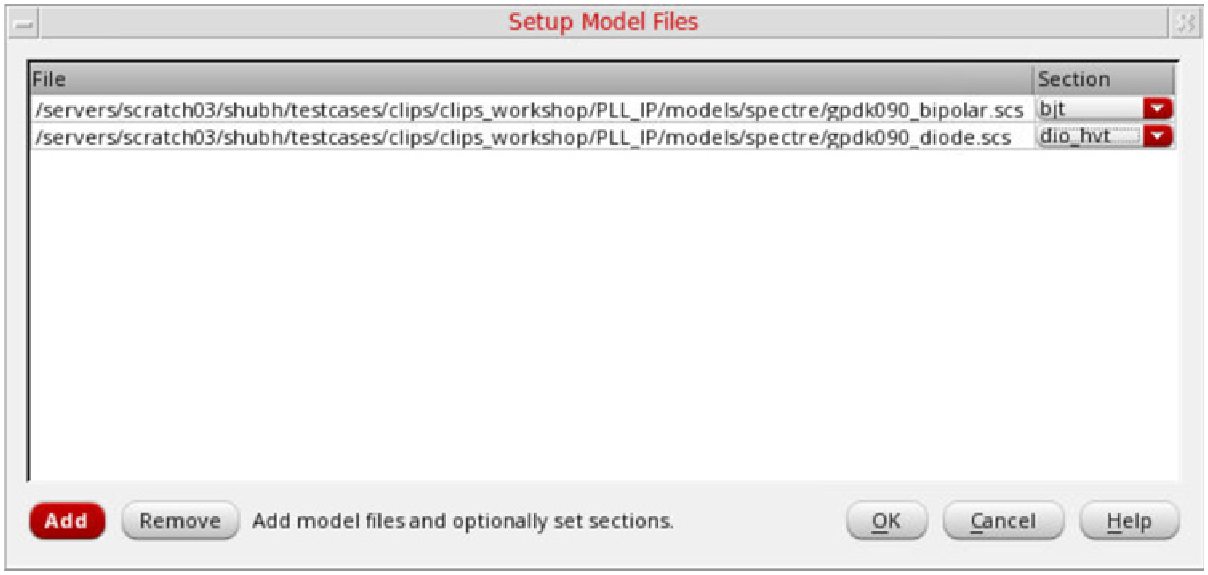
amsLDO_LDO_AON_1

4. Click the *Update* command on the toolbar to update the hierarchy.

Adding a Model File

To add a model file, perform the following steps.

1. Choose *Tools – Setup Model Files*. The Add/Edit Model Files form appears.

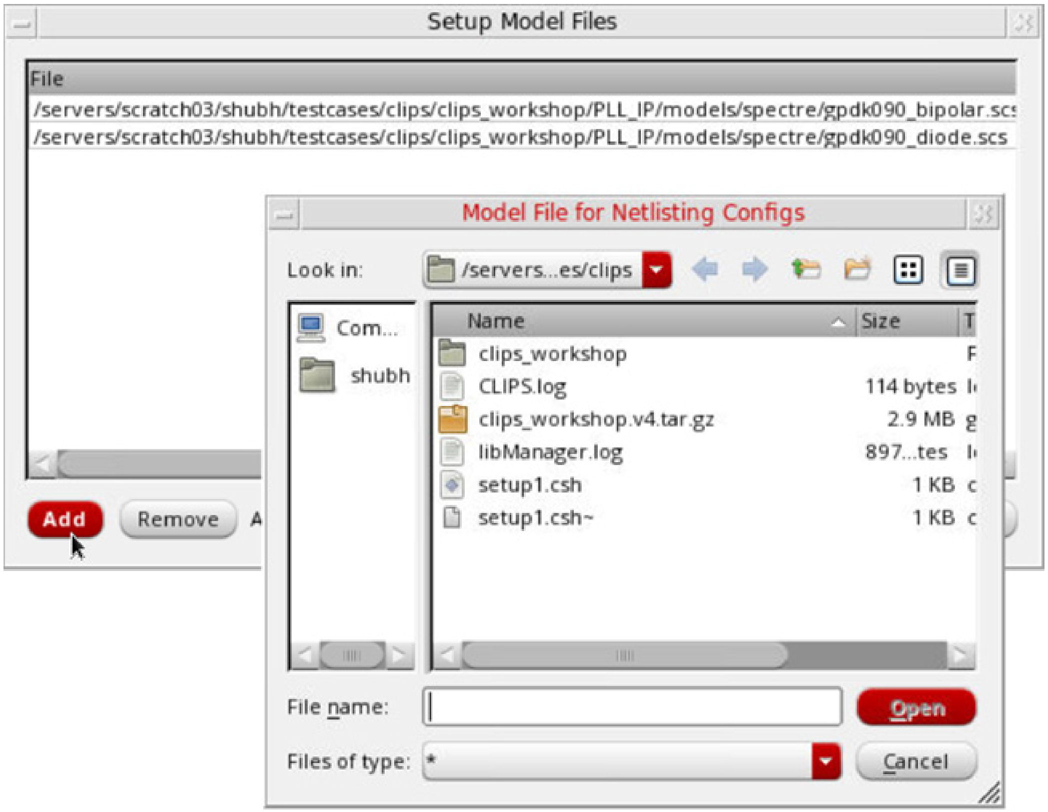


Setup Model Files

File	Section
/servers/scratch03/shubh/testcases/clips/clips_workshop/PLL_IP/models/spectre/gpdk090_bipolar.scs	bjt
/servers/scratch03/shubh/testcases/clips/clips_workshop/PLL_IP/models/spectre/gpdk090_diode.scs	dio_hvt

Add model files and optionally set sections.

2. Click *Add* to open the file browser and select model files.



Setup Model Files

File
/servers/scratch03/shubh/testcases/clips/clips_workshop/PLL_IP/models/spectre/gpdk090_bipolar.scs
/servers/scratch03/shubh/testcases/clips/clips_workshop/PLL_IP/models/spectre/gpdk090_diode.scs

Model File for Netlisting Configs

Look in: /servers...es/clips

Name	Size	T
clips_workshop		F
CLIPS.log	114 bytes	l
clips_workshop.v4.tar.gz	2.9 MB	g
libManager.log	897...tes	l
setup1.csh	1 KB	c
setup1.csh~	1 KB	c

File name:

Files of type: *

To remove an existing model file, select the file and click *Remove*.

3. Click *OK*.

Group Management

By default, all the instances in the hierarchy table inherit the global design settings, such as model files, design information, and netlisting options. If required, you can specify different values to these settings for one or more hierarchical instances. For that, you need to create a group that contains one or more hierarchical instances, and then, specify settings for the group.

In addition to the design settings mentioned above, you can define options to specify the pre-processing and post-processing options for a CLIPS run and directories to be used.

Related Topics

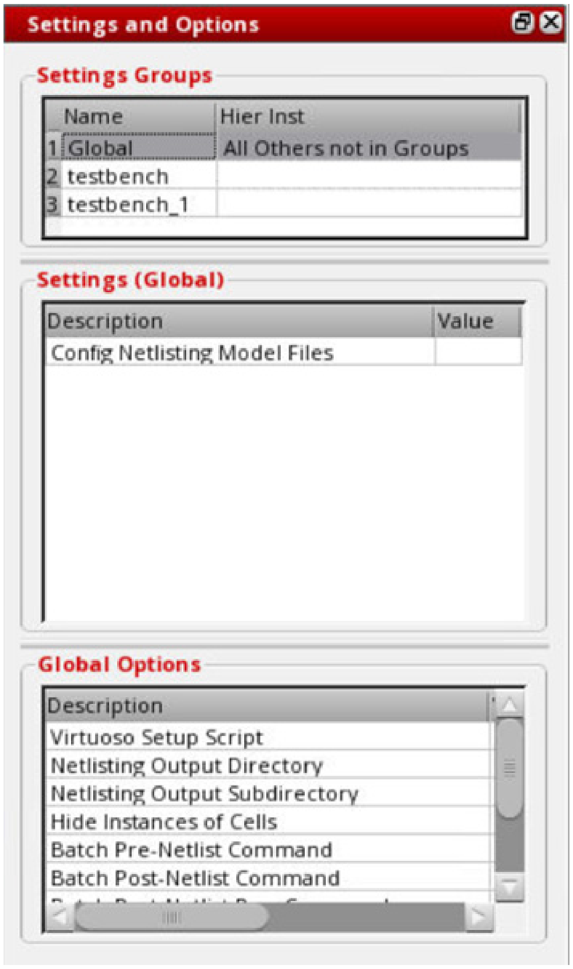
Creating a Group

Applying Settings to Specific Groups

Creating a Group

To create a group to include specific hierarchical instances, perform the following steps:

1. Choose *View — Show/Hide Settings* to open the Settings and Options assistant.



	Name	Hier Inst
1	Global	All Others not in Groups
2	testbench	
3	testbench_1	

Description	Value
Config Netlisting Model Files	

Description
Virtuoso Setup Script
Netlisting Output Directory
Netlisting Output Subdirectory
Hide Instances of Cells
Batch Pre-Netlist Command
Batch Post-Netlist Command

The available groups are displayed in the *Settings Groups* section.

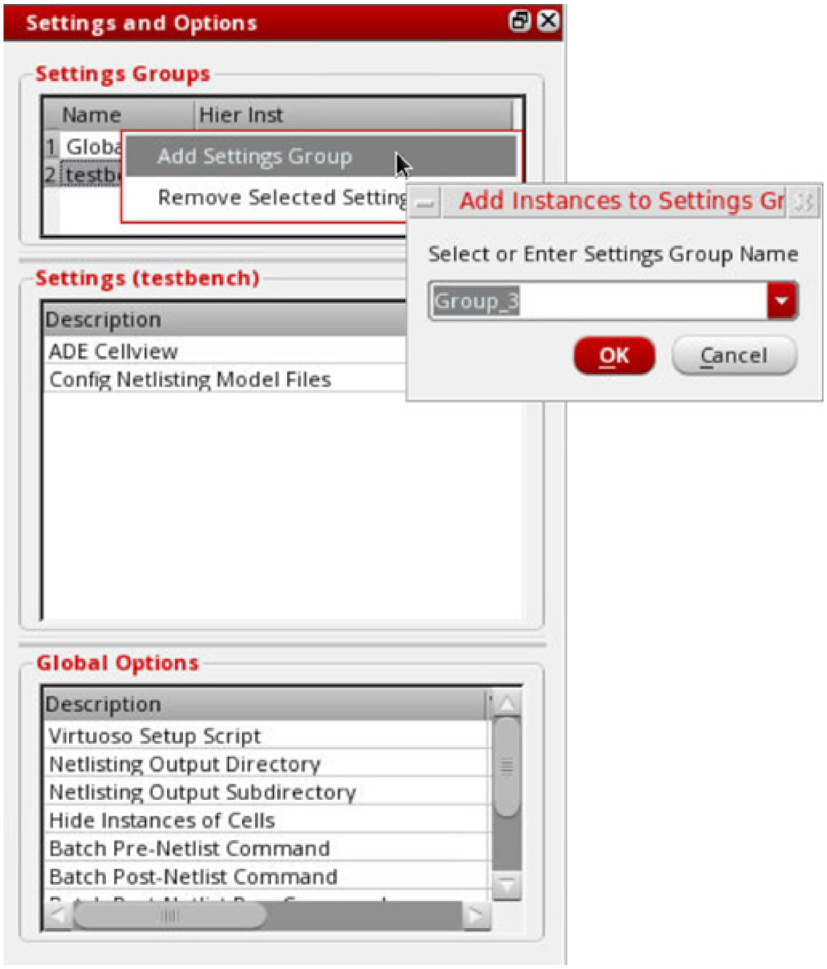
Note: By default, only one group named *Global* is created. All the hierarchical instances belong to this group and use the common settings.

2. To create a new group, right-click in the *Settings and Groups* section and choose *Add Settings Group* from the context-sensitive menu. The Add instances to Settings Group form is displayed.

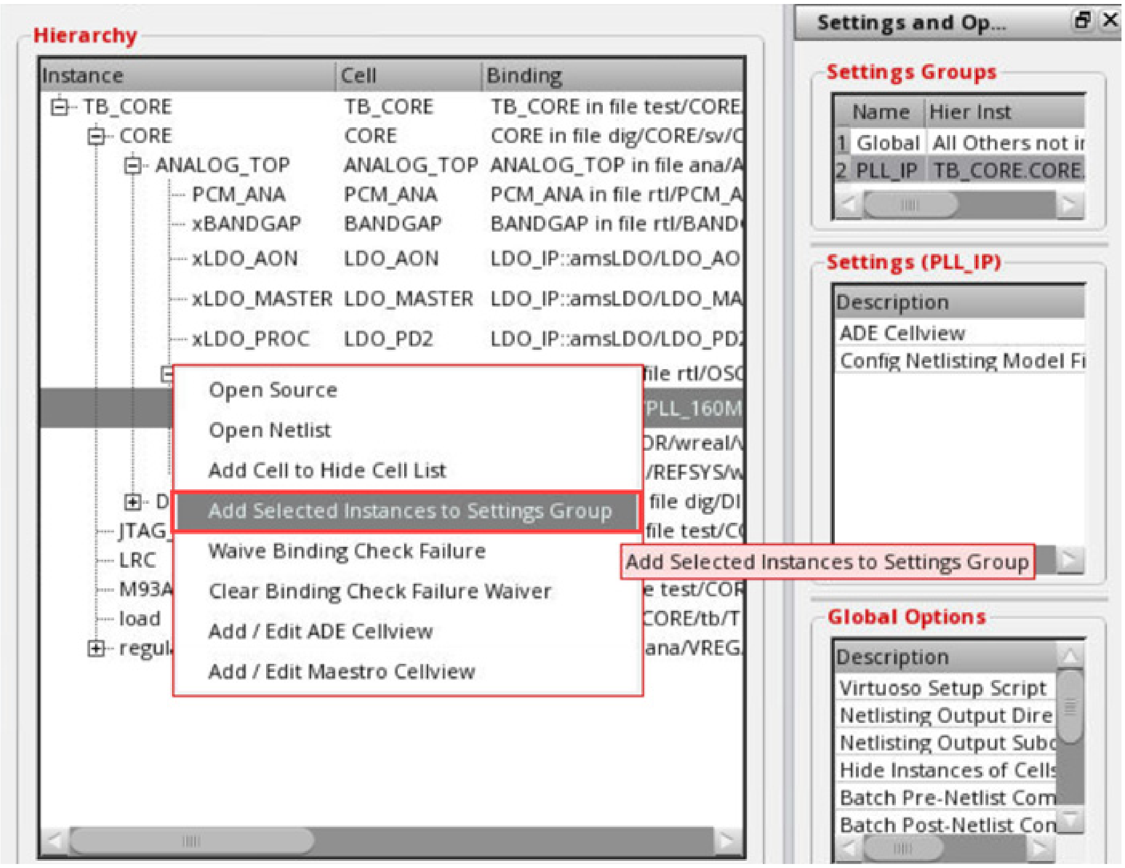
3. Specify a name for the new group in the *Select or Enter Settings Group Name* field.

4. Click OK to close the form.

A new group is displayed in the *Settings Groups* section.



5. To add an instance to this group, right-click an instance in the *Hierarchy* sections of the CLIPS window and choose *Add Selected Instances to the Settings Group*.


Hierarchy
Settings and Op...
Settings Groups
Settings (PLL_IP)
Global Options
Open Source
Open Netlist
Add Cell to Hide Cell List
Add Selected Instances to Settings Group
Waive Binding Check Failure
Clear Binding Check Failure Waiver
Add / Edit ADE Cellview
Add / Edit Maestro Cellview
Add Selected Instances to Settings Group

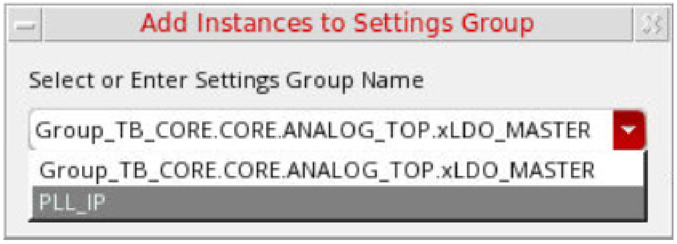
Instance	Cell	Binding
TB_CORE	TB_CORE	TB_CORE in file test/CORE
CORE	CORE	CORE in file dig/CORE/sv/C
ANALOG_TOP	ANALOG_TOP	ANALOG_TOP in file ana/A
PCM_ANA	PCM_ANA	PCM_ANA in file rtl/PCM_A
xBANDGAP	BANDGAP	BANDGAP in file rtl/BAND
xLDO_AON	LDO_AON	LDO_IP::amsLDO/LDO_AO
xLDO_MASTER	LDO_MASTER	LDO_IP::amsLDO/LDO_MA
xLDO_PROC	LDO_PD2	LDO_IP::amsLDO/LDO_PD

	Name	Hier Inst
1	Global	All Others not ir
2	PLL_IP	TB_CORE.CORE.

Description
ADE Cellview
Config Netlisting Model Fi

Description
Virtuoso Setup Script
Netlisting Output Dire
Netlisting Output Subc
Hide Instances of Cells
Batch Pre-Netlist Com
Batch Post-Netlist Con

The Add Instances to Settings Group form is displayed.



Select or Enter Settings Group Name

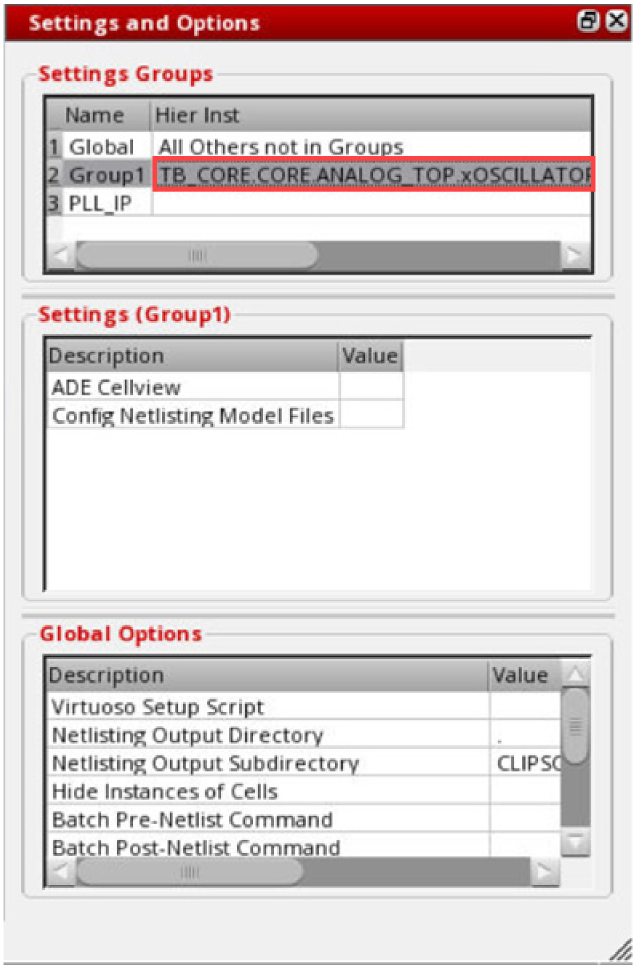
Group_TB_CORE.CORE.ANALOG_TOP.xLDO_MASTER

Group_TB_CORE.CORE.ANALOG_TOP.xLDO_MASTER
PLL_IP

6. In the *Select or Enter Settings Group Name* drop-down list, select the name of the group to which you want to add the instance.

Note: You can add more than one instance to the group.

The instance name is displayed in the *Hier Inst* column of the *Settings Groups* section, as shown below.



Settings and Options

Settings Groups

	Name	Hier Inst
1	Global	All Others not in Groups
2	Group1	TB_CORE.CORE.ANALOG_TOP.xOSCILLATO
3	PLL_IP	

Settings (Group1)

Description	Value
ADE Cellview	
Config Netlisting Model Files	

Global Options

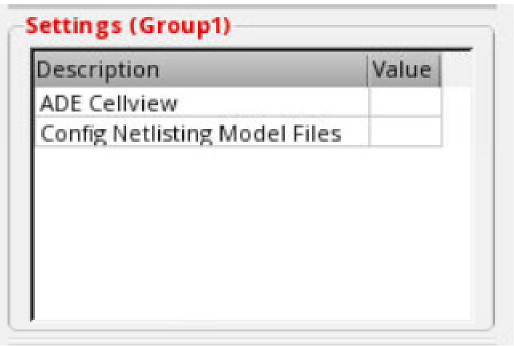
Description	Value
Virtuoso Setup Script	
Netlisting Output Directory	.
Netlisting Output Subdirectory	CLIPSC
Hide Instances of Cells	
Batch Pre-Netlist Command	
Batch Post-Netlist Command	

Applying Settings to Specific Groups

Settings specified for the Global group are applied to all the instances that are not a part of any other group.

To apply different settings to some specific instances, perform the following steps:

1. In the *Settings Groups* section, select the group name that contains the hierarchical instances for which you need to modify the settings. The properties of the selected groups are shown in the *Settings (<group-name>)* section.



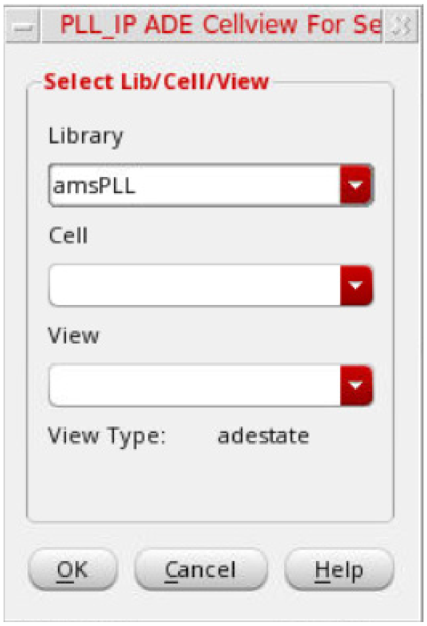
Settings (Group1)

Description	Value
ADE Cellview	
Config Netlisting Model Files	

2. Double-click the *Value* cell corresponding to the setting you want to modify.

3. Enter the value to the form that is displayed.

   Note: Different forms are displayed to specify the ADE cellview or model files. For ADE cellviews, the *<inst-name> ADE Cellview for Selected Instances* form is displayed, as shown below.



PLL_IP ADE Cellview For Se

Select Lib/Cell/View

Library: amsPLL

Cell:

View:

View Type: adestate

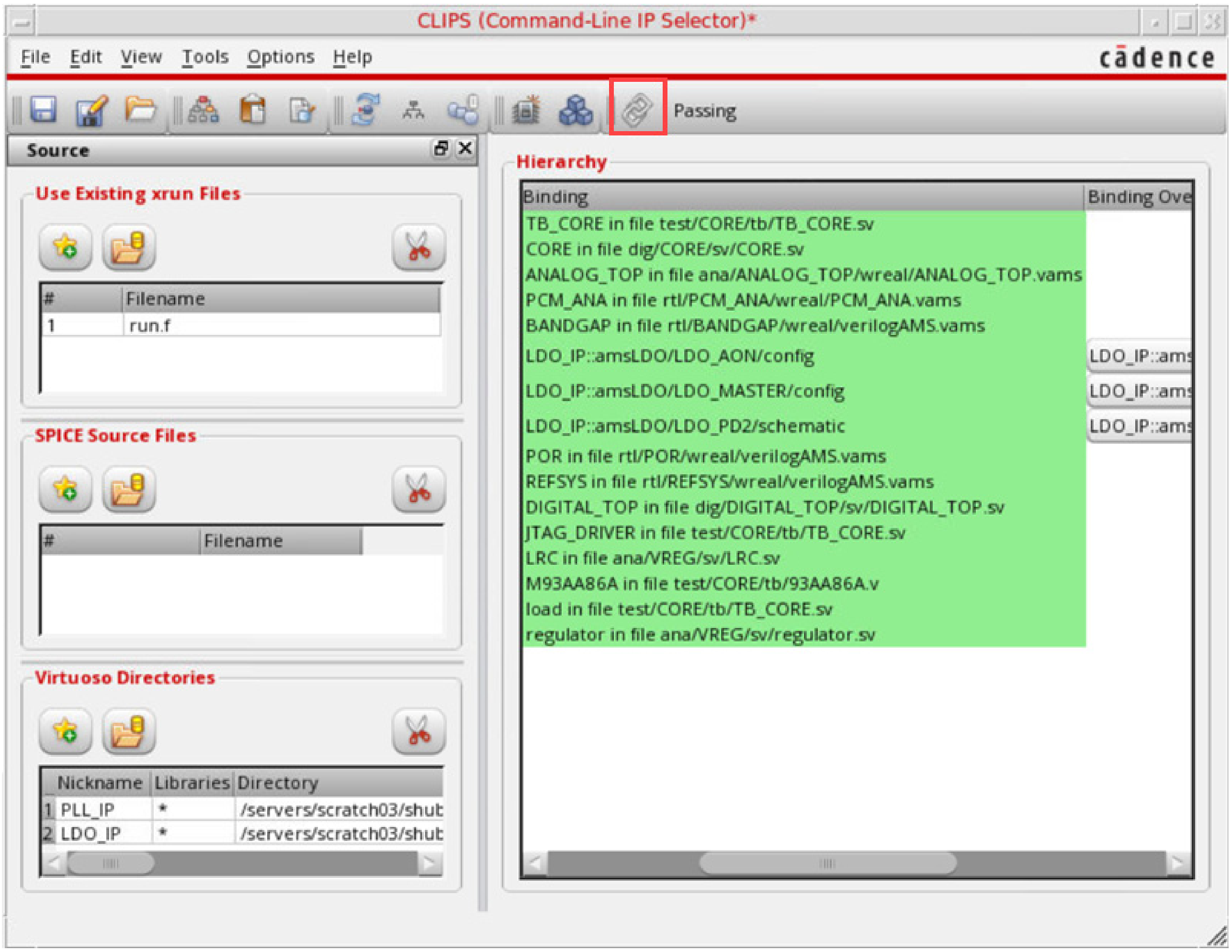
For `Config Netlisting Model files`, the *Setup Model* form is displayed.

Binding Check

Mostly, the top-level bindings (outside the IP block) are the native bindings by the Verilog language, which means that while searching for Virtuoso cellviews, it follows the search order set in the Xcelium simulation. This kind of native binding gets changed during a binding switch because new libraries are introduced.

Therefore, it is highly recommended to check whether the switched bindings are correctly used by the elaborator before running the post-CLIPS simulation. The existing bindings for all the remaining instances must remain intact.

You can perform a binding check by using the *Check Bindings* command on the toolbar. The pre-CLIPS and post-CLIPS elaboration results are compared, and any mismatch found is highlighted. The correct bindings are highlighted in green. If an incorrect binding is found, it is highlighted in red.



#	Filename
1	run.f

#	Filename

	Nickname	Libraries	Directory
1	PLL_IP	*	/servers/scratch03/shub
2	LDO_IP	*	/servers/scratch03/shub

Binding	Binding Ove
TB_CORE in file test/CORE/tb/TB_CORE.sv	
CORE in file dig/CORE/sv/CORE.sv	
ANALOG_TOP in file ana/ANALOG_TOP/wreal/ANALOG_TOP.vams	
PCM_ANA in file rtl/PCM_ANA/wreal/PCM_ANA.vams	
BANDGAP in file rtl/BANDGAP/wreal/verilogAMS.vams	
LDO_IP::amsLDO/LDO_AON/config	LDO_IP::ams
LDO_IP::amsLDO/LDO_MASTER/config	LDO_IP::ams
LDO_IP::amsLDO/LDO_PD2/schematic	LDO_IP::ams
POR in file rtl/POR/wreal/verilogAMS.vams	
REFSYS in file rtl/REFSYS/wreal/verilogAMS.vams	
DIGITAL_TOP in file dig/DIGITAL_TOP/sv/DIGITAL_TOP.sv	
JTAG_DRIVER in file test/CORE/tb/TB_CORE.sv	
LRC in file ana/VREG/sv/LRC.sv	
M93AA86A in file test/CORE/tb/93AA86A.v	
load in file test/CORE/tb/TB_CORE.sv	
regulator in file ana/VREG/sv/regulator.sv	

The result of binding check is displayed in the toolbar, where *Passing* indicates that correct bindings are confirmed.

Note: If you have enabled binding check in the GUI and saved it to the `.clips` file, the binding check will be performed when CLIPS is run in batch mode, unless `-checkbind` no is passed on the command line.

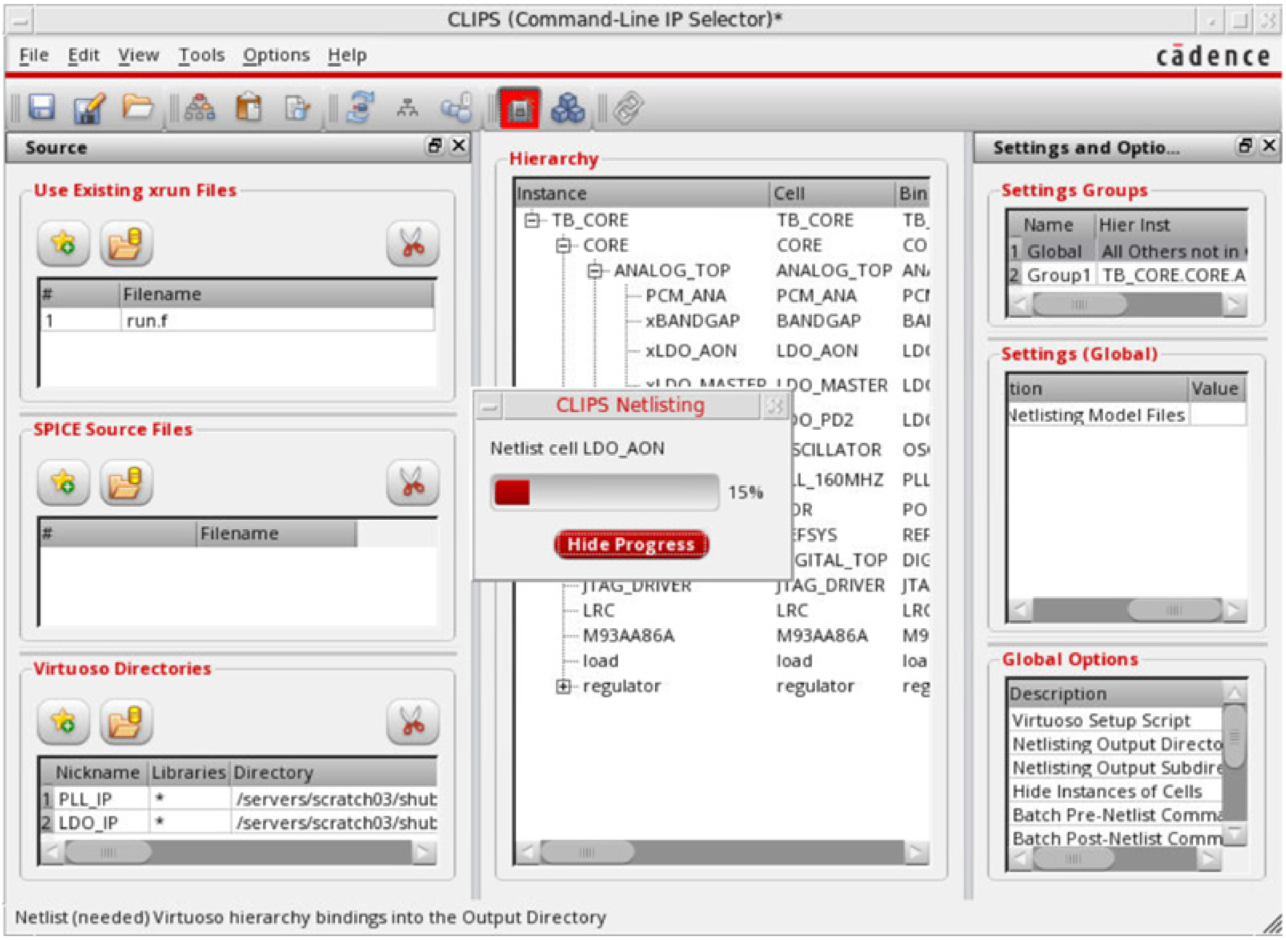
Related Topics:

Overriding Instance Bindings

Creating a Netlist for Config or Schematic Views

Once you have updated the hierarchy after switching the instance bindings

➡ Click the *Netlist* command on the toolbar to netlist the config or schematic views.



Use Existing xrun Files

#	Filename
1	run.f

SPICE Source Files

#	Filename

Virtuoso Directories

	Nickname	Libraries	Directory
1	PLL_IP	*	/servers/scratch03/shut
2	LDO_IP	*	/servers/scratch03/shut

Hierarchy

Instance	Cell	Bin
TB_CORE	TB_CORE	TB_
CORE	CORE	CO
ANALOG_TOP	ANALOG_TOP	AN/
PCM_ANA	PCM_ANA	PCI
xBANDGAP	BANDGAP	BAI
xLDO_AON	LDO_AON	LD(
xLDO_MASTER	LDO_MASTER	LD(
	)O_PD2	LD(
	SCILLATOR	OS
	L_160MHZ	PLL
	DR	PO
	EFSYS	REF
	GITAL_TOP	DIG
JTAG_DRIVER	JTAG_DRIVER	JTA
LRC	LRC	LR(
M93AA86A	M93AA86A	M9
load	load	loa
regulator	regulator	reg

CLIPS Netlisting

Netlist cell LDO_AON

15%

Hide Progress

Settings Groups

	Name	Hier Inst
1	Global	All Others not in
2	Group1	TB_CORE.CORE.A

Settings (Global)

tion	Value
Netlisting Model Files	

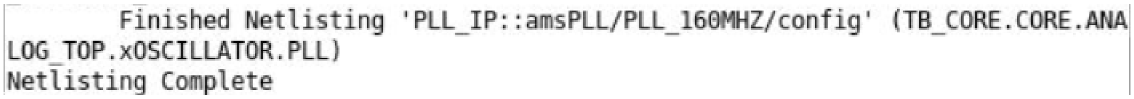
Global Options

Description
Virtuoso Setup Script
Netlisting Output Directo
Netlisting Output Subdire
Hide Instances of Cells
Batch Pre-Netlist Comma
Batch Post-Netlist Comm

Netlist (needed) Virtuoso hierarchy bindings into the Output Directory

It generates an incremental file, `clips.f`, along with the other necessary files.

When netlisting completes, messages are displayed in the terminal, as shown below:

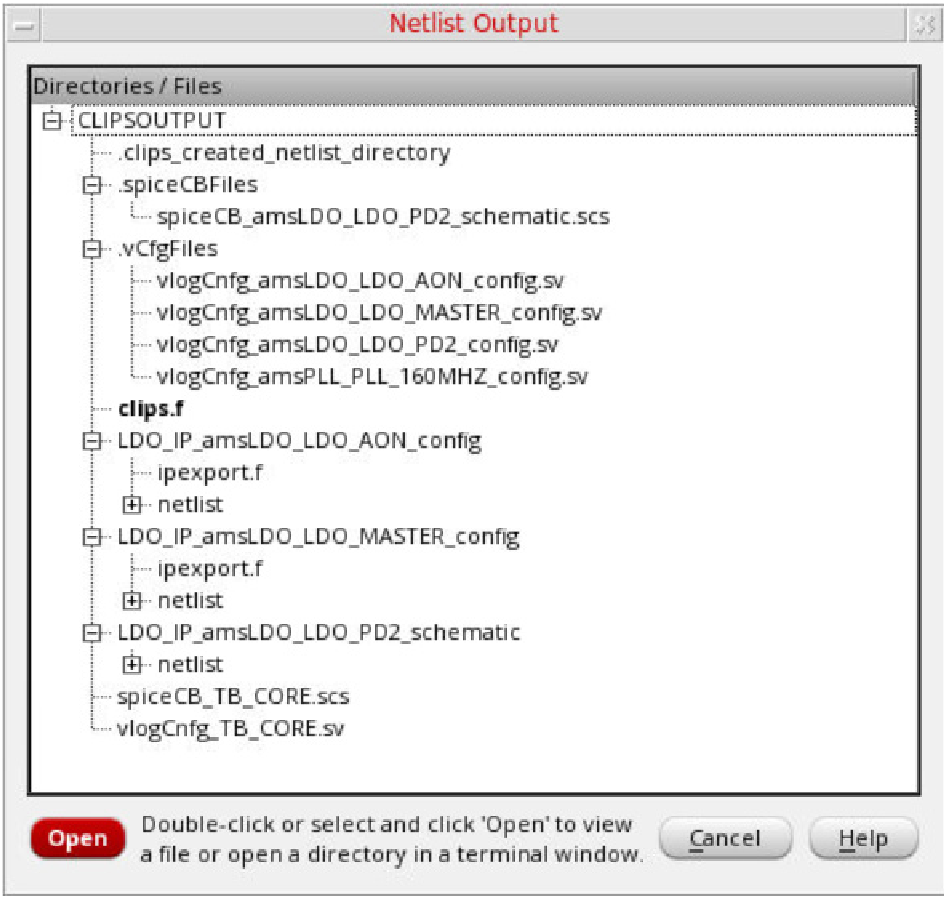


```
        Finished Netlisting 'PLL_IP::amsPLL/PLL_160MHZ/config' (TB_CORE.CORE.ANA
LOG_TOP.xOSCILLATOR.PLL)
Netlisting Complete
```

Viewing the Netlist Output

By default, all switched config or schematic views are netlisted in the `./CLIPSOUTPUT` directory.

1. To view the contents of this directory, click the *View Output* command on the toolbar. The directory structure is displayed in the Netlist Output window, as shown below:



```
⊟ CLIPSOUTPUT
    .clips_created_netlist_directory
  ⊟ .spiceCBFiles
      spiceCB_amsLDO_LDO_PD2_schematic.scs
  ⊟ .vCfgFiles
      vlogCnfg_amsLDO_LDO_AON_config.sv
      vlogCnfg_amsLDO_LDO_MASTER_config.sv
      vlogCnfg_amsLDO_LDO_PD2_config.sv
      vlogCnfg_amsPLL_PLL_160MHZ_config.sv
    clips.f
  ⊟ LDO_IP_amsLDO_LDO_AON_config
      ipexport.f
    ⊞ netlist
  ⊟ LDO_IP_amsLDO_LDO_MASTER_config
      ipexport.f
    ⊞ netlist
  ⊟ LDO_IP_amsLDO_LDO_PD2_schematic
    ⊞ netlist
    spiceCB_TB_CORE.scs
    vlogCnfg_TB_CORE.sv
```

Double-click or select and click 'Open' to view a file or open a directory in a terminal window.

2. To view the contents of the files displayed in the above window, double-click the file, or select the file and click *Open*.

3. To open the terminal in a directory, double-click the directory, or select the directory and click *Open*.

   To set your own output directory, perform the following steps.

   a. Choose *Edit—Netlist Output Directory*

      The Select Directory to Create Netlist Directory form is displayed.

   b. In this form, select a directory and click *Choose*.

4. To view the list of contents of this directory on the command line, enter the following command in the terminal:

```
% ls ./CLIPSOUTPUT/
```

Related Topics

Contents of the CLIPSOUTPUT Directory

Contents of the CLIPSOUTPUT Directory

In the `CLIPSOUTPUT` directory, separate directories are generated for all the config or schematic views bound to any of the instances. The `clips.f` file, which contains information about all the Virtuoso configurations, is also placed in this directory.

The following example shows the content available in the `clips.f` file:

```
# 'PLL_IP::amsPLL/PLL_ARST_DIG/config' first used for 'testbench.DUT.IP1'
-f /grid/cic/nsdpe-6/qingyu/project/clips/clips_demo_v2/TB/CLIPSOUTPUT/
PLL_IP_amsPLL_PLL_ARST_DIG_ip_0/ipexport.f
# 'PLL_IP::amsPLL/PLL_ARST/config' first used for 'testbench.DUT.IP2'
-f /grid/cic/nsdpe-6/qingyu/project/clips/clips_demo_v2/TB/CLIPSOUTPUT/
PLL_IP_amsPLL_PLL_ARST_ip_1/ipexport.f
-top vlogCnfg_testbench
/grid/cic/nsdpe-6/qingyu/project/clips/clips_demo_v2/TB/CLIPSOUTPUT/
vlogCnfg_testbench.sv
-compcnfg
```

Additionally, a SystemVerilog configuration file, `vlogCnfg_testbench.sv`, is generated to bind all the configurations under the top-level testbench.

```
config vlogCnfg_testbench;
        design testbench;
        default liblist worklib;
        //verilog 2001 binding for this IP
        cell PLL_ARST_DIG use cds_amsconfiglib.PLL_ARST_DIG:ip_0;
        cell PLL_ARST use cds_amsconfiglib.PLL_ARST:ip_1;
endconfig
```

In each Virtuoso configuration directory, the config or schematic view is netlisted in the same way as it is netlisted in the UNL flow. However, it is better packaged with an `ipexport.f` file, which contains the details of the Virtuoso config or schematic view netlisted by CLIPS.

Viewing Logs

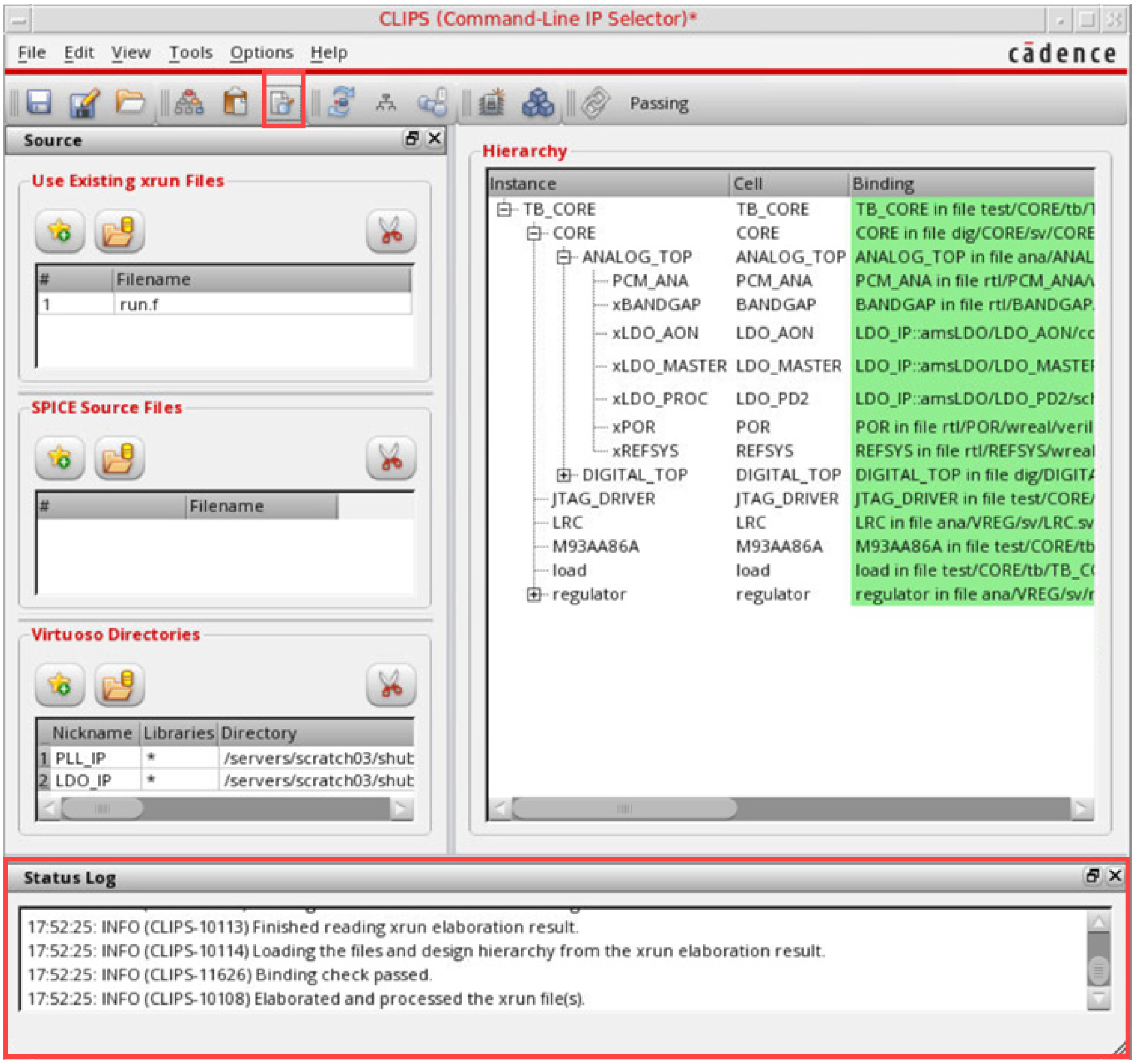
CLIPS logs show useful information about everything happening in the current session.

You can show and hide the log area of CLIPS.

To view the log pane, do one of the following

- Choose *View — Show / Hide Log.*

- Click the *Show/Hide Status Log* command on the toolbar.



#	Filename
1	run.f

#	Filename

	Nickname	Libraries	Directory
1	PLL_IP	*	/servers/scratch03/shut
2	LDO_IP	*	/servers/scratch03/shut

Instance	Cell	Binding
TB_CORE	TB_CORE	TB_CORE in file test/CORE/tb/T
CORE	CORE	CORE in file dig/CORE/sv/CORE
ANALOG_TOP	ANALOG_TOP	ANALOG_TOP in file ana/ANAL
PCM_ANA	PCM_ANA	PCM_ANA in file rtl/PCM_ANA/
xBANDGAP	BANDGAP	BANDGAP in file rtl/BANDGAP.
xLDO_AON	LDO_AON	LDO_IP::amsLDO/LDO_AON/cc
xLDO_MASTER	LDO_MASTER	LDO_IP::amsLDO/LDO_MASTEI
xLDO_PROC	LDO_PD2	LDO_IP::amsLDO/LDO_PD2/scl
xPOR	POR	POR in file rtl/POR/wreal/veril
xREFSYS	REFSYS	REFSYS in file rtl/REFSYS/wreal
DIGITAL_TOP	DIGITAL_TOP	DIGITAL_TOP in file dig/DIGIT/
JTAG_DRIVER	JTAG_DRIVER	JTAG_DRIVER in file test/CORE/
LRC	LRC	LRC in file ana/VREG/sv/LRC.sv
M93AA86A	M93AA86A	M93AA86A in file test/CORE/tb
load	load	load in file test/CORE/tb/TB_C(
regulator	regulator	regulator in file ana/VREG/sv/r

```
17:52:25: INFO (CLIPS-10113) Finished reading xrun elaboration result.
17:52:25: INFO (CLIPS-10114) Loading the files and design hierarchy from the xrun elaboration result.
17:52:25: INFO (CLIPS-11626) Binding check passed.
17:52:25: INFO (CLIPS-10108) Elaborated and processed the xrun file(s).
```

To capture the log files and intermediate files in a specified directory, perform the following steps:

1. Set the environment variable `CLIPS_DEBUG="dbg"`.
2. Create a local directory. For example, create a directory `local_tmp` using the command:

   ```
   mkdir local_tmp
   ```

3. Use the following option in CLIPS command-line:

   ```
   -tmpworkpath ./local_tmp
   ```

4. The content of virtuoso_<tempID>/directory is saved in the locally-defined directory `local_tmp`.

Differences Between CLIPS and Unified Netlisting (UNL)

CLIPS netlisting procedure works differently as compared to the UNL in ADE. The differences between the two netlisting procedures are summarized below:

- Files generated and included in Virtuoso IP netlisting:
  - All Whitebox Design Units (WDU) are netlisted in the same way as they are during UNL in ADE.
- Files generated but modified in Virtuoso IP netlisting:
  - `spiceModels.scs` and `scopeSpiceModels.scs`:
    - Path in the `spiceModels.scs` file is an absolute path.
    - `modelincdir` is not applied.
    - The scope-setting in the `scopeSpiceModels.scs` file is for the analog IP, which means that models inside this IP only search for the model files in this file (and its included files).
  - `cds_globals`
    - For the original `cds_globals.vams` from the SoC simulation setup (possibly from the UNL result), if there is no `-top` specified in the original `xrunArgs` file, CLIPS will add `-top cds_globals` in the `clips.f` file.
    - For the CLIPS netlisted `cds_globals`, a unique ID is generated and added to the name of each global module file (for example, `cds_globals_ip_0`). The name of the referencing parameter/signal is also added to the name of the module (for example, `cds_globals_ip_0.res`, `cds_globals.\VDD!`).
  - Files generated but not included in Virtuoso IP netlisting:
    - `svPkgTextInputs`/`vhdlPkgTextInputs` – It is suggested that you set this in the SoC simulation setup configuration, `xrunArgs`.
    - `ie_card.scs –` Since Interface Element (IE) cards are not supported in CLIPS, `ie_card.scs` is netlisted but commented out from the `ipexport.f`. If needed, you can manually modify the `ipexport.f` to include the `ie_card.scs` file.

    - `probe.tcl` – It is suggested that you set this in the SoC simulation setup configuration, `xrunArgs`.
    - `amsControlSpectre.scs` – This file contains only the analog simulation options. It is suggested that you set this in the SoC simulation setup configuration with an additional `xrunArgs` file.

- Options generated from an ADE L state but not included in the Virtuoso IP netlisting:
  - `-modelincdir`
    - You can set this only once. For adding multiple paths, use a colon `':'` to separate the paths.
    - Global setting changes the search order of model files.
    - You can set this in the SoC simulation setup (outside the testbench), if needed.
    - You can set the full path or the relative path from the Virtuoso invoking directory (where the `cds.lib` file is placed) for all the model files.
  - `-indir` or `+incdir` – Can be set sparsely, but all of the settings are applied globally.
  - `-reflib` and `-makelib`
    - These options are not printed in the `ipexport.f` file.
    - The `runtimeCompileFiles` file is generated in the netlist directory, but you need to add it in the `ipexport.f` file manually.

A

CLIPS Environment Variables

This section describes the public shell environment variables that control the characteristics of CLIPS. You can customize the operation and behavior of the tool by changing the value of a particular environment variable.

- CLIPS_CDS_BROWSE_MODEL
- CLIPS_DEBUG
- CLIPS_HIDE_CELLS
- CLIPS_INST_CNFG
- CLIPS_MODELFILE
- CLIPS_NETLISTING_TIMEOUT
- CLIPS_VIRTUOSO_BOURNE_SHELL
- CLIPS_VIRTUOSO_TIMEOUT

CLIPS_CDS_BROWSE_MODEL

```
CLIPS_CDS_BROWSE_MODEL "t_modelFilePath"
```

Description

Prompts to add model files to be used with Virtuoso for netlisting. This would happen when a Virtuoso session is added.

Examples

```
setenv CLIPS_CDS_BROWSE_MODEL "my_virtuoso_model/gpdk090.scs"
```

CLIPS_DEBUG

```
CLIPS_DEBUG "t_debugFileName"
```

Description

Saves the debugging logs locally in a user-defined log file.

Examples

```
setenv CLIPS_DEBUG "my_debug.log"
```

CLIPS_HIDE_CELLS

```
CLIPS_HIDE_CELLS "l_cellNames"
```

Description

Specifies a comma or space-separated list of cell or module names to be excluded from the hierarchy tree.

The default value is a blank list.

Examples

```
setenv CLIPS_HIDE_CELLS "cell1,cell2"
```

CLIPS_INST_CNFG

```
CLIPS_INST_CNFG {"YES"|"NO"}
```

Description

Generates all the switched bindings in explicitly instance-based configurations in Verilog-2001 configuration file.

The default value is `NO`.

Examples

```
setenv CLIPS_INST_CNFG "YES"
```

CLIPS_MODELFILE

```
CLIPS_MODELFILE "t_fileName"
```

Description

Predefines model files to be used within CLIPS in the same format as the `-modelfile` command-line argument. These model files are used for netlisting configurations.

The following syntax is used for defining model files:

```
<model1[:model2(sectionName)]>
```

The default value is a blank list.

Examples

```
setenv CLIPS_MODELFILE "/my_model_path/models.scs"
```

CLIPS_NETLISTING_TIMEOUT

```
CLIPS_NETLISTING_TIMEOUT "x_time"
```

Description

Specifies the maximum duration (in seconds) allowed for netlisting before timing out.

The default value is `300`.

Examples

```
setenv CLIPS_NETLISTING_TIMEOUT "150"
```

CLIPS_VIRTUOSO_BOURNE_SHELL

```
CLIPS_VIRTUOSO_BOURNE_SHELL {"YES"|"NO"}
```

Description

Ensures the parsing or analysis of a `.clips` file containing Bourne shell commands. By default, the cdsPre script uses C shell syntax.

The default value is "`NO`"

Examples

```
setenv CLIPS_VIRTUOSO_BOURNE_SHELL "YES"
```

CLIPS_VIRTUOSO_TIMEOUT

```
CLIPS_VIRTUOSO_TIMEOUT "x_time"
```

Description

Specifies the maximum duration for which CLIPS can attempt to connect to a Virtuoso session.

Possible values are any integer between `4` and `999` (seconds)

The default value is `90`.

Examples

```
setenv CLIPS_VIRTUOSO_TIMEOUT "50"
```

B

CLIPS Assistants

This section describes the following assistants available in the Command-Line IP Selector (CLIPS) tool.

- Settings and Options Assistant
- Source Assistant

Settings and Options Assistant

The Settings and Options assistant can be used to:

- Create a group to include specific hierarchical instances
- Apply different settings to specific instances
- Specify the global pre- and post-processing options for simulations and the directories to be used for simulation runs.

It has three sections.

Settings Groups	Lets you create settings groups and view the details related to it.
Settings (<Group-Name>)	Lets you view the properties of the selected groups.
Global Options	Lets you specify the global pre- and post-processing options for simulations and the directories to be used for the simulation runs

Settings Groups

The following table describes the command in the *Settings Groups* section.

Command name	Description
Add Settings Groups	Lets you create a new group. **Note:** This option is displayed on right-clicking in the *Settings and Groups* section.

Settings (<Group-Name>)

The following table describes the columns available in the *Settings (`<group-name>`)* section.

Column name	Description
Description	Displays the properties of the selected groups.
Value	Lets you modify the settings related to the properties displayed for a group.

Global Options

The following table describes the options in the *Global Options* section.

Option name	Description
Virtuoso Setup Script	Specifies the path to the UNIX script to set up Virtuoso Environment Variables.
Netlisting Output Directory	Specifies the name of the output directory.
Netlisting Output Subdirectory	Specifies the name of the subdirectory inside the output directory where the netlist is saved
Hide Instances of Cells	Specifies the list of cell names for which the instances are hidden from the hierarchy.
Batch Pre-Netlist Command	Specifies the UNIX command to be run before generating a netlist in the batch mode.
Batch Post-Netlist Command	Specifies the UNIX command to be run after generating a netlist in batch mode. This command is always run, irrespective of the pass or fail status of netlist creation.
Batch Post-Netlist Pass Command	Specifies the UNIX command to be run after successful creation of a netlist in batch mode.
Batch Post-Nestlist Fail Command	Specifies the UNIX command to be run after creation of a netlist fails in the batch mode.
Liblist in the Verilog2001 Configuration File	Lets you specify a new or modified list of libraries. If this field is empty, the default library list, `worklib` is used

Related Topics

Applying Settings to Specific Groups

Group Management

Creating a Group

Source Assistant

The Source assistant allows you to manage xrun files, SPICE source files, and virtuoso directories. It has three sections.

Use Existing xrun Files	Lets you manage the xrun files in CLIPS.
SPICE Source Files	Lets you manage the Spectre or SPICE text files.
Virtuoso Directories	Lets you specify the location of the directories that contain the Virtuoso library files, `cds.lib`. These cds.lib files contain the paths to the libraries where design files are saved.

Use Existing xrun Files

The following table describes the buttons in the *Use Existing xrun Files* section.

Icon	Command name	Description
★	*Add new xrun -f files*	Opens the CLIPS Add xrun -f Files form that you can use to select the xrun files from where you need to import the SoC simulation setup. The selected files are added to the table in this section.
📂	*View selected xrun file*	Opens the selected `xrun` file to show its content.
✂	*Remove selected xrun file*	Deletes the selected `xrun` script file.

SPICE Source Files

The following table describes the buttons in the *SPICE Source Files* section.

Icon	Command name	Description
★	*Add new SPICE source files*	Opens the CLIPS Add SPICE Source File(s) form that you can use to select SPICE source files from where you need to import the SoC simulation setup. The selected files are added to the table in this section.

Icon	Command name	Description
📂	*View Selected SPICE source file*	Opens the selected SPICE source file to show its content.
✂	Remove selected SPICE source file	Deletes the selected SPICE source file.

Virtuoso Directories

The following table describes the buttons in the *Virtuoso Directories* section.

Icon	Command name	Description
⭐	--	Opens a file browser in which you can select the Virtuoso library file, `cds.lib,` to import a Virtuoso directory that contains the mixed-signal IP configurations (Virtuoso config or schematic views).
📂	--	Opens the selected `cds.lib` file in the Virtuoso Library Manager.
✂	--	Deletes the selected `cds.lib` file.

Related Topics

xrun Files Management

Virtuoso Directories Management

Searching a Library