Agilent Technologies

Advanced Design System 2011.01

Feburary 2011
RFIC Dynamic Link

About RFIC Dynamic Link

Agilent Technologies and Cadence Design Systems both offer powerful EDA design tools. Many of today's design engineers prefer to use a combination of these tools to take advantage of the strengths of both design environments. Because of this desire to use multiple tools, Agilent Technologies has developed the *RFIC Dynamic Link for Cadence.* The Dynamic Link enables both *top-down* and *bottom-up* design and simulation in Advanced Design System (ADS) using IC designs from the Cadence database.

Advanced Design System

Advanced Design System has been developed specifically to simulate the entire communications signal path. This unique solution integrates the widest variety of proven RF, DSP, and electromagnetic design tools into a single, flexible environment. Building on years of expertise developing new technologies for our EDA tools, Advanced Design System provides a broad range of high-performance capability. This makes it easy to explore design ideas, then model the electrical and physical design of the best candidates.

Virtuoso Schematic Composer

The Virtuoso Schematic Composer from Cadence Design Systems is a hierarchical design entry tool used by RFIC circuit designers. Useful for both analog and digital designs, the database created is accessible by the Cadence simulation and physical layout tools. The tool supports multi-sheet schematics, including cross-referencing, symbol creation, automatic HDL cell template generation, global nets and hierarchical property definition for most database objects. The tool also provides hierarchical checking of connectivity, consistency of different cell representations and label attachments.

RFIC Dynamic Link

RFIC Dynamic Link is an EDA framework integration software product based on *Inter-Process Communication* (IPC), rather than data file translation, maximizing data integrity and ease of use.This documentation describes how to install and configure the RFIC Dynamic Link product and assists you in designing and analyzing analog mixed-signal and RF circuits via Dynamic Link. *Getting Started Tutorial* (dynlnkug) is provided to help you quickly get started with using the RFIC Dynamic Link. For information on Library Customization, refer to the *RFIC Dynamic Link Library Guide*.

RFIC Dynamic Link Use Model

The RFIC Dynamic Link use-model coincides with that of both Cadence and ADS, with only a few exceptions. Essentially, the *Virtuoso Analog Design Environment* user interface is replaced with Advanced Design System and all of its functionality. The Virtuoso features that are not directly replaced by ADS are provided on the *DynamicLink* pull-down menu in the Cadence Virtuoso Schematic window.

Note
If the *DynamicLink* pull-down menu does not appear in the Cadence Virtuoso Schematic window, choose **Launch > ADS Dynamic Link** (or **Tools > ADS Dynamic Link** for IC 5.1.41) in the Cadence schematic window.

Usage assumes basic familiarity with the Cadence *Virtuoso Design Environment* , including *Virtuoso* schematic capture and *Virtuoso Analog Design Environment* , as well as basic familiarity with design and simulation in the Advanced Design System.

Additional Information

- Wherever a shell variable is set, this documentation uses the Korn shell syntax.
- Unless otherwise mentioned, assume case sensitivity.

What's in this Documentation

The goal of this documentation is to help you get started, providing relevant examples that teach you how to use the software, and show you where you can get more information as you need it. This documentation contains:

- *Configuring RFIC Dynamic Link* (dynlnkug) describes the system requirements and how to install and configure the software.

- *Getting Started Tutorial* (dynlnkug) steps you through the process of simulating a circuit using components from the Dynamic Link analogLib library. Other examples are also included to help you become more familiar with the product.

- *Using RFIC Dynamic Link* (dynlnkug) provides information on launching ADS from a Cadence Schematic window, performing some basic operations and closing the Dynamic Link between ADS and Cadence.

- *Netlisting, Simulating, and Displaying Data* (dynlnkug) describes the procedures for netlisting and simulating a design as well as viewing the netlist from either ADS or a Cadence schematic window. Information on net, instance and expression name mapping is also provided.

- *Running a DSP and Analog/RF Cosimulation with RFIC Dynamic Link* (dynlnkug) provides and example of how to run an ADS Ptolemy Cosimulation using RFIC Dynamic Link.

- *Tuning and Optimizing Designs* (dynlnkug) provides information on tuning and optimizing designs using the ADS tuning and optimization capabilities.
- *Using Additional Features of RFIC Dynamic Link* (dynlnkug) includes a collection of Dynamic Link features such as "Freezing" selective subcircuits. Compatibility features covering support for Duplicate Pin Names, Bus-ports, Buses and Bundles, Unnamed Nets and pPar and iPar are also discussed.
- *Using Switch Views, Stop Views and the Hierarchy Editor* (dynlnkug) provides information on using switch views, stop views and the Hierarchy Editor in Dynamic Link.
- *Troubleshooting* (connectmui) provides information that can help you resolve common problems with RFIC Dynamic Link.
- *Command Reference* (dynlnkug) describes the function of each of the menu selections available for using RFIC Dynamic Link.

Command Reference for RFIC Dynamic Link

This section describes the function of each menu selection provided in:

- Advanced Design System (**DynamicLink** Menu)
- Cadence Schematic window (**Tools > ADS Dynamic Link** & **DynamicLink** Menus) and
- Cadence CIW (**Tools > ADS Dynamic Link** Menu item) while using the RFIC Dynamic Link.

ADS Schematic Window DynamicLink Menu

Menu Item	Description
Instance	
Add Instance of Cellview...	Add a symbol of a Cadence design to the current Advanced Design System Schematic window. For example, see *Adding a Symbol of the Cadence Cellview* (dynlnkug).
Update Instance of Cellview	Discards existing <*lib*>_<*cell*>_<*view*> ADS cellview, regenerates an ADS instance from the Cadence schematic cellview, and deletes the $HOME/simulation/<*cell*> directory. Deleting the $HOME/simulation/<*cell*> directory forces ADS to recreate a new netlist the next time ADS requires a netlist for a Cadence subcircuit.
Design Variables	
Get Design Variables	Add design variables from a Cadence Cellview to an Advanced Design System Schematic window. For example, see *Adding Design Variables* (dynlnkug).
Update Design Variables to Cellview	Update the value of the Advanced Design System schematic design variables to the Cadence Cellview. For example, see *Updating Cadence Design Variables* (dynlnkug).
Add Netlist File Include	Enables duplication of the Definition, Stimulus, and Model Library File include used in Cadence/Affirma. For more information, refer to *Adding Model Files* (dynlnkug).
Top-level Design Netlist	Generate and display the Advanced Design System netlist for the ADS cellview in the current ADS schematic window. For example, see *Viewing Netlists from Advanced Design System* (dynlnkug).
Annotate	
Annotate DC Voltages to Selected Cellview	Display the DC node voltages, generated by the ADS simulator, on the Cadence Schematic Window. For example, see *Performing a DC Simulation* (dynlnkug).
Annotate Operating Points to Selected Cellview	Display the DC operating points, generated by the ADS simulator, on the Cadence Schematic Window. For example, see *Annotating DC Operating Points to a Selected Cellview* (dynlnkug).
Clear DC Annotation in Selected Cellview	Erase DC node voltages and operating points in the Cadence Schematic Window.
Set Cadence Project Directory	This is the Cadence project directory and not the ADS workspace directory. The subcircuit netlist for a Cadence cellview is saved as: <*Cadence_Proj_Dir*>/<*cellName*>/adsDL/<*viewName*>/netlist/netlist.DL The Cadence subcircuit netlist will be merged with the top-level ADS netlist under the ADS workspace directory and used for Dynamic Link simulation. DC back annotation data for Dynamic Link is saved in the PSF files under: <*Cadence_Proj_Dir*>/<*cellName*>/adsDL/<*viewName*>/psf
Close Connection	Closes ADS and terminates the link between Cadence and ADS. For example, see *Ending the Session* (dynlnkug).

Cadence Schematic Window DynamicLink Menu

Menu Item	**Description**
Setup Options ...	Set parameters such as Switch View List and Stop View List. The default values for this dialog are obtained from the configuration file. For more information, consult your Cadence documentation.
New Design ...	Select a new design to include in the Cadence Cellview. Set parameters such as Library Name, Cell Name and View Name. The dialog also offers a browse feature to search for existing designs. For more information, consult your Cadence documentation.
Design Variables ...	Modify the Component Description Format (CDF) information for a component so that it works with ADS. For example, see *Using Design Variables* (dynlnkug). For more information, consult your Cadence documentation.
Annotate	Choose among annotation labels, after having run a successful Dynamic Link simulation and annotated results from DynamicLink menu items in ADS schematic window. Note that selecting the **DynamicLink > Annotate > Annotate DC Voltages to Selected Cellview** and the **DynamicLink > Annotate > Annotate DC Operating Points to Selected Cellview** from ADS schematic window will generate the (dcOp.dc and dcOpInfo.info, respectively) PSF file (based on the ADS dataset) and display results in Cadence schematic window; while selecting **DynamicLink > Annotate > DC Node Voltages** and **DynamicLink > Annotate > DC Operating Points** menu items in the Cadence schematic window will only display results if the corresponding PSF file exists and contains valid data.
Return to Parent Cellview	Bring the parent ADS schematic window to the front, after having pushed down from ADS to the current Cadence schematic cellview
Encrypt Cellview for ADS Dynamic Link ...	Encrypt the netlist of the current Cadence schematic cellview into an ADS library, create a Cadence component representing the encrypted ADS library, and create a wrapper schematic cellview containing the component to be used exclusively in ADS.
Subcircuit Netlist	Generate and display the Advanced Design System subnetwork netlist for the Cadence design displayed in a particular Cadence Schematic window. For example, see *Viewing Netlists from the Cadence Schematic Window* (dynlnkug). For more information, consult your Cadence documentation.

Note
If the *Dynamic Link* pull-down menu does not appear in the Cadence Virtuoso Schematic window, choose **Tools > ADS Dynamic Link > Add Dynamic Link menu to all schematic windows** in the Cadence Command Interpreter Window (CIW).

CIW Tools > ADS Dynamic Link Menu Item

There are two additional ADS sub-menu items accessible from the Cadence CIW.

Menu Item	**Description**
Start ADS Dynamic Link	Launch Advanced Design System without opening a Cadence cellview.
Add Dynamic Link menu to all schematic windows	Adds the *Dynamic Link* menu item to the Cadence schematic window with only the *Subcircuit Netlist* menu item enabled. This option enables you to create a Cadence subcircuit netlist without launching Advanced Design System.

Configuring RFIC Dynamic Link

This section describes system requirements and how to install and configure the software. You may require help from a UNIX or EDA Administrator to complete these tasks. For general system requirements, refer to *Check the System Requirements* (install) in the UNIX and Linux Installation documentation.

System Requirements

This section provides a reference to the minimum hardware and operating system requirements and also describes the EDA Framework and License requirements necessary for using the RFIC Dynamic Link.

Hardware Requirements

For information on the minimum hardware requirements for installing and using RFIC Dynamic Link, refer to *Check the System Requirements* (install) in the UNIX and Linux Installation documentation.

Software Requirements

Dynamic Link supports Cadence Virtuoso Design Environment versions on UNIX & Linux operating system versions from vendors which run this Cadence software. For a summary of supported platforms, refer to *Check the System Requirements* (install) in the UNIX and Linux Installation documentation, or contact Cadence Design Systems Inc.

Note that the following Cadence context files must be present under your Cadence installation directory in order to run RFIC Dynamic Link:

```
<Cadence Installation Dir>/tools/dfII/etc/context/spectrei.cxt
```

```
<Cadence Installation Dir>/tools/dfII/etc/context/asimenv.cxt
```

If either of these files is missing, please contact Cadence Design Systems Inc. for information on how to install it.

License Requirements

In addition to your standard Advanced Design System licenses, the following additional product licenses are required.

RFIC Dynamic Link License

- trans_idf

Cadence Licenses

- OASIS_Simulation_Interface
- 34510 - Virtuoso(R) Analog Design Environment
- 300 - Virtuoso(R) Layout Editor (if using layout)

> **Note**
> You must purchase all required Cadence licenses from Cadence Design Systems.

Installing RFIC Dynamic Link

The RFIC Dynamic Link installation procedure continues to be improved to make it easier for you to install and configure.
To install RFIC Dynamic Link:

1. Follow instructions in the *UNIX and Linux Installation* (install) documentation to run the SETUP utility and load the *install* program.
2. After the *Agilent EEsof Installation Manager* starts, you are prompted to select one of the following installation options:
    - **Complete** - If you choose a *Complete* installation, the RFIC Dynamic Link will be automatically installed.
    - **Custom** - If you choose a *Custom* installation, you must select *ADS programs and tools* component from the Choose Install Set scroll-down list of the *InstallAnywhere* dialog box.
3. After the installation is complete, you can run the *idfConfigCadence* script to configure Cadence for use with RFIC Dynamic Link. This script is located under: $HPEESOF_DIR/bin/idfConfigCadence

For more information on installation procedures, refer to the*UNIX and Linux Installation* (install) documentation.

Configuring the Cadence Installation

After the installation procedure (see Installing RFIC Dynamic Link), you can run the *idfConfigCadence* script if you want to customize your Cadence installation for Dynamic Link once and for all. This script configures the Cadence installation directory for use with ADS by adding or modifying files as required by Cadence.

If you prefer not to modify your existing Cadence installation(s), refer to the section on Avoiding Modifications to Your Cadence Installation.

Note
You must have write access to your Cadence install directory in order to run the *idfConfigCadence* script.

The following files are created:

<Cadence Installation Dir>/tools/dfII/etc/tools/ads/ .cdsenv

<Cadence Installation Dir>/tools/dfII/etc/tools/ADSsim/ .cdsenv

<Cadence Installation Dir>/tools/dfII/etc/tools/adsDL/ .cdsenv

<Cadence Installation Dir>/share/cdssetup/hierEditor/templates/ads

<Cadence Installation Dir>/tools/dfII/etc/skill/hnl/ ads.ile

<Cadence Installation Dir>/tools/dfII/etc/skill/si/caplib/ads.ile

An additional file, *<Cadence Installation Dir>* /tools/dfII/etc/tools/auCore/.cdsenv, is edited to add *ads* to the Tool Filter list of simulators.

Note
The files described in this section are created or modified based on detailed instructions provided in the Cadence *OASIS Direct Integrator's Guide*. The information in the Cadence documentation describes how to integrate a simulator into the Cadence *Analog Design Environment* using the Cadence *direct toolkit* .
Important While deviating from the information described in the Cadence *OASIS Direct Integrator's Guide* may work for RIFC Dynamic Link, the files have been specifically set up to follow the Cadence documentation. Agilent Technologies does not support deviations from this information.
For more information, refer to the Cadence documentation:
- *OASIS Integrator's Guide* , Product Version 4.4.5, December, 1999 - Pages 4-12, 4-13, 6-2 and 6-6.
- *OASIS Direct Integrator's Guide* , Product Version 4.4.6, September, 2001- Pages 67, 101 and 105.

Customizing a Cadence Installation

Before running the *idfConfigCadence* script, set HPEESOF_DIR to your ADS installation directory and set your PATH to include Cadence software.

The *idfConfigCadence* command uses the following general syntax:

```
idfConfigCadence {-h | -ls | -rm}
```

Simply entering *idfConfigCadence* at the command line with no options runs the script to configure Cadence for RFIC Dynamic Link. The following table shows a list of the options and definitions used by the *idfConfigCadence* command.

Option Definitions for idfConfigCadence

Option	Definition
-h	This option can be used to display the idfConfigCadence help file.
-ls	This option can be used to list the Cadence configuration files for ADS/RFIC Dynamic Link and indicate whether Cadence is ready for ADS RFIC Dynamic Link or not.
-rm	This option removes the Cadence configuration for ads/RFIC Dynamic Link by deleting the files previously installed with the idfConfigCadence script and removing ads from the Cadence tool Filter. For more information on the Cadence Tool Filter, refer to your Cadence documentation.

Understanding the idfConfigCadence Script

When the *idfConfigCadence* script is executed without any options, the script performs the following operations:

- Copies
  `$HPEESOF_DIR/idf/config/ adsCdsenvFile`
  to
  `<Cadence Installation Dir>/tools/dfII/etc/tools/ads/ .cdsenv`
- Copies
  `$HPEESOF_DIR/idf/config/ ads.hierEd`
  to
  `<Cadence Installation Dir>/share/cdssetup/hierEditor/ templates/ads`
- Creates two new empty files called *ads.ile* in the following locations:
  `<Cadence Installation Dir>/tools/dfII/etc/skill/hnl/ ads.ile`
  `<Cadence Installation Dir>/tools/dfII/etc/skill/si/caplib/ads.ile`

Customizing Multiple Cadence Installations

The *idfConfigCadence* script configures the Cadence installation directory for use with Dynamic Link. However, if a new version of the Cadence software is subsequently installed or you have multiple Cadence installations, neither will have the Dynamic Link configuration. In this case, you would need to run the *idfConfigCadence* script as described in Configuring the Cadence Installation for each Cadence installation.

Avoiding Modifications to Your Cadence Installation

If you do not want to change your Cadence installation or create a Cadence shadow directory tree for RFIC Dynamic Link, there is an alternative approach. You will need to set the following environment variable before starting Cadence Virtuoso Design Environment:

```
CDS_LOAD_ENV=CSF
```

In addition to the above, you will need to add the following line to your *setup.loc* file:

```
$HPEESOF_DIR/idf/ads_site
```

Sourcing the setCSF.ksh (or setCSF.csh) sample script under `$HPEESOF_DIR/bin` is one method of accomplishing the tasks above. Assuming you are using the Korn shell, the following is a list of steps for using the sample scripts to setup your environment for Dynamic Link in order to work with your original Cadence installation:

1. Change to a directory where you have write permission. For example,
   ```
   cd /tmp
   ```
2. Set your HPEESOF_DIR to your ADS installation root directory.
3. Add $HPEESOF_DIR/bin to your path. For example,
   ```
   PATH=$HPEESOF_DIR/bin:$PATH
   ```
4. Ensure your Cadence and Agilent licenses are set correctly for your environment.
5. Ensure that the Cadence <*cadence_installation_root*> `/tools/bin` and <*cadence_installation_root*> `/tools/dfII/bin` directories are in your PATH.
6. Set CDS_LOAD_ENV=CSF in the environment and add $HPEESOF_DIR/idf/ads_site to setup.loc file. For example, if using the Korn Shell,
   ```
   . $HPEESOF_DIR/bin/setCSF.ksh
   ```
   If you are using the C Shell, use the following command,
   ```
   source $HPEESOF_DIR/bin/setCSF.ksh
   ```
7. If you will use adsLib components, ensure that you include the adsLib library. For example,
   ```
   echo INCLUDE \$HPEESOF_DIR/cdslibs/rfde.lib > cds.lib
   ```

8. Ensure that the Dynamic Link startup file ads.ini will be loaded upon starting Cadence Virtuoso Design Environment. For example,

```
cp $HPEESOF_DIR/idf/examples/.cdsinit .
```

9. Launch Cadence by entering `virtuoso&` (`icms&` or `msfb&` for Cadence IC 5.1.41.)

If Cadence fails to locate the Dynamic Link OASIS files under your Cadence installation, the software will look for these OASIS files under `$HPEESOF_DIR/idf/ads_site`.

The alternate search location will be used because of the way CDS_LOAD_ENV and setup.loc are set in step 6 above. Dynamic Link will still operate normally with your original Cadence installation. This may be the prefered approach when you have multiple Cadence installations, in which case you would need to add Dynamic Link OASIS files to every individual installation or create a shadow directory for each installation.

According to the Cadence *Application Infrastructure User Guide, Product Version 5.01* , Cadence looks for the Cadence setup search file, `$CDS_SITE/cdssetup/setup.loc`, if $CDS_SITE is defined in the environment; otherwise, it will look for *<your_installation_directory>* `/share/cdssetup/setup.loc`.

For more information about CDS_LOAD_ENV, CSF, setup.loc, or OASIS, refer to your Cadence documentation or contact your Cadence customer support representative.

Configuring the Software

This section describes the various aspects of configuring and/or modifying the software to provide additional flexibility.

Configuring the UNIX Environment

There are several UNIX environment variables relevant to Dynamic Link. These are described in the following table.

UNIX Environment Variables

Environment Variable	**Description**
IDF_CONFIG_FILE	The name of the configuration file (only the file name, not the entire path). Default value is *idf.cfg.*
IDF_ADS_PROJ_DIR	If this environment variable is defined, ADS will open the specified workspace when RFIC Dynamic Link launches ADS.
IDF_DEBUG_MODE	If set to TRUE debugging will be turned on and additional log messages will be written to the CIW.
PATH	The UNIX path variable.

Configuring the ADS Install Directory

The installation procedure configures the ADS install directory by automatically adding, modifying or replacing some files and/or directories.
The following files and/or directories are created, modified or replaced:

```
$HPEESOF_DIR/bin/idfmp
$HPEESOF_DIR/bin/idf
$HPEESOF_DIR/bin/idfConfigCadence
$HPEESOF_DIR/idf
$HPEESOF_DIR/idf/ael
$HPEESOF_DIR/cdslibs
$HPEESOF_DIR/idf/config
$HPEESOF_DIR/idf/examples
$HPEESOF_DIR/idf/skill
$HPEESOF_DIR/tools/lib/dpkg/info/
```

Modifying the RFIC Dynamic Link-Cadence Initialization File

The RFIC Dynamic Link-Cadence initialization file (*.cdsinit)* is located in the directory *$HPEESOF_DIR/idf/config/.cdsinit*. Append it to or load it from the first available .cdsinit file from the list below:

- *<Cadence Installation Dir>* /tools/dfII/local/.cdsinit
- ./.cdsinit
- $HOME/.cdsinit

Modifying the Configuration File

Dynamic Link comes with the default configuration file *$HPEESOF_DIR/idf/config/ idf.cfg* . This file is used to set various site-specific or user-specific options. It is searched for and read sequentially from the following locations in the order given, so that settings in files read later override those of earlier files:

```
$HPEESOF_DIR/idf/config/

$HOME/hpeesof/config/

./
```

The name of the configuration file can be set via the UNIX environment variable IDF_CONFIG_FILE. By default the configuration file is named *idf.cfg* .
The configuration file consists of lines in the form < *parameter* > = < *value* >. The various parameters that can be set in the configuration file are listed below, with brief descriptions and an example for each. If no configuration file is found or some parameters are not set, internal default values are used. Note that wherever a file name is required for a configuration parameter value, it may be specified with a path prefix that is a UNIX environment variable value or using standard UNIX conventions such as ~. A complete example configuration file can be found at the end of this section.

- Switch View List: If an instance has none of the views listed in the switch view list, the netlister reports an error. The default is *"spectre ads sch.model schematic"* .
  Example:
  IDF_SWITCH_VIEW_LIST = "spectre ads schematic extracted config"

- Stop View List: The netlister identifies primitives with a stop list. When it reaches a view that is listed in both the switch view and stop view lists, the instance is netlisted and no expansion occurs below this level. The default Stop View list is *"spectre ads"* .
  Example:
  IDF_STOP_VIEW_LIST = "spectre"

- Project Path: This indicates the path to the ADS workspace to open upon starting ADS via RFIC Dynamic Link.

  Note
  Unless Cadence and ADS are installed on the same workstation where you run RFIC Dynamic Link, using this parameter is not recommended.

  Example:
  IDF_ADS_PROJ_DIR = /tmp/test_wrk

- Netlist Filter: When you have site customization that is not performed by the supplied netlister, this option enables you to specify the name of the program or script used to post-process the netlist generated by the Dynamic Link netlister.
  Example:
  IDF_NETLIST_FILTER = "$HOME_DIR/bin/myfilter"
  If you want to change all "vss!" in the netlist to 0 (zero or the ground in ADS), the script (*myfilter*) would look similar to the one below:
  `sed -e s/\"vss!\"/0/g`

- Debug Mode: This option enables you to turn debugging messages on or off. These messages appear in order to help you determine the cause and/or location of

problems.
By default, debugging is turned off. To enable debugging, set this option to *TRUE* .
Example:
IDF_DEBUG_MODE = TRUE

- Symbol Generation: This option enables you to specify whether to generate a missing symbol using the Cadence symbol generator or the Advanced Design System symbol generator. The Cadence symbol generator is used as the default. Example:
IDF_CADENCE_SYMBOL = FALSE

- User AEL Files: Users can define their own AEL functions and load them into the ADS environment via a list of comma separated file names. These files get loaded just after the Dynamic Link environment is initialized. Example:
IDF_USER_AEL_FILES = "file1.ael, $HOME/file2.atf"

- Expression Mapping: This causes sub-strings in Cadence expressions to be mapped to corresponding sub-strings in ADS expressions in the netlist file and/or in the design variable values used. Example:
IDF_EXPR_MAP = "foo bar"

- Message Timeout: This specifies the timeout period in seconds for message actions initiated in ADS to complete in Cadence Virtuoso Design Environment. The default is 45; however, you may need to set it to a value larger than the default for netlisting very large Cadence circuits. Example:
IDF_MSG_TIMEOUT = 200

- Other Parameters: The following parameters should not be altered.
IDF_AEL_FILES = "globals.atf, utils.atf, commands.atf, callbacks.atf, symbol.atf"
IDF_PDE_EXEC = hpeesofde
IDF_PDE_ARGS = "-env de_sim"
IDF_SCALE_FACTOR = 2.0

Example Configuration File

```
IDF_SWITCH_VIEW_LIST = "spectre ads config schematic extracted"

IDF_STOP_VIEW_LIST = "spectre ads"

IDF_NETLIST_FILTER = "$HPEESOF_DIR/idf/examples/pcell_workaround.py"
```

Managing Workspaces

Your Cadence designs will remain in their original locations. They are not copied, translated, or otherwise modified. When ADS starts up in Dynamic Link mode, it puts you in a workspace defined by the IDF_ADS_PROJ_DIR environment variable. If this variable is not defined, only the ADS Main window will open. This is the recommended option.

To change this behavior, you can do one of the following:

- Specify your own startup workspace by defining IDF_ADS_PROJ_DIR in your UNIX environment or via the configuration file *idf.cfg.*
- After the Advanced Design System has come up, go to the Main window and open a new or existing workspace (**File > New > Workspace** or **File > Open > Workspace**).

For more information on ADS workspace, refer to *Things You Must Know About ADS 2011* (oaqkref).

Getting Started with RFIC Dynamic Link

This tutorial steps you through the process of simulating a circuit using components from the Agilent version of the analogLib library. Other examples are also included to help you become familiar with the product.

Both the drop-down menus and icons are described to help familiarize you with the Advanced Design System environment.

Setting up the Examples Directory

To set up the examples directory, first ensure that:

- The HPEESOF_DIR environment variable is set to your ADS installation directory
- $HPEESOF_DIR/bin is in your PATH
- Your environment is set up to run the Cadence virtuoso (for IC 6.1.1 through 6.1.4) or msfb (for IC 5.1.41) tool

From any directory of your choice, enter:

```
cp -r $HPEESOF_DIR/idf/ examples ./

cd examples
```

Note
This must be done before attempting the Getting Started Tutorial. The *cds.lib* file under the examples directory defines libraries provided by Dynamic Link.
The *.cdsinit* file under this directory loads the Dynamic Link *.cdsinit* file which then loads the context files required to run Dynamic Link.

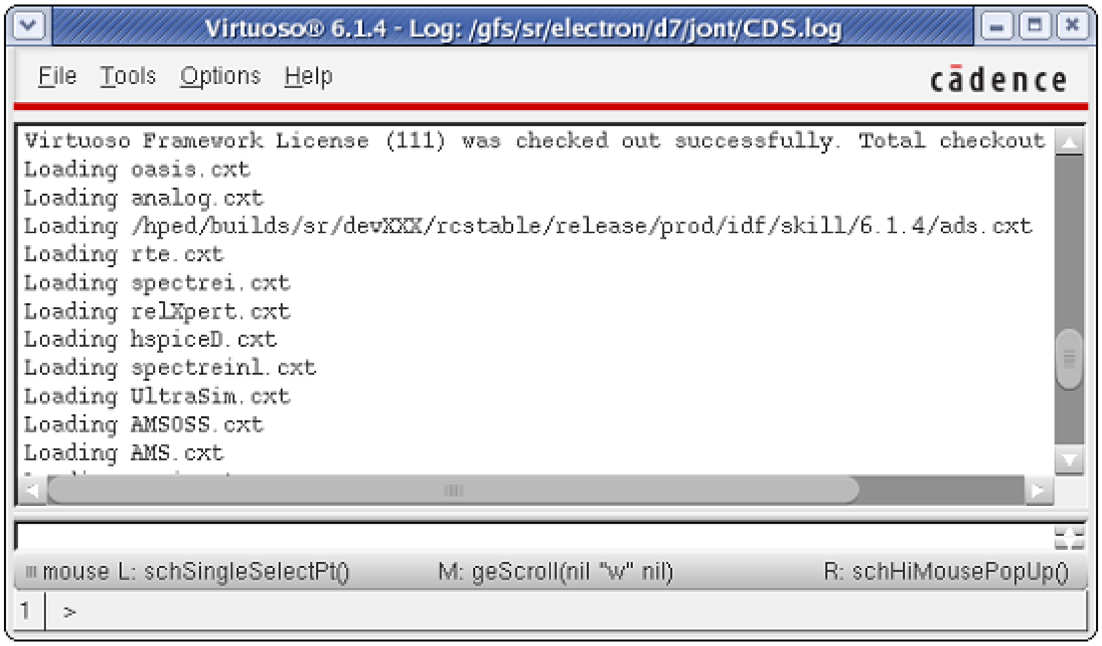
Define Cadence CSF and include $HPEESOF_DIR/idf/ads_site in ./setup.loc by sourcing $HPEESOF_DIR/bin/setCSF.ksh or $HPEESOF_DIR/bin/setCSF.csh depending on your SHELL. For example, under Korn Shell, enter:

```
. setCSF.ksh
```

Starting the Cadence Virtuoso Design Environment

Ensure that you are in the examples directory then launch Cadence Custom IC Design Tools by typing the appropriate command (*virtuoso*, *msfb* or *icms*). The Cadence *Command Interpreter Window* (CIW) appears.

Figure: Cadence CIW Window



```
Virtuoso Framework License (111) was checked out successfully. Total checkout
Loading oasis.cxt
Loading analog.cxt
Loading /hped/builds/sr/devXXX/rcstable/release/prod/idf/skill/6.1.4/ads.cxt
Loading rte.cxt
Loading spectrei.cxt
Loading relXpert.cxt
Loading hspiceD.cxt
Loading spectreinl.cxt
Loading UltraSim.cxt
Loading AMSOSS.cxt
Loading AMS.cxt
```

mouse L: schSingleSelectPt() M: geScroll(nil "w" nil) R: schHiMousePopUp()

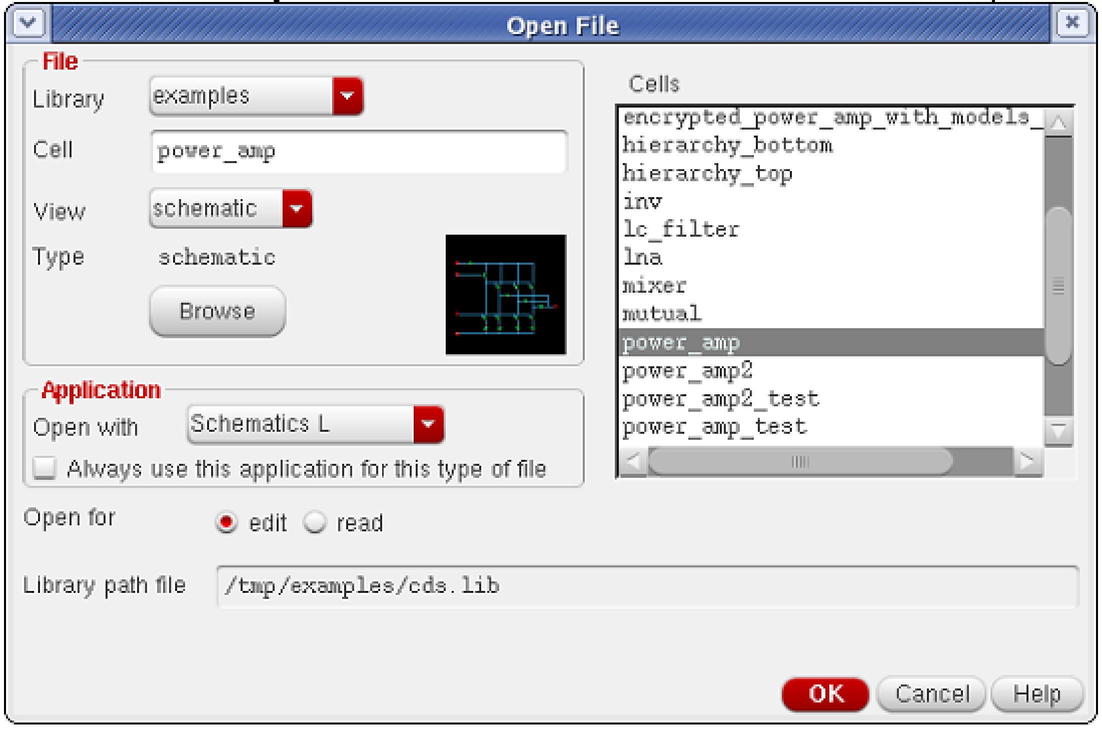
1 >

Note
Non-standard or customized start-up scripts for Cadence Virtuoso Design Environment may not be supported. If you have difficulties, contact your system administrator.

Opening a Cadence Virtuoso Schematic

To open a schematic cellview in Cadence Virtuoso Environment:

1. Choose **File > Open** from the Cadence CIW. The Cadence Open File form appears.



Library: examples
Cell: power_amp
View: schematic
Type: schematic

Open with: Schematics L
Always use this application for this type of file

Cells:
encrypted_power_amp_with_models_
hierarchy_bottom
hierarchy_top
inv
lc_filter
lna
mixer
mutual
power_amp
power_amp2
power_amp2_test
power_amp_test

Open for: [✓ edit] read

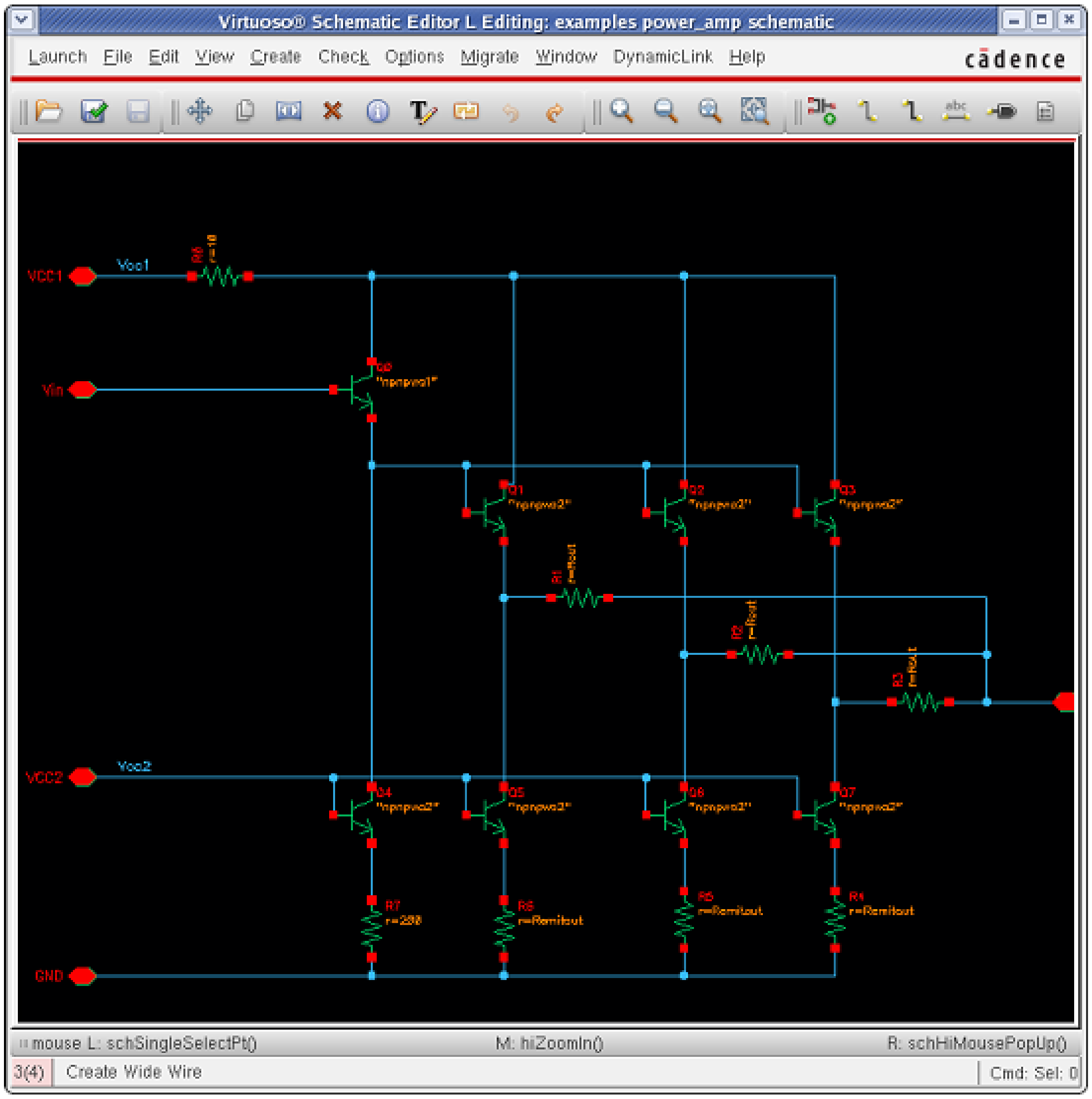
Library path file: /tmp/examples/cds.lib

2. Select *examples* from the Library Name drop-down list.
3. Click *power_amp* in the Cell Names list. This sets the Cell Name field to power_amp.
4. Select *schematic* from the View Name drop-down list if not already selected.
5. Select the *edit* Mode if not already selected.

	Note To open a file in edit mode you must have write permission.

6. Click **OK** . The Cadence *examples, power_amp* schematic cellview appears.

Figure: Cadence Virtuoso Schematic Window

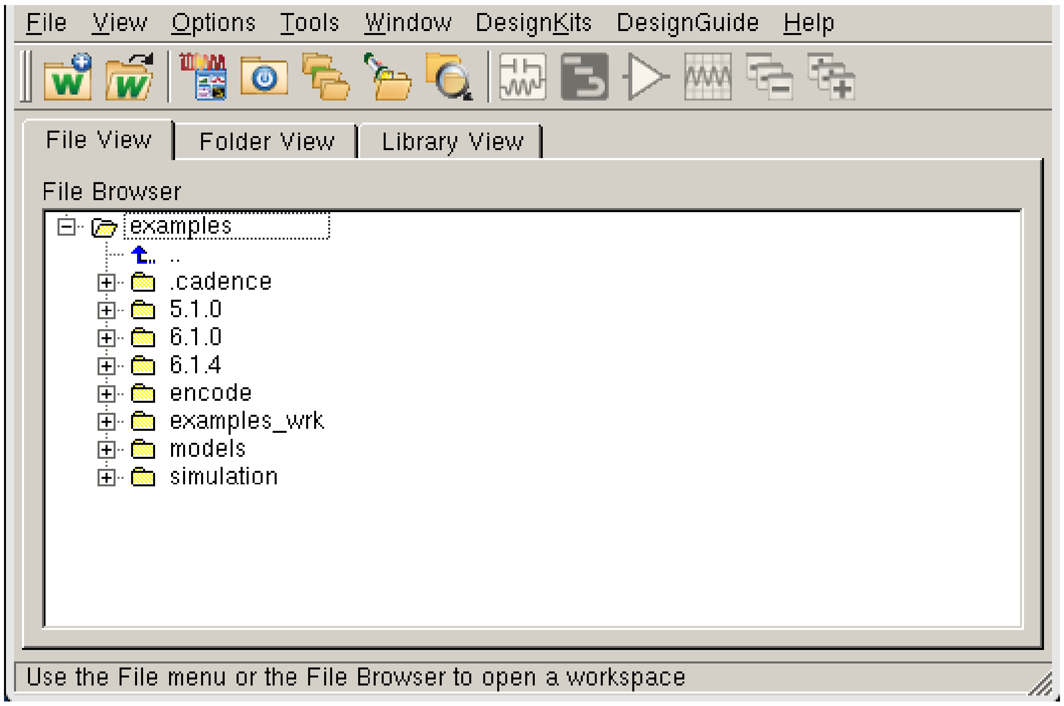

Virtuoso® Schematic Editor L Editing: examples power_amp schematic
Launch File Edit View Create Check Options Migrate Window DynamicLink Help
cādence
VCC1
Vcc1
Vin
VCC2
Vcc2
GND
mouse L: schSingleSelectPt()
M: hiZoomIn()
R: schHiMousePopUp()
3(4)
Create Wide Wire
Cmd: Sel: 0

Linking with Advanced Design System

To link the Cadence design environment to Advanced Design System:

1. Choose **Launch > ADS Dynamic Link** from the menu bar in the Cadence Schematic window. In a few moments, the Advanced Design System Main window appears in your display.

Figure: Advanced Design System Main Window

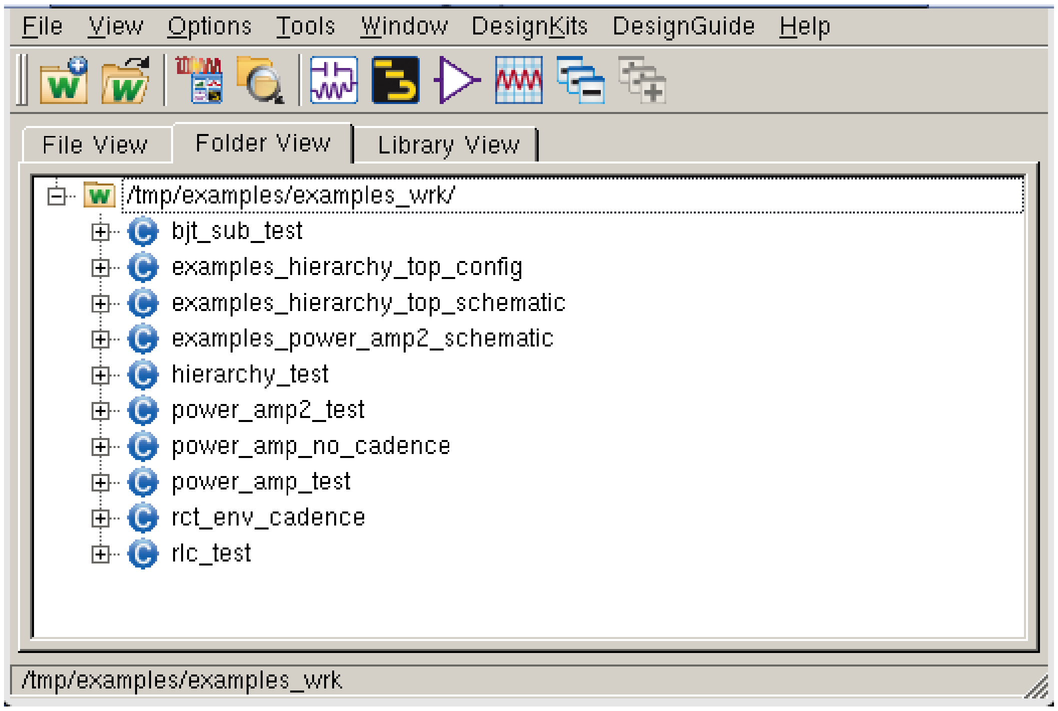


Use the File menu or the File Browser to open a workspace

> **Note**
> Depending on your system, it may take a few moments for the ADS windows to appear. View the Cadence CIW window for the link status.

2. From the ADS Main window, choose **File > Open > Workspace** and select the examples_wrk workspace. A list of cells appear in the Folder View tab of the ADS Main window (see the following figure).

Figure: Advanced Design System Main Window Folder View



/tmp/examples/examples_wrk/
bjt_sub_test
examples_hierarchy_top_config
examples_hierarchy_top_schematic
examples_power_amp2_schematic
hierarchy_test
power_amp2_test
power_amp_no_cadence
power_amp_test
rct_env_cadence
rlc_test

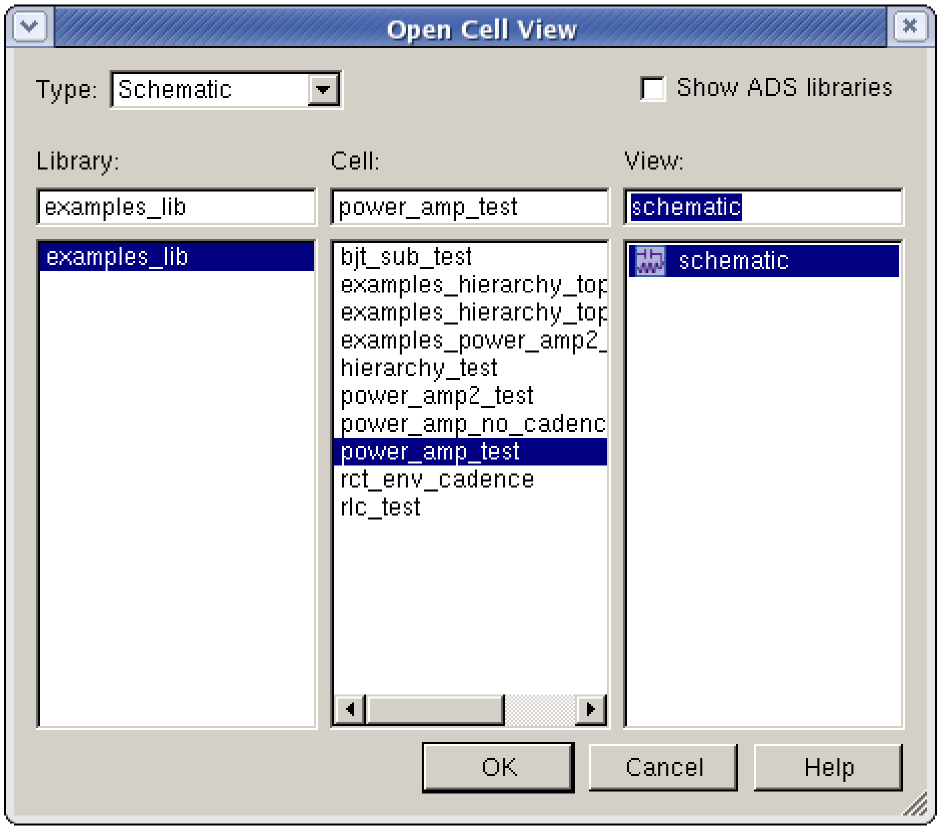
/tmp/examples/examples_wrk

At this point, ADS and the Cadence Virtuoso Design Environment are working together.

Opening a Test Schematic Design

To open a test schematic design:

1. Choose **File > Open > Schematic** in the ADS Main window to display the Open Cell View dialog box. Use this dialog box to select the design you wish to simulate.



Library: examples_lib
Cell: power_amp_test
View: schematic

examples_lib

bjt_sub_test
examples_hierarchy_top
examples_hierarchy_top
examples_power_amp2_
hierarchy_test
power_amp2_test
power_amp_no_cadenc
power_amp_test
rct_env_cadence
rlc_test

schematic

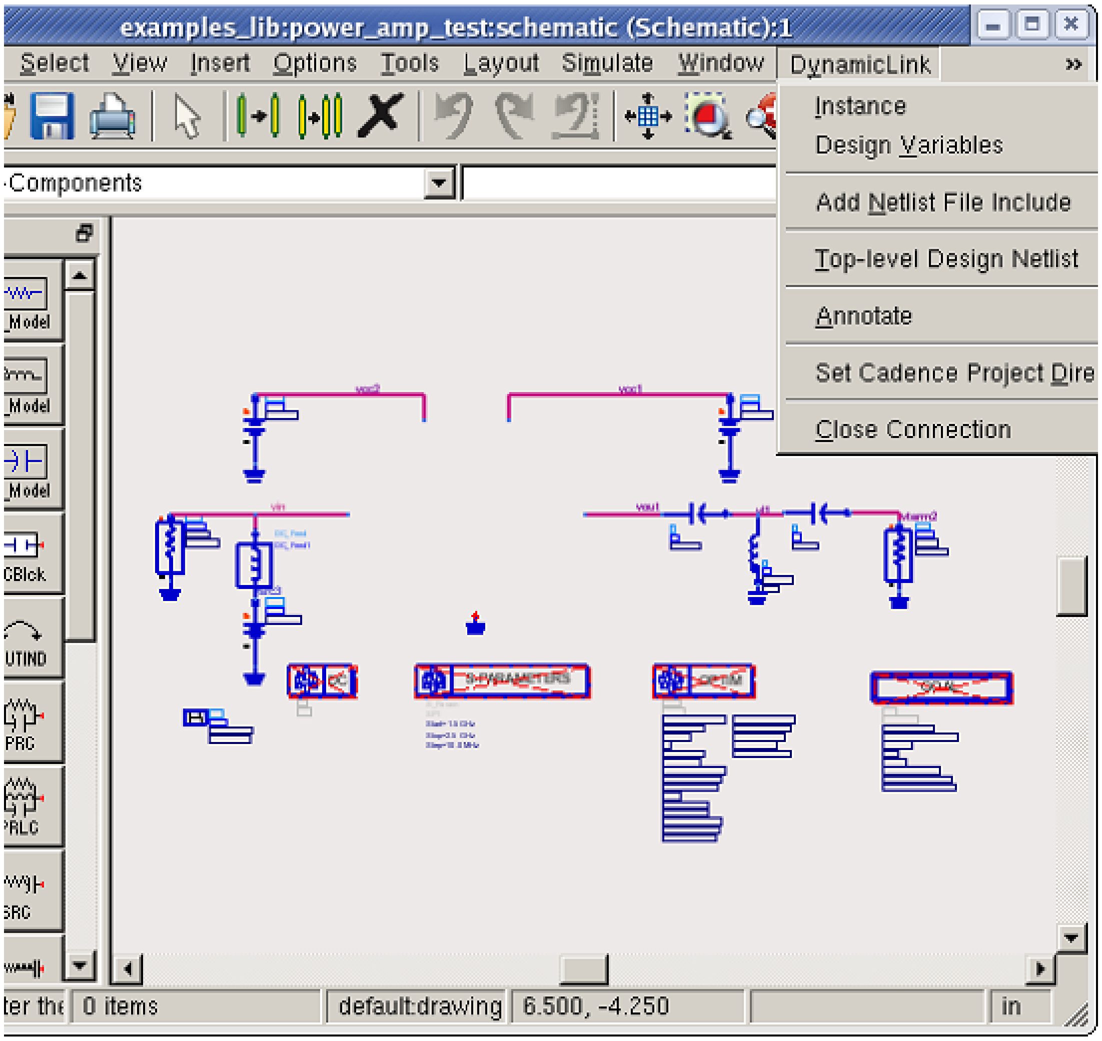
2. The *examples_lib* selection in the Library list and the *schematic* in the View list are selected by default in the Open Cell View dialog box.
3. Select *power_amp_test* from the Cell list. The *power_amp_test* schematic contains simulation components that can be selectively activated or deactivated.
4. Click **OK** to include the *power_amp_test* schematic in the ADS Schematic window.

Adding an Instance of the Cadence Cellview

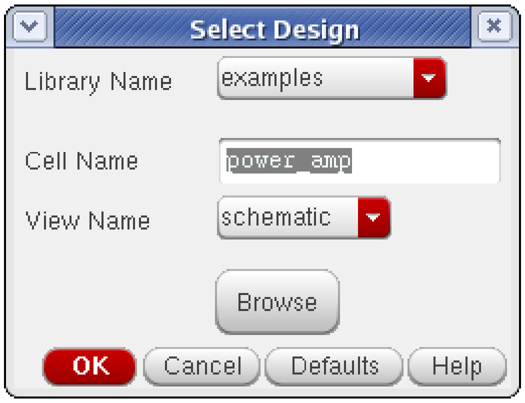
To add a symbol of the Cadence cellview in the Advanced Design System Schematic window:

1. Choose **DynamicLink > Instance > Add Instance of Cellview** in the ADS Schematic window.

Figure: Adding a Cellview



A Select Design dialog box appears, enabling you to select the Cadence Cellview to simulate.


Select Design

Library Name	examples
Cell Name	power_amp
View Name	schematic

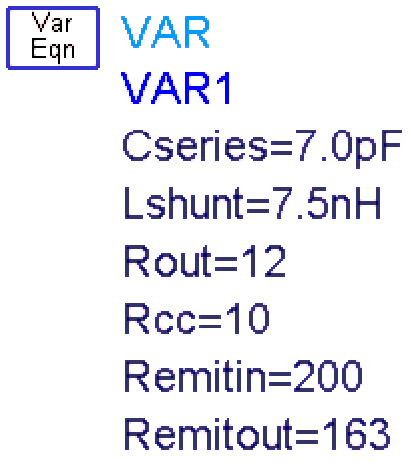
1. In the *Cell Name* field of this dialog, verify the entry or type the name of the Cellview you want to simulate (in this case *power_amp*). Alternatively, you can use the *Browse* button and library manager to select the name.
2. Click **OK** . A symbol of your Cadence Cellview is automatically generated.
3. An instance of the symbol is attached to the cursor for you to place. In the Advanced Design System Schematic window, click the left mouse button to place the symbol as

desired.

4. You may continue placing more instances of the same cellview, or, in this case, choose the *Cancel Command And Return To Select Mode* icon to proceed with the next step. Similarly, you may place instances of other Cadence designs.

Adding Design Variables

To add the design variables from the Cadence Cellview to the ADS schematic window choose **DynamicLink > Design Variables > Get Design Variables** .
This places a corresponding *VAR* component on the ADS schematic containing the design variables from Cadence (i.e. Rcc, Rout, Remitin and Remitout).

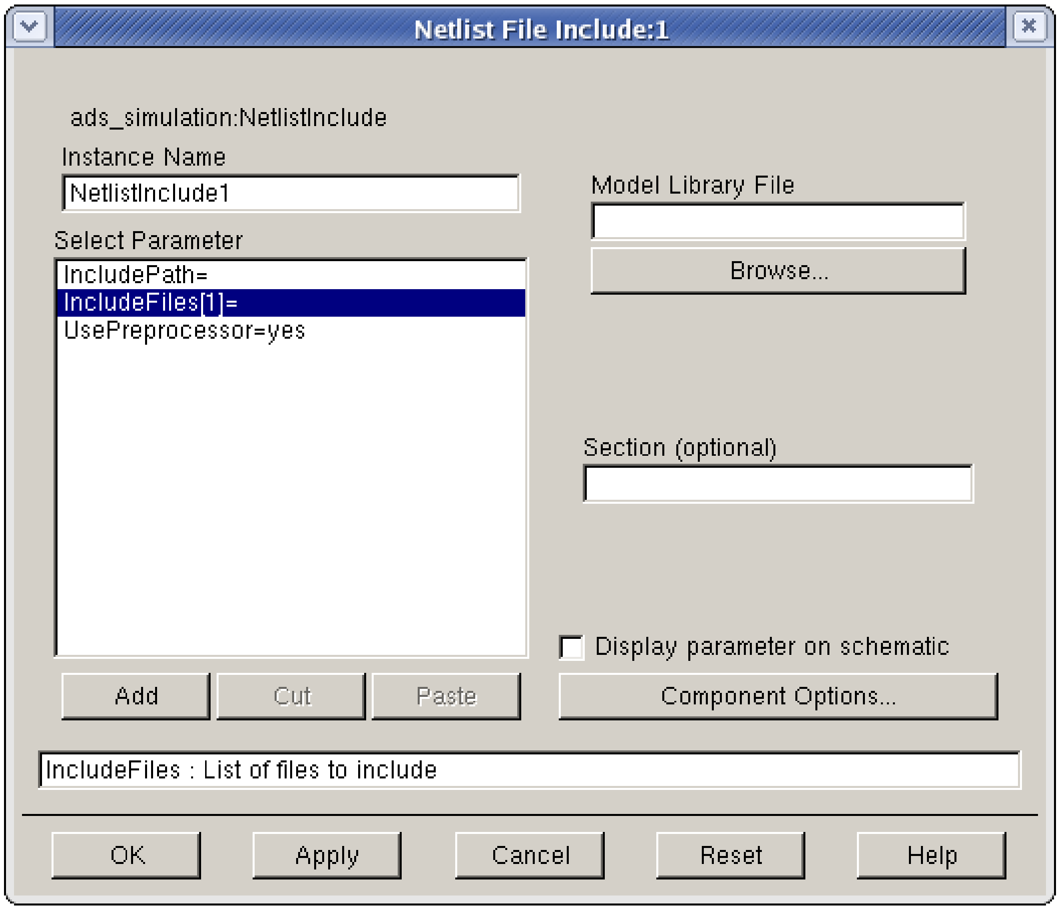


VAR
VAR1
Cseries=7.0pF
Lshunt=7.5nH
Rout=12
Rcc=10
Remitin=200
Remitout=163

Adding Model Files

To add a model file

1. Choose **DynamicLink > Add Netlist File Include**
2. Place the *NetlistInclude* component in an open area on the schematic.
3. Double click the include component icon. The *Netlist File Include* dialog box appears.


Netlist File Include:1

ads_simulation:NetlistInclude

Instance Name: NetlistInclude1

Select Parameter:
IncludePath=
IncludeFiles[1]=
UsePreprocessor=yes

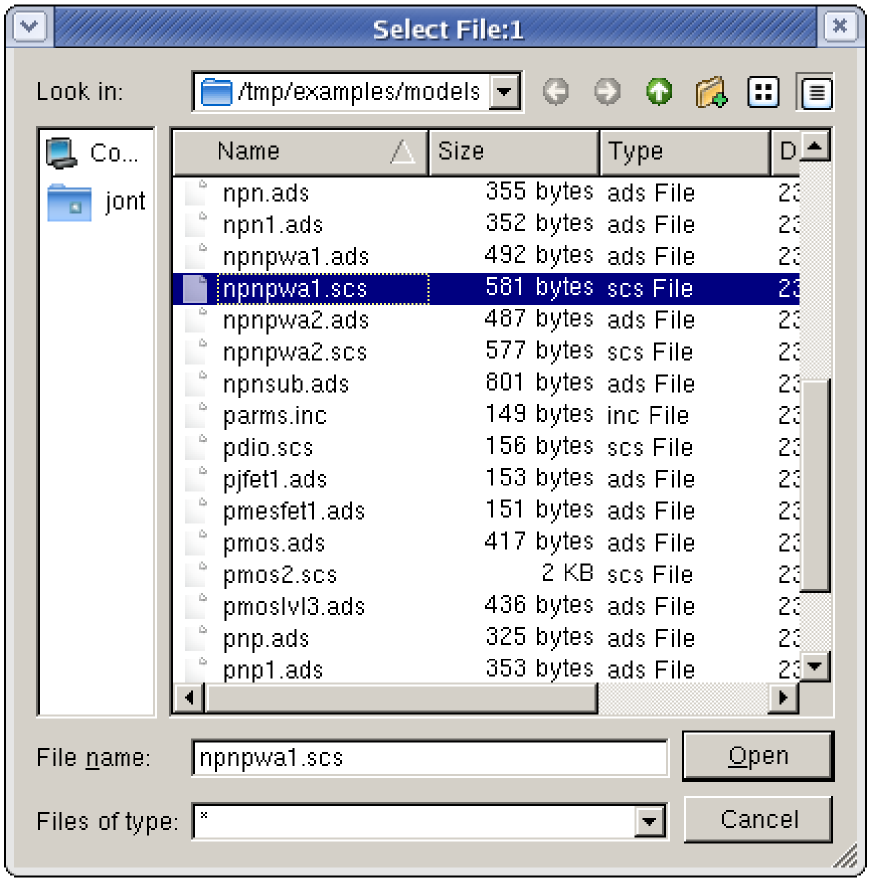
Model Library File:

Section (optional):

☐ Display parameter on schematic

IncludeFiles : List of files to include

4. In the Netlist File Include dialog box, click **IncludeFiles** in the Select Parameter list box. The *Model Library File* field appears in the dialog box.
5. Click **Browse** just below the *Model Library File* field to locate the first model library file. The Select File dialog box appears.

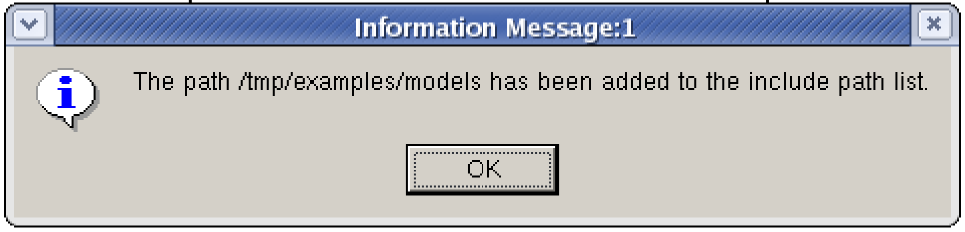


Name	Size	Type	D
npn.ads	355 bytes	ads File	23
npn1.ads	352 bytes	ads File	23
npnpwa1.ads	492 bytes	ads File	23
npnpwa1.scs	581 bytes	scs File	23
npnpwa2.ads	487 bytes	ads File	23
npnpwa2.scs	577 bytes	scs File	23
npnsub.ads	801 bytes	ads File	23
parms.inc	149 bytes	inc File	23
pdio.scs	156 bytes	scs File	23
pjfet1.ads	153 bytes	ads File	23
pmesfet1.ads	151 bytes	ads File	23
pmos.ads	417 bytes	ads File	23
pmos2.scs	2 KB	scs File	23
pmoslvl3.ads	436 bytes	ads File	23
pnp.ads	325 bytes	ads File	23
pnp1.ads	353 bytes	ads File	23

File name: npnpwa1.scs

Files of type: *

6. In the Select File dialog box, use the *Directories* field to locate the *models* directory. *<your_current_working_dir>* /examples/ models
   This sets the path for the location of the model library files.
7. In the Select File dialog box, use the *Files* field to locate and click the **npnpwa1.scs** Cadence spectre model file, then click **OK** . An information message appears stating that a new path has been added to the include path list.



The path /tmp/examples/models has been added to the include path list.

Click **OK** in the *Information Message* dialog box. You are returned to the Netlist File Include dialog box.
8. In the Netlist File Include dialog box, notice that the Model Library File field now contains the *npnpwa1.scs* model file. Click **Apply** to add the *npnpwa1.scs* model file.
9. Click **Browse** again to locate the second model library file. The Select File dialog box appears.
10. In the Select File dialog box, use the *Files* field to locate and click the **npnpwa2.scs** model file, then click **OK** . You are returned to the Netlist File Include dialog box.
11. In the Netlist File Include dialog box, notice that the Model Library File field now contains the *npnpwa2.scs* model file. Click **Add** to add the *npnpwa2.scs* model file.
12. The *Select Parameter* field should now contain the information below.
    IncludeFiles[1]=npnpwa1.scs
    IncludeFiles[2]=npnpwa2.scs.
13. Click **OK** in the Netlist File Include dialog box.
    For more information on the Netlist File Include Component, refer to *Adding Model*

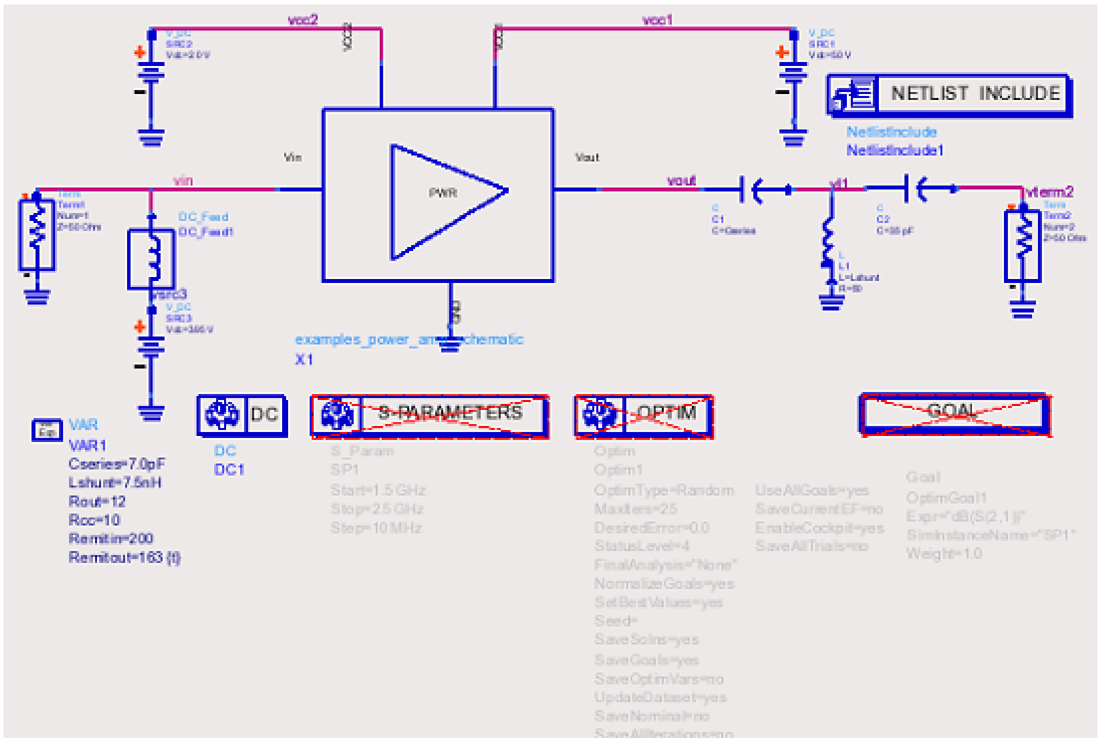
Files (dynlnkug).

Performing a DC Simulation

To run a DC simulation on an ADS schematic and then annotate the results to the Cadence schematic Window:

1. Choose **Edit > Component > Deactivate/Activate** then click the DC component in the ADS Schematic window to toggle and activate the component. Alternatively, you can choose the *Deactivate/Activate Components* icon to activate the DC component.

Figure: Activating Components in an ADS Schematic Window



VAR
VAR1
Cseries=7.0pF
Lshunt=7.5nH
Rout=12
Rcc=10
Remitin=200
Remitout=163 {t}

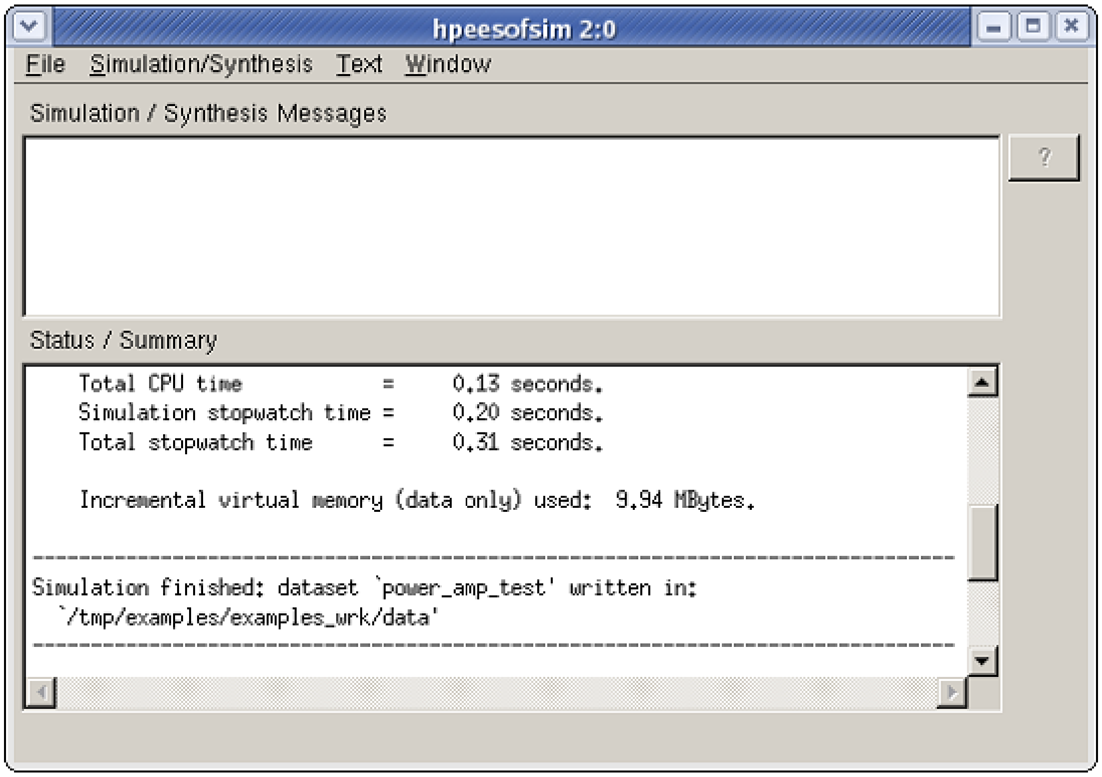
DC
DC1

S_Param
SP1
Start=1.5 GHz
Stop=2.5 GHz
Step=10 MHz

Optim
Optim1
OptimType=Random
MaxIters=25
DesiredError=0.0
StatusLevel=4
FinalAnalysis="None"
NormalizeGoals=yes
SetBestValues=yes
Seed=
SaveSolns=yes
SaveGoals=yes
SaveOptimVars=no
UpdateDataset=yes
SaveNominal=no
SaveAllIterations=no
UseAllGoals=yes
SaveCurrentEF=no
EnableCockpit=yes
SaveAllTrials=no

Goal
OptimGoal1
Expr="dB(S(2,1))"
SimInstanceName="SP1"
Weight=1.0

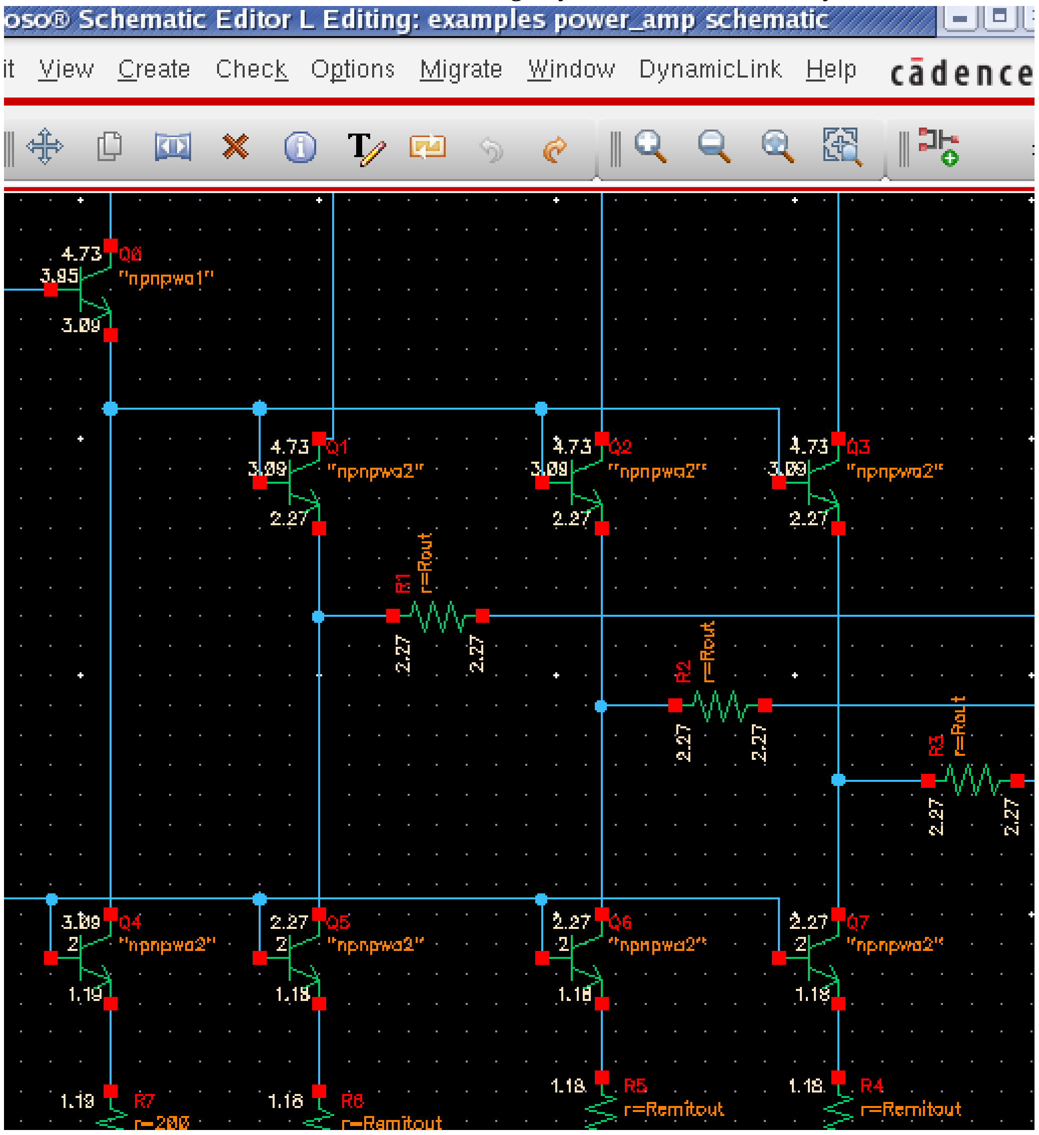
2. Choose **Simulate > Simulate** or choose the *Simulate* icon to run a simulation. A simulation dialog box appears in your display.


hpeesofsim 2:0
File Simulation/Synthesis Text Window
Simulation / Synthesis Messages
?
Status / Summary

```
    Total CPU time             =     0.13 seconds.
    Simulation stopwatch time =     0.20 seconds.
    Total stopwatch time       =     0.31 seconds.

    Incremental virtual memory (data only) used:  9.94 MBytes.

-------------------------------------------------------------------------------
Simulation finished: dataset `power_amp_test' written in:
  `/tmp/examples/examples_wrk/data'
-------------------------------------------------------------------------------
```

3. After the simulation is complete, a Data Display window titled *power_amp_test* automatically appears. Close this window using the **File > Close Window** menu option.
4. Click the *power_amp* schematic symbol in the ADS Schematic window.
5. Choose **DynamicLink > Annotate > Annotate DC Voltages to Selected Cellview** . This displays the DC node voltages on the Cadence schematic.

Figure: Cadence Schematic with Annotated Display

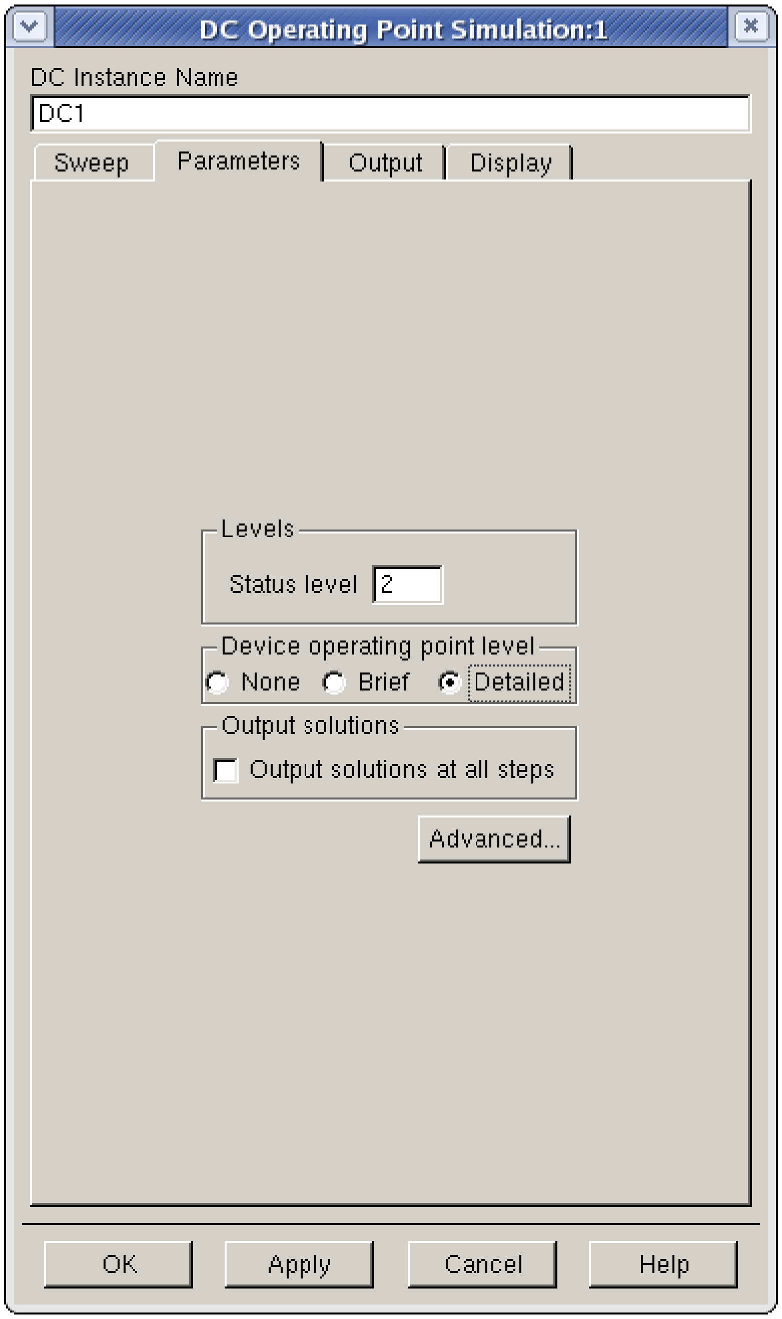


Annotating DC Operating Points to a Selected Cellview

Any simulation that includes a DC analysis produces DC operating point information for most active and some passive devices in the circuit. This data includes currents, power, voltages, and linearized device parameters of the selected device.

To run a DC Operating Point Simulation on an ADS schematic and then annotate the results to the Cadence schematic Window:

1. Double-click the DC Simulation component in the ADS schematic window. The DC Operating Point Simulation dialog box appears.



DC Instance Name: DC1

Levels
Status level: 2

Device operating point level: None Brief [✓ Detailed]

Output solutions
☐ Output solutions at all steps

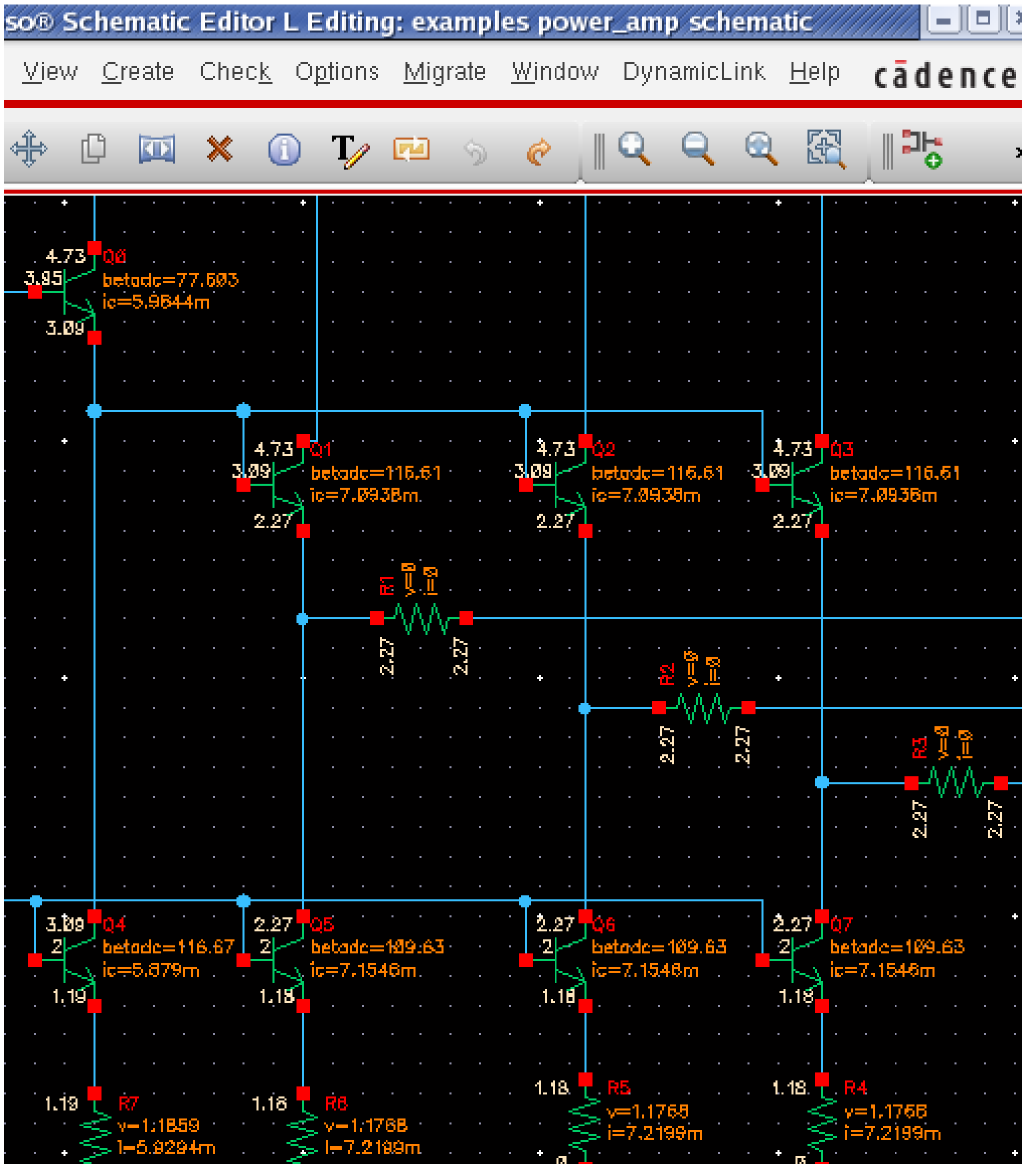
2. Click the *Parameters* tab in the DC Operating Point Simulation dialog box.
3. Click **Detailed** in the *Device operating point level* section of the DC Operating Point

Note
For a subset of the detailed DC Operating Point Simulation information that covers most common parameters, click **Brief** in the *Device operating point level* section of the DC Operating Point Simulation dialog box.

4. Choose **Simulate > Simulate** or choose the *Simulate* icon to run a simulation. A simulation dialog box appears in your display.
5. After the simulation is complete, a Data Display window titled *power_amp_test* automatically appears. Close this window using the **File > Close Window** menu option.
6. Click the *power_amp* schematic symbol in the ADS Schematic window.
7. Choose **DynamicLink > Annotate > Annotate Operating Points to Selected Cellview** . This displays the DC operating point values for each component on the

Cadence schematic.

Figure: Annotated Cellview with DC Operating Points

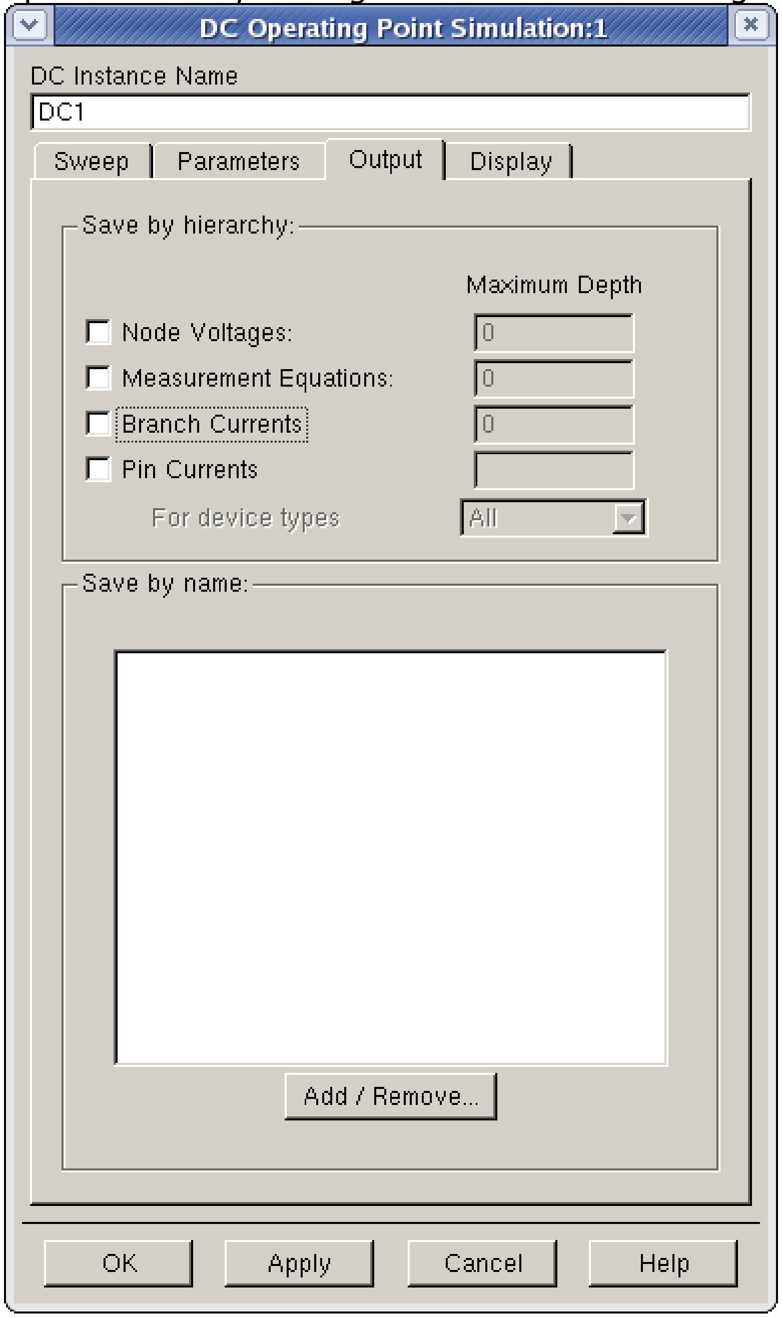

so® Schematic Editor L Editing: examples power_amp schematic
View Create Check Options Migrate Window DynamicLink Help
cādence
Q0 betadc=77.693 ic=5.9644m
Q1 betadc=116.61 ic=7.0938m
Q2 betadc=116.61 ic=7.0938m
Q3 betadc=116.61 ic=7.0938m
Q4 betadc=116.67 ic=5.879m
Q5 betadc=109.63 ic=7.1546m
Q6 betadc=109.63 ic=7.1546m
Q7 betadc=109.63 ic=7.1546m
R1 v=0 i=0
R2 v=0 i=0
R3 v=0 i=0
R7 v=1.1859 i=5.9294m
R6 v=1.1768 i=7.2189m
R5 v=1.1768 i=7.2199m
R4 v=1.1768 i=7.2199m
4.73
3.95
3.09
2.27
2
1.19
1.18

Saving Node Voltages and Pin Currents with RFIC Dynamic Link

With the RFIC Dynamic Link Probing feature, you can select nodes or pins of interest in a Cadence subcircuit and display their voltages or currents in an ADS Data Display window.

The following is an example session using the Dynamic Link Probing feature with the power_amp example.

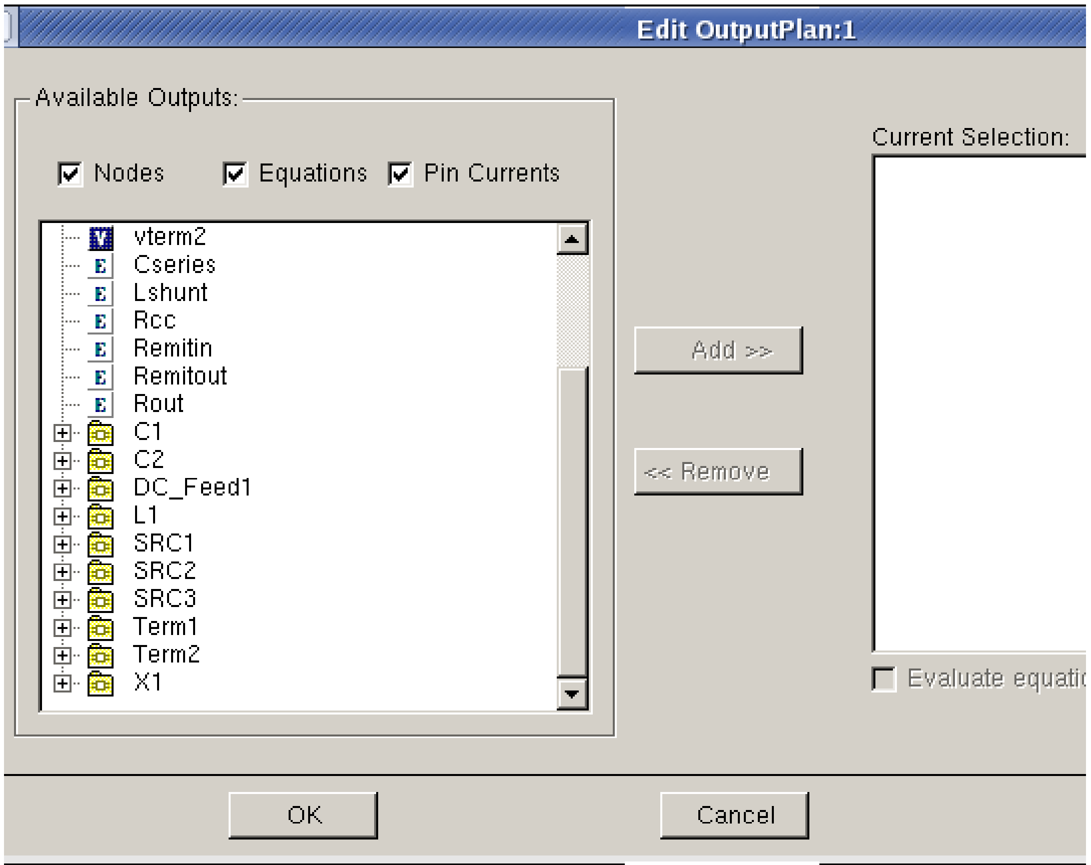
1. Double click the DC Simulation controller component in the ADS schematic window to open the *DC Operating Point Simulation* dialog box.



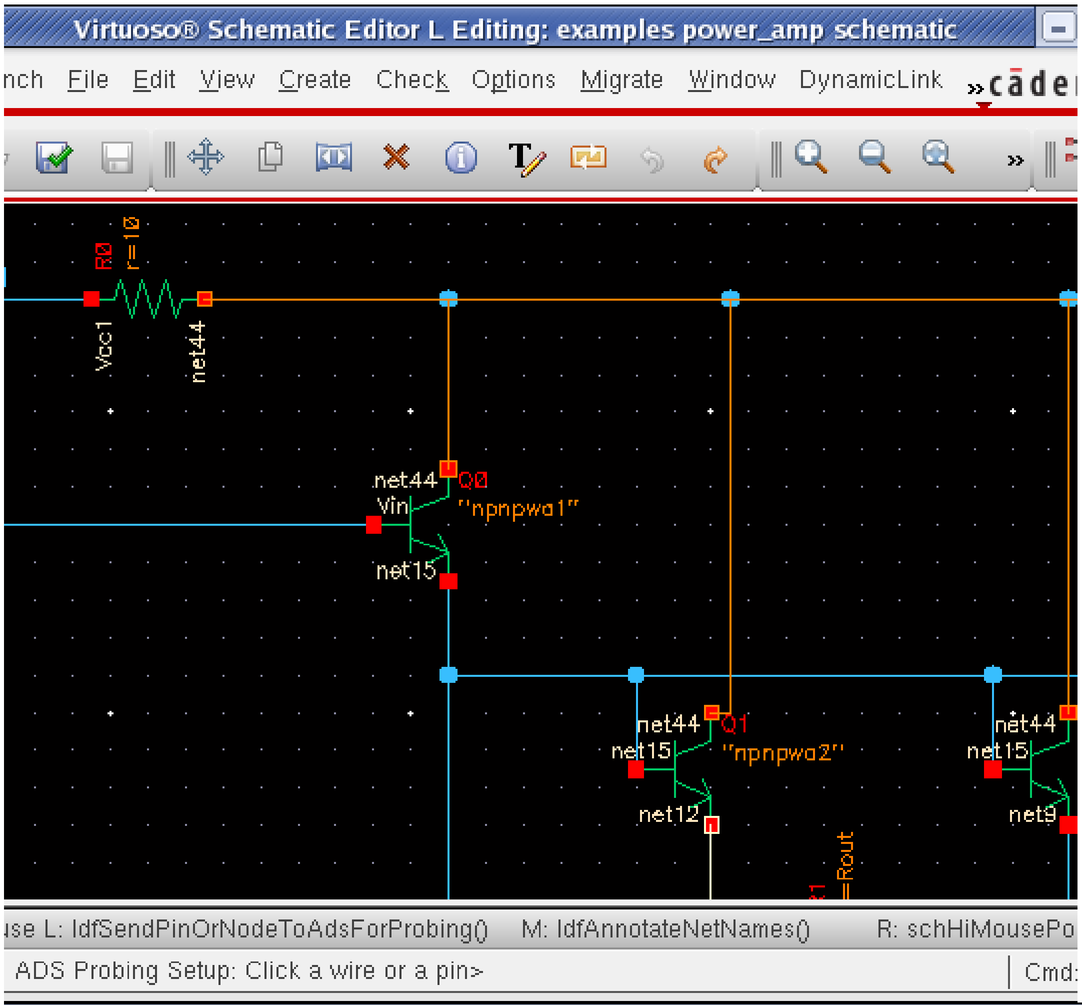
DC1

☐ Node Voltages:	0
☐ Measurement Equations:	0
☐ Branch Currents	0
☐ Pin Currents	
For device types	All

2. Click the **Output** tab in the *DC Operating Point Simulation* dialog box.
3. Click **Add/Remove** in the *Output* section of the *DC Operating Point Simulation* dialog box to open the *Edit OutputPlan*dialog box.

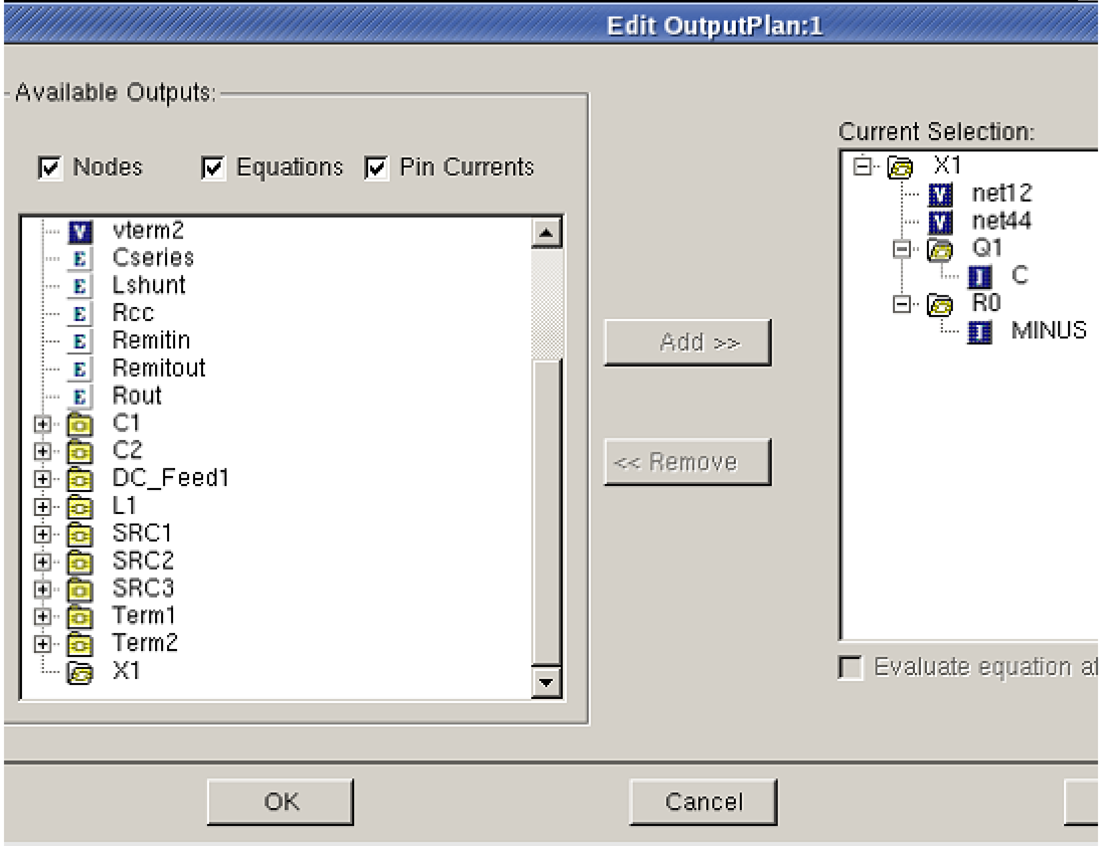


4. Click the **+** sign to the left of the X1 component in the *Available Outputs*field to start Dynamic Link Node Probing setup. This raises the Cadence schematic window containing the power_amp cellview. The prompt in the Cadence schematic window is changed to: **ADS Probing Setup: Click a wire or a pin>.**
5. Click the right pin of the R0 resistor on the upper-left region of the schematic window. "X1.R0.MINUS.pinCurrent.i" appears in the *Current Selection* field on the right side of the ADS *Edit OutputPlan* dialog box.
6. Click the collector pin at the top of the Q1 BJT. "X1.Q1.C.pinCurrent.i" appears in the *Current Selection* field on the right side of the ADS *Edit OutputPlan* dialog box.
7. Click the vertical net connected from above Q1 to the collector pin of the Q1 BJT. "X1.net44" appears in the *Current Selection* field on the right side of the ADS *Edit OutputPlan* dialog box.
8. Click the vertical net connected from below Q1 to the emitter pin of the Q1 BJT. "X1.net12" appears in the *Current Selection* field on the right side of the ADS *Edit OutputPlan* dialog box.


Virtuoso® Schematic Editor L Editing: examples power_amp schematic
nch File Edit View Create Check Options Migrate Window DynamicLink
R0
r=10
Vcc1
net44
net44
Vin
Q0
"npnpwa1"
net15
net44
net15
Q1
"npnpwa2"
net12
net44
net15
net9
=Rout

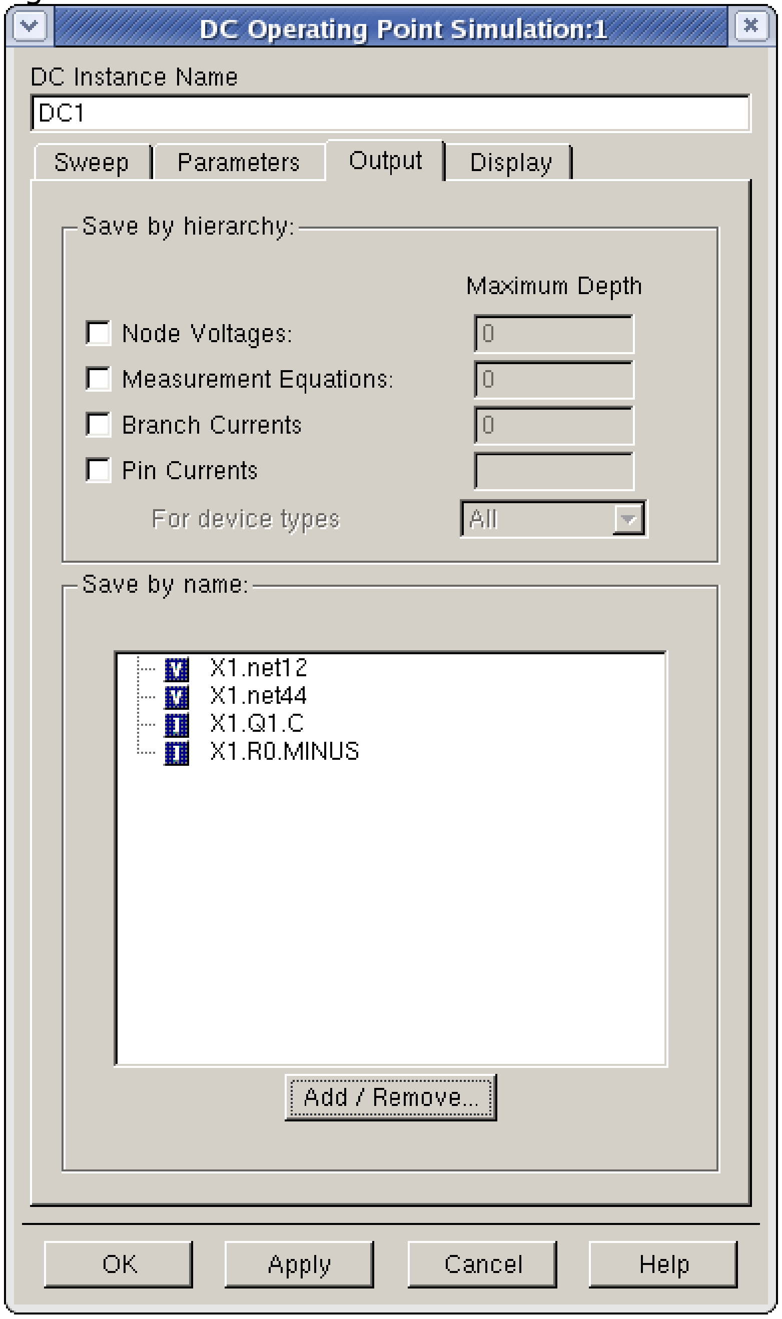
use L: ldfSendPinOrNodeToAdsForProbing() M: ldfAnnotateNetNames() R: schHiMousePo

ADS Probing Setup: Click a wire or a pin>



9. Click **OK** in the ADS *Edit OutputPlan* dialog box to close the dialog box. The *DC*

Operating Point Simulation dialog box should now appear similar to the following figure.



DC Operating Point Simulation:1

DC Instance Name: DC1

Tabs: Sweep | Parameters | Output | Display

Save by hierarchy:

	Maximum Depth
☐ Node Voltages:	0
☐ Measurement Equations:	0
☐ Branch Currents	0
☐ Pin Currents	
For device types	All

Save by name:

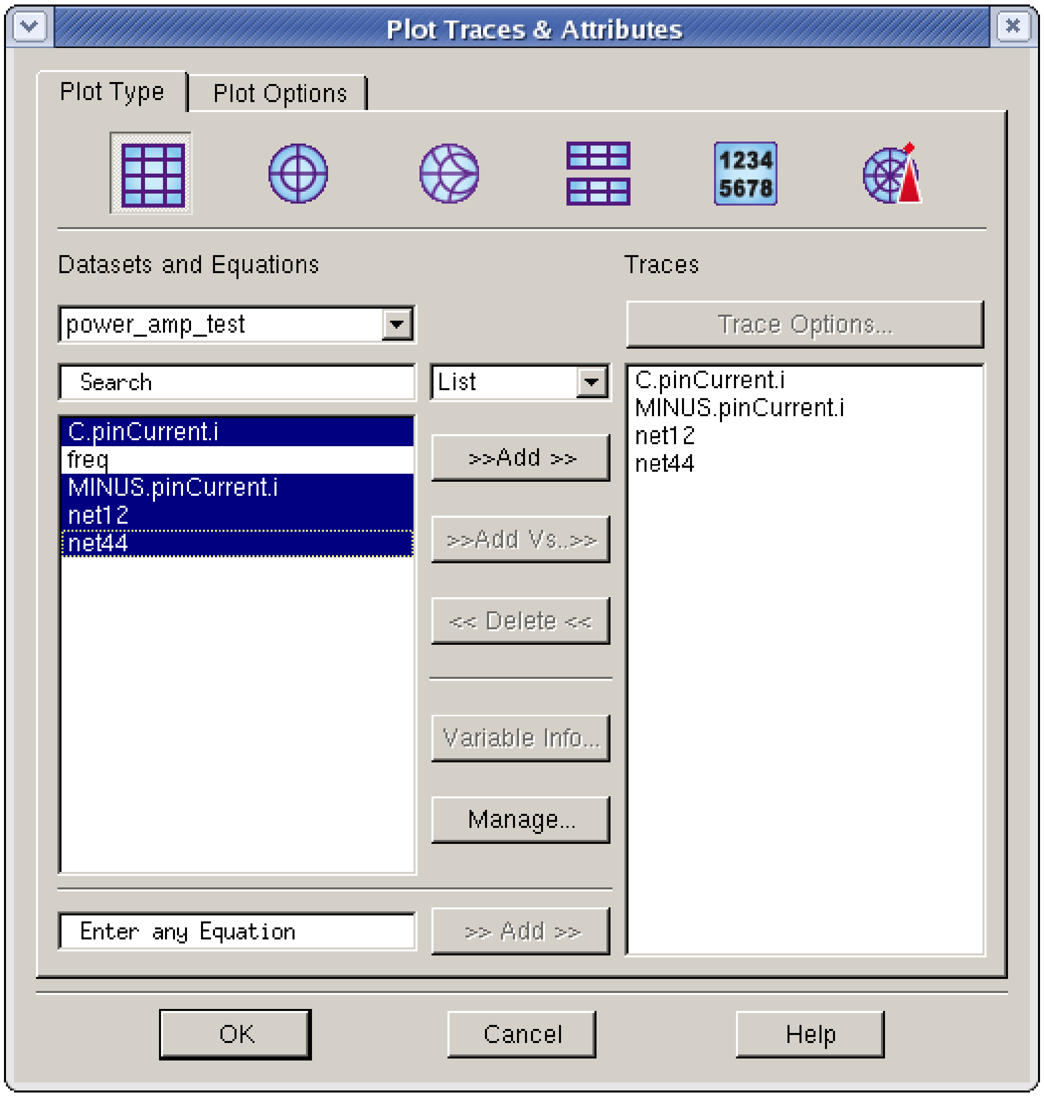
- V X1.net12
- V X1.net44
- I X1.Q1.C
- I X1.R0.MINUS

Add / Remove...

OK | Apply | Cancel | Help

10. Click **OK** in the *DC Operating Point Simulation* dialog box to instruct ADS to save currents for **X1.R0.MINUS** and **X1.Q1.C** in addition to voltage values for **X1.net12** and **X2.net44** after DC simulation.
11. Click the Simulate icon or select **Simulate > Simulate** menu item in the ADS schematic window to run a DC Simulation. The ADS Data Display window appears automatically once the ADS DC Simulation is complete.
12. Click the *List* icon on the left of the Data Display window, then click a desired location in the ADS Data Display window to bring up the *Plot Traces & Attributes* dialog box.
13. In the *Plot Traces & Attributes* dialog box, press the Ctrl key and click the following four items on the left, then click >>**Add**>> in the middle to add these four items to the *Traces* field on the right.

C.pinCurrent.i
MINUS.pinCurrent.i
net12
net44



Datasets and Equations: power_amp_test

Search

List

C.pinCurrent.i
freq
MINUS.pinCurrent.i
net12
net44

Enter any Equation

Traces:
C.pinCurrent.i
MINUS.pinCurrent.i
net12
net44

14. Click **OK** to close the *Plot Traces & Attributes* dialog box. The currents of *R0.MINUS* and *Q1.C* pins and voltages of *net12* and *net44* are immediately listed in the ADS Data Display window.

Additional Notes:

- When the **Add/Remove** button in the *DC Operating Point Simulation* dialog box is clicked to bring up the *Edit OutputPlan* dialog box, as described in step 6 above, all of the named nodes and subcircuit components at current circuit level appear in the *Available Outputs* field on the left. You can select a named node from the *Available Outputs* field and then click **Add>>** to add it to the *Current Selection* field on the right. You can also remove a named node from the *Current Selection* field by clicking **<<Remove** after selecting it.
- Clicking the + sign to the left of a subcircuit component in the *Available Outputs* field displays all the named nodes and subcircuit components within that subcircuit. In

step 7 above, nothing was displayed initially because there was no named node or subcircuit in the power_amp cellview.

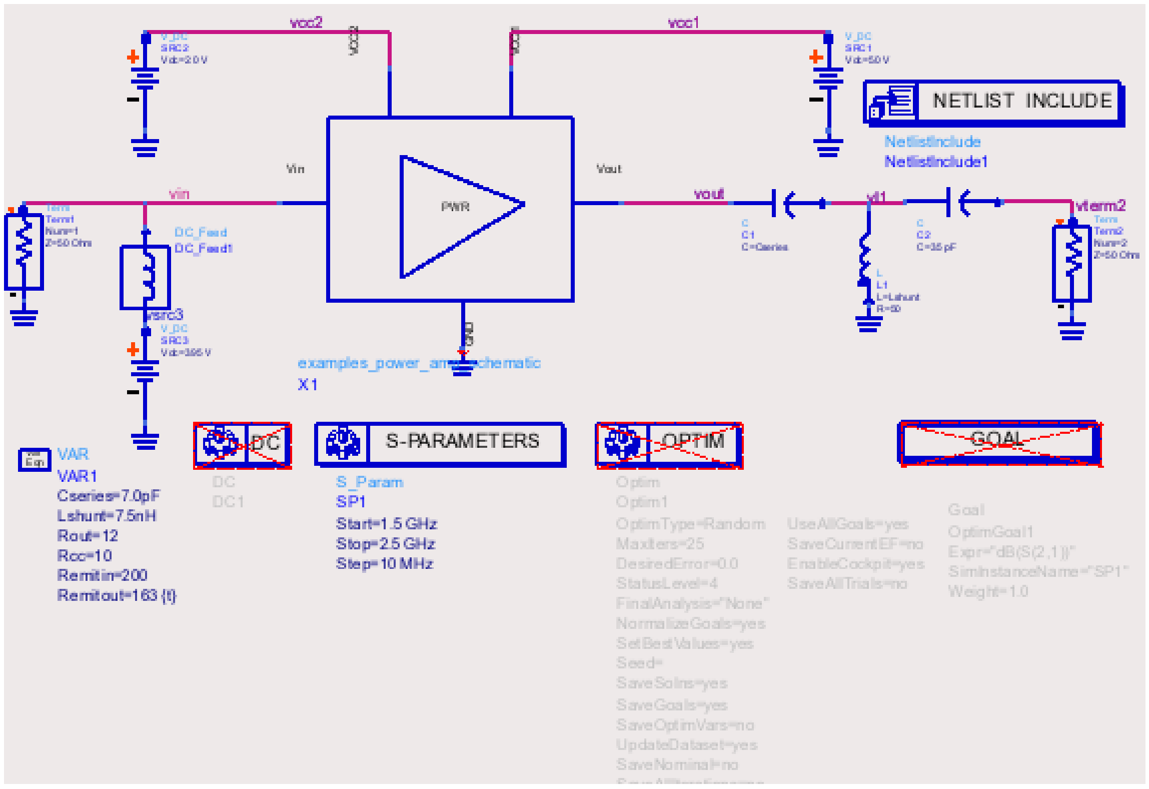
- After a Cadence schematic window is raised as a result of clicking the + sign mentioned above, any named or unnamed node in the Cadence subcircuit hierarchy can be selected by clicking the left mouse button at any point on the wire. To select a node in Cadence subcircuit, choose **Design > Hierarchy > Descend Edit** menu item in the Cadence schematic window, click **OK** in the form popped up, and then click a wire of interest. Valid selections are limited to the current window and the current circuit hierarchy.
- During the Node Probing operation, the left mouse button in the Cadence Virtuoso schematic window is mapped to a Dynamic Link SKILL procedure. Do not bind any function to the left mouse button during this period. Any bindkey function previously mapped to the left mouse button will not work until the tune mode ends.

Performing an S-parameter Simulation

To perform an S-parameter Simulation on an ADS schematic:

1. Deactivate the DC component using the *Deactivate/Activate* toggle.
2. Activate the S-parameter component using the *Deactivate/Activate* toggle.The ADS schematic is now ready to simulate an S-parameter.

Figure: Activating Components in an ADS Schematic Window



VAR
VAR1
Cseries=7.0pF
Lshunt=7.5nH
Rout=12
Rcc=10
Remitin=200
Remitout=163 {t}

DC
DC1

S_Param
SP1
Start=1.5 GHz
Stop=2.5 GHz
Step=10 MHz

Optim
Optim1
OptimType=Random
MaxIters=25
DesiredError=0.0
StatusLevel=4
FinalAnalysis="None"
NormalizeGoals=yes
SetBestValues=yes
Seed=
SaveSolns=yes
SaveGoals=yes
SaveOptimVars=no
UpdateDataset=yes
SaveNominal=no
UseAllGoals=yes
SaveCurrentEF=no
EnableCockpit=yes
SaveAllTrials=no

Goal
OptimGoal1
Expr="dB(S(2,1))"
SimInstanceName="SP1"
Weight=1.0

3. Choose **Simulate > Simulate** or choose the *Simulate* icon to run the S-parameter

simulation.

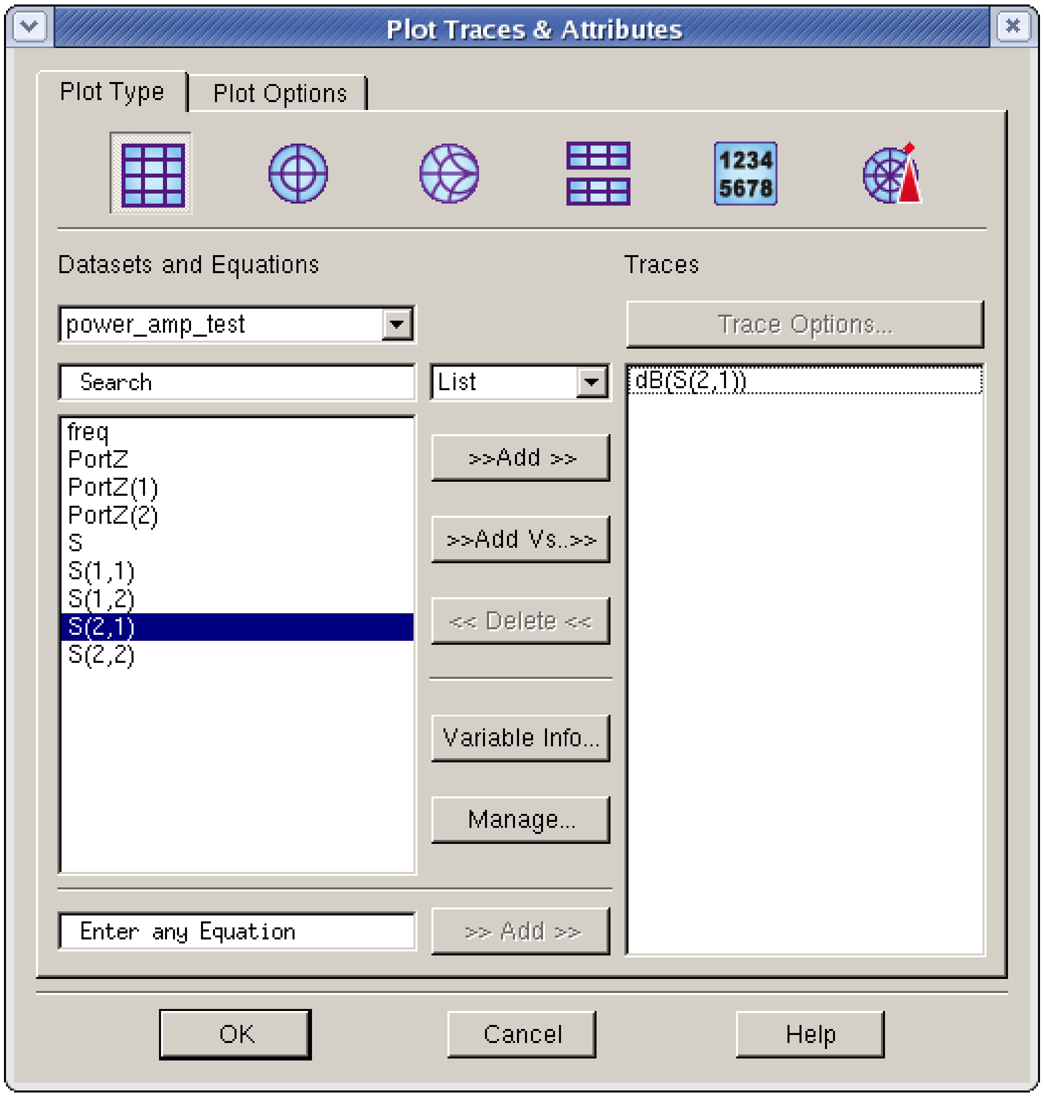
After the simulation is complete, a Data Display window titled *power_amp_test* automatically appears.

Note
The amplifier used in this tutorial is not a traditional microwave amplifier for use by itself. It is meant to be used as part of an RFIC, where it takes a small voltage signal and converts it into a current for driving a larger-signal amplifier, off chip. It has not been designed to have S_{11} and S_{22} less than 1. The example described in this document is only illustrating the RFIC Dynamic Link for Cadence.

Displaying Your Results

To view the results of your simulation in a plotted Data Display:

1. Select *power_amp_test* from the *Default Dataset* drop-down list if not already selected.
2. From the Data Display window, choose **Insert > Plot** , move the frame to an appropriate location within the window and click. This anchors a frame for your plot. Similarly, you can choose the *Rectangular Plot* icon to drag and drop the plot frame.
3. The *Plot Traces & Attributes* dialog box appears. Select *S(2,1)* and then click **Add** . Select *dB* from the dialog box then click **OK** .



Dataset: power_amp_test

Search

List

freq
PortZ
PortZ(1)
PortZ(2)
S
S(1,1)
S(1,2)
S(2,1)
S(2,2)

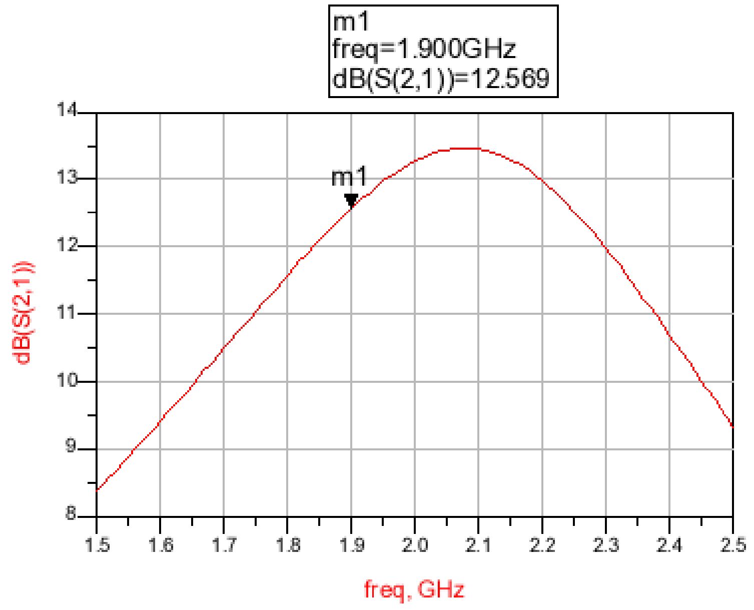
Enter any Equation

dB(S(2,1))

4. Click **OK** again to view the *Data Display* .

Note
The following figure shows the forward gain, S(2,1), at 1.9 GHz to be approximately 12.569 dB. By varying the value of *Remitout* in this case, you can modify the circuit to achieve the desired results. After you have completed the tutorial, take some time to experiment with different values of *Remitout* .

Figure: Data Display with S-parameter Simulation Results

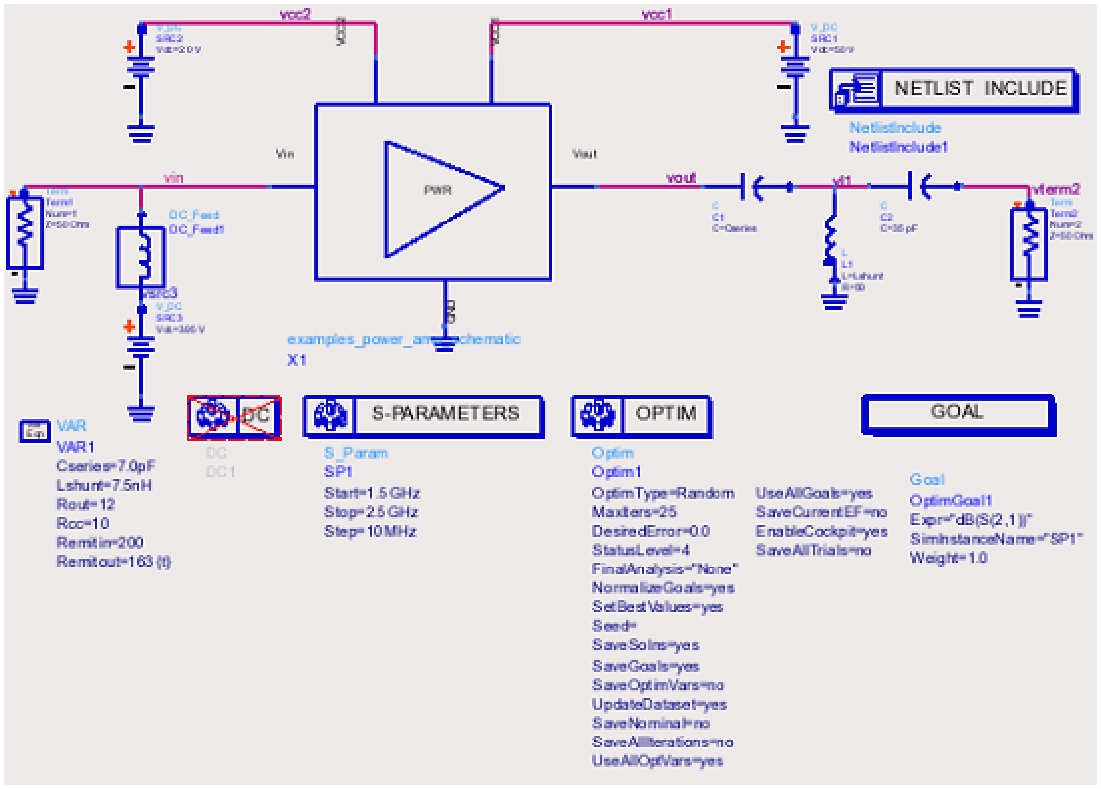


Performing a Parameter Optimization

To optimize the parameters of *Var1* in the ADS schematic:

1. Choose the *Deactivate/Activate Components* icon then place the cross hair over the *Nominal Optimization* component to toggle and activate the component. Repeat this step for the *Goal* component. The ADS schematic is now ready to optimize the S-parameter.

Figure: Activating Components in an ADS Schematic Window



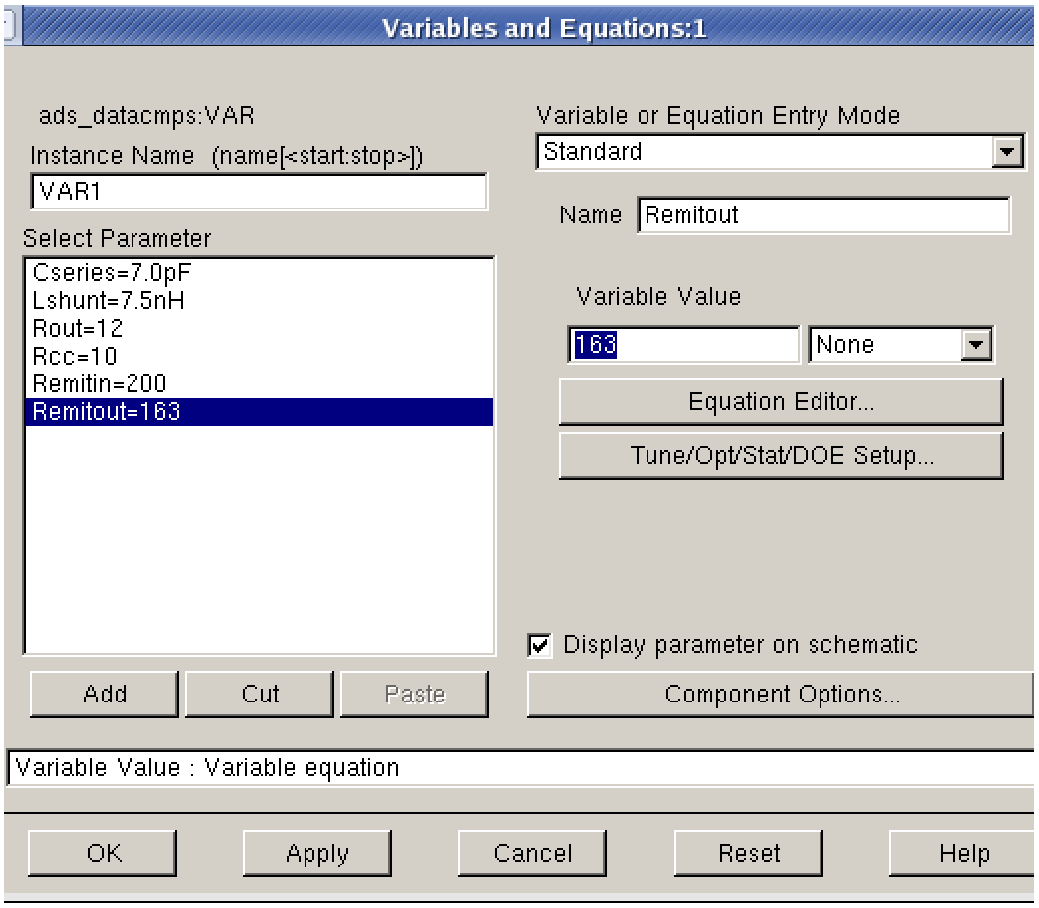
VAR
VAR1
Cseries=7.0pF
Lshunt=7.5nH
Rout=12
Rcc=10
Remitin=200
Remitout=163 {t}

S_Param
SP1
Start=1.5 GHz
Stop=2.5 GHz
Step=10 MHz

Optim
Optim1
OptimType=Random
MaxIters=25
DesiredError=0.0
StatusLevel=4
FinalAnalysis="None"
NormalizeGoals=yes
SetBestValues=yes
Seed=
SaveSolns=yes
SaveGoals=yes
SaveOptimVars=no
UpdateDataset=yes
SaveNominal=no
SaveAllIterations=no
UseAllOptVars=yes
UseAllGoals=yes
SaveCurrentEF=no
EnableCockpit=yes
SaveAllTrials=no

Goal
OptimGoal1
Expr="dB(S(2,1))"
SimInstanceName="SP1"
Weight=1.0

2. Choose **Edit > Component > Edit Component Parameters** and then click the VAR component. The *Variables and Equations* dialog box appears.
3. In the *Variables and Equations* dialog box, select the parameter *Remitout* in the *Select Parameter* field.



ads_datacmps:VAR

Instance Name (name[<start:stop>])
VAR1

Select Parameter
Cseries=7.0pF
Lshunt=7.5nH
Rout=12
Rcc=10
Remitin=200
Remitout=163

Variable or Equation Entry Mode
Standard

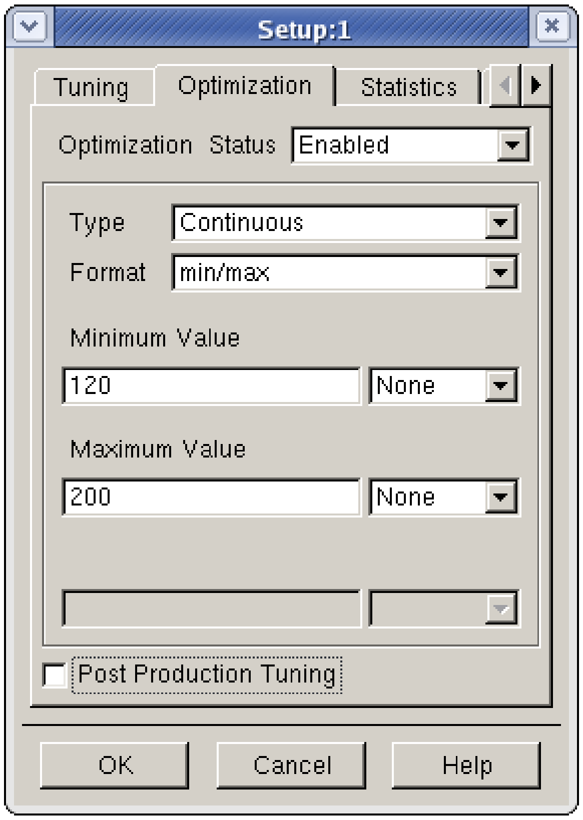
Name Remitout

Variable Value
163 None

[✓] Display parameter on schematic

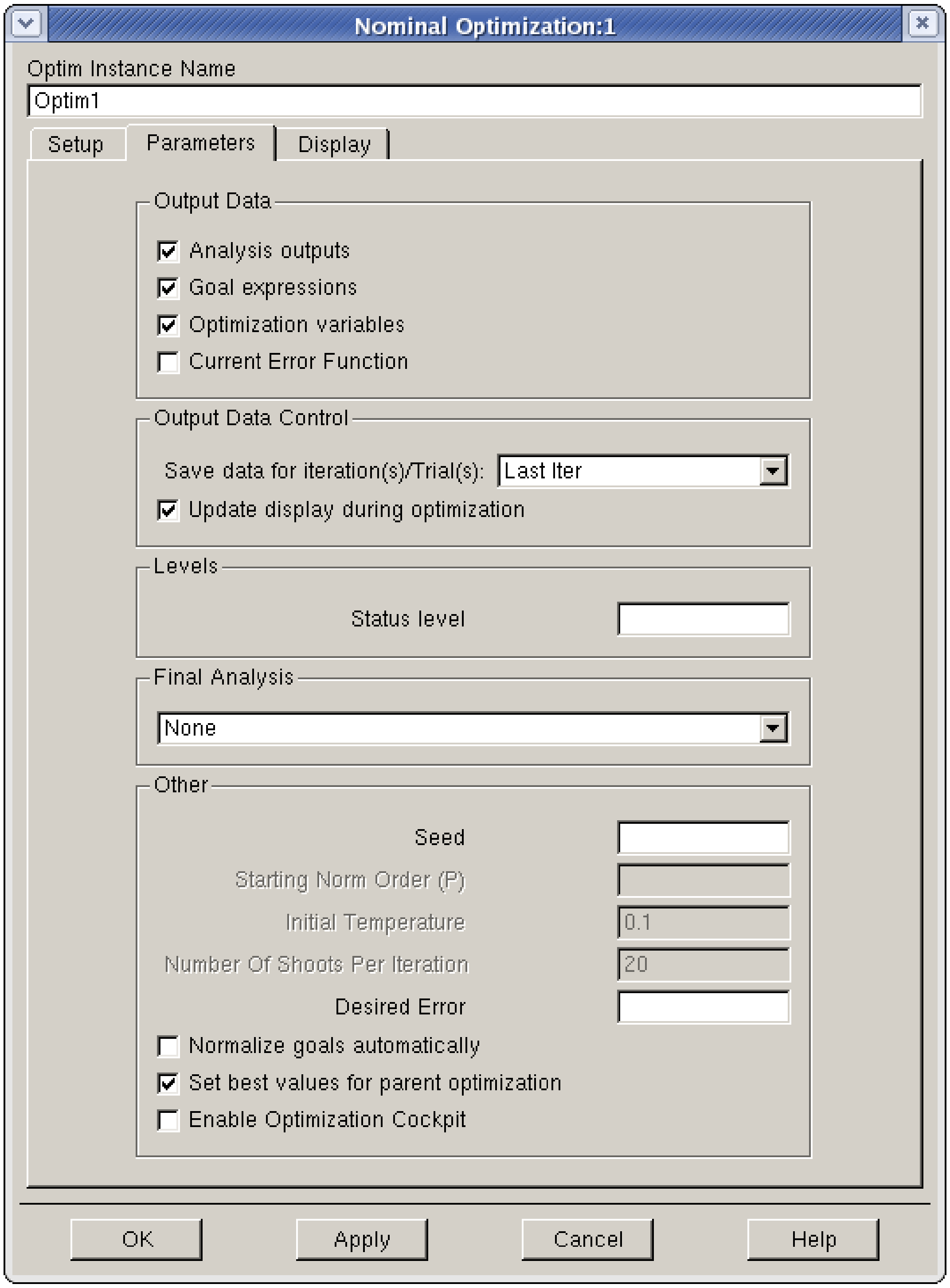
Variable Value : Variable equation

4. Click the **Tune/Opt/Stat/DOE Setup** button. The Setup dialog box appears.
5. Click the **Optimization** tab and set the **Optimization status** to *Enabled* .
6. Set the **Minimum Value** to *120* and the **Maximum Value** to *200* .



Optimization Status: Enabled
Type: Continuous
Format: min/max
Minimum Value: 120 None
Maximum Value: 200 None
Post Production Tuning

7. Click **OK** twice, once in the *Setup* dialog box and once in the *Variables and Equations* dialog box. Note that the *Remitout* parameter on the ADS schematic now displays: Remitout= 163 opt{120 to 200}
8. Click the OPTIM optimization controller and then select *Last Iter* in the **Save data for iteration(s)/Trial(s):** field and uncheck the **Enable Optimization Cockpit** in the *Nominal Opitimization* dialog box.


Nominal Optimization:1

Optim Instance Name: Optim1

Output Data
- [✓] Analysis outputs
- [✓] Goal expressions
- [✓] Optimization variables
- [ ] Current Error Function

Output Data Control
- Save data for iteration(s)/Trial(s): Last Iter
- [✓] Update display during optimization

Levels
- Status level:

Final Analysis
- None

Other
- Seed:
- Starting Norm Order (P):
- Initial Temperature: 0.1
- Number Of Shoots Per Iteration: 20
- Desired Error:
- [ ] Normalize goals automatically
- [✓] Set best values for parent optimization
- [ ] Enable Optimization Cockpit

9. Choose **Simulate > Optimize** or choose the *Optimize* icon. This netlists each Cadence subcircuit in the *Virtuoso Analog Design Environment* , as well as the top-level ADS schematic, and starts the Advanced Design System simulator.

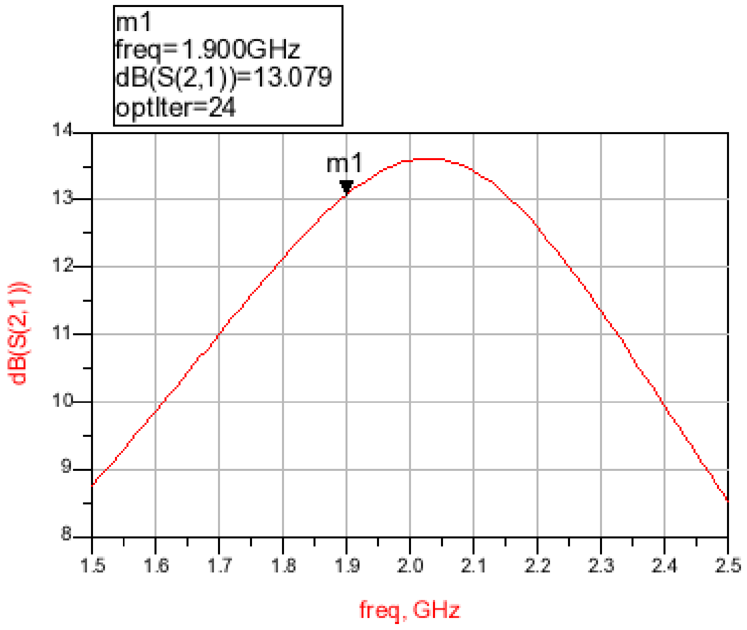
Note
A simulation status window appears, reporting the status of the simulation; depending on your system, this may take some time. Check this status window to see if any errors occurred during netlisting or simulation.

After the simulation is complete, a Data Display window titled *power_amp_test* automatically appears. For information on configuring the Data Display, refer to Displaying

Your Results.

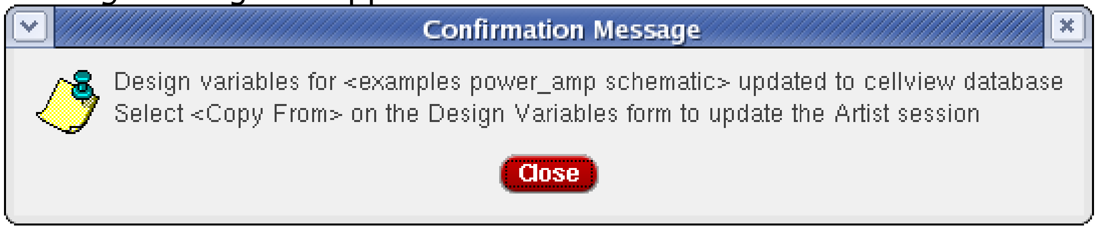
Note
The following figure shows the optimized forward gain, S(2,1), at 1.9 GHz to be approximately 13.08dB as set by the *Goal* component in the ADS Schematic window.

Figure: Data Display with Optimized Simulation Results



Updating Cadence Design Variables

1. Choose **Simulate > Update Optimization Values** to update the optimized values. This changes the value of *Remitout* in the *VAR* component to the optimized value.
2. Choose **DynamicLink > Design Variables > Update Design Variables to Cellviews** to update the optimized value to the Cadence cellview. A *Confirmation Message* dialog box appears.

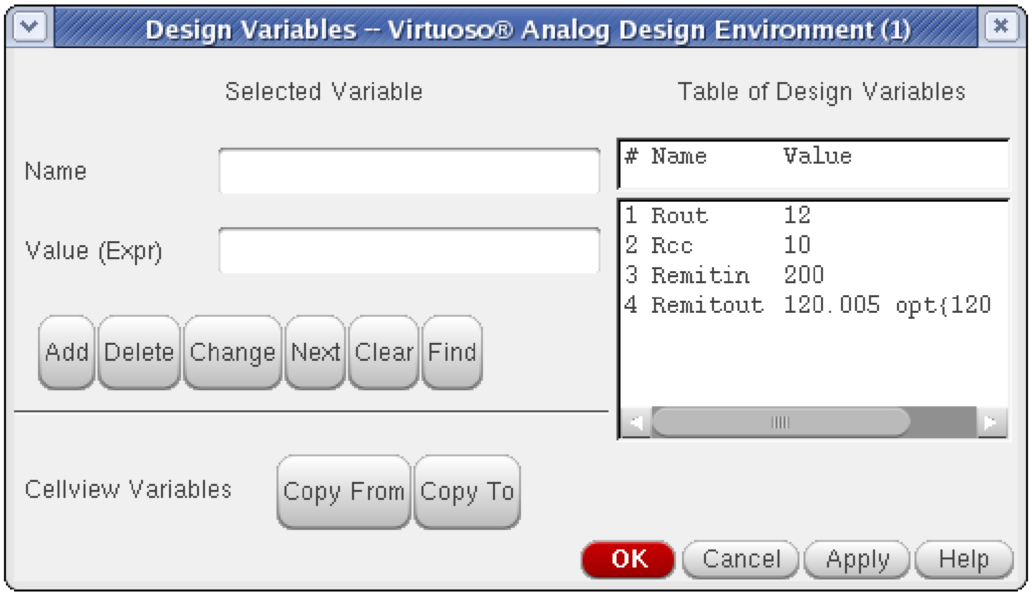


Design variables for <examples power_amp schematic> updated to cellview database
Select <Copy From> on the Design Variables form to update the Artist session

3. Click the **Close** button.
4. From the Cadence menu bar, choose **DynamicLink > Design Variables** . A *Design Variables* form appears.

Note
If the *DynamicLink* pull-down menu does not appear in the Cadence Virtuoso Schematic window, choose **Launch > ADS Dynamic Link** in the Cadence schematic window.

5. Click **Copy From** in the *Cellview Variables* section. This enables you to update the design variables to the OASIS session.



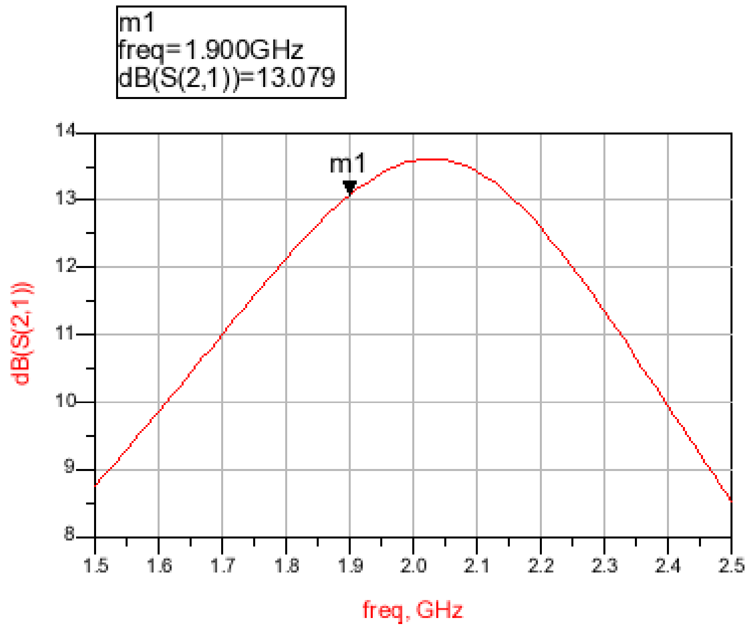
```
# Name      Value
1 Rout      12
2 Rcc       10
3 Remitin   200
4 Remitout  120.005 opt(120
```

Verifying Your Results

You may now verify the results of your optimization by using the optimized value of *Remitout* in an S-parameter simulation . Set the value of Remitout = 120 (approximate optimum value) and repeat the steps in *Performing an S-parameter Simulation* with the new value of *Remitout.*

Note
The following figure again shows the optimized forward gain, S(2,1), at 1.9 GHz to be approximately 13.08 dB as set by the *Goal* component in the ADS Schematic window. By using the optimized value of *Remitout* , you have verified the optimum desired results of the circuit.

Figure: Data Display with Optimized Simulation Results



Ending the Session

Use the following steps to exit the Dynamic Link environment and close both ADS and Cadence.

1. Choose the Advanced Design System Schematic menu option **DynamicLink > Close Connection** . This closes ADS and terminates the link between Cadence and the Advanced Design System.
2. Exit from the Cadence CIW.
3. Congratulations... you have now successfully completed the Tutorial.

Glossary for RFIC Dynamic Link

ADS (Advanced Design System)

Advanced Design System is an EDA System for high-frequency circuit and system design.

AEL (Cadence Analog Expression Language)

In the Cadence context, AEL is the syntax and API (available in Skill or C) to support full or partial expression evaluation for repetitious circuit simulation.

AEL (ADS Application Extension Language)

This is a C-like interpretive programming language to configure, customize and enhance the Advanced Design System design environment.

bindkeys

Settings used to map individual keystrokes to a particular function within the software.

callback

A function or expression that gets evaluated when certain events occur; for example, clicking on a menu item.

CDF (Component Description Format)

The CDF is Cadence's mechanism to interactively define and evaluate parameters and attributes for individual components and designs.

CIW (Command Interpreter Window)

The CIW is Cadence's command window.

colormap

Indexed color table where each entry is a combination of R, G, and B pixel intensity values for UNIX X-windows display. Table size (number of colors) per software application is limited by the number of display bits per pixel, commonly eight.

DFII (Design Framework II)

Cadence's overall IC design environment.

EDA (Electronic Design Automation)

Software and services that give customers a distinct advantage by improving time-to-market, quality and productivity in the design of electronic products.

GUI (Graphical User Interface)

The interface between the user and the application.

HB (Harmonic Balance Simulation)

An iterative method of analysis that is based on the assumption that for a given sinusoidal excitation, there exists a steady-state solution that can be approximated to satisfactory accuracy using a finite Fourier series.

iPar()

The function used in an AEL expression for a parameter which is a function of another parameter of the same instance. For example, for MOSFET instances we might use AD=iPar("w")*5u.

IPC (Inter-Process Communication)

The protocol for passing messages between two or more processes.

OASIS

Open Analog Simulation Integration Socket. The procedural interface for simulator integration into the Cadence simulation environment.

optimization

Mechanism by which a simulator finds the optimal value of a global parameter within a user-supplied range of values.

OS (Operating system)

Such as HP-UX, Solaris, or Win2000.

pPar()

The function used in a Cadence AEL expression for a parameter which is a function of some parameter of the parent instance. For example, for CMOS inverters we might use W=pPar(wp) (where *wp* is a parent instance parameter) on one of the pull-up FETs, enabling use of the same inverter symbol for different size inverters.

PSF

Parameter Storage Format. This is a Cadence-defined file format for storing complex structured data.

Ptolemy

A design environment that supports simultaneous mixtures of different computation models. Ptolemy, named after the second-century Greek astronomer, mathematician, and geographer, was developed at the University of California at Berkeley. For detailed information, refer to the *ADS Ptolemy Simulation* documentation.

RFIC Dynamic Link (Dynamic Link)

RFIC Dynamic Link (Dynamic Link) for Cadence is an EDA framework integration software product. The product enables both *tops-down* and *bottoms-up* design and simulation in *Advanced Design System(ADS)* using IC designs from the Cadence database. RFIC Dynamic Link is based on IPC rather than data file translation maximizing data integrity and ease of use.

SKILL

Cadence's C/lisp-like interpretive programming language for framework and database integration.

testbench

Top-level schematic used to analyze a sub-circuit using a circuit simulator.

tuning

Mechanism by which a simulator can quickly re-simulate a circuit using new values for a number of parameters without having to re-input the netlist and recreate its data structures.

Virtuoso Analog Design Environment

Cadence's analog design and simulation environment for the Virtuoso custom design platform. It is a task-based environment for simulating and analyzing full-custom, analog, and RF IC designs.

Netlisting, Simulating, and Displaying Data

This section describes the procedures for netlisting and simulating a design as well as viewing the netlist from either Advanced Design System or a Cadence Schematic window. Information on net, instance and expression name mapping is also provided.

Netlisting and Simulating a Design

Netlisting automatically occurs when you simulate your schematic in Advanced Design System. The complete netlist is sent to the simulator, stored in memory, and written to a netlist.log file in the Workspace directory of ADS. You can view the netlist in the *netlist.log* file as needed.

To netlist and simulate a schematic in Advanced Design System:

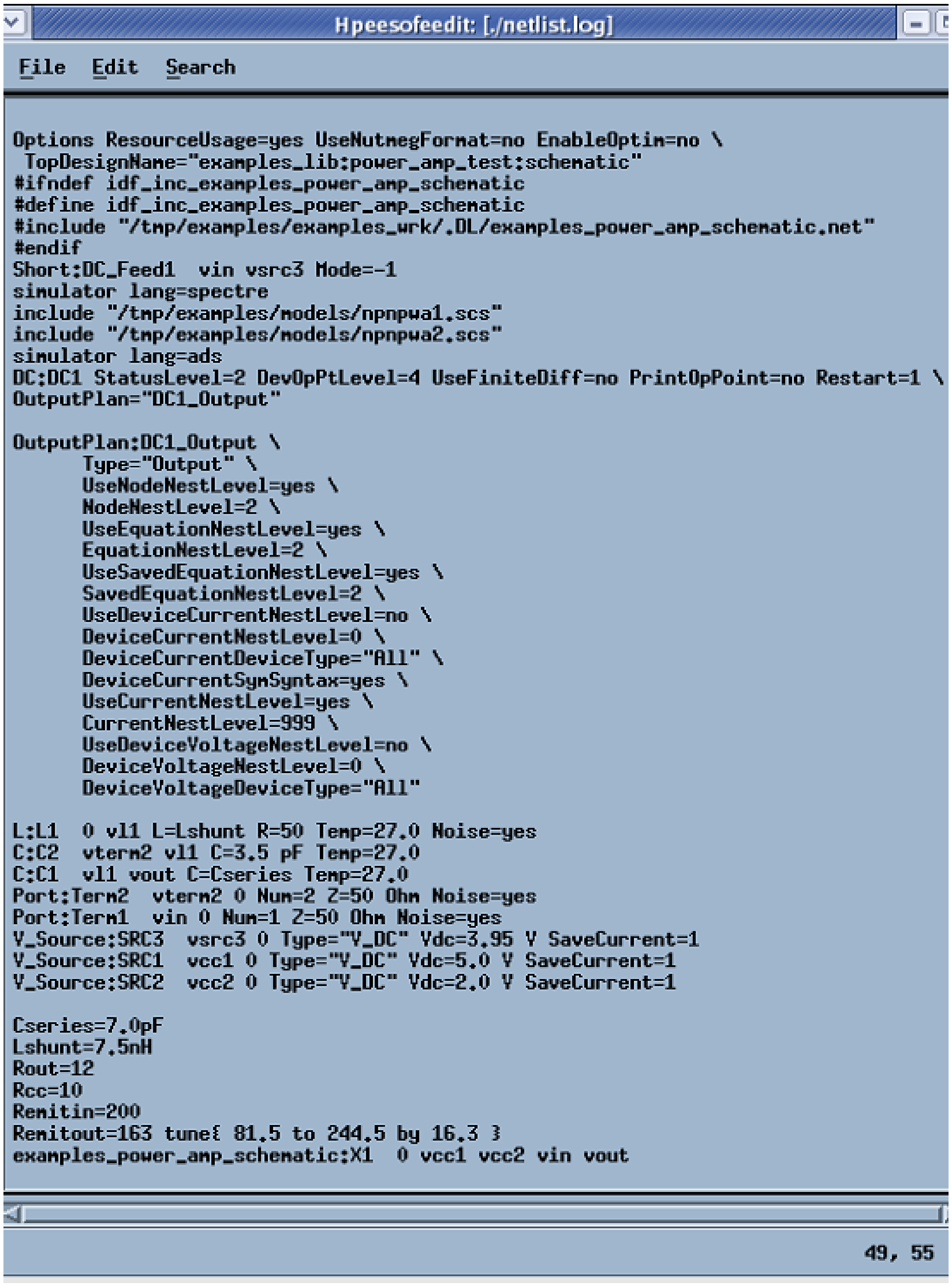
1. In the ADS Schematic window, choose the *Simulate* icon or choose the menu item **Simulate > Simulate** .
   A Simulation window appears, indicating the netlisting status and listing any errors encountered. If the netlisting is successful, the design is then simulated; otherwise, act on the errors displayed in the Simulation window and repeat step one above.
2. For information on configuring and viewing the simulation results in the Data Display window, refer to *Data Display Basics* (data) in the *Data Display* (data) documentation.

Viewing Netlists

This section describes how to view the top-level netlist from Advanced Design System as well as how to view an ADS subnetwork netlist for a Cadence design from a Cadence Schematic window.

Viewing Netlists from Advanced Design System

To generate and display the entire top-level ADS netlist, select **DynamicLink > Top-level Design Netlist** menu item in the ADS Schematic window.



```
Options ResourceUsage=yes UseNutmegFormat=no EnableOptim=no \
 TopDesignName="examples_lib:power_amp_test:schematic"
#ifndef idf_inc_examples_power_amp_schematic
#define idf_inc_examples_power_amp_schematic
#include "/tmp/examples/examples_wrk/.DL/examples_power_amp_schematic.net"
#endif
Short:DC_Feed1  vin vsrc3 Mode=-1
simulator lang=spectre
include "/tmp/examples/models/npnpwa1.scs"
include "/tmp/examples/models/npnpwa2.scs"
simulator lang=ads
DC:DC1 StatusLevel=2 DevOpPtLevel=4 UseFiniteDiff=no PrintOpPoint=no Restart=1 \
OutputPlan="DC1_Output"

OutputPlan:DC1_Output \
      Type="Output" \
      UseNodeNestLevel=yes \
      NodeNestLevel=2 \
      UseEquationNestLevel=yes \
      EquationNestLevel=2 \
      UseSavedEquationNestLevel=yes \
      SavedEquationNestLevel=2 \
      UseDeviceCurrentNestLevel=no \
      DeviceCurrentNestLevel=0 \
      DeviceCurrentDeviceType="All" \
      DeviceCurrentSymSyntax=yes \
      UseCurrentNestLevel=yes \
      CurrentNestLevel=999 \
      UseDeviceVoltageNestLevel=no \
      DeviceVoltageNestLevel=0 \
      DeviceVoltageDeviceType="All"

L:L1  0 vl1 L=Lshunt R=50 Temp=27.0 Noise=yes
C:C2  vterm2 vl1 C=3.5 pF Temp=27.0
C:C1  vl1 vout C=Cseries Temp=27.0
Port:Term2  vterm2 0 Num=2 Z=50 Ohm Noise=yes
Port:Term1  vin 0 Num=1 Z=50 Ohm Noise=yes
V_Source:SRC3  vsrc3 0 Type="V_DC" Vdc=3.95 V SaveCurrent=1
V_Source:SRC1  vcc1 0 Type="V_DC" Vdc=5.0 V SaveCurrent=1
V_Source:SRC2  vcc2 0 Type="V_DC" Vdc=2.0 V SaveCurrent=1

Cseries=7.0pF
Lshunt=7.5nH
Rout=12
Rcc=10
Remitin=200
Remitout=163 tune{ 81.5 to 244.5 by 16.3 }
examples_power_amp_schematic:X1  0 vcc1 vcc2 vin vout
```

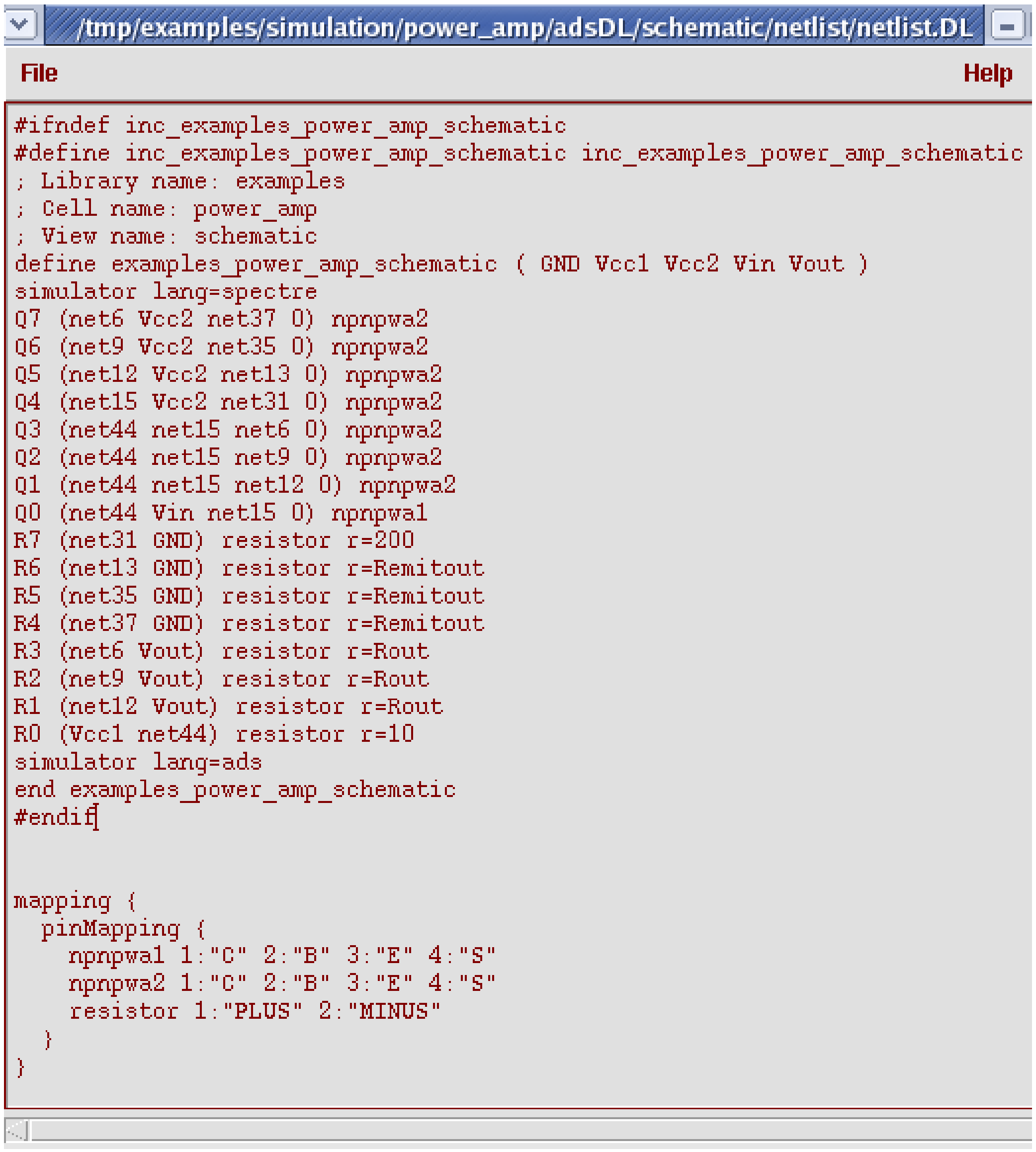
Viewing Top-Level ADS Netlist

Viewing Netlists from the Cadence Schematic Window

To generate and display the ADS subnetwork netlist for the Cadence design displayed in a particular Cadence Schematic window:

1. From the Cadence Schematic window menu bar, select **DynamicLink > Subcircuit Netlist**. Netlisting progress is displayed in the Cadence CIW.

   Note
   If the *Dynamic Link* pull-down menu does not appear in the Cadence Virtuoso Schematic window, choose **Tools > ADS Dynamic Link > Add Dynamic Link menu to all schematic windows** in the Cadence Command Interpreter Window (CIW).


/tmp/examples/simulation/power_amp/adsDL/schematic/netlist/netlist.DL
File
Help

```
#ifndef inc_examples_power_amp_schematic
#define inc_examples_power_amp_schematic inc_examples_power_amp_schematic
; Library name: examples
; Cell name: power_amp
; View name: schematic
define examples_power_amp_schematic ( GND Vcc1 Vcc2 Vin Vout )
simulator lang=spectre
Q7 (net6 Vcc2 net37 0) npnpwa2
Q6 (net9 Vcc2 net35 0) npnpwa2
Q5 (net12 Vcc2 net13 0) npnpwa2
Q4 (net15 Vcc2 net31 0) npnpwa2
Q3 (net44 net15 net6 0) npnpwa2
Q2 (net44 net15 net9 0) npnpwa2
Q1 (net44 net15 net12 0) npnpwa2
Q0 (net44 Vin net15 0) npnpwa1
R7 (net31 GND) resistor r=200
R6 (net13 GND) resistor r=Remitout
R5 (net35 GND) resistor r=Remitout
R4 (net37 GND) resistor r=Remitout
R3 (net6 Vout) resistor r=Rout
R2 (net9 Vout) resistor r=Rout
R1 (net12 Vout) resistor r=Rout
R0 (Vcc1 net44) resistor r=10
simulator lang=ads
end examples_power_amp_schematic
#endif


mapping {
  pinMapping {
    npnpwa1 1:"C" 2:"B" 3:"E" 4:"S"
    npnpwa2 1:"C" 2:"B" 3:"E" 4:"S"
    resistor 1:"PLUS" 2:"MINUS"
  }
}
```

Viewing a Subnetwork Netlist

2. A log window pops up, displaying the netlist results. Once you have viewed the results, you can select **File > Close Window** to exit this window.

Net and Instance Name Mapping

Since Advanced Design System nomenclature rules differ from those of Cadence, nets, instances, etc. must be properly mapped. This mapping is done automatically as part of the netlisting function. The mapping rules are as follows.

- Advanced Design System keywords used as net or instance names are mapped by appending an underscore (_) to the name.

Name	**Map**
then	then_
else	else_
elseif	elseif_
endif	endif_
equals	equals_
notequals	notequals_
and	and_
not	not_
or	or_
global	global_
model	model_
define	define_
end	end_
parameters	parameters_

- Any non-alphabetical character (e.g. not a-z) in a net or instance name is mapped to an under bar (_).
- Advanced Design System uses a single name space for all names, regardless of object type (net, instance, etc.). This may necessitate name mapping in addition to the above.

Expression Name Mapping

Most Cadence *Analog Expression Language* (AEL) expressions contain constants, functions and suffixes with equivalents in ADS. In most cases the names of these equivalents are identical, requiring no mapping. As far as possible, Cadence expressions are pre-evaluated in the Cadence environment, prior to netlisting and prior to getting design variables from Cadence. This leaves only a few built-in function names to map (i.e. names that are not identical in the two environments).

Cadence	**ADS**
complex	cmplx
fabs	abs
log	ln
log10	log

Some built-in operator and function names in the Cadence *Affirma Analog Circuit Design Environment* as yet do not map to anything in the ADS environment.

Non-Mapping Operators

Cadence	Description
-	unary minus
~	unary one's complement
%	modulo
<<	left shift
>>	right shift
&	bitwise AND
|	bitwise OR
^	bitwise XOR
?:	conditional expression

For these non-mapping functions, custom equivalents in ADS need to be written and mapped until they are available as *built-ins* in ADS. Custom mapping is enabled via the configuration file option IDF_EXPR_MAP. For more information, refer to Expression Mapping in *Modifying the Configuration File* (dynlnkug).

Non-Mapping Functions

Cadence	Description
acosh	inverse hyperbolic cosine
asinh	inverse hyperbolic sine
atanh	inverse hyperbolic tangent
aelCheckRange	determines if a number falls within a range
conjgate	complex conjugate
floor	floor of a real number
ceil	ceiling of real number
mag	magnitude
db10	10 times log10
db20	20 times log10

Using Global Nodes

Cadence designs typically use implicit global nodes (names ending in *!*) for substrate power and ground connections. This notation is now supported by Advanced Design System. If the exclamation point suffix is used, a globalnode does not need to be placed in the ADS schematic.

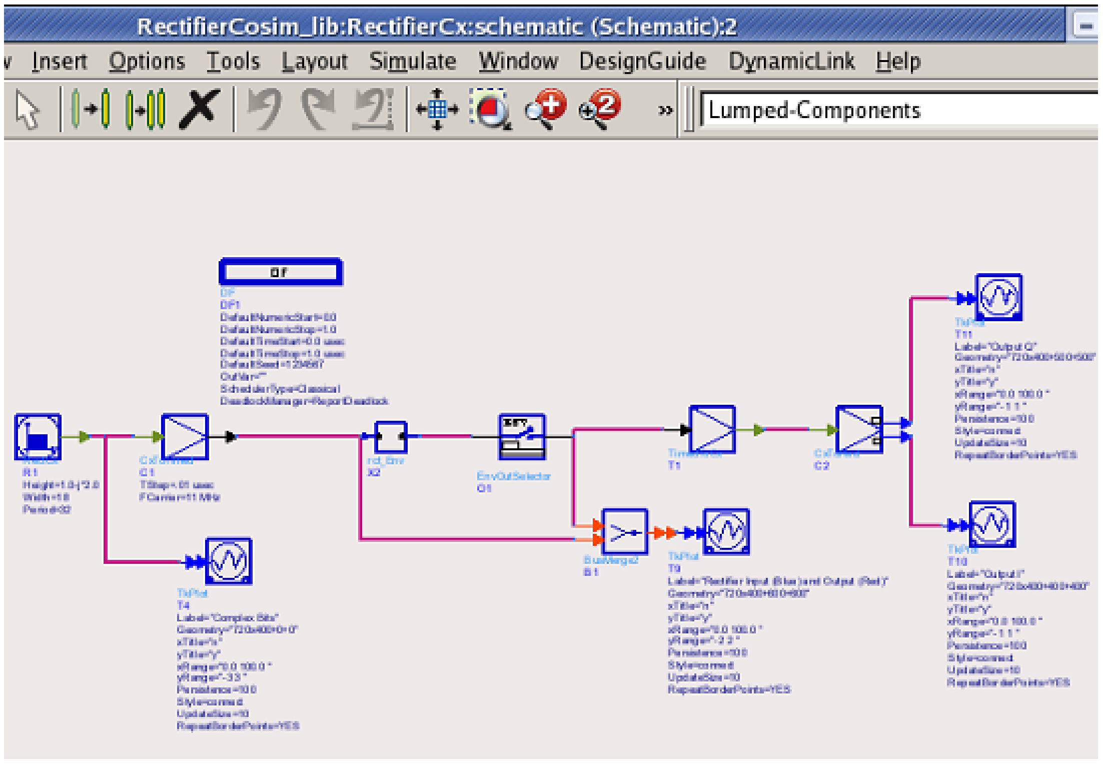
Running a DSP and Analog/RF Cosimulation with RFIC Dynamic Link

This section describes how to run cosimulation of behavioral Digital Signal Processing (DSP) designs along with Analog/RF circuit designs. Using RFIC Dynamic Link, you can perform a cosimulation with DSP designs in Advanced Design System (ADS) and your Analog/RF designs in Cadence. To run a cosimulation using RFIC Dynamic Link, you will need to create an intermediate layer ADS design. This intermediate design must contain a dummy ADS design representing the Cadence cellview in ADS and also include either an ADS *Transient* (Tran) or *Envelope* simulation controller.

For more detailed information about ADS Cosimulation, refer to the *ADS Ptolemy Simulation* (ptolemy) documentation.

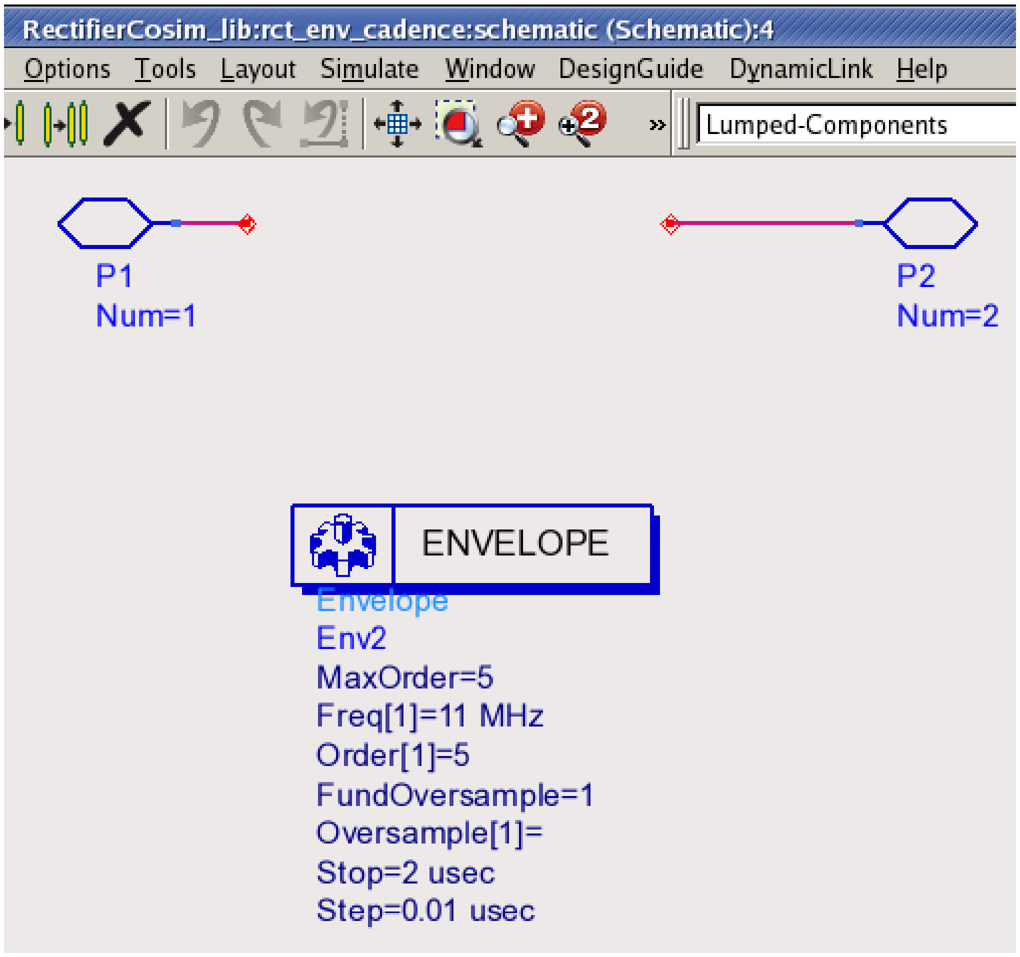
RFIC Dynamic Link DSP and Analog/RF Cosimulation Example

The following example uses the *RectifiedCx* schematic cellview (see the following figure) provided in the ADS Communication Systems *RectifierCosim_wrk* example workspace to demonstrate running a DSP and Analog/RF cosimulation using RFIC Dynamic Link.



Communications Systems *RectifiedCx* Example

The intermediate ADS design used in this example is a modified version of the *rct_Env* subcircuit design called *rct_env_cadence* (see the following figure).


RectifierCosim_lib:rct_env_cadence:schematic (Schematic):4
Options Tools Layout Simulate Window DesignGuide DynamicLink Help
Lumped-Components
P1
Num=1
P2
Num=2
ENVELOPE

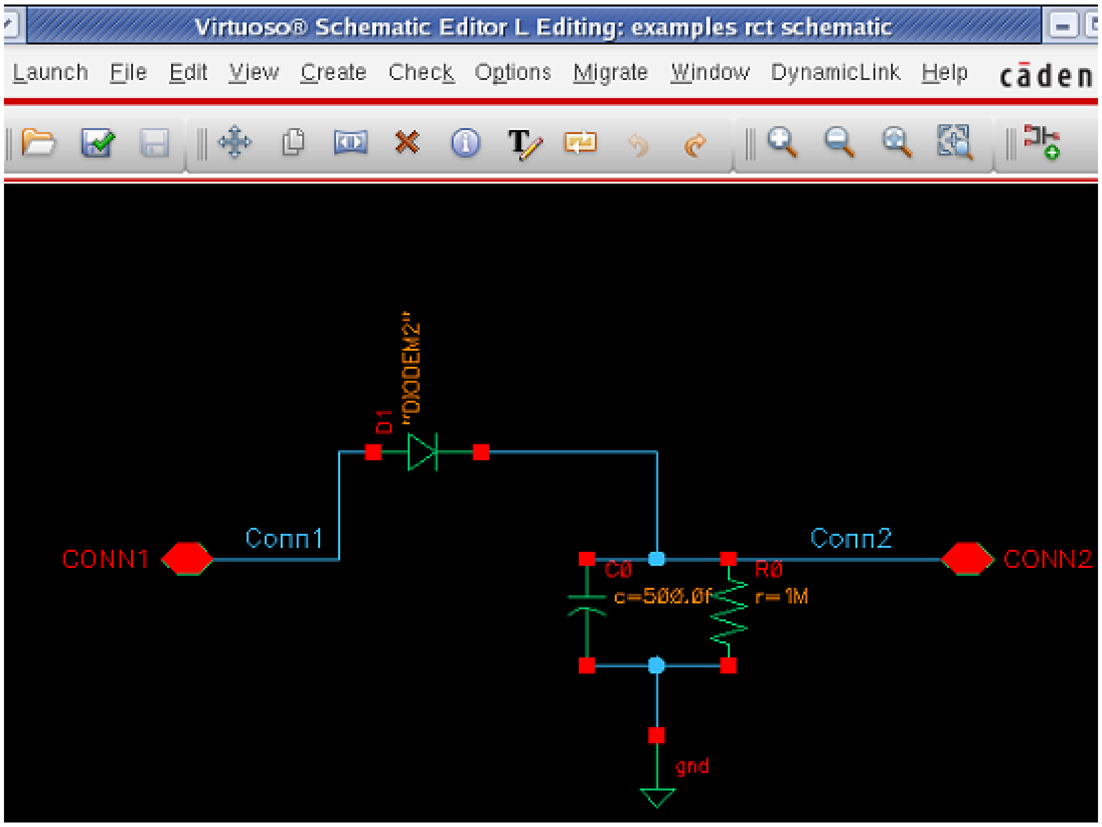
Envelope
Env2
MaxOrder=5
Freq[1]=11 MHz
Order[1]=5
FundOversample=1
Oversample[1]=
Stop=2 usec
Step=0.01 usec

The rct_env_cadence Intermediate Design

Before you begin, ensure you are set up to run the RFIC Dynamic Link examples as described in *Setting up the Examples Directory* (dynlnkug).

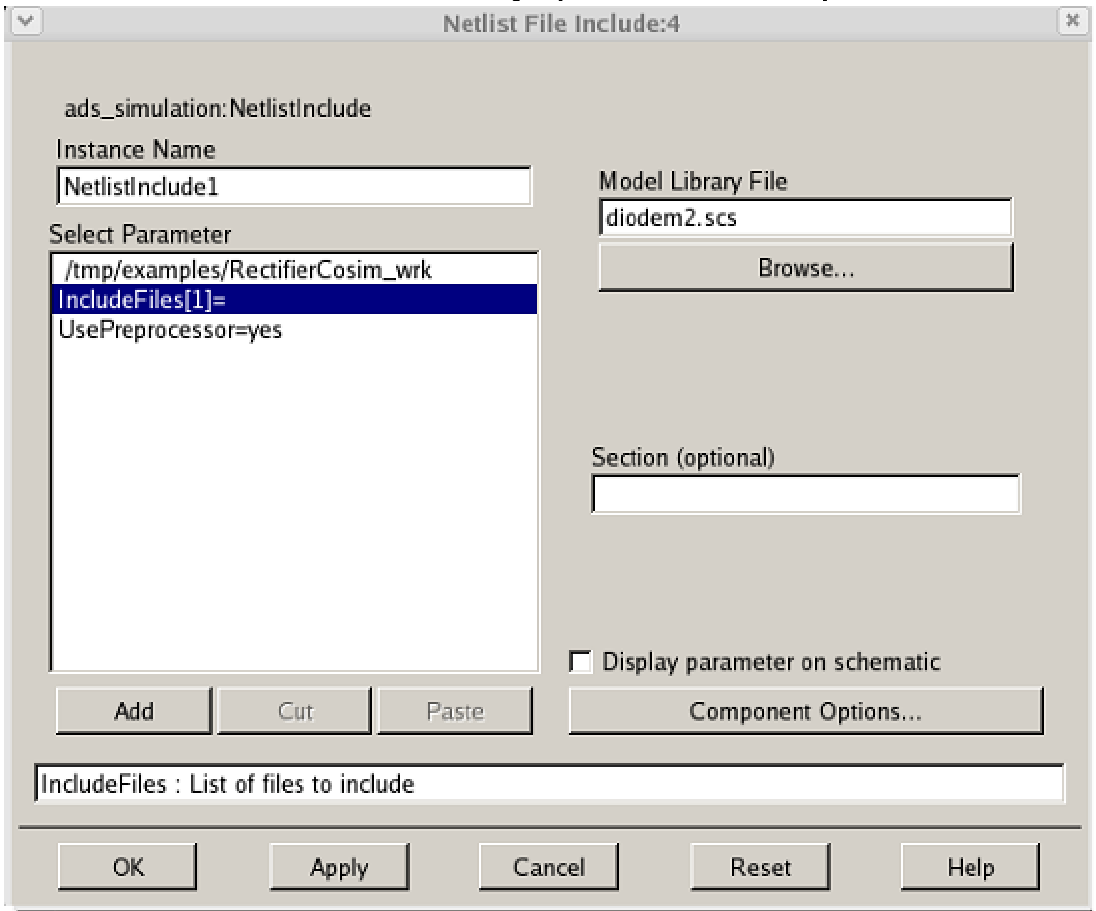
1. Open the *rct* cellview under the examples library. This cellview is a duplicate of the Advanced Design System *rct_Env* in the ADS example workspace $HPEESOF_DIR/examples/Com_Sys/RectifierCosim_wrk.

   Note
   As an alternative to performing steps 2 to 4 below, you can simply execute the *get_cosim_example.sh* script in the examples directory at your system prompt.


Virtuoso® Schematic Editor L Editing: examples rct schematic
Launch File Edit View Create Check Options Migrate Window DynamicLink Help
cāden
D1
"DIODEM2"
Conn1
CONN1
Conn2
CONN2
C0
c=500.0f
R0
r=1M
gnd

The *rct* Cell in the *examples* Library

2. Copy the ADS example workspace $HPEESOF_DIR/examples/Com_Sys/RectifierCosim_wrk to a local directory.
3. Copy $HPEESOF_DIR/idf/examples/examples_wrk/networks/rct_env_cadence.* to the local RectifierCosim_wrk/networks directory. This is the intermediate ADS design for this example.
4. Copy $HPEESOF_DIR/idf/examples/models/diodem2.ads to the local *RectifierCosim_wrk* directory. This model file contains the same ADS netlist as generated for the diode DIODEM2 in the original rct_Env subcircuit.
5. Start ADS by selecting **Tools > ADS Dynamic Link** in the Cadence schematic window.
6. Open the local *RectifierCosim_wrk* from the ADS Main window, then open the *rct_env_cadence* ADS design.
7. Choose **DynamicLink > Instance > Add Instance of Cellview** to place the *examples_rct_schematic* design between the two ports.
8. Add a NetlistInclude component to the rct_env_cadence design by choosing **DynamicLink** > **Add Netlist File Include** .
9. Double click the NetlistInclude component and apply the *diodem2.ads* model file to the *IncludeFiles* parameter.


Netlist File Include:4

ads_simulation:NetlistInclude

Instance Name: NetlistInclude1

Select Parameter:
- /tmp/examples/RectifierCosim_wrk
- IncludeFiles[1]=
- UsePreprocessor=yes

Model Library File: diodem2.scs

Browse...

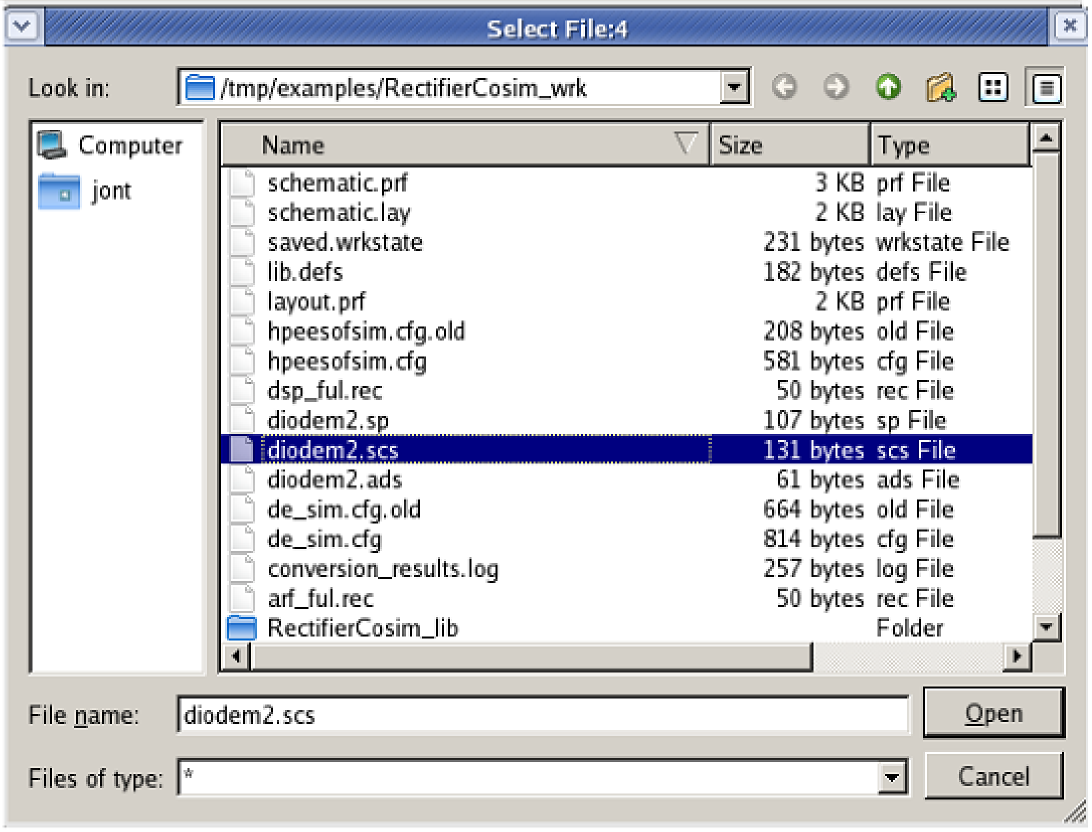
Section (optional):

Display parameter on schematic

Add | Cut | Paste | Component Options...

IncludeFiles : List of files to include

OK | Apply | Cancel | Reset | Help


Select File:4

Look in: /tmp/examples/RectifierCosim_wrk

Computer
jont

Name	Size	Type
schematic.prf	3 KB	prf File
schematic.lay	2 KB	lay File
saved.wrkstate	231 bytes	wrkstate File
lib.defs	182 bytes	defs File
layout.prf	2 KB	prf File
hpeesofsim.cfg.old	208 bytes	old File
hpeesofsim.cfg	581 bytes	cfg File
dsp_ful.rec	50 bytes	rec File
diodem2.sp	107 bytes	sp File
diodem2.scs	131 bytes	scs File
diodem2.ads	61 bytes	ads File
de_sim.cfg.old	664 bytes	old File
de_sim.cfg	814 bytes	cfg File
conversion_results.log	257 bytes	log File
arf_ful.rec	50 bytes	rec File
RectifierCosim_lib		Folder

File name: diodem2.scs

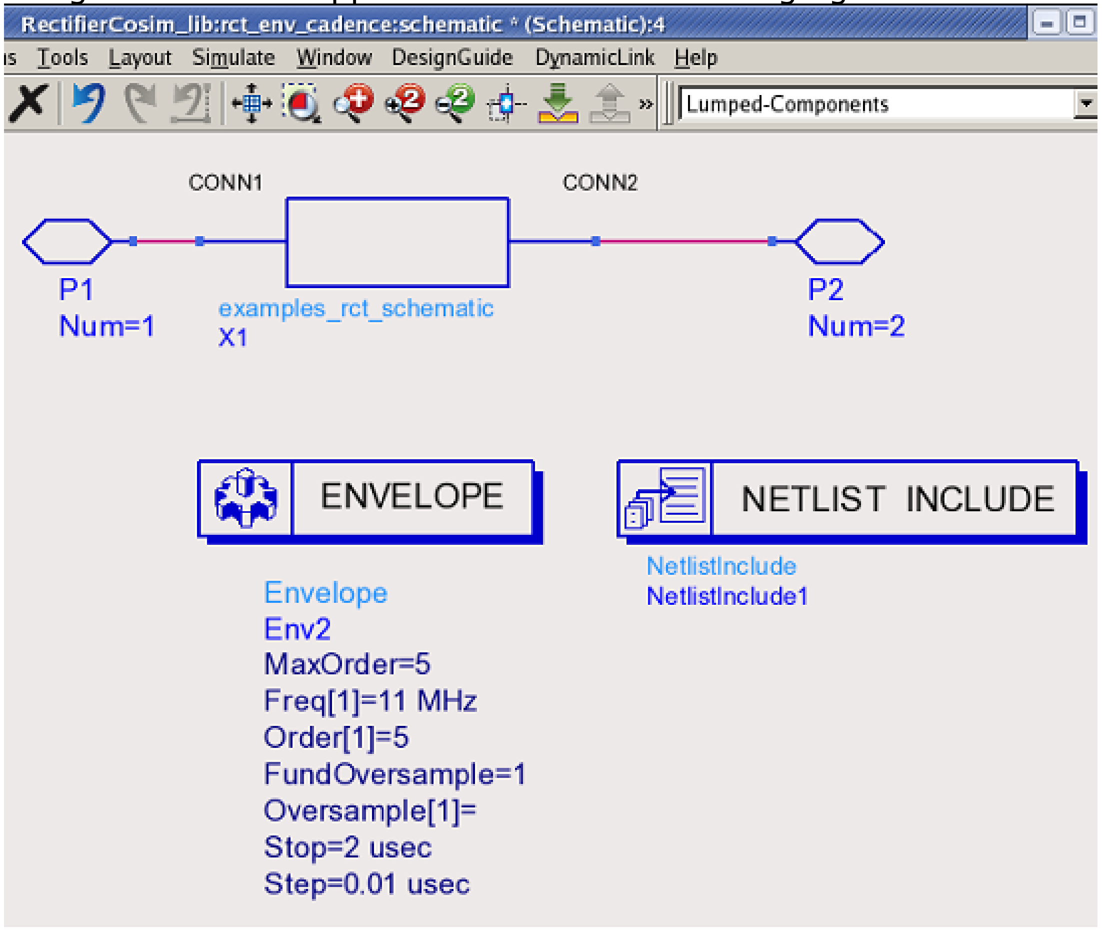
Files of type: *

Open | Cancel

Adding the NetlistInclude Component

10. Click **OK** to update the NetlistInclude component. The modified *rct_env_cadence*

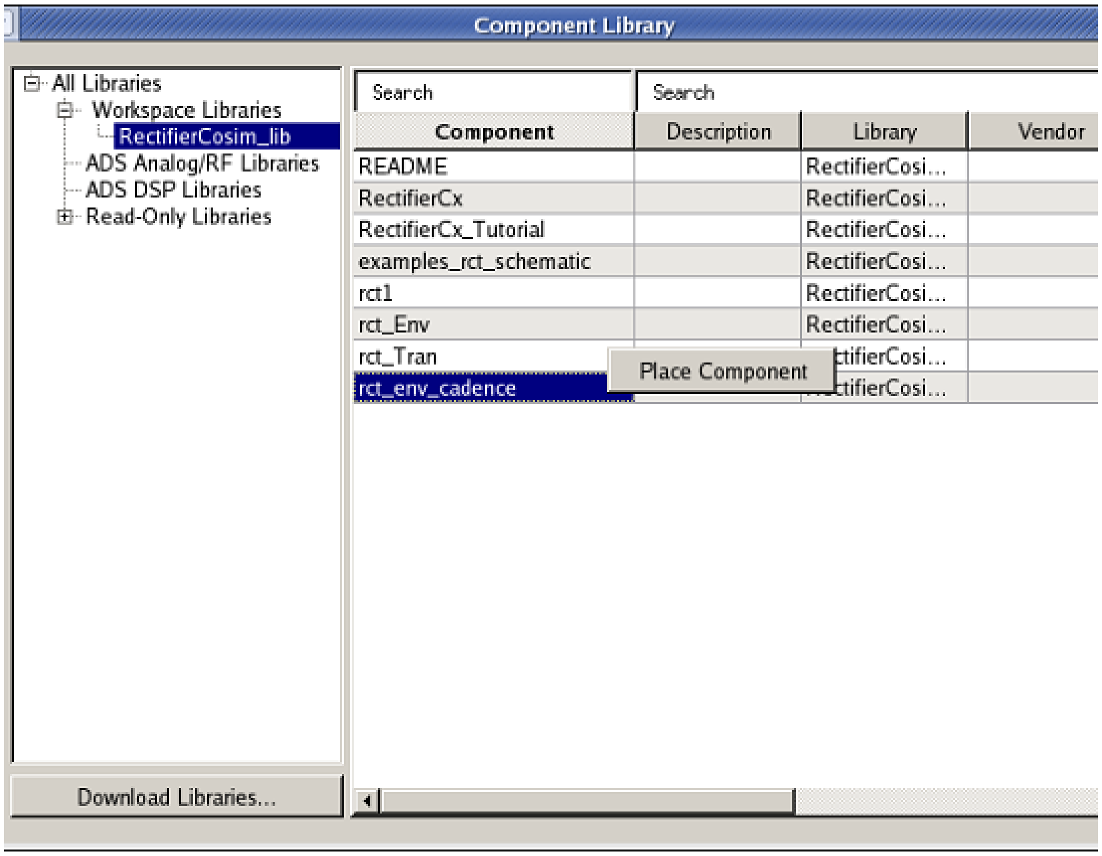
design should now appear similar to the following figure.


RectifierCosim_lib:rct_env_cadence:schematic * (Schematic):4
ls Tools Layout Simulate Window DesignGuide DynamicLink Help
Lumped-Components
CONN1
CONN2
P1
Num=1
examples_rct_schematic
X1
P2
Num=2
ENVELOPE
NETLIST INCLUDE
NetlistInclude
NetlistInclude1

Envelope
Env2
MaxOrder=5
Freq[1]=11 MHz
Order[1]=5
FundOversample=1
Oversample[1]=
Stop=2 usec
Step=0.01 usec

The Modified *rct_env_cadence* Design

11. Replace the *rct_Env* subcircuit component with *rct_env_cadence* in the *RectifiedCx* design by choosing **Insert > Component > Component Library** and then right-clicking the rct_env_cadence from the RectifierCosim_lib library in Library Browser to place an instance in the ADS schematic window.

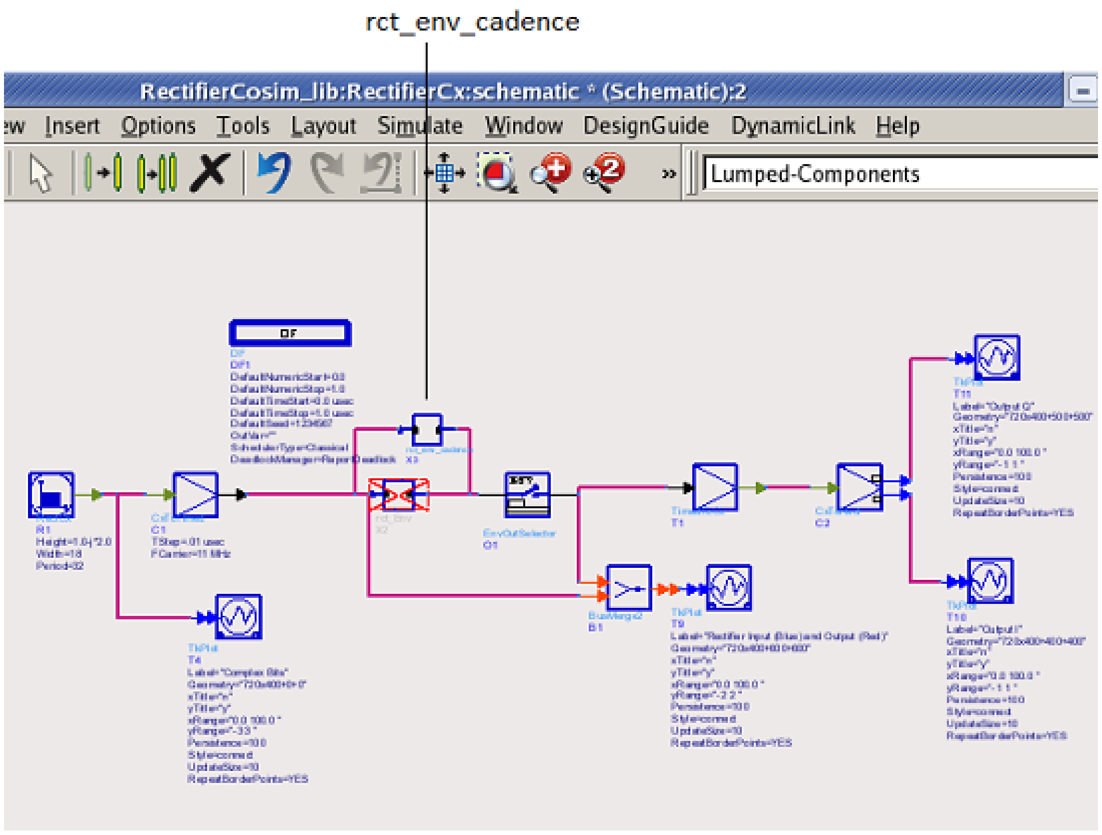

Component Library

- All Libraries
  - Workspace Libraries
    - RectifierCosim_lib
  - ADS Analog/RF Libraries
  - ADS DSP Libraries
  - Read-Only Libraries

Download Libraries...

Component	Description	Library	Vendor
README		RectifierCosi...	
RectifierCx		RectifierCosi...	
RectifierCx_Tutorial		RectifierCosi...	
examples_rct_schematic		RectifierCosi...	
rct1		RectifierCosi...	
rct_Env		RectifierCosi...	
rct_Tran		RectifierCosi...	
rct_env_cadence		RectifierCosi...	

Note
If a *Failed to locate the component definition* error occurs, you may need to create a symbol. Open the rct_env_cadence design and select **View > Create/Edit Schematic Symbol** to create a new symbol.



The RectifiedCx Design with rct_env_cadence

12. Choose **Simulate > Simulate** in the ADS schematic window of RectifierCx to simulate the design in ADS.
    Simulation results of the modified RectifierCx design with rct_env_cadence subcircuit should be identical with that of the original RectifierCx Design where the rct_Env subcircuit was included.

Troubleshooting RFIC Dynamic Link

This section provides troubleshooting information that can help you resolve common problems is also provided in this section.

All errors, warnings, and other messages are directed to the Cadence CIW. When a new message is written to the CIW, the window is raised to the top of your window stack so that new messages are always visible. Error messages may also be logged in a file, *idf.log*.

Some known problems and solutions are listed in the following section. You may find this information helpful in determining how to resolve a particular problem however, if you're unable to resolve a problem with the RFIC Dynamic Link using the information provided, contact Agilent EEsof EDA customer support.

Known Problems and Solutions

Problem:
Multiple Cadence instances containing the same pcell with different parameter values can cause incorrect simulation results when using Dynamic Link.

Solution: Add the following line in './idf.cfg' and then re-start Dynamic Link:

```
IDF_NETLIST_FILTER=$HPEESOF_DIR/idf/examples/pcell_workaround.py
```

When multiple distinct Cadence instances are placed in an ADS schematic window, each instance is netlisted into a netlist file in an individual Dynamic Link netlist session independent of the netlist sessions for the other Cadence instances. If multiple Cadence instances contain the same Cadence pcell instance but each with a different set of parameter values, it is possible for the pcell instance to be netlisted with the same Cadence subckt name but with a different subckt definition. In this case, only one version of the duplicate definitions will be used for all instances in the ADS simulation. This can cause the simulation results to be wrong.

When a Cadence subckt is redefined, ADS issues a warning similar to the following:

```
Warning detected by hpeesofsim during netlist parsing.
     In file `/tmp/work/examples_wrk/.DL/examples_pcell_1_schematic.net' at,
or just before, line 24.
     Subcircuit `polyhres_pcell_1' is redefined. Discarding the earlier definition
     from `/tmp/work/examples_wrk/.DL/examples_pcell_2_schematic.net:29'.
```

To avoid duplicate conflicting pcell subckt definitions upon seeing the above message, add the following line to the ./idf.cfg file and then re-start Dynamic Link:

```
IDF_NETLIST_FILTER=$HPEESOF_DIR/idf/examples/pcell_workaround.py
```

The 'pcell_workaround.py' Python script assumes that the name of the spectre subckt created for each pcell contains a suffix 'pcell' followed by an optional '_' *and then a series of digits. The top Cadence cell's '_<library>_<cell>*_<view>' will be appended after this suffix to avoid pcell subckt name conflicts across different netlist files.

The Python tool must be in your PATH before starting Cadence Virtuoso. If the Python in your PATH is older than version 2.4, use the Python tool shipped with ADS by inserting the path among the following list that suits your platform:

- $HPEESOF_DIR/tools/linux_x86_64/bin
- $HPEESOF_DIR/tools/linux_x86/bin
- $HPEESOF_DIR/tools/sun_sparc_64/bin
- $HPEESOF_DIR/tools/sun_sparc/bin

Problem: By default, ADS does not create its own private color map, which may lead to unpredictable color behavior and/or menu buttons in place of icons.

Solution: Try one or more of the following:

- Set *HPEESOF_COLORMAP = private* in the ADS configuration file *$HOME/hpeesof/config/hpeesof.cfg* or *$HPEESOF_DIR/config/hpeesof.cfg* .
- Set *CDS_NUM_USER_COLORS = 16* in your *.profile* or *.cshrc* file.
- Restart Dynamic Link after exiting all other color-intensive applications.

Problem: When you remake a symbol, even when the old symbol is deleted, ADS does not allow you to create another symbol with the same name.

Solution: Click the instance in the ADS schematic window, then choose **DynamicLink > Instance > Update Instance of Cellview** . If the problem persists:

- Delete the *<lib>* _ *<cells>* _ *<view>* symbol in the ADS Design.
- Save the ADS design.
- Exit ADS.
- Delete all the *<ADS_proj_dir>* /.DL/ *<lib>* _ *<cells>* _ *<view>* .* files.
- Restart ADS.

Problem: **Error* Could not find `nlpglobals/ads' in library `basic'. The nlpglobals' cell view is required. Netlisting aborted.* This error occurs either while netlisting in Analog Artist or after clicking *Simulate* in the ADS.

Solution: Copy the spectre view in the nlpglobals cell to create an ads view.

Problem: There is no distinction between a design variable X from *Design A* and a design variable X from a different *Design B* .

Solution: Use unique design variable names for different designs, unless you really intend them to be the same variable, in which case there's no problem.

Problem: Symbol generation via Cadence symbol duplication does not reproduce arcs.

> **Solution:** Use line segments instead of arcs.

Problem: *Could not spawn master program* . This message appears in your parent terminal window upon attempting to use the ADS link.

> **Solution:** Ensure that *$* HPEESOF_DIR/ *bin* is in your *PATH* and that *$* HPEESOF_DIR/ *bin/idfmp* is a valid executable. If this does not work, ask your UNIX System Administrator to reboot your system or otherwise determine if a socket address is in use.

Problem: The UNIX environment does not set up properly when an in-house script for DFII is used.

> **Solution:** Starting DFII using an in-house script may not set up the UNIX environment properly for RFIC Dynamic link. Work with your System Administrator to ensure that you understand what environment variables need to be set in the in-house script and modify your script accordingly.

Installation and Use Checklist

This section provides a checklist that can be used to help you resolve problems with RFIC Dynamic Link. You can use the questions below to help determine what the cause of a particular problem might be.

- Is RFIC Dynamic Link installed? If yes, the following files/directories should exist:
    - `$HPEESOF_DIR/bin/idf`
    - `$HPEESOF_DIR/bin/idfmp`
    - `$HPEESOF_DIR/idf/ads_site/`
    - `$HPEESOF_DIR/idf/ael/`
    - `$HPEESOF_DIR/idf/config/`
    - `$HPEESOF_DIR/idf/examples/`
    - `$HPEESOF_DIR/idf/skill/`
- Is Cadence configured for RFIC Dynamic Link? Run *idfConfigCadence -h* (RFIC Dynamic Link 2002) or *$HPEESOF_DIR/idf/config/configCadence -h* (RFIC Dynamic Link 2001).
- Is HPEESOF_DIR set? Run *idfenv* (RFIC Dynamic Link 2002 or later).
- Is *$HPEESOF_DIR/bin* in your $PATH?
- Is *icms* in your $PATH?
- Are *cds_root* and *cdsd* both in your $PATH?
- Is PATH, AGILEESOF_LICENSE_FILE or LM_LICENSE_FILE set in ~/.cshrc, ~/.profile, ~/.kshrc, or other scripts sourced by these scripts? Execute *echo $SHELL* or *finger* to display your login shell.
- Is your ADS *license.lic* file in the default *$HPEESOF_DIR/licenses/* directory or is it set in $AGILEESOF_LICENSE_FILE?
- Is there a *trans_idf* (RFIC Dynamic Link 2002) or *Idf_c_interface* (previous versions of RFIC Dynamic Link) feature in the ADS *license.lic* or *license.dat* file? Has this

feature expired?

- Which item in the following list defines the Cadence license file?
    - `$CDS_LIC_FILE`
    - *<Cadence_install_dir>* `/share/license/clients`
    - `$LM_LICENSE_FILE`
    - *<Cadence_install_dir>* `/share/license/license.dat`
- Does the Cadence license contain the valid features listed below:
    - `OASIS_Simulation_Interface`
    - `34510`
    - `300 (only if using layout)`
- Are all seats of the above ADS and Cadence licenses taken?
- Does $HPEESOF_DIR/idf/config/.cdsinit exist?
- Is $HPEESOF_DIR/idf/config/.cdsinit loaded in <Cadence_installation_directory>/tools/dfII/local/.cdsinit?
- If the above file does not exist, is $HPEESOF_DIR/idf/config/.cdsinit loaded in ./.cdsinit?
- If the above files do not exist, is $HPEESOF_DIR/idf/config/.cdsinit loaded in $HOME/.cdsinit?
- If none of the *.cdsinit* files exists, `cp $HPEESOF_DIR/idf/examples/.cdsinit ./`
- Does the *.cdsinit* file that loads RFIC Dynamic Link's *.cdsinit* file change PATH, CDS_INST_DIR, LM_LICENSE_FILE?
- Does the *icms* , *icfb* , or *msfb* script change your PATH or LM_LICENSE_FILE? Run *idfenv* at the UNIX prompt, start Cadence, enter *system("idfenv")* in the Cadence CIW, and then compare the output of *idfenv* before and after the Cadence script is executed.
- What is in the *./cds.lib* ? Do all of the libraries appear in blue in Cadence Library Path Editor form? If you are using the Agilent Technologies version of *analogLib* and *analogLib* appears in red, check to see if IDF_CDS_VERSION is used in *cds.lib* and it is not defined in the effective *.cdsinit* file.
- Does an *ads* view exist for the components in use? Use the Cadence Library Manager to check to see if the *analogLib/cap* cell contains an *ads* view.
- Does the *ads simInfo* section exist in the Cadence Component Description Format (CDF) for the components in use? Is all of the information in the *ads simInfo* correct? Use the Edit Component CDF form (**Tools > CDF > Edit**) or `cdfDump("<lib>" "/tmp/<cell>.cdf" ?cellName "<cell>" ?edit t)` function to see if the *ads simInfo* section exists in a cellview (leave out the '?cellName "<cell>"' for dumping the library CDF instead of the cell CDF).
- Does the *ads* view exist for the global cellview *nlpglobals* under the *basic* library? If not, open the Cadence Library Manager, select *symbol* or *spectre* view, press middle mouse button, choose **Copy** from the popup list, and copy the *symbol* or *spectre* view to the *ads* view.
- If RFIC Dynamic Link failed to create an ADS netlist, can the Cadence netlister create a *spectre* netlist? Select **Tools > Analog Environment** from Cadence schematic window, select **Setup >Simulator/Directory/Host** from the Affirma DE window and ensure that *spectre* is selected as the Simulator, then select **Simulation > Netlist > Create** to generate a *spectre* netlist.
- Has the Cadence symbol changed since the instance was placed in ADS? Select the instance in the ADS 2002 schematic window, then choose **DynamicLink > Instance > Update Instance to Cellview** to recreate ADS symbol and remove the

<Cadence_project_dir> / *<cell>* / directory to force Dynamic Link to regenerate the netlist. With RFIC Dynamic Link 2001, remove *<lib>* _ *<cell>* _ *<view>* .* under *<ads_wrk>* /.DL/, delete *<Cadence_project_dir>* / *<cell>* , then delete the instance in the ADS schematic and replace the instance.

- Is the *de_sim.cfg* file under the ADS workspace directory corrupted?
- Is there any local ADS customization in $HOME/hpeesof/config/de_sim.cfg? Search for *AEL* .
- Is there site-wide ADS customization in $HPEESOF_DIR/config/de_sim.cfg or custom/config/? For example, the line that reads "+ dynamic_link" in the *de_sim.cfg* file includes the file *dynamic_link.cfg* .
- If some components require models, are the associated ADS model files included with the *modelLibraryFiles* parameter in the *idfInclude* component?
- For debugging purposes, if the *./idf.log* file appears to be shorter than expected for debugging, is IDF_DEBUG_MODE set to TRUE in the UNIX environment?
- If the current GMT in *./idf.log* and *~/CDS.log* do not match, is there a lock on *~/CDS.log* ? Is it *~/CDS.log.1* or *~/CDS.log.2* that is the Cadence log file for this session?
- Are there any suspicious options set in any of the following four Cadence *.cdsenv* files?
    - *<cds>* `/tools/dfII/etc/tools/` *<application>* `/.cdsenv`
    - *<cds>* `/tools/dfII/local/.cdsenv`
    - `$HOME/.cdsenv`
    - `./.cdsenv`

    The above files contain user preference options.
- If DC Operating Point Back Annotation failed, check the following in the order given:
    - *<ads_wrk>* `/data/` *<ads_dsn_name>* `.ds`
    - *<ads_wrk>* `/psf/` *<lib>* *{}* *_<cell>* *{}* *_<view>* `/dcOpInfo.info`
    - *<Cadence_proj_dir>* `/` *<cell>* `/adsDL/` *<view>* `/psf/dcOpInfo.info`
    - View the contents of the last file to see if there are any reasonable numbers.
- If DC Back Annotation failed, check the following in the order given:
    - *<ads_wrk>* `/psf/` *<lib>* *{}* *_<cell>* *{}* *_<view>* `/dcOp.dc`
    - *<Cadence_proj_dir>* `/` *<cell>* `/adsDL/` *<view>* `/psf/dcOp.dc`
    - View the contents of the last file to see if there are any reasonable numbers.

Tuning and Optimizing Designs

This chapter provides information on tuning and optimizing designs using the Advanced Design System tuning and optimization capabilities.

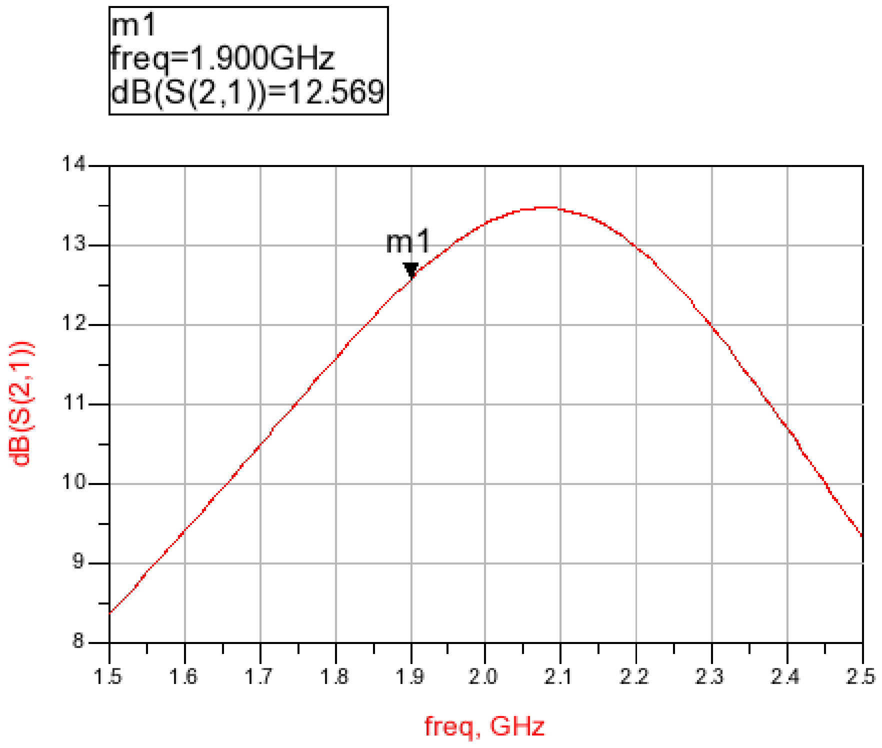
Tuning Cadence Instance Parameters and Design Variables

The ADS tuning capability enables you to change one or more design parameter values and see its effect on the output without re-simulating the entire design again from the beginning. Dynamic Link extends the ADS tuning function to instance parameters and design variables in Cadence subcircuits. For more information on using the ADS tuning feature, refer to *Tuning* in the *Tuning, Optimization, and Statistical Design* (optstat).

This section uses the *power_amp* example used in *Getting Started Tutorial* (dynlnkug) for demonstrating tuning of Cadence instance parameters and design variables.

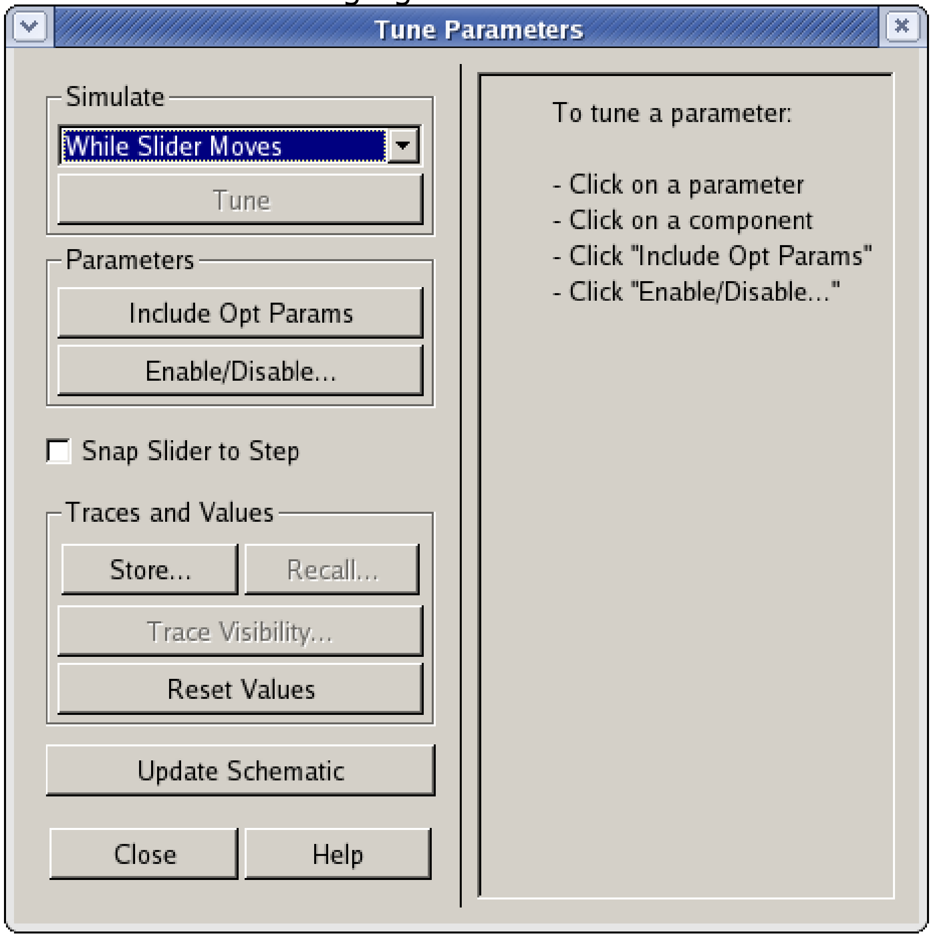
A Dynamic Link For Cadence Tuning Example

1. Follow the steps listed in *Performing an S-parameter Simulation* (dynlnkug) in the *Getting Started Tutorial* (dynlnkug). Plot S(2,1) in the ADS Data Display window after ADS S-parameter simulation is completed. The S(2,1) plot should resemble the following figure.



Data Display with S(2,1) of the *power_amp_test* Design.

2. In the ADS Schematic window, choose **Simulate** > **Tuning** or click the Tune Parameters icon (tuning fork) in the toolbar.
3. Wait for the initial analysis to complete. The *Tune Parameters* dialog box appears as shown in the following figure.

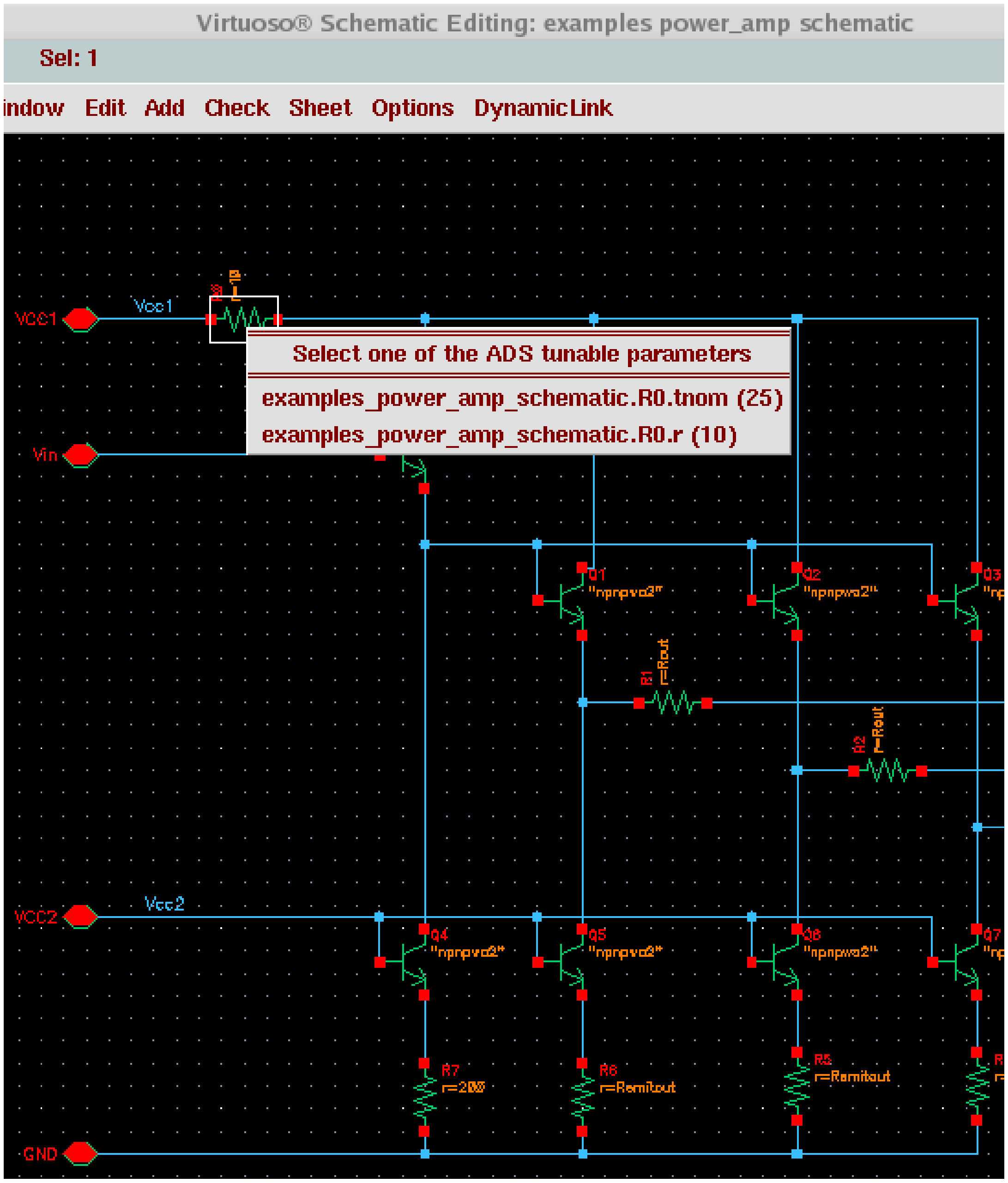


To tune a parameter:

- Click on a parameter
- Click on a component
- Click "Include Opt Params"
- Click "Enable/Disable..."

The Initial ADS Tune Parameters Dialog Box.

1. Move your cursor over resistor *R7* in the Cadence schematic window and click the left mouse button (left click). A slider control for *R7.R* immediately appears in the ADS Tune Parameters dialog box. Note that only integer and floating point parameters are tunable. *R7* contains only one tunable parameter *R* (*r* in Cadence spectre), the resistance.
2. Left click resistor *R0* in the Cadence schematic window. A menu pops up beneath *R0* as illustrated in the following figure. The popup menu is displayed because there is more than one tunable parameter in *R0* , *R* and *Tnom* . Cadence instance parameters are sent to ADS one at a time.



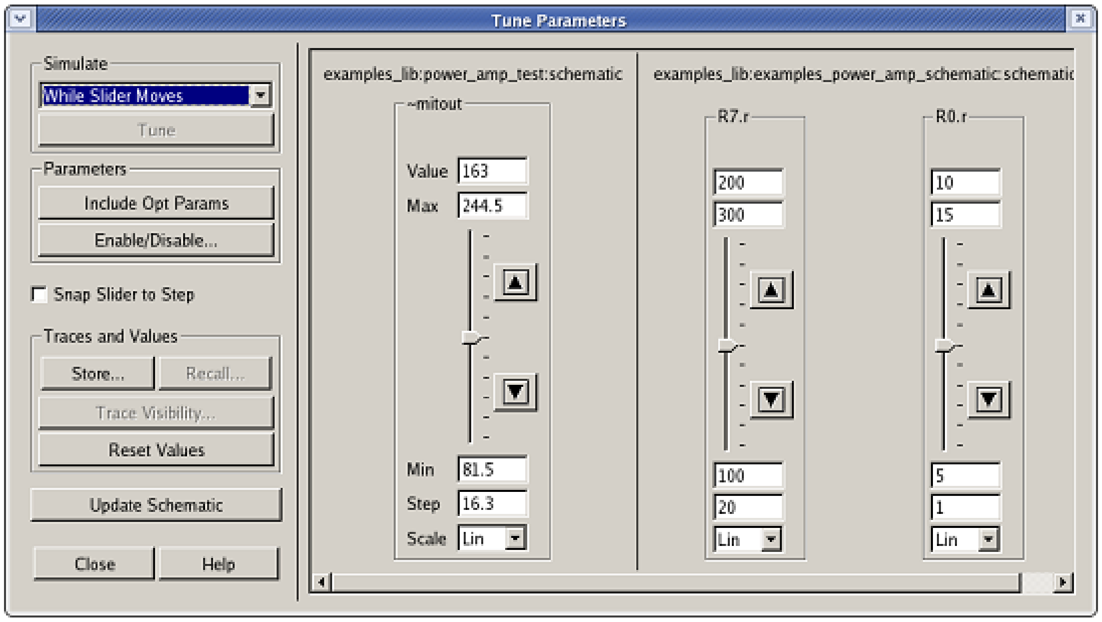
Select one of the ADS tunable parameters

examples_power_amp_schematic.R0.tnom (25)

examples_power_amp_schematic.R0.r (10)

Select Tunable Parameter Pop-up Menu in the Cadence Virtuoso Schematic Window.

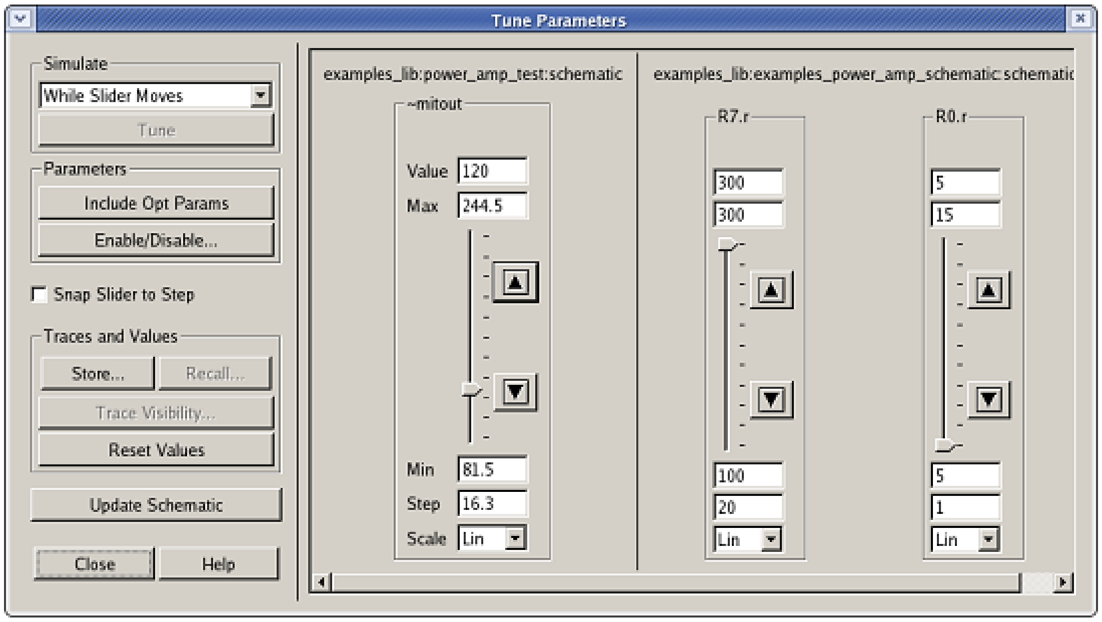
3. Select *examples_power_amp_schematic.R0.R(10)* in the pop-up menu. This creates a slider control for *R0.R* in the ADS Tune Parameters dialog box.
4. In the ADS Schematic window, click the design variable *Remitout* in the VAR1 block. The following figure shows the ADS Tune Parameters dialog box with *VAR1.Remitout* , *R7.R* , and *R0.R* being selected. Recall that *VAR1.Remitout* is a design variable originated from the Cadence subcircuit.



ADS Tune Parameters Dialog Box

Notice the distinction that a Cadence design variable is selected directly from the *VAR1* block in the ADS Schematic window, while an instance parameter is selected from the Cadence Schematic window.

5. In the ADS *Tune Parameters* dialog box, select a tune analysis mode from the *Simulate* drop-down list. This tells ADS when you want tuning to occur. For this example, choose *After each change* if not already selected.
   After you finish with all of the steps in this example, try each tuning analysis method (*After Each Change* , *After Pressing Tune* , *While Slider Moves*) to see which one works best for you.
6. Drag the slider for *examples_power_amp_schematic.R7.R* to 300 (Ohms). You also can change the tunable parameters by doing the following:
   - Click the up or down arrows.
   - Type the value directly into the box.
     Observe the results in Data Display window each time you release the mouse button after dragging a slider or changing a value in the Tune Parameters dialog box.
7. Drag the slider for *examples_power_amp_schematic.R0.R* to 5 (Ohms).
8. Finally, enter a value for *power_amp_test.VAR1.Remitout* of 120 (Ohms). The Tune Parameters dialog box should now appear similar to the following figure.


Tune Parameters

Simulate: While Slider Moves

Tune

Parameters: Include Opt Params | Enable/Disable...

Snap Slider to Step

Traces and Values: Store... | Recall... | Trace Visibility... | Reset Values

Update Schematic

Close | Help

examples_lib:power_amp_test:schematic

~mitout: Value 120, Max 244.5, Min 81.5, Step 16.3, Scale Lin

examples_lib:examples_power_amp_schematic:schematic

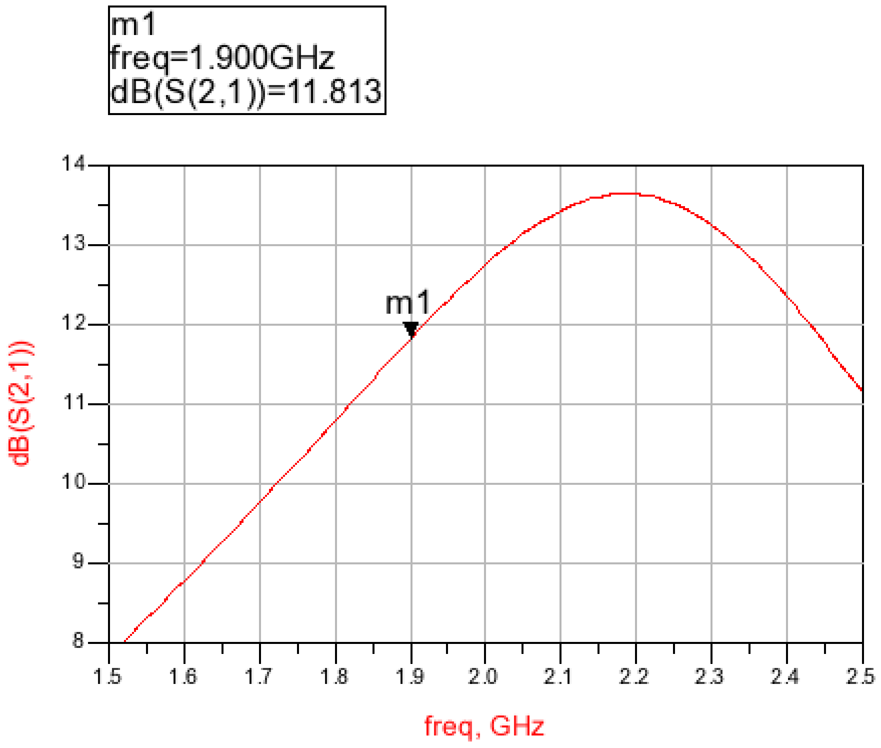
R7.r: 300, 300, 100, 20, Lin

R0.r: 5, 15, 5, 1, Lin

ADS Tune Parameters Dialog Box (With *VAR1.Remitout* , *R7.R* , and *R0.R* values Changed).

The next figure shows the resultant S(2,1) curve displayed in the Data Display window after changing *VAR1.Remitout* , *R7.R* , and *R0.R* in the Tune Parameters dialog box. You can compare this to the initial results in the figure Data Display with S(2,1) of the power_amp_test Design.

9. You can click the **Reset Values** button to restore all the sliders to their original values. The **Update Schematic** button in the Tune Parameters dialog box enables you to write the instance parameter values currently displayed in the dialog box into the Cadence Schematic window. For Cadence design variables, such as *VAR1.remitout* , you still need to select **DynamicLink > Design Variables > Update Design Variables to Cellviews** in the ADS schematic window and then follow the instruction in the Confirmation Message form to complete the update. Because there is no undo function for the *Update* operation, do not click the **Update Schematic** button if you do not want to change values in the Cadence subcircuit.



Results of S(2,1)

(as *R7.R* , *R0.R* and *VAR1.Remitout* are changed in the Tune Parameters dialog box)

10. Click the **Close** button in the Tune Parameters dialog box to end the tuning session.

Note
For more information on using the ADS tuning feature, refer to *Tuning in Advanced Design System* (optstat) in *Tuning, Optimization, and Statistical Design* (optstat).

This example demonstrated three types of tuning operations in Dynamic Link:

- Clicking a Cadence instance with a single tunable parameter causes that parameter to be sent to ADS for tuning.
- Clicking a Cadence instance with multiple tunable parameters results in a pop-up menu being displayed. Selecting an item from the pop-up menu causes the parameter associated with that menu item to be sent to ADS for tuning.
- After obtaining Cadence design variables by selecting **DynamicLink > Design Variables > Get Design Variables** in the ADS schematic window, clicking a Cadence design variable in the ADS VAR1 block sends that variable for tuning.

All the above three types of operation act like a toggle switch. Selecting an item already on the Tune Parameters dialog box removes it from the dialog box.

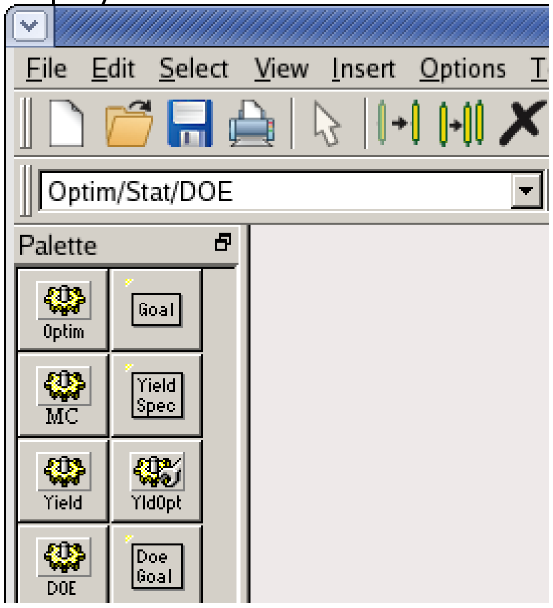
You can descend down a Cadence design hierarchy to find an instance parameter for tuning. You can then return to higher level Cadence design hierarchy to select another instance parameter.

Note
During the tuning operation, the left mouse button in the Cadence Virtuoso schematic window is mapped to a Dynamic Link SKILL procedure. Do not bind any function to the left mouse button during this period. Any bindkey function previously mapped to the left mouse button will not work until the tune mode ends.

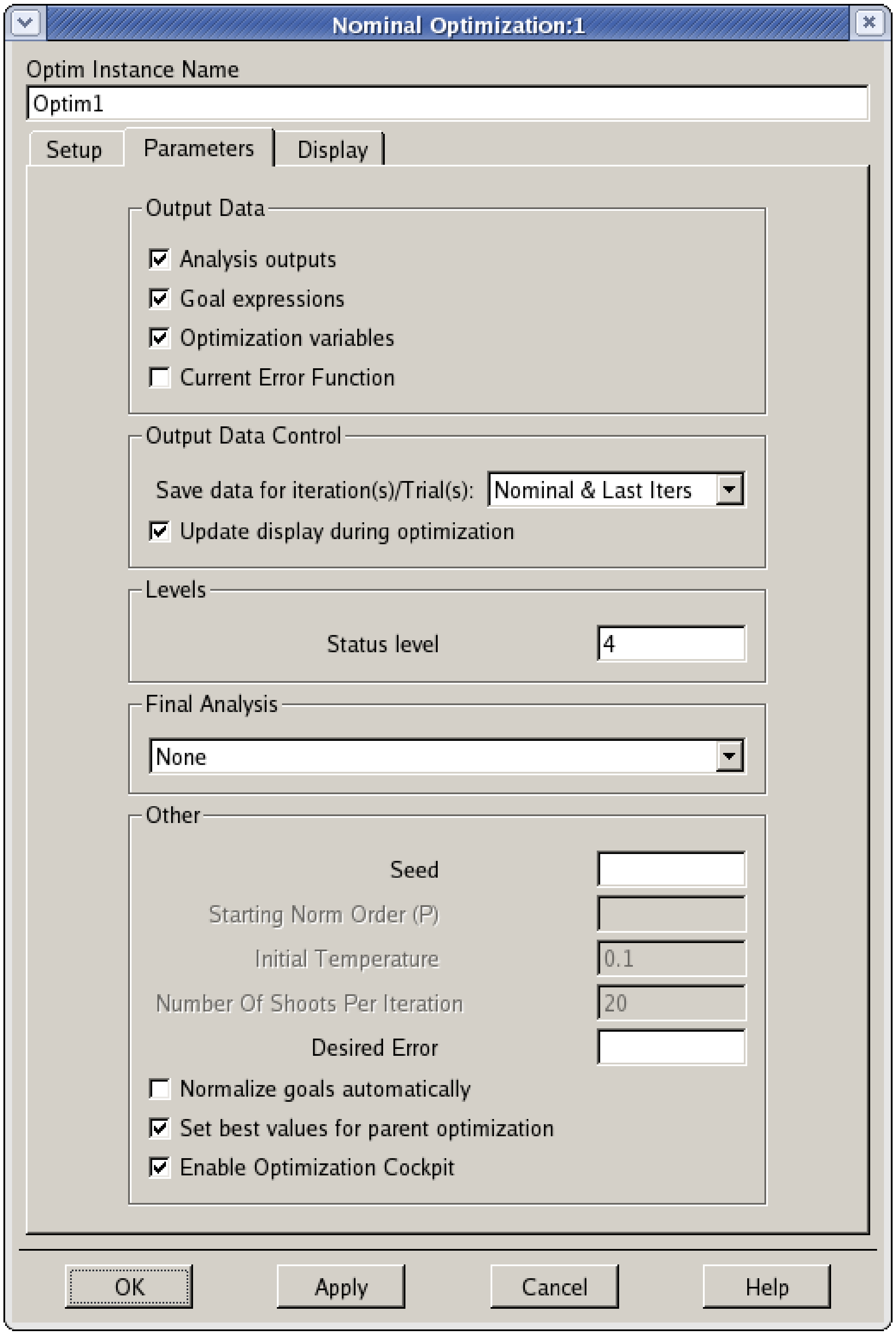
If you have a problem while tuning and need to discontinue the operation, enter *IdfMpsTuneEnd* in the Cadence CIW input area. This will end the Dynamic Link Tune Mode operation.

Optimizing Designs

Performance optimization enables you to specify a range of device or component values. The software can then automatically compute the nominal values that best meet your performance goals or specifications. A family of optimizers come with ADS, each with a different mathematical effect or use. For more information on performance optimization, refer to *Performing Nominal Optimization* in the *Tuning, Optimization, and Statistical Design* (optstat).
To optimize a design in the Advanced Design System:

1. In the Schematic window containing the design you want to optimize, choose **Optim/Stat/DOE** from the component palette. ** The Optim/Stat/DOE palette is displayed.


The Optim/Stat/DOE Component Palette
2. Set the options (*Goal, Nominal Optimization,* etc.) as desired.When a Nominal Optimization component is first added, you need to enable the output to the dataset. Click the *Nominal Optimization* component and choose **Edit > Component > Edit Component Parameters** . ** This brings up a *Nominal Optimization* dialog box. Select the *Parameters* tab and ensure that the *Analysis outputs* and *Optimization variables* are checked. Click **OK** .



Optim Instance Name: Optim1

Output Data
- [✓] Analysis outputs
- [✓] Goal expressions
- [✓] Optimization variables
- [ ] Current Error Function

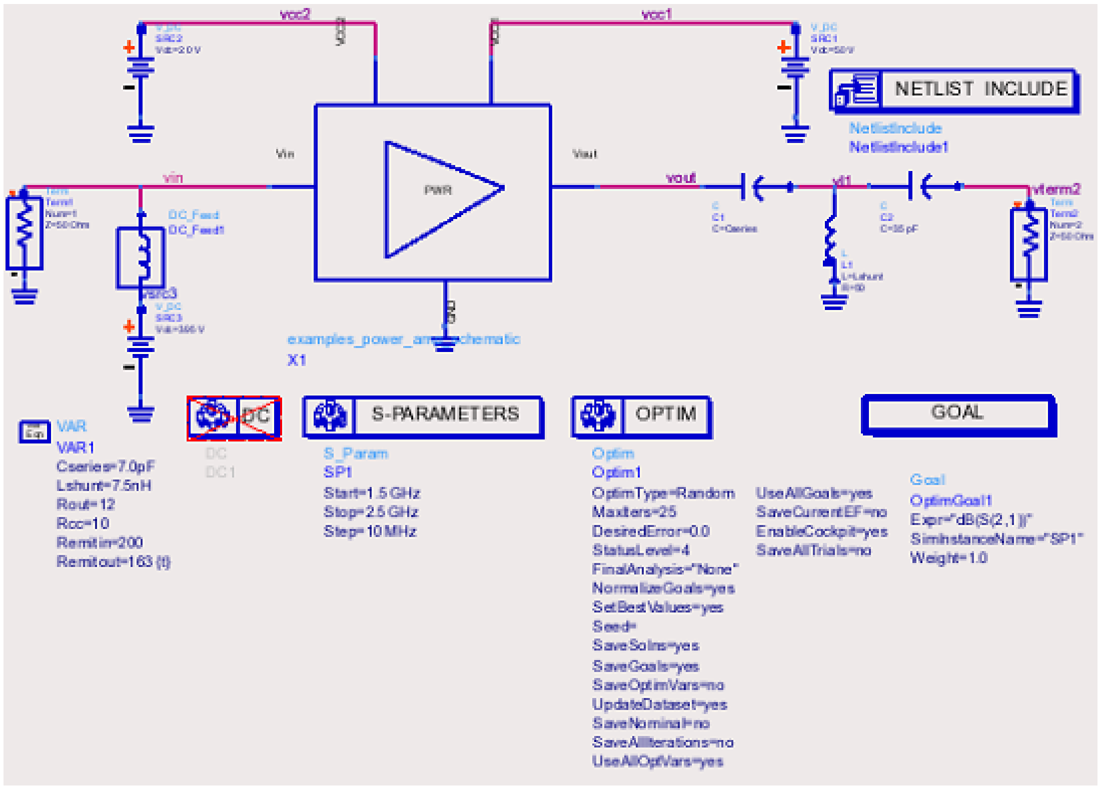
Output Data Control
- Save data for iteration(s)/Trial(s): Nominal & Last Iters
- [✓] Update display during optimization

Levels
- Status level: 4

Final Analysis
- None

Other
- Seed:
- Starting Norm Order (P):
- Initial Temperature: 0.1
- Number Of Shoots Per Iteration: 20
- Desired Error:
- [ ] Normalize goals automatically
- [✓] Set best values for parent optimization
- [✓] Enable Optimization Cockpit

3. Proceed with the Advanced Design System optimization.



```
VAR1
Cseries=7.0pF
Lshunt=7.5nH
Rout=12
Rcc=10
Remitin=200
Remitout=163 {t}
```

```
S_Param
SP1
Start=1.5 GHz
Stop=2.5 GHz
Step=10 MHz
```

```
Optim
Optim1
OptimType=Random          UseAllGoals=yes
MaxIters=25               SaveCurrentEF=no
DesiredError=0.0          EnableCockpit=yes
StatusLevel=4             SaveAllTrials=no
FinalAnalysis="None"
NormalizeGoals=yes
SetBestValues=yes
Seed=
SaveSolns=yes
SaveGoals=yes
SaveOptimVars=no
UpdateDataset=yes
SaveNominal=no
SaveAllIterations=no
UseAllOptVars=yes
```

```
Goal
OptimGoal1
Expr="dB(S(2,1))"
SimInstanceName="SP1"
Weight=1.0
```

An Optimization Setup

Statistical Variations

Both the *Monte Carlo* and *Yield* simulation controllers include options for specifying statistical variations. Note that *process* and *mismatch* variations are only used for ADS RFIC Dynamic Link.

Statistical Variation Options

option	Description
stat{...}	ADS syntax for statistical variations.
process †	For process only analysis, select the process checkbox and unselect mismatch checkbox. For process and mismatch analysis, select both the process and mismatch checkboxes.
mismatch †	For mismatch only analysis, unselect the process checkbox and select mismatch checkbox. For process and mismatch analysis, select both the process and mismatch checkboxes.
† These options use Spectre syntax and are provided only for use with RFIC Dynamic Link. For more information on process and mismatch variations, refer to the section on *Monte Carlo Analysis* in Chapter 6 of the Cadence _Virtuoso Spectre Circuit Simulator User Guide, Product Version 5.1.41_ .	

Updating the Cadence Cellview

Once the optimum value of a variable is computed by an ADS simulation, you can update the value to the Cadence cellview. To update the Cadence cellview:

1. In the Schematic window, choose **DynamicLink > Design Variables > Update Design Variables to Cellviews** .

Note
Only the nominal values of variables get updated to Cadence; any range values are ignored. Variables to be optimized should not be assigned a value in Cadence; they may be assigned a nominal value and a range in ADS.

Using Additional Features of RFIC Dynamic Link

This section describes some of the additional features provided by the RFIC Dynamic Link. Some of the issues related to compatibility between Advanced Design System and the Cadence tools are also discussed in this section.

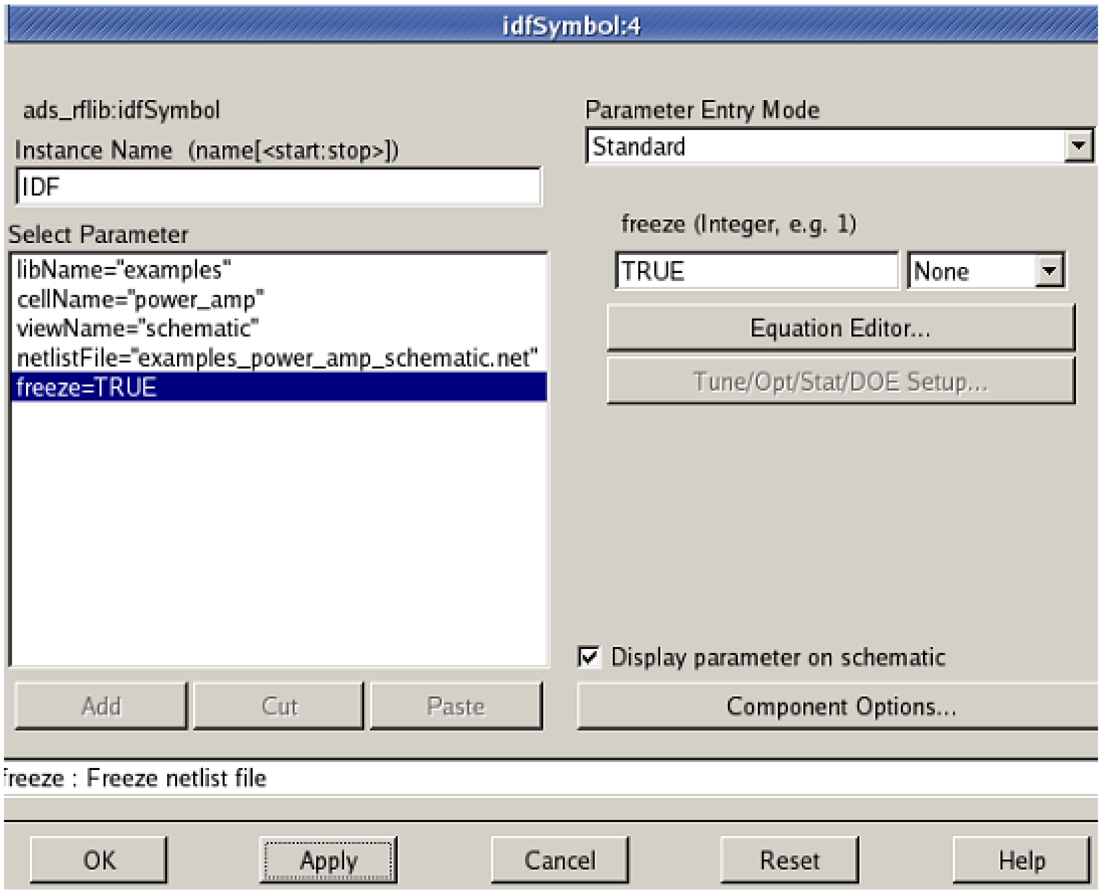
Freezing Selected Subcircuits

The RFIC Dynamic Link *Freeze* mode enables you to keep Cadence from generating a new netlist each time you simulate in Advanced Design System. This helps to avoid unnecessary time-outs caused by re-netlisting a large Cadence subcircuit.

Setting the Freeze Parameter

If you want to keep a Cadence Cellview from being netlisted and a netlist already exists:

1. Edit the ADS dummy design for that Cellview which would have a name of the form <*lib*>_<*cell*>_<*view*>. In this example, the name of the file is examples_power_amp_schematic.
2. Double click the idfSymbol in the schematic dummy design (see the following figure) and set the *freeze* parameter to "yes" or "TRUE". Note that the idfSymbol contains the library name, cell name, view name and netlist file information


idfSymbol:4

ads_rflib:idfSymbol

Instance Name (name[<start:stop>])

IDF

Select Parameter

libName="examples"
cellName="power_amp"
viewName="schematic"
netlistFile="examples_power_amp_schematic.net"
freeze=TRUE

Parameter Entry Mode: Standard

freeze (Integer, e.g. 1)

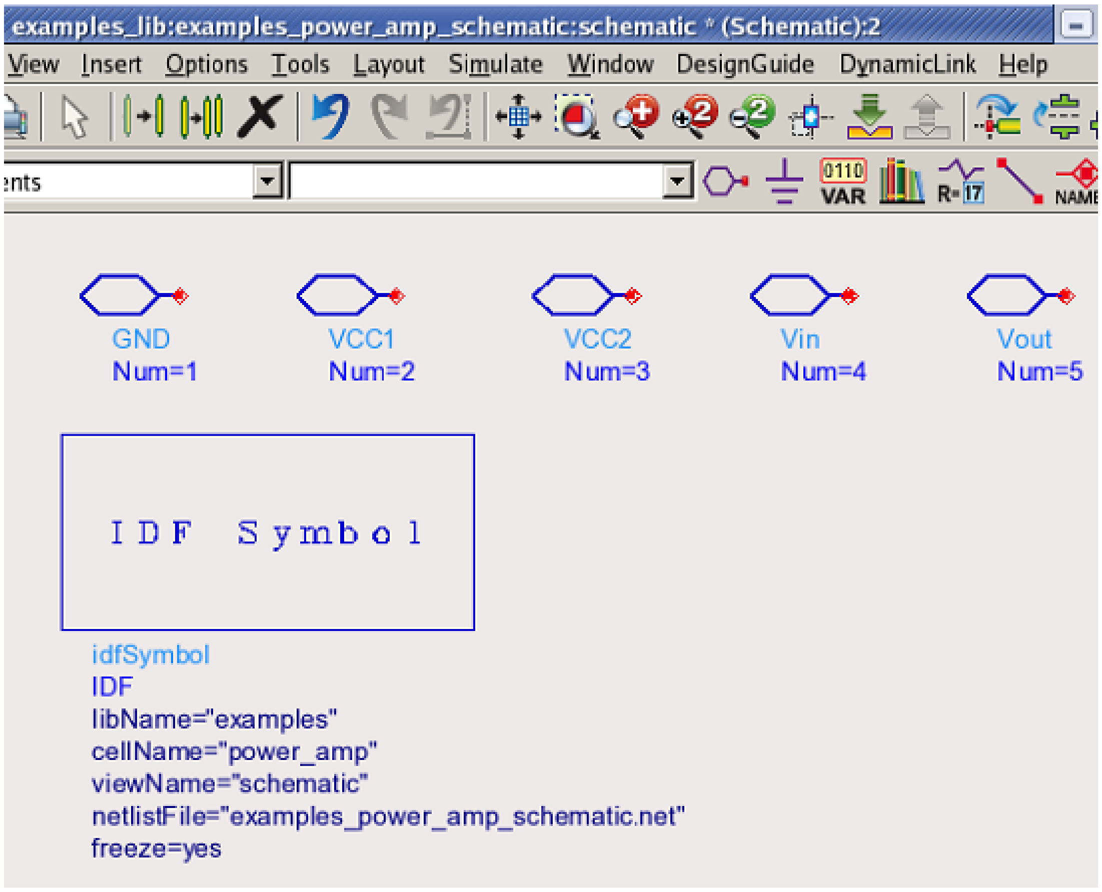
TRUE None

☑ Display parameter on schematic

freeze : Freeze netlist file

Freeze Parameter Setting in idfSymbol

Refer to the example schematic in the following figure for the location of the *freeze* and *netlistFile* parameter settings. To turn the freeze parameter off, set the parameter to "no" or "FALSE". The default value for the freeze parameter is "FALSE".


examples_lib:examples_power_amp_schematic:schematic * (Schematic):2
View Insert Options Tools Layout Simulate Window DesignGuide DynamicLink Help
GND
Num=1
VCC1
Num=2
VCC2
Num=3
Vin
Num=4
Vout
Num=5
IDF Symbol

```
idfSymbol
IDF
libName="examples"
cellName="power_amp"
viewName="schematic"
netlistFile="examples_power_amp_schematic.net"
freeze=yes
```

Defining the Freeze Parameter

To freeze all Cadence subcircuits, see *Modifying the Configuration File* (dynlnkug).

Generating a Cadence Subcircuit Netlist

In order to run Dynamic Link in Freeze mode, the Cadence subcircuit netlist must exist. If a Cadence subcircuit netlist does not exist, you can generate a new Cadence subcircuit netlist before running your ADS Simulation. Choose **DynamicLink > Subcircuit Netlist** from your Cadence Schematic window.

Note
If the *Dynamic Link* pull-down menu does not appear in the Cadence Virtuoso Schematic window, choose **Tools > ADS Dynamic Link > Add Dynamic Link menu to all schematic windows** in the Cadence Command Interpreter Window (CIW).

This generates the Cadence subcircuit netlist. You can now run your ADS Simulation in *Freeze* mode.

Note
If you set the *Freeze* parameter to *TRUE* but the Cadence subcircuit has never been netlisted, the Cadence subcircuit will automatically be netlisted the first time an ADS Simulation is attempted.

Setting the netlistFile Parameter

The *netlistFile* parameter is used to specify the location of the ADS netlist of a frozen Cadence Cellview. On the dummy (placeholder) design, set the *netlistFile* parameter to point to the appropriate netlist file. For example:

netlistFile="examples_power_amp_schematic.net"

The default location for storing the ADS netlist of a frozen Cadence Cellview is:

<current_ADS_Workspace_directory>/.DL/

If you want to copy the netlist file elsewhere, set the *netlistFile* parameter to point to the full path and file name of the new location. For example:

netlistFile="/tmp/my_design.net"

Refer to the example schematic in Defining the Freeze Parameter for the location of the *netlistFile* parameter setting.

Using "Freeze" Mode to Simulate a Design in ADS Standalone

You can run Advanced Design System standalone (without Cadence DFII or RFIC Dynamic Link) using a frozen netlist from an earlier RFIC Dynamic Link session (see Freezing Selected Subcircuits). The parameter *Freeze=TRUE* must be set on all dummy designs in ADS that represent Cadence cellviews.

While RFIC Dynamic Link is not designed to operate on a PC, you can take an ADS netlist (created using Dynamic Link on your UNIX workstation) of your Cadence cellview and copy it to your PC for an ADS simulation. This type of operation is typically done for board design. Once you have done some minor configuration, you can then add simulation and control components externally to your design and resimulate on the PC.

To set up and simulate your Cadence cellview in Advanced Design System on a standalone Windows machine:

1. Check to make sure RFIC Dynamic Link is installed on your PC. If it is installed, $HPEESOF_DIR/idf/ael should contain five **.atf files. If those atf files are present but the *DynamicLink** menu is not present in ADS schematic window, select **Tools > Command Line** menu item in ADS Main window and then use the load() command to load each atf file in turn.
2. For each Cadence CellView, copy the netlist file into the .DL subdirectory under your

Workspace directory. Create a .DL directory, if it does not exist. For instance, copy the <UNIX_Workspace_directory>/.DL/examples_power_amp_schematic.net file into your <PC_Workspace_directory>/.DL directory.

3. For each Cadence CellView, copy the corresponding ADS *<lib>_<cell>_<view>* directory tree into into an ADS library under your Workspace directory on PC. For example, copy the examples_lib/examples_power_amp_schematic directory tree to a library under your Workspace on PC.
4. If the Cadence CellView contains design variables, you will need to manually enter them into a VAR block in the top level ADS schematic. To do this:
   - Choose *Data Items* from the Component Pallet.
   - Click the *Var Eqn* block to add the component and use the cursor to place an instance on the schematic. You may continue placing more instances of the *Var Eqn* block, or choose the *Cancel Command And Return To Select Mode* icon to proceed with the next step.
   - Enter the appropriate values into *Var Eqn* block.

It is important to note that any changes made to the design on your standalone machine will not be reflected in your original Cadence cellview. While you may add simulation and control elements externally, the fundamental design should not be changed if you want it to match your original Cadence design.

Compatibility between Advanced Design System and Cadence Tools

Some of the features provided by the RFIC Dynamic Link include support for compatibility issues related to differences between Advanced Design System and the various Cadence tools. This section addresses several of these compatibility issues.

Support for Duplicate Pin Names

It is typical for the top (chip) level schematic to have multiple pins for the same signal, usually power and ground connections. The netlister lists duplicate I/O ports only once in the subnetwork definition and likewise for the nets connected to an instance of the subnetwork. However, the netlister in ADS (which does the top-level netlisting), writes out the multiple connections to ports with the same name, causing a conflict to be reported by the ADS simulator while parsing the final netlist. To eliminate this conflict, when the symbol generator encounters duplicate pin names, it draws only one pin with a given name and issues a warning message. However, duplicate pins at lower levels in the Cadence schematic hierarchy are allowed, because no ADS symbol is involved.

Using Buses

For an example on using buses in RFIC Dynamic Link, refer to *Getting Started Tutorial* (dynlnkug) and use the *power_amp2* Cadence cellview in place of the *power_amp* cellview. Then use the *power_amp2_test* ADS design in place of the *power_amp_test* design. The power_amp2 example is the same as the power_amp example except that it uses bus wires.

For details on creating buses in ADS, refer to *Creating Designs* (usrguide) in the *Schematic Capture and Layout* (usrguide) documentation. For details on using buses within Cadence Virtuoso Design Environment, refer to your Cadence documentation.

Setting up Unnamed Nets

In ADS, unnamed nets begin with an *_net* prefix followed by an integer. All other *net* value are written out to the output dataset during simulation. By default, Cadence tools use the prefix *net* followed by an integer. By default, the dataset can get very large. To avoid this, set the *Net Name Prefix* in the Cadence schematic to *_net* instead of *net*. To set the default *Net Name Prefix* in the Cadence schematic:

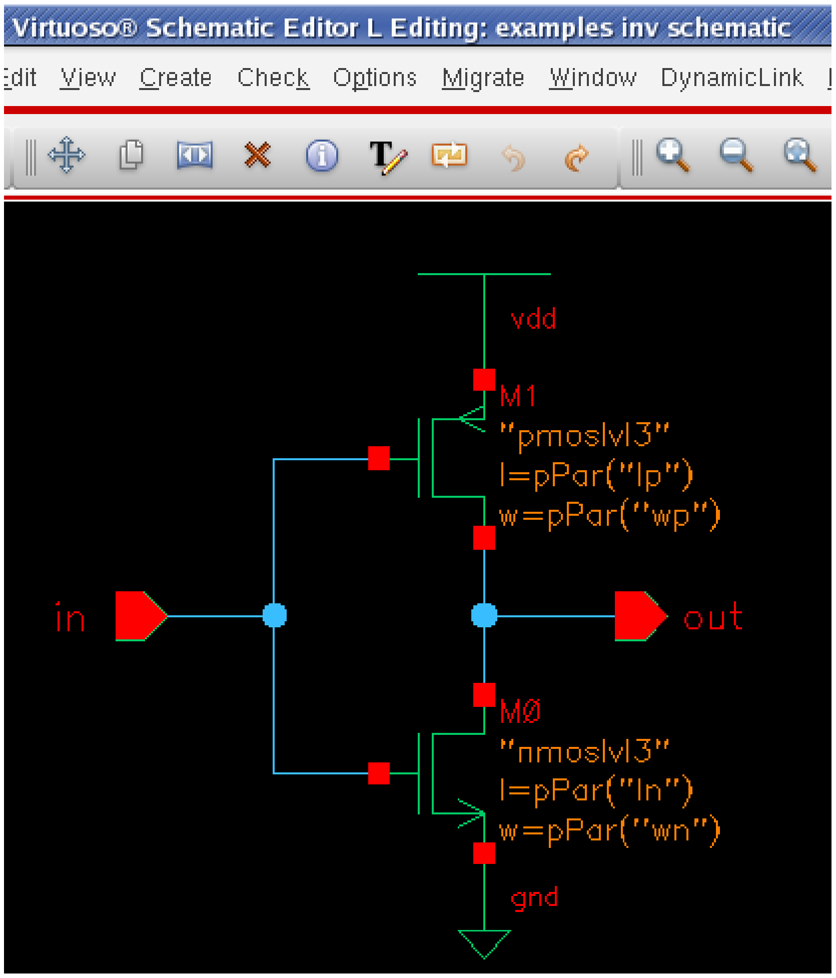
- Choose **Options** > **Editor**.
- In the *Editor Options* dialog box, enter *_net* in the *Net Name Prefix* field
- Click **OK**.
- Choose **Design** > **Check and Save** to save each related Cadence schematic.

Support for pPar and iPar

This section describes the general use of parent parameters (pPar) and instance parameters (iPar).

pPar()

The following figure shows an example of an inverter design that contains two CMOS transistors (M0 & M1).


Virtuoso® Schematic Editor L Editing: examples inv schematic
Edit View Create Check Options Migrate Window DynamicLink
vdd
M1
"pmoslvl3"
l=pPar("lp")
w=pPar("wp")
in
out
M0
"nmoslvl3"
l=pPar("ln")
w=pPar("wn")
gnd

Composer Schematic showing use of *pPar()*

This Composer circuit contains instances whose parameters are defined in terms of parent parameter values using *pPar()*. The parameters in this case are defined as,

l = pPar("ln")

w = pPar("wn")

for M0, and

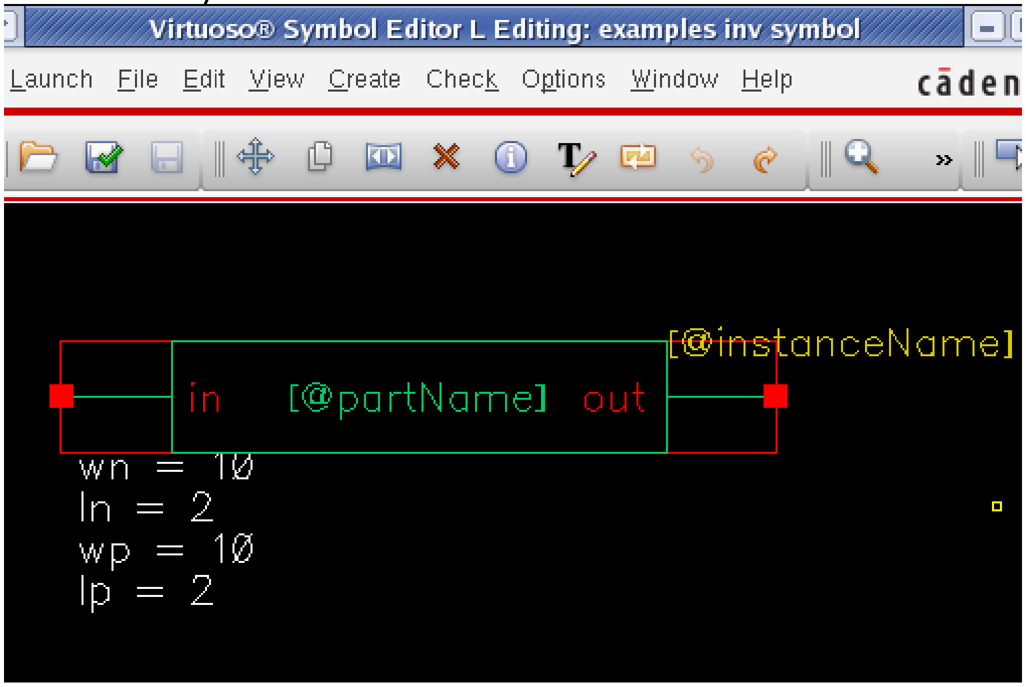
l = pPar("lp")

w = pPar("wp")

for M1.

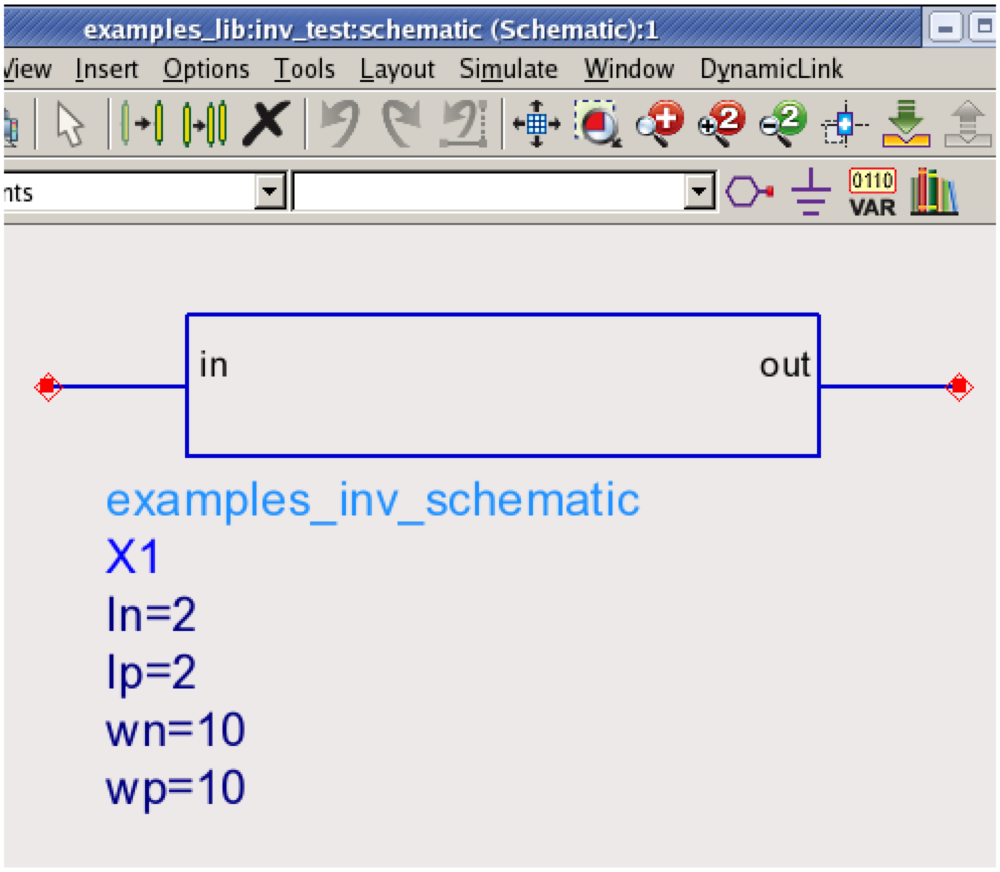
This inverter circuit also has an associated symbol view in Cadence Composer. The symbol view shown in the following figure is equivalent to a black box that displays the input, output and instance properties for the circuit in the previous figure.

The default values of *wn*, *ln*, *wp* and *lp* are displayed in the symbol view along with the associated symbol for the device.


Virtuoso® Symbol Editor L Editing: examples inv symbol
Launch File Edit View Create Check Options Window Help
cāden
[@instanceName]
in
[@partName]
out
wn = 10
ln = 2
wp = 10
lp = 2

Composer Symbol View showing default values for *pPar()*

The Composer symbol is instantiated in ADS via the Dynamic Link where the instance properties also appear in ADS Schematic (see the following figure).


examples_lib:inv_test:schematic (Schematic):1
View Insert Options Tools Layout Simulate Window DynamicLink
in
out
examples_inv_schematic
X1
ln=2
lp=2
wn=10
wp=10

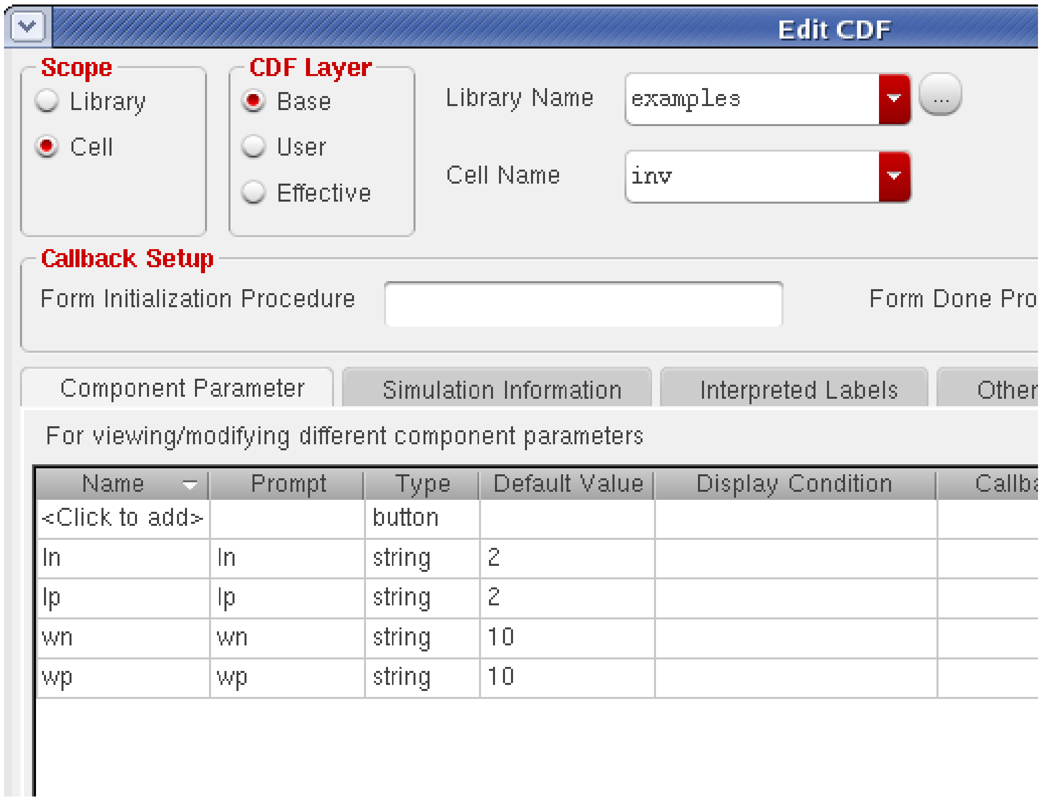
Inherited Symbol in ADS

The parameter values are reflected in the netlist that is sent to the simulator. These values can be viewed and edited using the Cadence *Edit Component CDF* form (see the following figure).

If the Cadence menu option **Design > Create Cellview > From Cellview** is used, the CDF for the schematic will be set up automatically. The Cadence software will traverse the hierarchy looking for *pPar* statements and automatically generate parameters. It will also set up netlisting data for known simulators.

If you are modifying an existing schematic and you have already generated a symbol for your schematic, it may be necessary to manually add the netlisting data for ADS. If this is the case, do the following:

1. Go to the *Simulation Information* section of the *Edit Component CDF* form (see the following figure) and click **Edit**.



Scope: Library [✓ Cell]

CDF Layer: [✓ Base] User Effective

Library Name: examples

Cell Name: inv

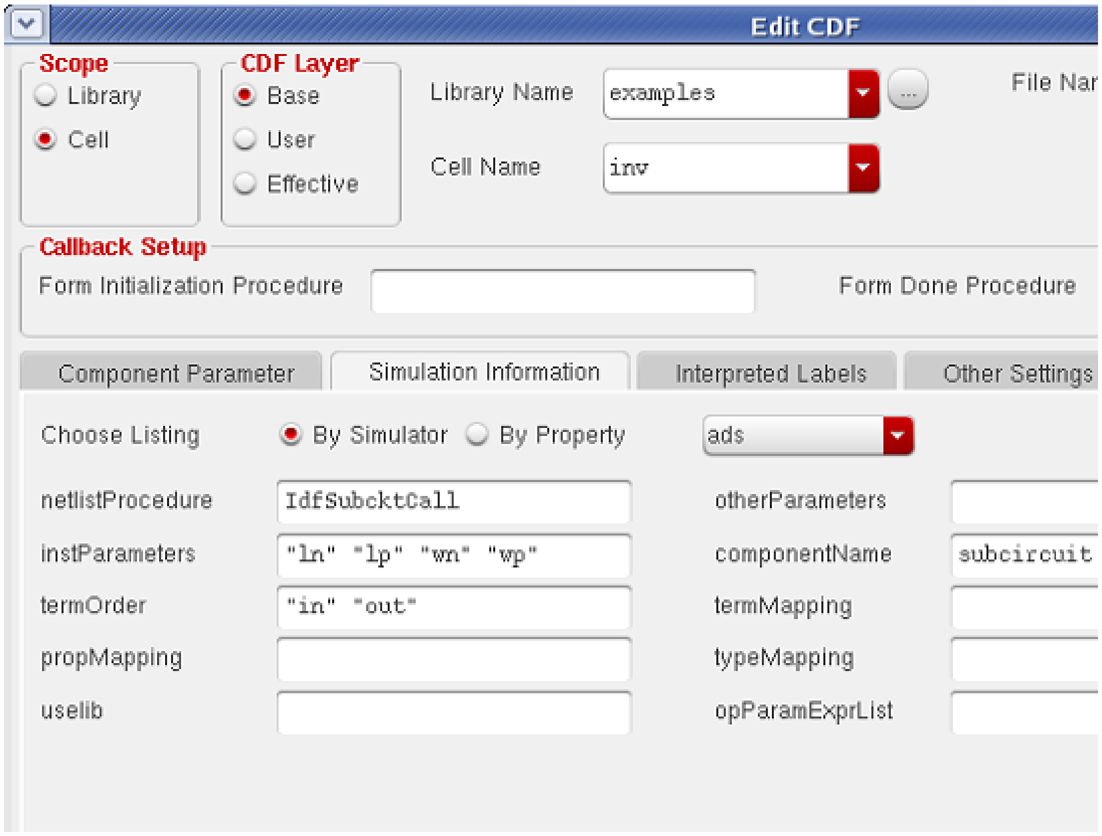
Form Initialization Procedure:

For viewing/modifying different component parameters

Name	Prompt	Type	Default Value	Display Condition	Callba
<Click to add>		button			
ln	ln	string	2		
lp	lp	string	2		
wn	wn	string	10		
wp	wp	string	10		

Edit Component CDF Form

2. When the *Edit Simulation Information* dialog (see the following figure) appears, change the *Simulator* to "ads".



Scope: Library [✓ Cell]

CDF Layer: [✓ Base] User Effective

Library Name: examples

Cell Name: inv

Form Initialization Procedure:

Choose Listing: [✓ By Simulator] By Property — ads

netlistProcedure	IdfSubcktCall	otherParameters	
instParameters	"ln" "lp" "wn" "wp"	componentName	subcircuit
termOrder	"in" "out"	termMapping	
propMapping		typeMapping	
uselib		opParamExprList	

Edit Simulation Information Dialog

Edit Simulation Information Dialog

3. In the *netlistProcedure* field enter *IdfSubcktCall*.
4. In the *instParameters* field enter the parameters you wish to have netlisted for ADS. For the inv circuit, this means entering "wn ln wp lp". The parameter order does not matter.
5. Change the *componentName* field entry to "subcircuit" or leave it blank.
6. Set the *termOrder* field to the order you wish to netlist the terminals in for ADS. You should have one entry for each terminal on your design. For the inv circuit, this is set to "in" "out". You can also quote the names, but it is not necessary.
7. If you wish to back annotate currents, make the appropriate entry in the *termMapping* field. This is done by specifying the name of a terminal, followed by the ADS pin number it will be. For the *inv* example, termMapping entry would be "nil in ": P1" out ": P2"".
   For more information on editing simulator information, refer to *Creating the Netlist Interface* (dynlnklc) in the *Cadence Library Integration* (dynlnklc) documentation.

iPar()

Similarly, the Dynamic Link supports the use of *iPar*. For any given instance, you can define an instance parameter as a function of another parameter of the same instance. For example, if the parameter w in the figure Composer Schematic showing use of pPar() were defined as *w=2*iPar("l")*, then if *l=10*, then *w=20*.

Using Inherited Connections

Inherited connections used in Dynamic Link must all be resolved within the Cadence hierarchy. For example, if you create a schematic in Cadence called *test*, that contains instances that have inherited connections with them, such as *nmos* in analogLib, the default connectivity is used in *test* if no netset properties have been placed on instances in the hierarchy of the top level circuit. If you have hierarchy above the top level Cadence schematic placed in the ADS design environment, you cannot place netset properties on those instances.

Using S-parameter File Devices from analogLib

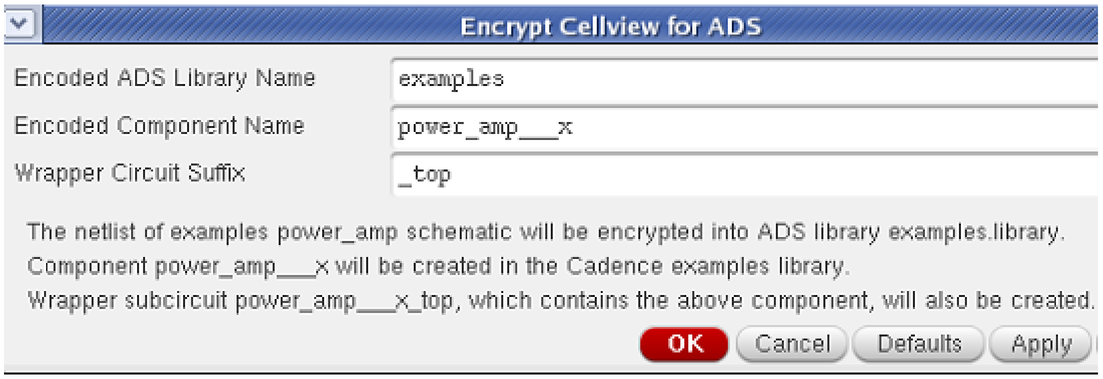
For information on S-parameter file components in the analogLib library, refer to *Using S-parameter File Devices from analogLib* (anloglib) in the *analogLib Components* (anloglib) documentation.

Encoding Cadence Designs for RFIC Dynamic Link

Two methods for encrypting a Cadence cellview are described below. The first method brings up a dialog box but not the second method. The callback function of the dialog box described in Method 1 automates the tedious and error-prone process of Method 2. Using Method 1 is recommended; however, if the automated function fails to meet any need, Method 2, being more flexible, may be applied to achieve the goal.

Method 1: Using the *Encrypt Cellview for ADS* Dialog Box

A relatively easy way of encoding a Cadence design is to use the *Encrypt Cellview for ADS* dialog box, which is opened by selecting the **DynamicLink > Encrypt Cellview for ADS Dynamic Link** menu item in the Cadence Virtuoso schematic window. For example, when started from a window with the *examples power_amp schematic* example cellview open, the dialog box will contain default values and descriptions as shown in the figure below:

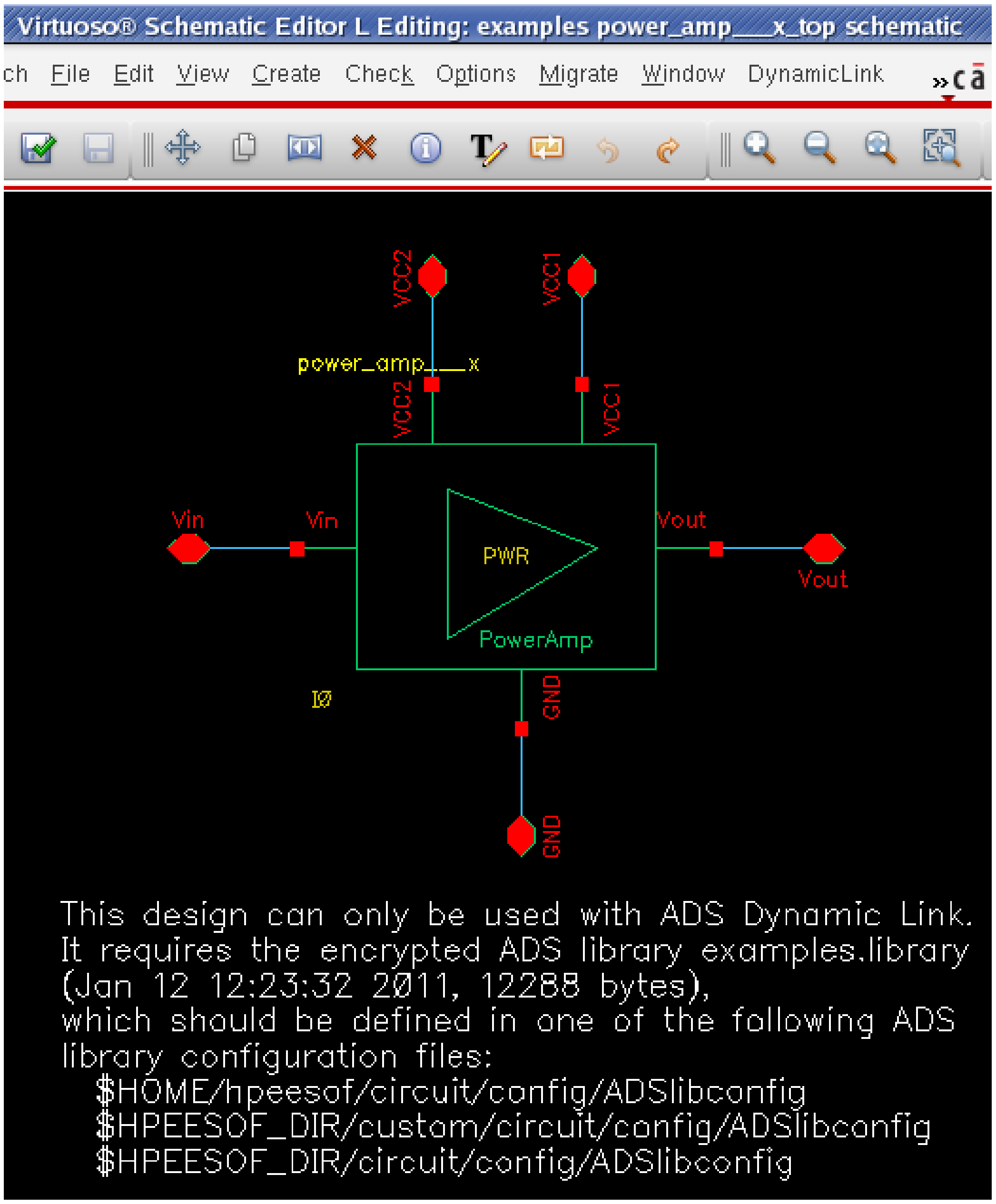


Encoded ADS Library Name: examples
Encoded Component Name: power_amp___x
Wrapper Circuit Suffix: _top

The netlist of examples power_amp schematic will be encrypted into ADS library examples.library.
Component power_amp___x will be created in the Cadence examples library.
Wrapper subcircuit power_amp___x_top, which contains the above component, will also be created.

Selecting the **OK** button in the dialog will encrypt the netlist of the schematic cellview into an ADS library, generate a Cadence component (under the same Cadence library) representing the encrypted ADS library, and generate a wrapper Cadence schematic design that contains the new component. The wrapper Cadence schematic design can then be instantiated in an ADS schematic window in place of the original Cadence design.

Upon successful encrypting the *examples power_amp schematic* cellview, a Cadence schematic window will open and display the generated *power_amp___x_top* wrapper schematic cellview as shown in the figure below:


Virtuoso® Schematic Editor L Editing: examples power_amp___x_top schematic
ch File Edit View Create Check Options Migrate Window DynamicLink
VCC2
VCC1
power_amp___x
VCC2
VCC1
Vin
Vin
PWR
PowerAmp
Vout
Vout
I0
GND
GND

This design can only be used with ADS Dynamic Link.
It requires the encrypted ADS library examples.library
(Jan 12 12:23:32 2011, 12288 bytes),
which should be defined in one of the following ADS
library configuration files:

```
   $HOME/hpeesof/circuit/config/ADSlibconfig
   $HPEESOF_DIR/custom/circuit/config/ADSlibconfig
   $HPEESOF_DIR/circuit/config/ADSlibconfig
```

If an instance of this wrapper schematic cellview is placed in the *Adding an Instance of the Cadence Cellview* (dynlnkug) step of the *Getting Started* (dynlnkug) Tutorial, the simulation results should be identical to those from using the un-encrypted *power_amp* schematic cellview. The difference is that the netlist of the *power_amp* cell is now hidden.

When encrypting the Cadence cellview, the tool adds an entry specifying the full path to the encoded ADS library in your *$HOME/hpeesof/circuit/config/ADSlibconfig* file. Users of your encoded library will have to add a similar library configuration line in an ***ADSlibconfig*** file under one of the following directories in their own environment:

```
$HOME/hpeesof/circuit/config/
$HPEESOF_DIR/custom/circuit/config/
$HPEESOF_DIR/circuit/config/
```

Note
For simulations launched from ADS schematic window, the ADSlibconfig file can also be under the *data* directory of the current ADS Workspace. For simulations started from Linux command line using *adssim* or *hpeesofsim*, the ADSlibconfig file can be placed in the directory where the command is issued.

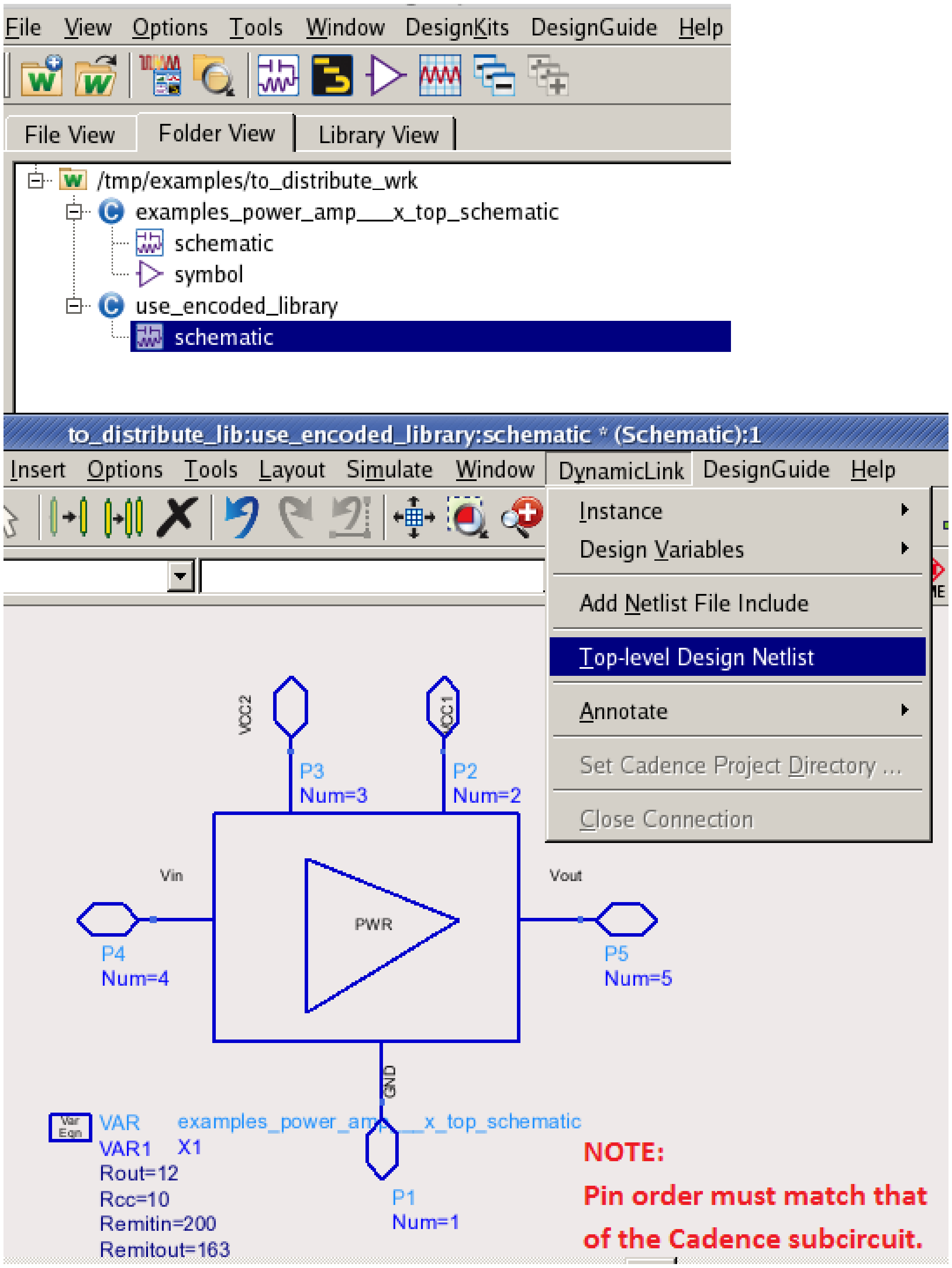
Distributing An Encoded Cadence Design

Once your Cadence design is encoded, users no longer need access to Cadence or ADS RFIC Dynamic Link to use your encoded design in ADS. Follow the list of steps below to distribute your encoded Cadence design to ADS users in an ADS Workspace:

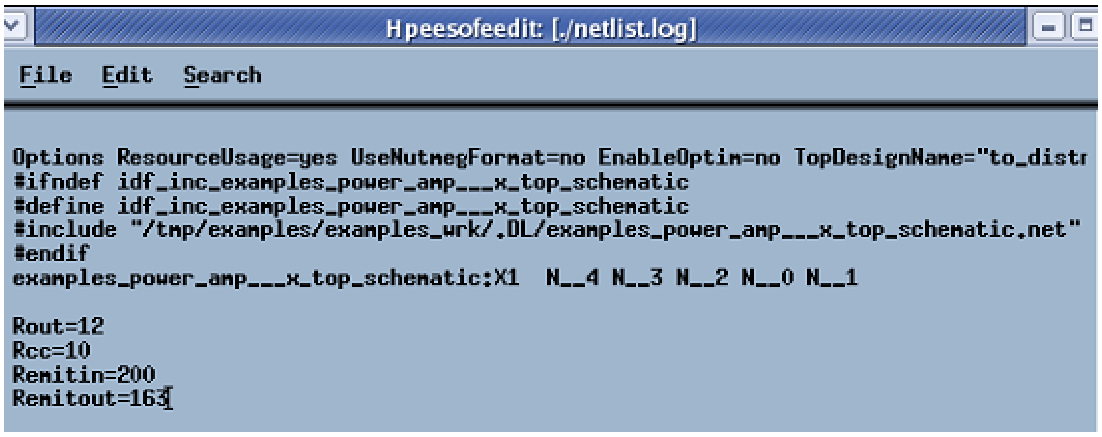
1. Start Dynamic Link from a Cadence schematic window with the wrapper schematic cellview open.
2. Create an ADS Workspace.
3. Create an ADS schematic cell.
4. Place an instance of the Cadence wrapper schematic cellview in the ADS schematic cell.
5. For each port in the Dynamic Link instance, insert a pin and make a connection.

   Note
   Pins must be placed in exactly the same order as in the original design. The first pin placed will have the property Num=1 and will appear first in the netlist.

6. Get design variables from Cadence, if exist.


File View Options Tools Window DesignKits DesignGuide Help
File View
Folder View
Library View
/tmp/examples/to_distribute_wrk
examples_power_amp___x_top_schematic
schematic
symbol
use_encoded_library
schematic
to_distribute_lib:use_encoded_library:schematic * (Schematic):1
Insert Options Tools Layout Simulate Window DynamicLink DesignGuide Help
Instance
Design Variables
Add Netlist File Include
Top-level Design Netlist
Annotate
Set Cadence Project Directory ...
Close Connection
VCC2
VCC1
P3
Num=3
P2
Num=2
Vin
Vout
PWR
P4
Num=4
P5
Num=5
GND
VAR
VAR1
Rout=12
Rcc=10
Remitin=200
Remitout=163
examples_power_amp___x_top_schematic
X1
P1
Num=1
NOTE:
Pin order must match that
of the Cadence subcircuit.

7. Select **DynamicLink > Top-level Design Netlist** to generate the netlist. It is critical to create this 'freeze mode' netlist for your users, so that they will not need to start Dynamic Link and use your original Cadence designs to create a netlist in order to simulate in ADS.



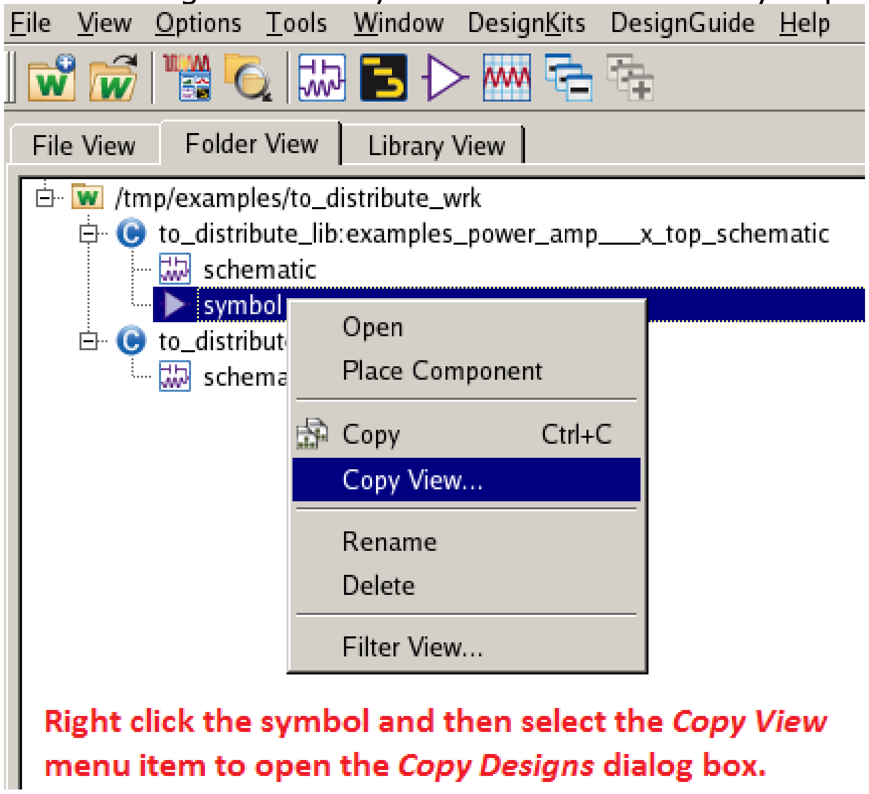
```
Options ResourceUsage=yes UseNutmegFormat=no EnableOptim=no TopDesignName="to_distr
#ifndef idf_inc_examples_power_amp___x_top_schematic
#define idf_inc_examples_power_amp___x_top_schematic
#include "/tmp/examples/examples_wrk/.DL/examples_power_amp___x_top_schematic.net"
#endif
examples_power_amp___x_top_schematic:X1  N__4 N__3 N__2 N__0 N__1

Rout=12
Rcc=10
Remitin=200
Remitout=163
```

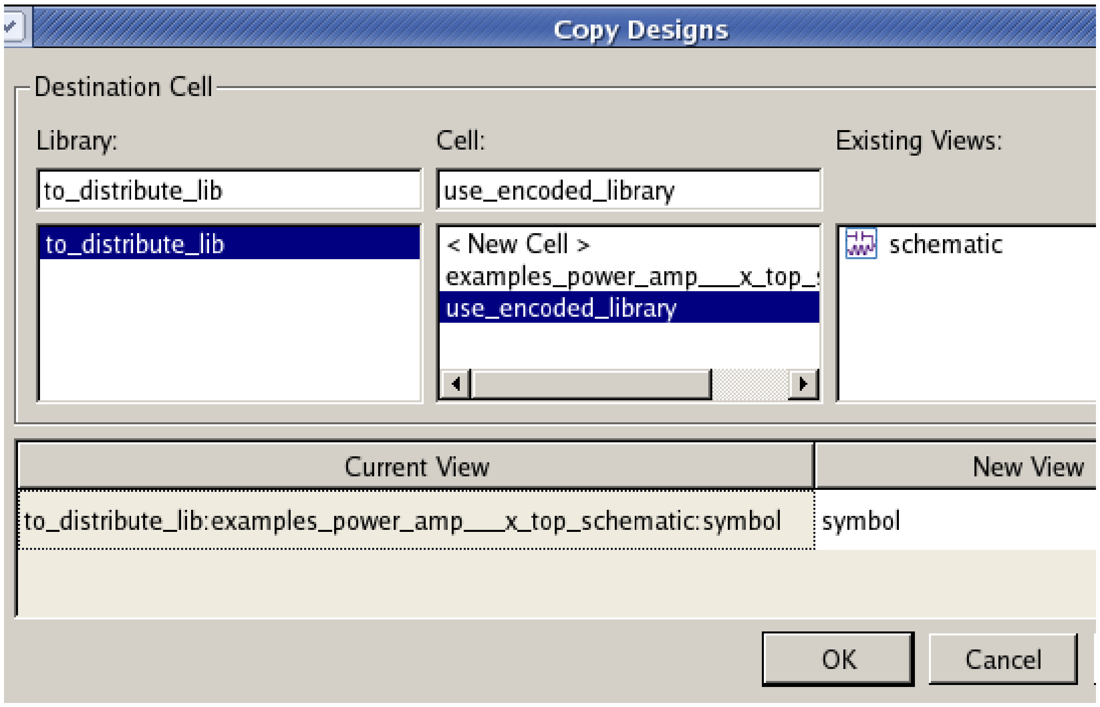
Notice that the netlist includes the netlist file of the Cadence wrapper subcircuit. The included netlist file (which is *examples_power_amp___x_top_scheamtic.net* in this example) is under the .DL sub-directory of the current Workspace. It contains the following lines for accessing the encrypted library:

```
define examples_power_amp___x_top_schematic ( GND VCC1 VCC2 Vin Vout )
 #uselib "examples", "power_amp___x"
 power_amp___x:I0 GND VCC1 VCC2 Vin Vout
end examples_power_amp___x_top_schematic
```

The first field after the **#uselib** pre-processor directive is the encrypted library name, which is the first field in the *Encrypt Cellview for ADS* dialog box. There must be a line defining its full path in one of your *ADSlibconfig* files.

8. Save the schematic cellview.
9. Create a symbol view by copying from the symbol view of the Dynamic Link instance, which was generated by the tool the first time you placed it.


File View Options Tools Window DesignKits DesignGuide Help
File View
Folder View
Library View
/tmp/examples/to_distribute_wrk
to_distribute_lib:examples_power_amp___x_top_schematic
schematic
symbol
to_distribut
schema
Open
Place Component
Copy Ctrl+C
Copy View...
Rename
Delete
Filter View...
Right click the symbol and then select the Copy View menu item to open the Copy Designs dialog box.



Library: to_distribute_lib
Cell: use_encoded_library

to_distribute_lib

< New Cell >
examples_power_amp___x_top_s
use_encoded_library

schematic

Current View	New View
to_distribute_lib:examples_power_amp___x_top_schematic:symbol	symbol

10. Copy the encoded library under $HOME/hpeesof/encode/ to the current Workspace.
11. Archive the Workspace and deliver it to your users.

Your users will need to add one line in one of the *ADSlibconfig* files described above to specify their full path to the encoded library included in the Workspace. In addition to the encoded library, the Workspace contains the following essential items for others to use your encoded library: two components and a Dynamic Link 'freeze mode' netlist file. Either component can be used in any ADS design except that the one generated by Dynamic Link does not contain the list of Cadence design variables.

Including Model Files for Encryption

Spectre model files obtained from the following SKILL function call will be included with the design for encryption:

```
asiGetEnvOptionVal(asiGetTool('spectre) "modelFiles")
```

If the PDK you are using defines the above Spectre environment option but you do not intend to include those Spectre model files in the encoded library, set IDF_INCLUDE_MODEL_FILES=NO in the environment. For example, type the following SKILL command in CIW:

```
setShellEnvVar("IDF_INCLUDE_MODEL_FILES=NO")
```

Method 2: Applying Non-GUI Approach To Encrypting A Cadence Cellview

The long procedure described below has been automated in the callback function of the dialog box explained in Method 1. It is recommended that you avoid this approach unless Method 1 does not work for you.

This example demonstrates the use of Advanced Design System's RF IP Encoder command-line interface with RFIC Dynamic Link. The Cadence and ADS designs provided in the RFIC Dynamic Link Tutorial are adopted here. The *power_amp* Cadence design and the ADS model files that are used in the design will be encoded using the RF IP Encoder, as a component in an ADS library. The encoded component will eventually be used in the power_amp_test ADS design for a simulation. For more information, refer to the *RF Intellectual Property Encoder* (rfipenc) documentation.

Familiarity with the Cadence environment and the RFIC Dynamic Link tutorial is assumed. The RFIC Dynamic Link tutorial is provided in *Getting Started Tutorial* (dynlnkug).

The following is a list of steps to encode a Cadence design and then use the encoded design in an ADS simulation via RFIC Dynamic Link:

1. Make a copy of the Dynamic Link examples directory and start Cadence from the directory created. For example, enter the following commands:

   ```
   cp -r $HPEESOF_DIR/idf/examples/tmp
   cd /tmp/examples
   virtuoso&    # or icms& for IC 5.1.41
   ```

   See *Setting up the Examples Directory* (dynlnkug) for details.
2. Open the power_amp cellview of the examples library in the Cadence schematic window.
3. Choose **Launch > ADS Dynamic Link** (or **Tools > ADS Dynamic Link** for IC 5.1.41) menu item in the Cadence schematic window to start ADS RFIC Dynamic Link.
4. Choose **DynamicLink > Subcircuit Netlist** menu item in the Cadence schematic window to generate the netlist for the power_amp Cadence design.
5. Choose **File > Save As** in the netlist window and save the netlist file as *examples_power_amp_schematic*. For demonstration purpose, it is more convenient to name the netlist file exactly as *examples_power_amp_schematic* here, i.e. name the netlist file as *<libName>_<cellName>_<viewName>* with no extension.
6. Close the Cadence schematic window and the netlist window.
7. Open the examples_power_amp_schematic file in an ACSCII text editor.
8. Insert the two model files examples/models/npnpwa1.scs and examples/models/npnpwa2.scs between the **end** and the **#endif** statements in the examples_power_amp_schematic file. Comments and the redundant *simulator lang* lines, and the trailing mapping() section can be removed from the model files as shown below. Save your changes and quit the editor.

   ```
   ...
   end examples_power_amp_schematic
   model npnpwa1 bjt type=npn bf=130 ikf=0.01085 \
   ```

```
 ise=7.56E-13 ne=2 vaf=25 nf=1.03 tf=8.91e-12 \
 xtf=3.35 itf=0.0217 ptf=18 xtb=2.2 \
 br=5.123 ikr=0.056 isc=2.01e-12 \
 nc=2 var=0 nr=1 tr=1.6e-9 eg=1.11 is=2.15e-16 \
 imax=1 xti=8 tnom=25 nkf=0.5 iss=3.28e-13 ns=1 \
 cjc=1.045e-13 vjc=0.75 mjc=0.5 xcjc=0.2292 \
 fc=0.5 cje=1.935E-13 vje=0.85 mje=0.4 \
 cjs=1.092e-13 vjs=0.7 mjs=0.5 rb=317 irb=0 \
 rbm=10.63 re=1.79 rc=37.6 af=1 \
 bnoisefc=1 struct=vertical
model npnpwa2 bjt type=npn bf=166 ikf=0.02325 \
 ise=1.218E-12 ne=2 vaf=25 nf=1.03 tf=7.97e-12 \
 xtf=3.16 itf=0.0465 ptf=18 xtb=2.2 \
 br=5.123 ikr=0.12 isc=4.08e-12 \
 nc=2 var=0 nr=1 tr=1.6e-9 eg=1.11 is=4.08e-16 \
 imax=1 xti=8 tnom=25 nkf=0.5 iss=4.96E-13 ns=1 \
 cjc=2.13e-13 vjc=0.75 mjc=0.5 xcjc=0.22 \
 fc=0.5 cje=3.123e-13 vje=0.85 mje=0.4 \
 cjs=1.65e-13 vjs=0.7 mjs=0.5 rb=59.2 irb=0 \
 rbm=14.1 re=0.833 rc=22 af=1 \
 bnoisefc=1 struct=vertical
#endif
```

9. Make a temporary working directory. For example, enter

```
mkdir /tmp/myip
cd /tmp/myip
```

10. Copy the netlist fragment to be encrypted to the temporary directory. Make sure the filename is the name that will be used to get the fragment out of the library and it does not contain an extension. For example, enter

```
cp /tmp/examples/examples_power_amp_schematic/tmp/myip/
```

11. Create a new file named ' library.cfg' that contains the following three lines:

```
Library name: power_amp_schematic
Library size: 2
examples_power_amp_schematic examples_power_amp_schematic
```

12. Create another file named ` ADSlibconfig' that contains the following line:

```
power_amp_schematic /tmp/myip/power_amp_schematic.library
```

13. Make sure LD_LIBRARY_PATH (Linux and SunOS) or SHLIB_PATH (HP-UX) is set to include shared libraries required to run ADS. For example, entering the following command at the Korn shell prompt will accomplish this:

```
. bootscripts.sh
```

14. Make sure power_amp_schematic.library does not already exist by entering:

```
rm power_amp_schematic.library
```

15. Enter the ` hpeesofencode' command to create power_amp_schematic.library.
16. Edit or create one of three *ADSlibconfig* files to add a line for mapping the library name to the full path to the.library file. The line to add for this tutorial should read:

```
power_amp_schematic /tmp/myip/power_amp_schematic.library
```

The library mapping line can be added to any of the *ADSlibconfig* files listed below:

```
$HOME/hpeesof/circuit/config/ADSlibconfig
$HPEESOF_DIR/custom/circuit/config/ADSlibconfig
$HPEESOF_DIR/circuit/config/ADSlibconfig
```

You can simply type the following command:

```
cat ADSlibconfig >> $HOME/hpeesof/circuit/config/ADSlibconfig
```

But make sure that no entry named *power_amp_schematic* already exists in any of the *ADSlibconfig* files.

17. Choose **File > New > Cellview** in the Cadence CIW to create a new examples_power_amp_schematic symbol in the examples library, that is, enter the following information in the Create New File form:

Library Name:	examples
Cell Name:	examples_power_amp_schematic
View Name:	symbol

Note that selecting Composer-Symbol in the Tool cyclic button will fill in symbol as the View Name.

18. In the Cadence Symbol Editor window, choose **Add > Import Symbol** to import the power_amp symbol in the examples library. Save the symbol and close the Symbol Editor window.
19. Use the Cadence CDF Editor or `cdfDump()` and `load()` SKILL commands to enter the following ads simInfo (Simulation Information) for the examples_power_amp_schematic cellview:

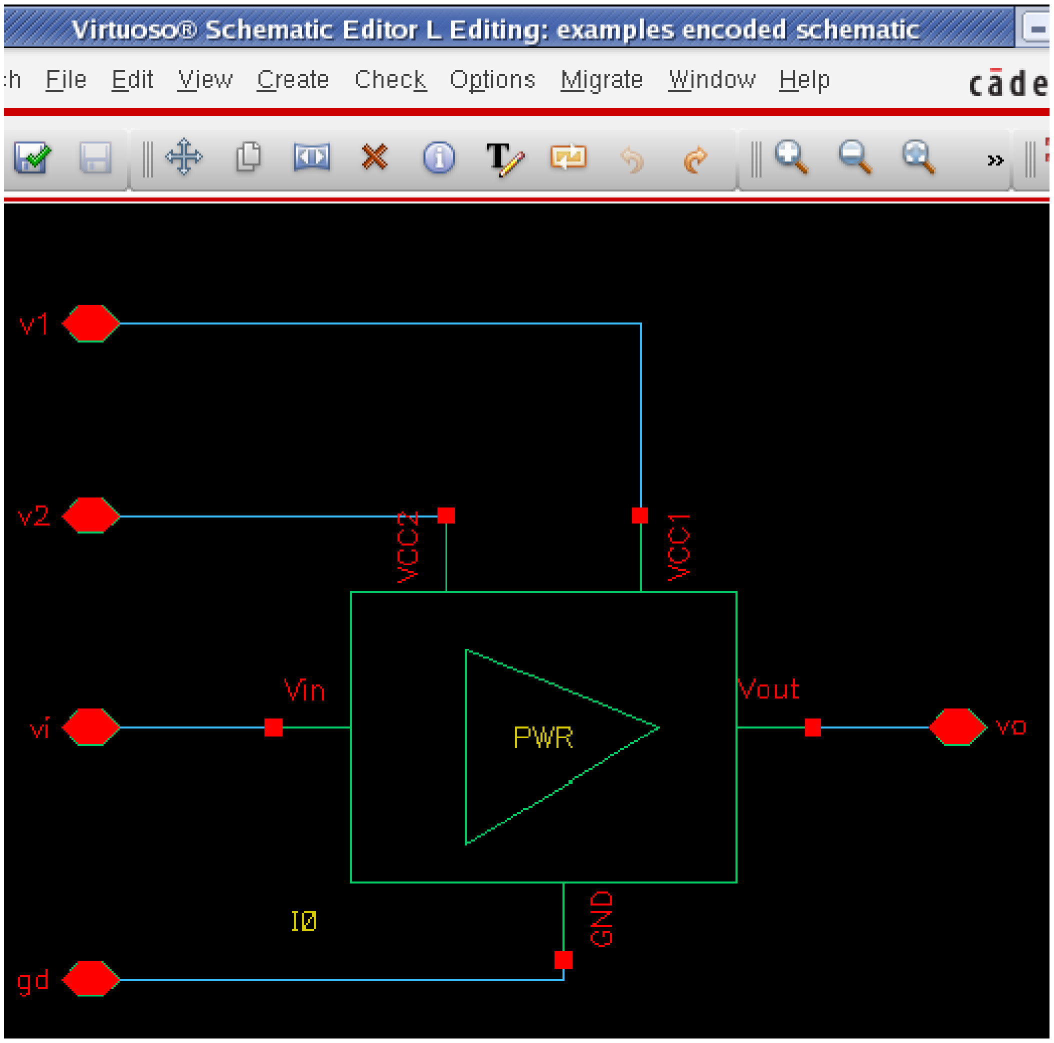
netlistProcedure	ADSsimCompPrim
termOrder	(GND VCC1 VCC2 Vin Vout)
uselib	power_amp_schematic

Note that the library name entered in the *uselib* field must be exactly the same as the name entered in the *ADSlibconfig* file.

20. Create an ads view by copying the examples_power_amp_schematic symbol view to an ads view. One way is to press and hold the middle mouse button at the symbol in the Cadence Library Manager and select Copy in the popup menu. Enter ads in the *View* field of the *To* group in the *Copy View* form and click **OK**.
21. Create a cellview in the Cadence schematic window. For example, enter the following information in the Create New File form:

Library Name:	examples
Cell Name:	encoded
View Name:	schematic

22. Place an examples_power_amp_schematic instance in the schematic window. Add five Input/Output pins and connect one to each pin of the instance as shown below.


Virtuoso® Schematic Editor L Editing: examples encoded schematic
h File Edit View Create Check Options Migrate Window Help
cāde
v1
v2
VCC2
VCC1
Vin
Vout
vi
PWR
vo
I0
GND
gd

23. Choose **Tools > ADS Dynamic Link** in the Cadence schematic window.
24. Choose **DynamicLink > Design Variables** in the Cadence schematic window in which the *encoded* cellview is displayed. Add the following four design variables:

Rout	12
Rcc	10
Remitin	200
Remitout	163

It may be necessary to click the **Copy To** button in the Design Variables form.
25. Choose **Design > Create Cellview > From Cellview** to create a symbol view for the examples/encoded/schematic cellview. Save the symbol created and close the Symbol Editor window.
26. Click the **Check and Save** icon in the schematic window.
27. Open the power_amp_test ADS design in the examples_wrk ADS Workspace.
28. Choose **DynamicLink > Instance > Add Instance of Cellview** to place the examples_encoded_schematic instance in the power_amp_test ADS design. Make an appropriate connection for each of the five pins as demonstrated in the Dynamic Link Tutorial. Make sure the v1 pin, which connects to the VCC1 pin of the examples_power_amp_schematic Cadence cellview, is connected to the SRC1 instance on the right side of the power_amp_test ADS design.
29. Choose **DynamicLink > Design Variables > Get Design Variables** to add the Rout, Rcc, Remitin, and Remitout Cadence design variables to the VAR1 block in the ADS schematic window.

30. Activate the DC and S_PARAMETERS simulation controllers and start an ADS simulation.

    Note
    Unlike in the Dynamic Link Tutorial, the model cards need not be included in a NetlistInclude component since they have already been included in the encoded power_amp_schematic library.

31. Plot S(2,1) to ensure the simulation results are the same as illustrated in the Dynamic Link Tutorial. If the simulation completed without error but the output plot looks different, check the pin order of the Cadence instance.
32. Choose **DynamicLink > Subcircuit Netlist** in the Cadence Virtuoso schematic window and observe the following lines in the netlist file:

    ```
    define examples_encoded_schematic \( gd v1 v2 vi vo \)
    #uselib "power_amp_schematic", "examples_power_amp_schematic"
    examples_power_amp_schematic:I0 gd v1 v2 vi vo
    end examples_encoded_schematic
    ```

    These lines are equivalent to the combination of the original subcircuit definition of the examples_power_amp_schematic and the model card include statements in the netlist of the Dynamic Link Tutorial.
33. Choose **DynamicLink > Close Connection** to end the Dynamic Link session. Exit Cadence.

Encoding Process Summary

When selecting the **DynamicLink > Subcircuit Netlist** menu item in a Cadence schematic window, Dynamic Link defines the Cadence subcircuit name as *<libName>_<cellName>_<viewName>*, which can be renamed as desired in an editor. The key is that a Cadence component with the same name as the subcircuit, with an ads view and appropriate uselib, must be created. The name must contain an underscore character. The whole string with the underscore character will be the encoded component name. The sub-string after the underscore will be the name of the new ADS library that contains the encoded component.

If the **Simulation > Netlist > Recreate** menu item in the RF Design Environment (RFDE) Analog Design Environment is used to create the initial netlist, a subcircuit definition (define *<subckt_name>* and end statements) must be inserted and undesired netlist fragments such as, Options and outputPlan, must be removed.

The encoded design will not work in spectre or other simulators integrated into the Cadence design environment.

In summary, here is the list of tasks to perform for creating and using an encoded Cadence design with Dynamic Link:

1. Create a subcircuit netlist for the Cadence design to be encoded. The subcircuit name must contain an underscore character. For the ensuing context, assuming ABC_XYZ

is the subcircuit name.

2. Merge model cards to be encoded with the above netlist file and save the file as the same name as the subcircuit with no suffix, i.e. ABC_XYZ.
3. Create a `library.cfg' file in a temporary working directory with the following three lines:

```
Library name: XYZ
Library size: 2
ABC_XYZ ABC_XYZ
```

4. Create another file named `*ADSlibconfig'* under the same temporary directory with the following line:

```
XYZ <full_path_to_temporary_directory>/XYZ.library
```

5. Create a Cadence symbol with as many input/output pins as the encoded subcircuit. This symbol should bear the same name as the subcircuit, i.e. ABC_XYZ.
6. Enter `. bootscript.sh` to set up shared library paths for ADS tools.
7. Enter `hpeesofencode' in the temporary directory to encode the ABC_XYZ netlist file into XYZ.library.
8. Copy the Cadence symbol view to create an ads view to indicate it being a primitive component in RFIC Dynamic Link.
9. Add an ads simInfo section to the CDF of the Cadence component. Enter ADSsimCompPrim as the netlistProcedure, the encoded library name (i.e. XYZ) as uselib, and appropriate termOrder entries that match the subcircuit definition.
10. Create a Cadence subcircuit to encapsulate an instance of the above primitive component. This step is not necessary for RFDE.
11. Instantiate an instance of the Cadence subcircuit cellview in the ADS schematic window via RFIC Dynamic Link to use the encoded Cadence design in ADS; or, simply instantiate an instance of the Cadence primitive component (i.e. the ABC_XYZ symbol) to use the encoded Cadence design in RFDE.
12. To distribute the encoded design to RFIC Dynamic Link users, deliver the encoded library (XYZ.library), the primitive cellview (ABC_XYZ), and the Cadence wrapper subcircuit created for encapsulating the primitive cellview. For RFDE users, only the first two items are necessary. The original Cadence design and its associated model cards are protected in the encoded library.
    Note that the user of the encoded library will need to add an entry with appropriate full path to define the encoded library in one of the user's *ADSlibconfig* files listed below:

```
$HOME/hpeesof/circuit/config/ADSlibconfig
$HPEESOF_DIR/custom/circuit/config/ADSlibconfig
$HPEESOF_DIR/circuit/config/ADSlibconfig
```

Note
The contents of these *ADSlibconfig* files must contain only ASCII text. Some Windows applications may introduce invisible encoding characters without your knowledge. These non-ASCII endcoding characters will cause a library name to be interpreted incorrectly resulting in a failure that may be very difficult to diagnose. It is highly recommended that you use an ASCII text editor such as vi or emacs to edit these files.

If you will deliver the library to users who will only use it on a PC, then instantiate

the Cadence wrapper instance in an ADS schematic window via Dynamic Link, and then set its Freeze Mode to *yes* after generating its ADS netlist. You can create an ADS Workspace with a design containing this instance and its design variables (in VAR1), place the encoded library under this ADS Workspace, and then send the archive of this ADS Workspace to your users. You will not need to send the Cadence library for PC (Dynamic Link Freeze Mode) users. The ADS design should contain:
the ADS encoded library *<cell>_<view>.library* and
the netlist file <Workspace_directory>/.DL/*<lib>*_*<cell>*_*<view>*.net
Note that the *<lib>*_*<cell>*_*<view>*.net file is the Cadence wrapper circuit's Freeze Mode netlist.

Using RFIC Dynamic Link

This section describes the procedures for:

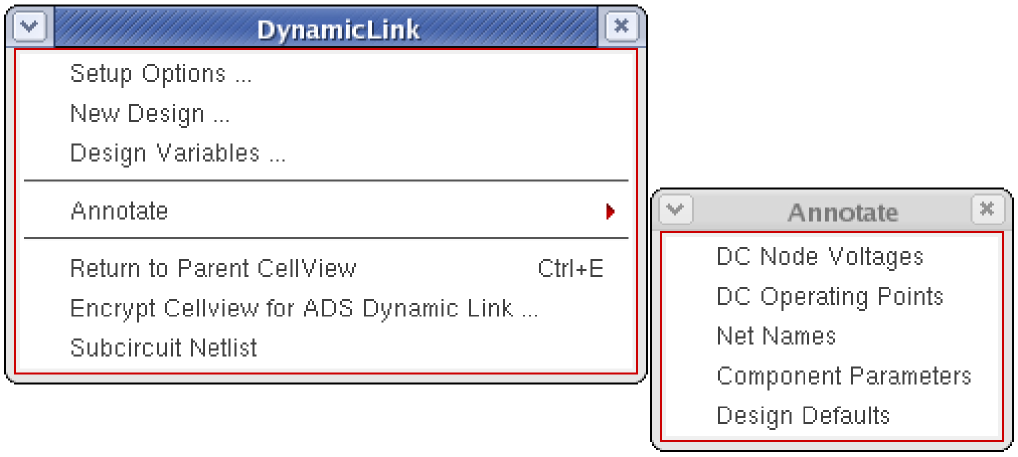
- Launching Advanced Design System
- Adding an Instance of a Cadence Design
- Pushing into the Design Hierarchy
- *Popping out of Cadence schematic window*
- Using Design Variables
- Adding Model Files
- Annotating a Cellview

Launching Advanced Design System

To run Advanced Design System from the Cadence Virtuoso Schematic window using RFIC Dynamic Link:

1. In the Cadence Virtuoso Schematic window, open the desired cellview.
2. Choose **Launch > ADS Dynamic Link** from the Cadence Virtuoso Schematic window. The Advanced Design System Main window appears in the upper left corner of your screen followed, to the right, by an empty ADS Schematic window (this may take some time).

The Cadence Virtuoso schematic window displays a **DynamicLink** pull-down menu. This menu provides some familiar, useful *Virtuoso Analog Design Environment* interface functionality. For further information about these options, consult your Cadence documentation.

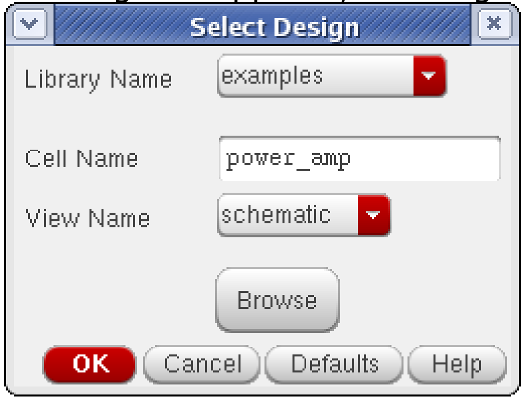


Note
The terminal output (*stderr*) of ADS gets redirected to the file *idf.log* in the $HOME/idflog/ directory. Once the link is started, subsequently opening a Cadence design will not involve the overhead of re-starting ADS, but you will need to select **Launch > ADS Dynamic Link** just to see the **DynamicLink** pull-down menu.

Adding an Instance of a Cadence Design

To add an instance of a Cadence design to an ADS test schematic, choose **DynamicLink > Instance > Add Instance of Cellview** .

A dialog box appears, allowing the selection of a Cadence design.



If a symbol already exists for the design in Cadence, the symbol geometry is duplicated in ADS; otherwise the Cadence symbol generator is automatically invoked to generate a Cadence symbol, which is then automatically duplicated in ADS.

Note
The generated symbol can be edited and modified if needed. If aesthetics are a concern, it is recommended that the symbol be manually created in Cadence and then automatically replicated in ADS as described above. The symbol of the Cadence design is given the following nomenclature *<library>* _ *<cell>* _ *<view>* . For example, *examples_power_amp_schematic* . There is a skeleton schematic of the Cadence design that also gets created in ADS. This is used as a *placeholder* to link with the actual Cadence design; you do not need to edit this.

Pushing into the Design Hierarchy

To view a design deeper in the Advanced Design System schematic hierarchy:

1. Select the component you want to push into in the ADS Schematic window.
2. Choose the *Push Into Hierarchy* icon . This downward arrow icon is located below the DynamicLink menu item in the ADS schematic window tool bar.
3. If the selected component is a Cadence cellview instance, the corresponding Cadence cellview is opened in the Cadence design editor. If this view is already open, it is simply raised to the top of the window stack.

Popping out of Cadence schematic window

Once you have pushed into a Cadence schematic window from a Cadence cellview instance in an ADS schematic window, you can return to the ADS design by choosing **DynamicLink > Return to Parent Cellview** menu item in the Cadence schematic window.

If you have set a Cadence schematic window bindkey for the Cadence **Edit > Hierarchy > Return** menu item, you can also use that bindkey to return from a Cadence schematic design to an ADS schematic design (provided that you have pushed down from ADS to Cadence earlier in the current Dynamic Link session.)

Using Design Variables

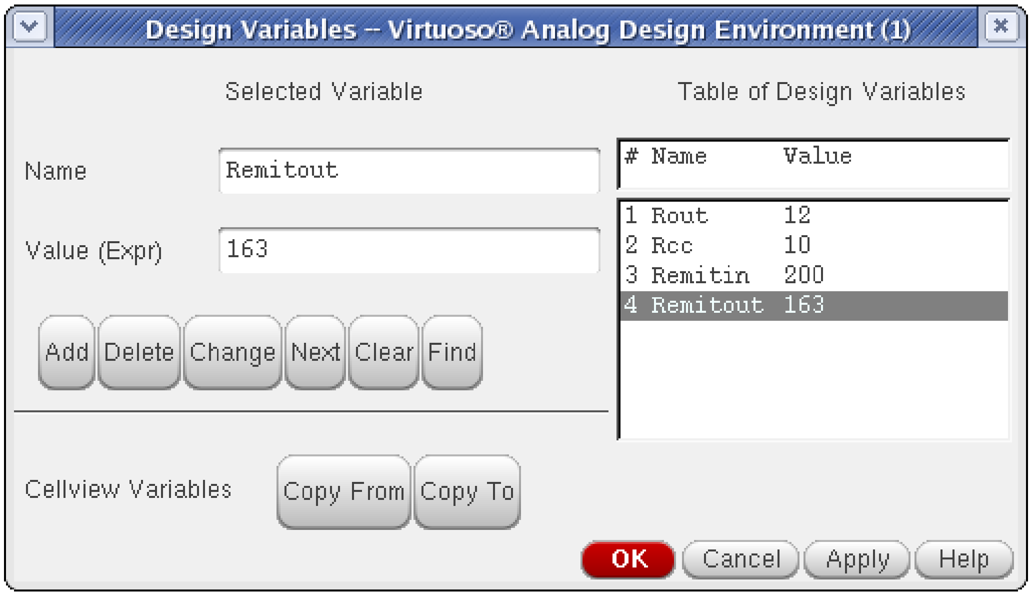
This section describes how to add and edit design variables in Advanced Design System and also update your Cadence design variables.

Cadence *Virtuoso Analog Design Environment* design variables are intended to be global in the context of a particular Artist session or Cellview. When you select **DynamicLink > Design Variables > Get Design Variables** , these variables are automatically mapped to corresponding variables in a *VAR* component in the Advanced Design System schematic. This mapping ensures that these variables can be used for optimization or statistical analysis in ADS.
All the design variables for each Cadence design are put into a single *VAR* component. Each time the menu item **DynamicLink > Design Variables > Get Design Variables** is selected, this component is updated with the most recent values from Cadence.

Adding and Editing Design Variables

To add or edit a design variable for the ADS schematic:

1. From the Cadence Virtuoso schematic window menu bar, choose **DynamicLink** > **Design Variables** . The Cadence Design Variables form appears.



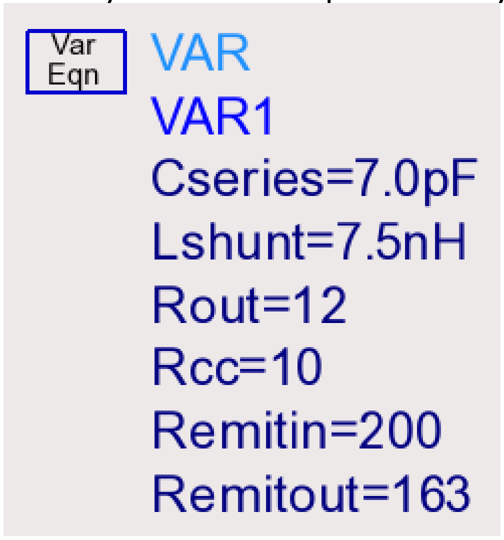
Name: Remitout

Value (Expr): 163

#	Name	Value
1	Rout	12
2	Rcc	10
3	Remitin	200
4	Remitout	163

For more information one using this form, refer to *Design Variables and Simulation* in your Cadence documentation.

2. In the ADS Schematic window, choose **DynamicLink > Design Variables > Get Design Variables** . This places a corresponding *VAR* component on the ADS schematic containing the design variables from Cadence. If the *VAR* component already exists it is updated only with variables and values that are not already there.



```
VAR
VAR1
Cseries=7.0pF
Lshunt=7.5nH
Rout=12
Rcc=10
Remitin=200
Remitout=163
```

VAR Block corresponding to Design Variables

Note
There is no way to distinguish a design variable of the same name coming from different Cadence cellviews. If a variable has different values in different cellviews, the value sent to ADS is chosen arbitrarily. Non-alphanumeric characters, like parentheses, in variable expressions must be preceded by a backslash (`\').

Updating Cadence Design Variables

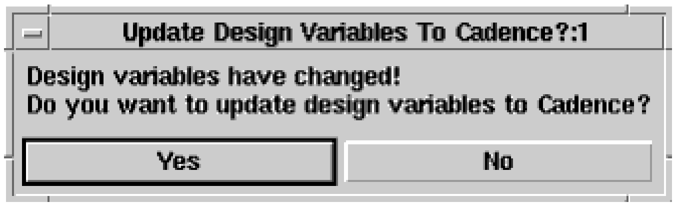
To update your Cadence design variables:

1. In the Advanced Design System Schematic window, choose **DynamicLink > Design Variables > Update Design Variables** .

> **Note**
> Design variables can be used for optimizations and sweeps in ADS. Variables used for this purpose should not be assigned a value or expression in Cadence; the Value (Expr) text entry box should be left blank.

Closing the Cadence Connection

If you have changed your design variables in Advanced Design System and attempt to close the Cadence connection before updating your design variables in Cadence, an *Update Design Variables To Cadence* message will appear prompting you to update design variables.


Update Design Variables To Cadence?:1

Design variables have changed!
Do you want to update design variables to Cadence?

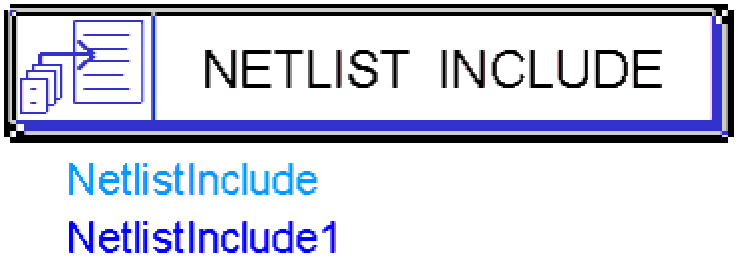
Adding Model Files

This section describes how to use the *NetlistInclude (Netlist File Include Component)* (ccsim) in RFIC Dynamic Link. The NetlistInclude component is provided as a means of utilizing external files in your design.

Adding a Netlist File Include Component

To place an instance of the NetlistInclude Component:

1. From the top-level ADS schematic window, choose **DynamicLink** > **Add Netlist File Include** . An instance of the NetlistInclude is attached to your cursor.
2. Move the cursor to where you want to place the component, then single click. A *Netlist Include* symbol is placed on the schematic as shown in the following figure.

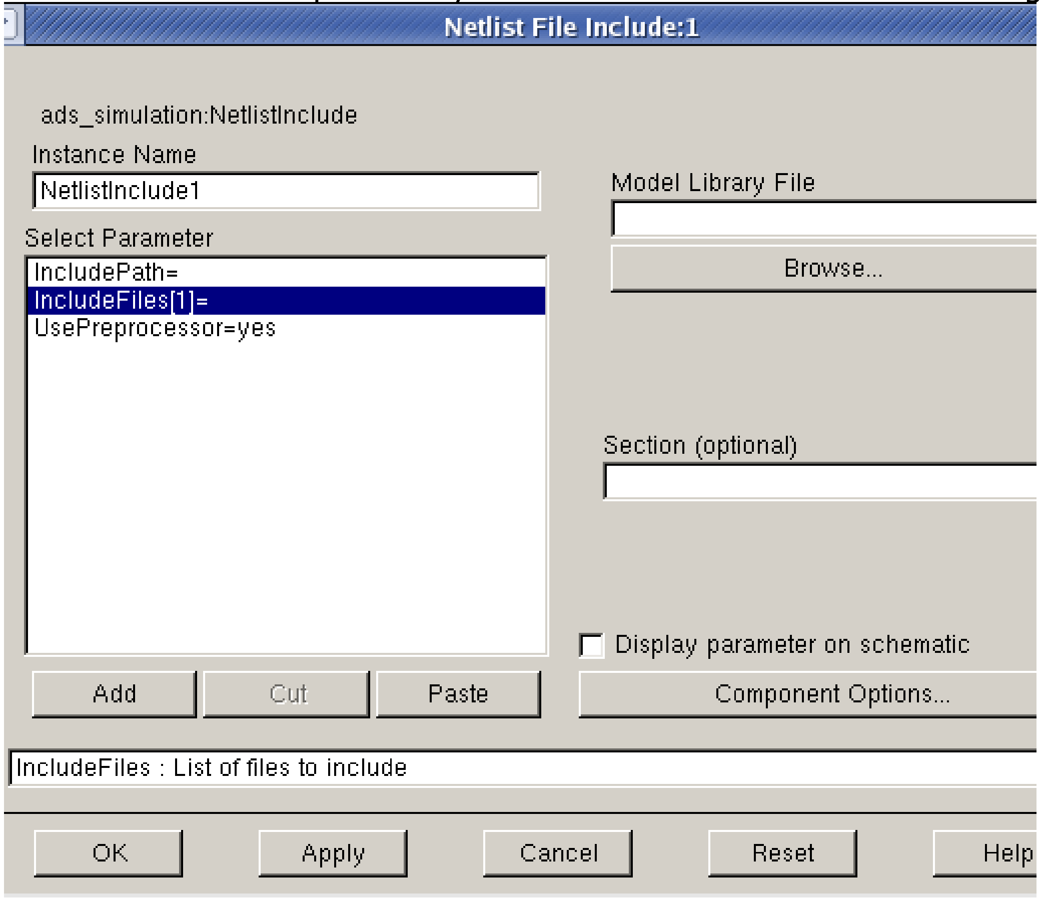


The Netlist Include Component Symbol

> **Note**
> Only *one* Netlist Include Component can be placed in a Dynamic Link design. This is to ensure that files are not multiply included (this cause's redefinition errors within the ADS simulator).

Accessing the Netlist File Include Dialog

To access and edit information in the Netlist File Include component, double click the Netlist Include component symbol. The *Netlist File Include* dialog box appears.


Netlist File Include:1

ads_simulation:NetlistInclude

Instance Name: NetlistInclude1

Select Parameter:
IncludePath=
IncludeFiles[1]=
UsePreprocessor=yes

Model Library File:

Section (optional):

Display parameter on schematic

IncludeFiles : List of files to include

Select Parameter

The *Select Parameter* list box displays a list of three parameters that enable you to create your include definition. Refer to each of the sections listed for detailed information on defining these parameters.

- Set the path to where your model files are located (IncludePath).
- Select the model files to include (see IncludeFiles).
- Enter an optional section designator (see Section (optional)).
- Determine the preprocessor setting (see UsePreprocessor).

IncludePath

The *includePath* parameter is a space delimited search path that is used to locate included model files. The include path needs to be set up for the simulation machine in order to work properly. However, there is an issue with this. The netlister searches through the include path to find files, and then outputs the values as expanded full paths (the simulator requires this). If the expanded full path on the netlisting machine is different from the expanded full path of the simulation machine, the simulator will not find the file to be included. If you want to do remote simulations, ensure that the expanded full path of your included file is the same on the netlisting machine and the simulation machine. Note that, in directory names, path prefixes such as '.', '..', '~', and '$' all have the usual UNIX interpretation.

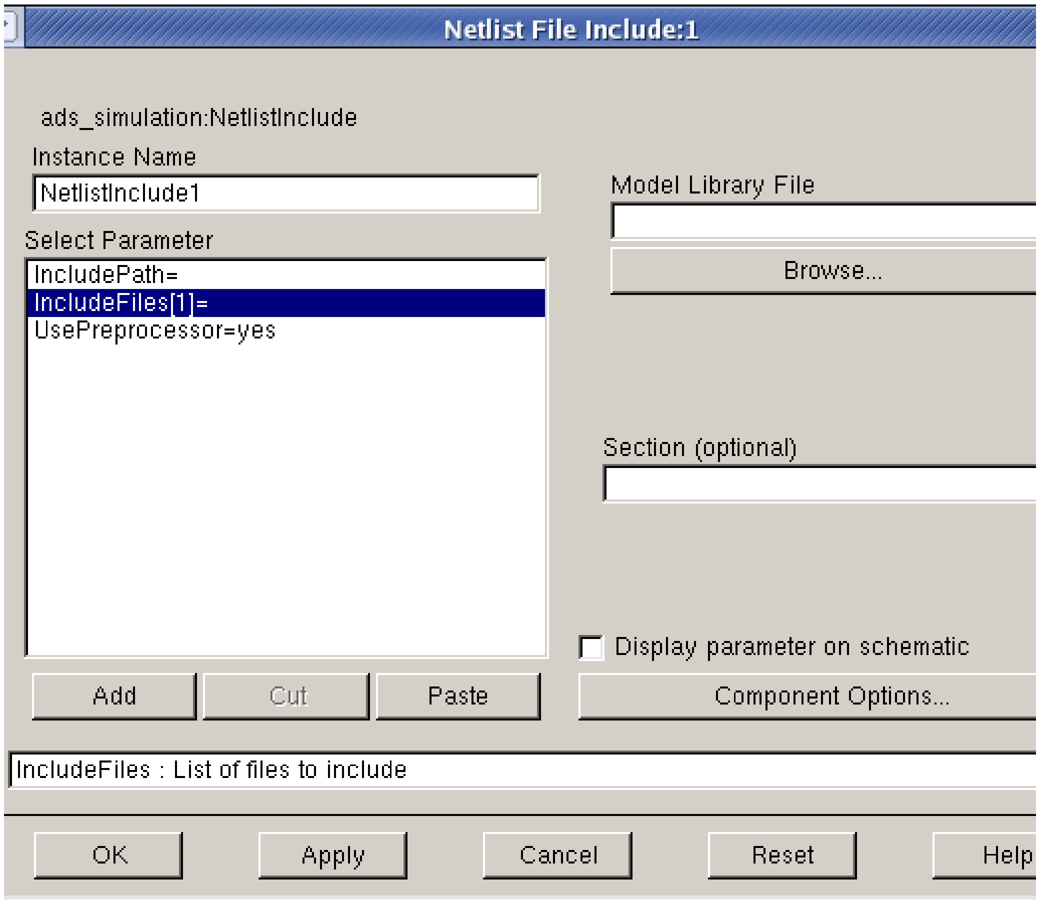
To enter a group of include paths:

1. Click the *includePath=* parameter in the *Select Parameter* list box.
2. Enter the name of the search path in the *Search Path for included files (space delimited)* field separating each search path by a space.

IncludeFiles

This parameter enables you to build a list of model files that you want to include.To specify the path and filename of a model file to include:

1. Click the *IncludeFiles[n]=* in the *Select Parameter* list box. This activates the *Model Library File* selection field.



ads_simulation:NetlistInclude

Instance Name: NetlistInclude1

Select Parameter:
IncludePath=
IncludeFiles[1]=
UsePreprocessor=yes

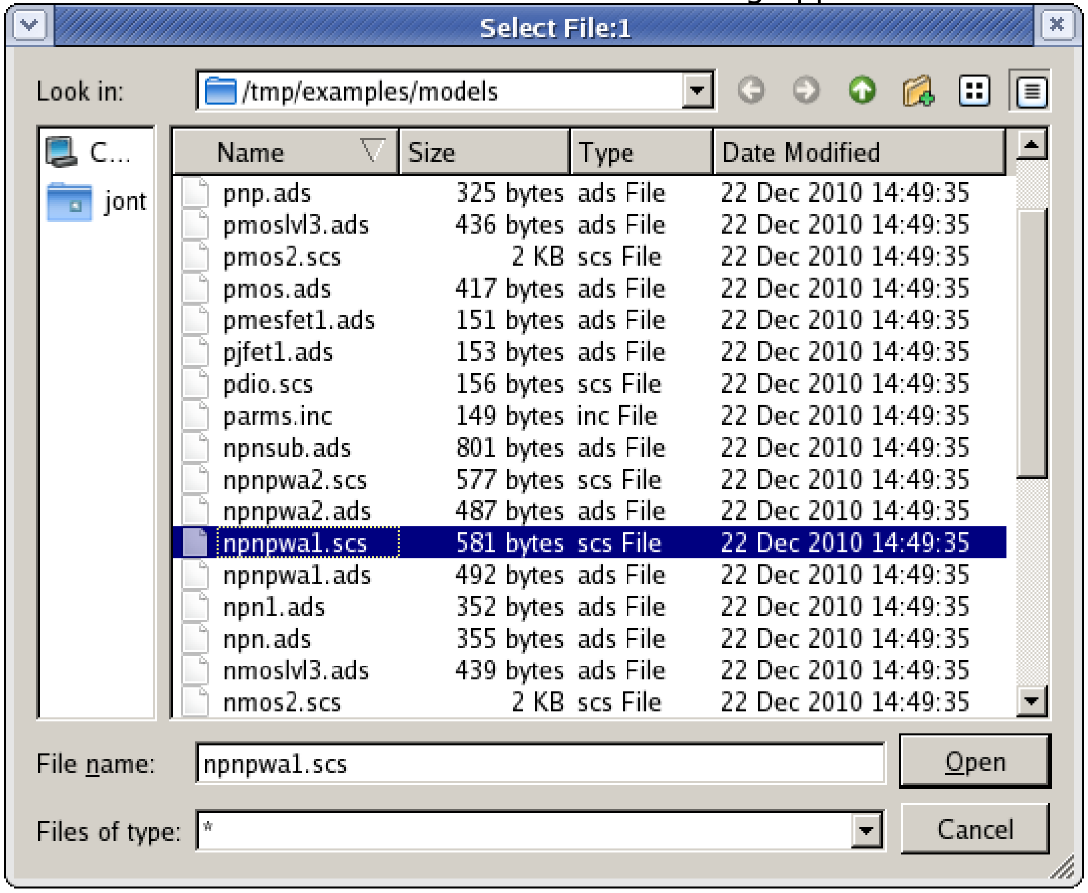
Model Library File

Section (optional)

Display parameter on schematic

IncludeFiles : List of files to include

2. Click the **Browse** button. The *Select File* dialog appears.



Look in: /tmp/examples/models

Name	Size	Type	Date Modified
pnp.ads	325 bytes	ads File	22 Dec 2010 14:49:35
pmoslvl3.ads	436 bytes	ads File	22 Dec 2010 14:49:35
pmos2.scs	2 KB	scs File	22 Dec 2010 14:49:35
pmos.ads	417 bytes	ads File	22 Dec 2010 14:49:35
pmesfet1.ads	151 bytes	ads File	22 Dec 2010 14:49:35
pjfet1.ads	153 bytes	ads File	22 Dec 2010 14:49:35
pdio.scs	156 bytes	scs File	22 Dec 2010 14:49:35
parms.inc	149 bytes	inc File	22 Dec 2010 14:49:35
npnsub.ads	801 bytes	ads File	22 Dec 2010 14:49:35
npnpwa2.scs	577 bytes	scs File	22 Dec 2010 14:49:35
npnpwa2.ads	487 bytes	ads File	22 Dec 2010 14:49:35
npnpwa1.scs	581 bytes	scs File	22 Dec 2010 14:49:35
npnpwa1.ads	492 bytes	ads File	22 Dec 2010 14:49:35
npn1.ads	352 bytes	ads File	22 Dec 2010 14:49:35
npn.ads	355 bytes	ads File	22 Dec 2010 14:49:35
nmoslvl3.ads	439 bytes	ads File	22 Dec 2010 14:49:35
nmos2.scs	2 KB	scs File	22 Dec 2010 14:49:35

File name: npnpwa1.scs

Files of type: *

3. Double-click as needed to locate the directory containing your model file or enter the full path and file name in the *Selection* field. Click **OK** to return to the Netlist File

Include dialog.

4. Once selected, the filename of the model file is displayed in the Library Model File field. Note that the path is appended to the *includePath* parameter and the file name is added to the *IncludeFile* parameter definition.
   Example:

   includePath= *Path1 Path2 ... ModelFilePath*

   IncludeFiles[*n*]= *filename*
5. To add additional model files, click **Add** . This creates additional model file parameter definitions in the *Select Parameter* list box. Repeat steps 1 through 4 to define the path and file name. You can continue adding model file parameters as needed. You can also use the **Cut** and **Paste** buttons to move or delete any model file parameters.

Section (optional)

You may only have a single file for each *IncludeFiles[n]* parameter - unlike the prior parameters - this is not a space delimited list. Each model file can have a *Section* designator. This enables you to include only a portion of a model file for *corner analysis* , provided your model file has been set up properly. The section designator is optional; if it is left empty, the entire file will be included (provided it has no dependencies on needing a particular section set up).
To properly set up a model file to utilize the section directive, you must bracket the sections using **#ifdef** *<section>/* **#endif** C- Pre-Processor(CPP) directives. The netlister automatically defines and undefines a variable with the name section before and after the **#include** statement. As an example, if you wanted to have a file with corner cases, and had a *Nominal* section, you would make the file as follows:

```
#ifdef Nominal
; Nominal section
R:R1 in out R=50
#endif
```

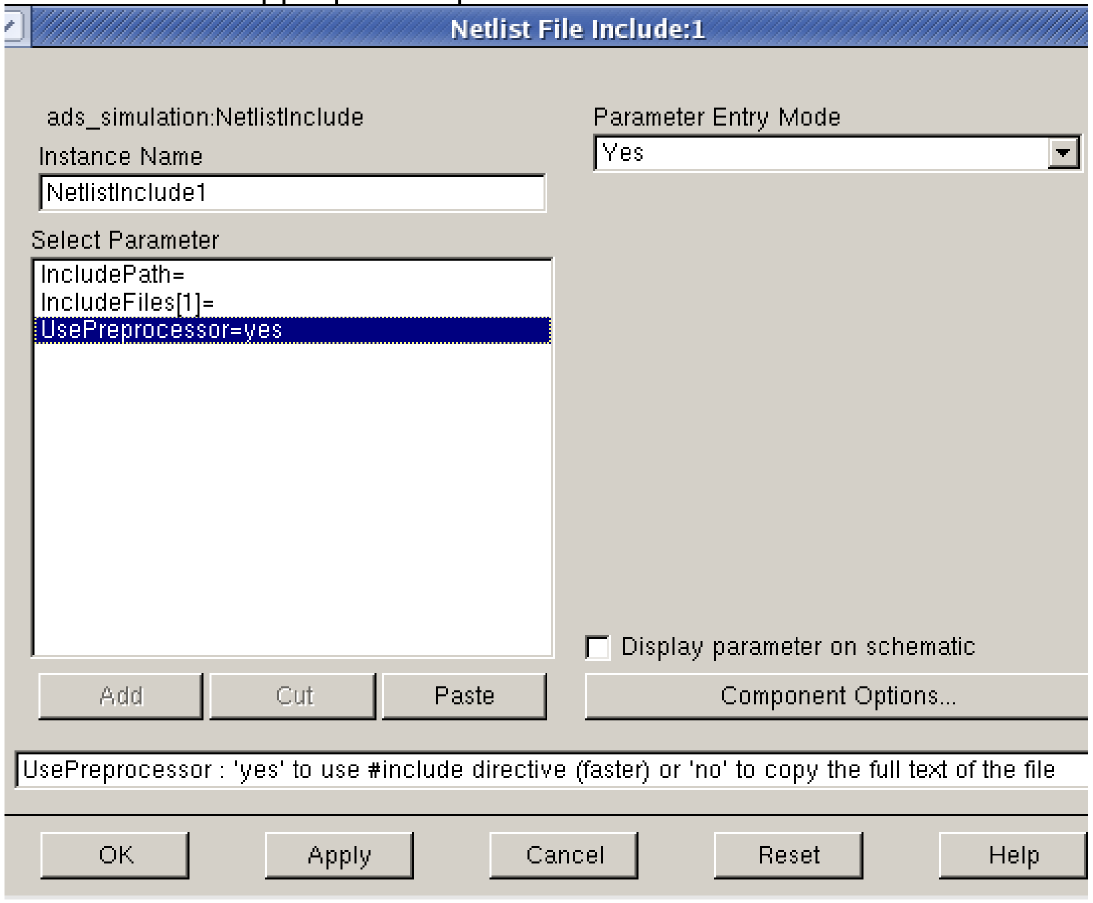
If the same library file is named, with a different section, a single **#include** is generated, with multiple **#define** statements around it.

UsePreprocessor

The *UsePreprocessor* parameter defaults to " *yes*" . Setting this parameter to *yes* causes a preprocessor directive (**#include**) to be used which results in faster behavior by not copying the full text of the file. This parameter can also be set to *no* to provide a slower, but more compatible, full-text inclusion behavior.
To set the UsePreprocessor parameter,

1. Click the *UsePreprocessor=yes* parameter in the *Select Parameter* field of the Netlist File Include dialog box. The *Parameter Entry Mode* pull-down menu appears.

2. Click the *Parameter Entry Mode* pull-down menu in the Netlist File Include dialog box and select the appropriate option.



ads_simulation:NetlistInclude
Instance Name: NetlistInclude1
Parameter Entry Mode: Yes
Select Parameter:
IncludePath=
IncludeFiles[1]=
UsePreprocessor=yes
Display parameter on schematic: ☐
UsePreprocessor : 'yes' to use #include directive (faster) or 'no' to copy the full text of the file

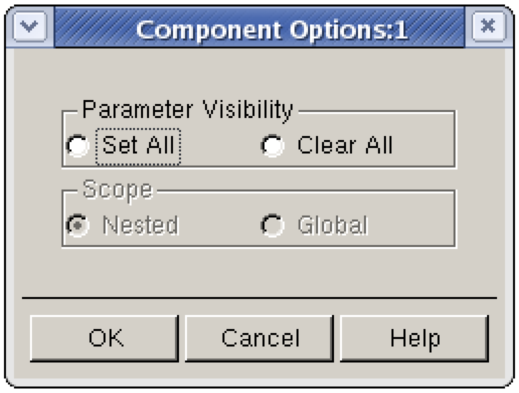
The *UsePreprocessor* parameter was developed for delivering increased speed to old ADS designs that use the *geminiInclude* , *spiceInclude* or *csvInclude* component and for delivering increased flexibility to old ADS designs using the *idfInclude* component.

Display parameter on schematic

The *Display parameter on schematic* check box in the Netlist File Include dialog box is used to list the individual parameters and their associated values on the schematic. If you want to display the parameters, activate the check box.

Component Options

The *Component Options* dialog box enables you to change the visibility of the component parameters on a schematic and/or reference items in hierarchical designs. To access the Component Options, click the Component Options button on the Netlist File Include dialog box. A *Component Options* dialog box appears.



Changing the Visibility of Component Parameters on a Schematic

You can change the visibility status of all parameters of the Netlist File Include component through the Component Options dialog box.

- Set All-Displays all parameters for this component on the schematic. Use this option to display all, or almost all, parameters for this component. To display most-but not all-parameters, select *Set All* and then go back and turn off the display of individual parameters as desired.
- Clear All-Clears the display of all parameters for this component from the schematic. Use this option to turn off the display of all, or almost all, parameters for this component. To display a small subset of parameters, select *Clear All* and then go back and turn on the display of individual parameters as desired.

Referencing VAR Data Items and Model Items in Hierarchical Designs

The *Scope* option applies to the VAR (Variables and Equations) data item and most model items (such as R_Model, BJT_Model, BSIM3_Model). Exception: it does not apply to multi-layer models. Scope indicates the levels, from a hierarchical standpoint, that recognize the expressions defined in the VAR data item or model item.

- Nested-VAR or model item expressions are recognized within the design containing the VAR or model item, as well as within any subnetworks (designs at lower levels) referenced by the design containing the VAR or model item.
  Global-VAR or model item expressions are recognized throughout the entire design, no matter what level in the design hierarchy the VAR or model item is placed.

Summarizing the Netlist File Include Component

For all of the Netlist File Include component parameters, a single include statement is netlisted for each file. The netlister checks to see if a file has already been output, to avoid having multiple definitions of files. The precedence is that model files are output first, so that the segment directives can be placed around the **#include** .

If you are using the Netlist File Include component, it is not putting out **#ifdef** <*file*> statements to further ensure that files are not multiply included. If you use a Netlist File Include component, you should not additionally use other file *include* components to avoid multiple inclusions which will cause a simulator redefinition error.
Example:

Parameter settings

```
includePath=". ./models"
definitionFiles="functions.def"
stimulusFiles="vccdef.stim"
modelLibraryFiles\[1\]="resistor.lib Nominal"
```

Netlist File Output (Note that *.* is */users/default/default_wrk* in this example):

```
#define Nominal
#include "/users/default/default_wrk/models/resistor.lib"
#undef Nominal
#include "/users/default/default_wrk/models/functions.def"
#include "/users/default/default_wrk/models/vccdef.stim"
```

It is worth noting that, once the Netlist File Include component is netlisted, the simulator makes no differentiation between definition, stimulus, or model files. Each file will generate the **#include** __ statements.

You may want to use the *IncludeFiles* parameter for all of your files so that you can put corner case statements into all of your model files.

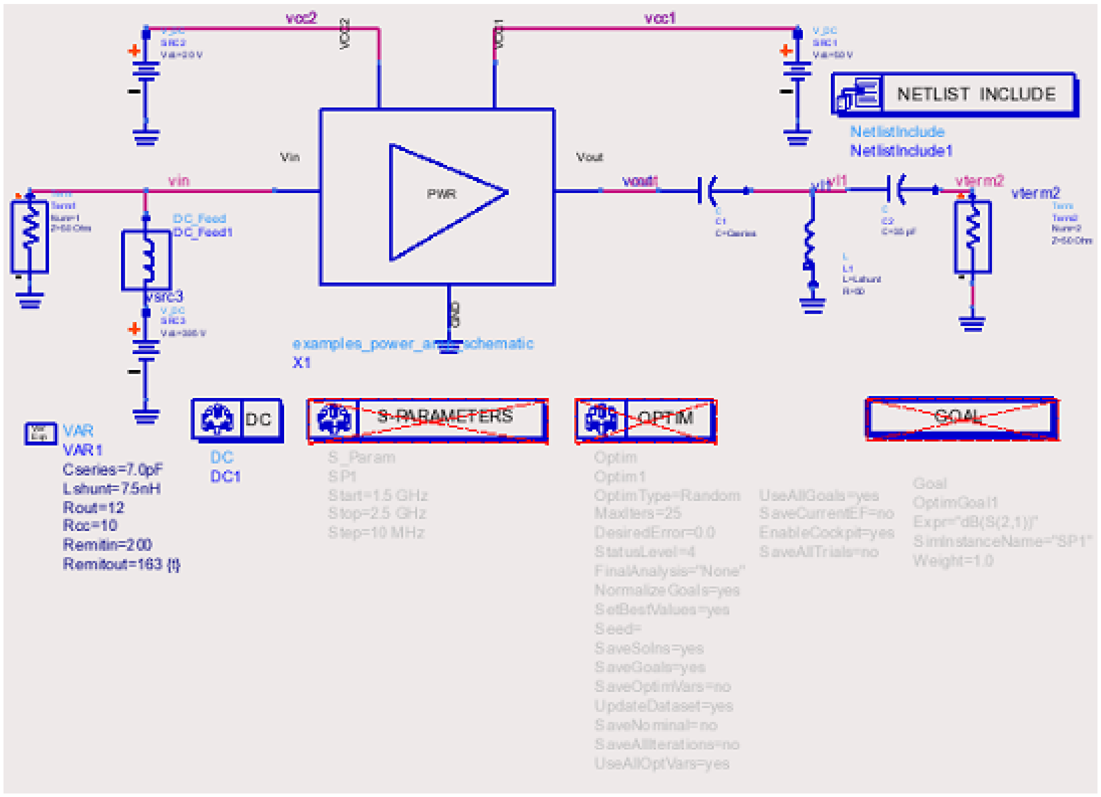
Annotating a Cellview

This section describes how to annotate your simulation results in Advanced Design System to a Cadence cellview.

Annotating DC Voltages to a Cadence Cellview

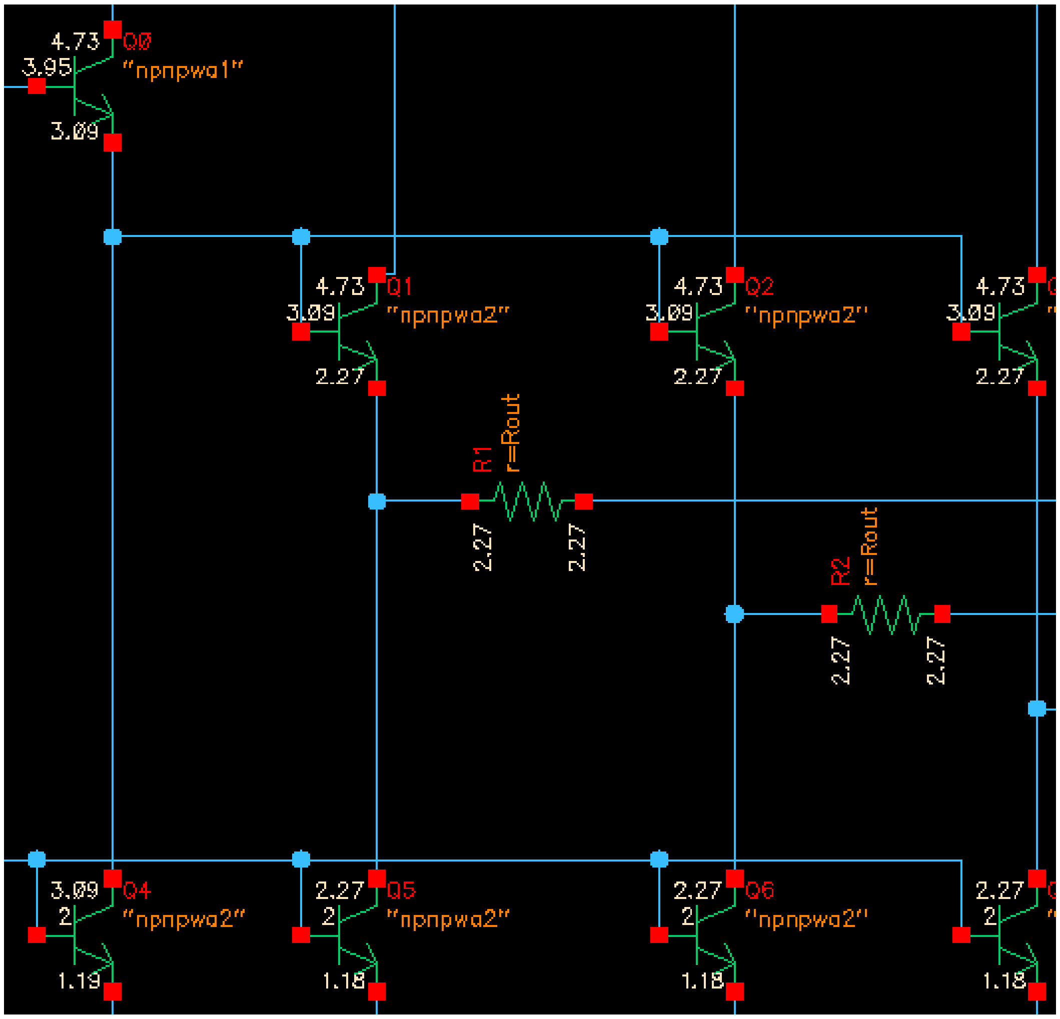
To annotate a DC voltage solution in Advanced Design System to the Cadence cellview:

1. In Advanced Design System, set up and simulate your schematic. This schematic must contain a *DC* Simulation Component as shown in Example setup for DC Simulation.



Example setup for DC Simulation

2. Select the schematic symbol in the ADS schematic that represents the Cadence circuit you want to back annotate. For example, the amplifier block in the preceding figure.
3. From the ADS Schematic window, choose **DynamicLink > Annotate > Annotate DC Voltages to Selected Cellview** .
4. The voltages are then displayed on the Cadence Virtuoso schematic window as shown in the following figure.

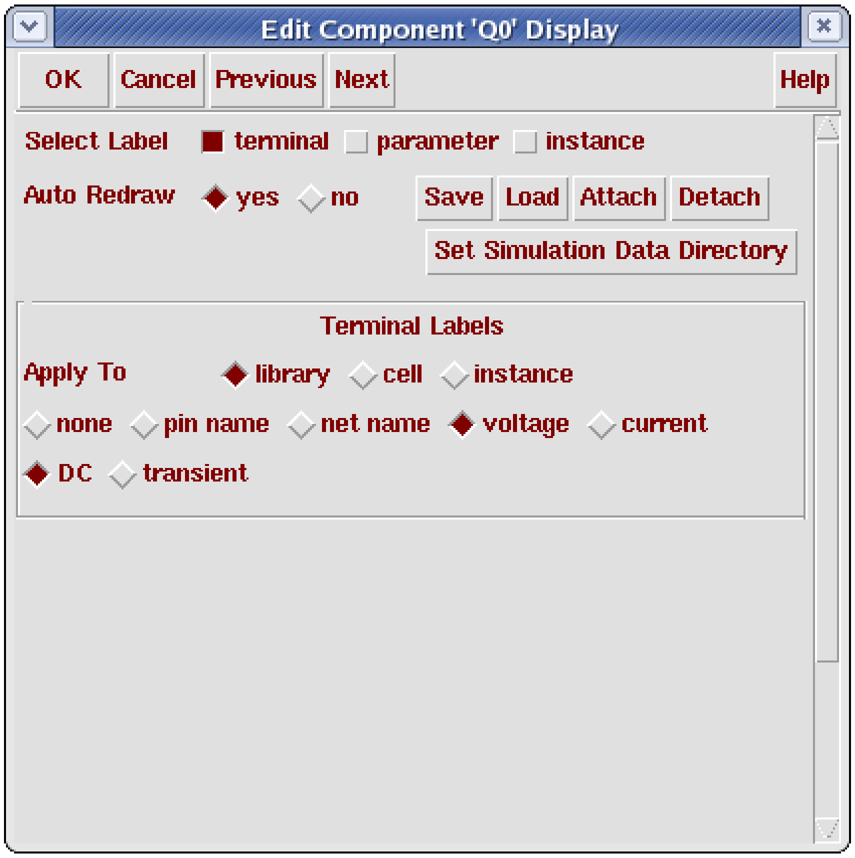


DC Voltage Annotation on the Cadence Virtuoso Schematic window

Annotating DC Currents to a Cadence Cellview

To annotate a DC current solution in Advanced Design System to the Cadence cellview:

1. First annotate the DC voltages as described in Annotating DC Voltages to a Cadence Cellview.
2. From the Cadence Virtuoso schematic window, choose **Edit** > **Component Display** . The *Edit Component Display Options* form appears.
3. Click an instance in the Cadence Virtuoso Schematic window. For this example, *Q0* was selected. Note that the title of the Edit Component Display Options form changes to include the component selected. In this case, the title *Edit Component `Q0' Display* appears.
4. Click the *terminal* checkbox from the *Select Label* options. Notice that the form now displays a *Terminal Labels* section. This section shows that DC and voltage is currently selected as seen in Edit Component `Q0' Display form showing DC Voltage.


Edit Component 'Q0' Display

Select Label [✓ terminal] parameter instance

Auto Redraw [✓ yes] no

Terminal Labels

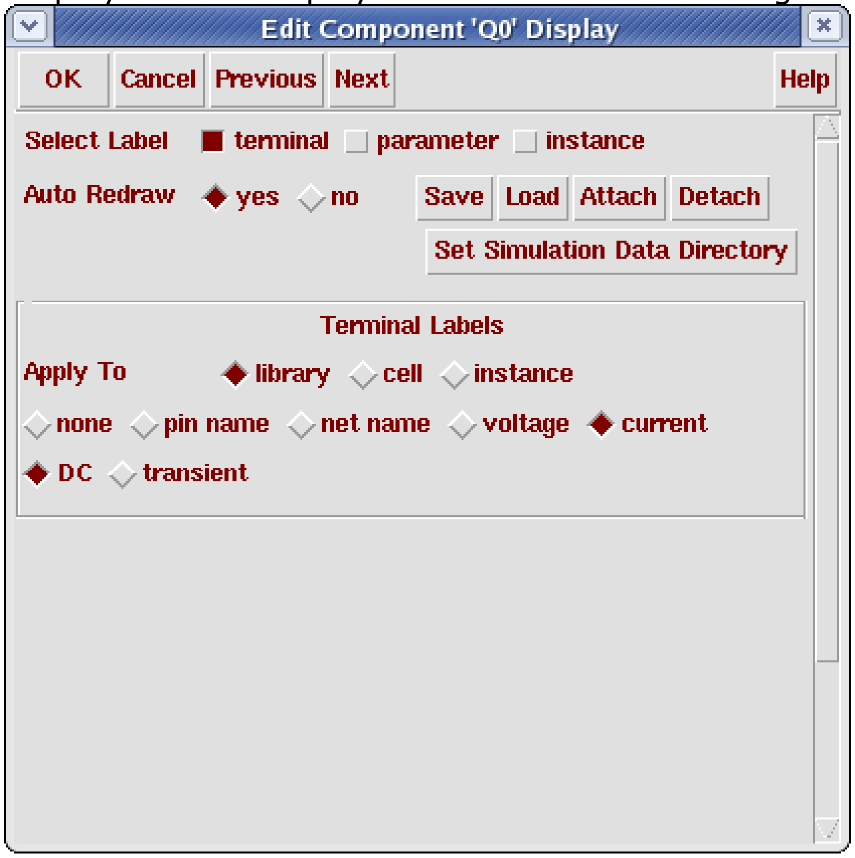
Apply To [✓ library] cell instance

none pin name net name [✓ voltage] current

[✓ DC] transient

Edit Component `Q0' Display form showing DC Voltage

5. Click the *currents* checkbox in the *Terminal Labels* section of the Edit Component Display form to display currents instead of voltage.


Edit Component 'Q0' Display

Select Label [✓ terminal] parameter instance

Auto Redraw [✓ yes] no

Terminal Labels

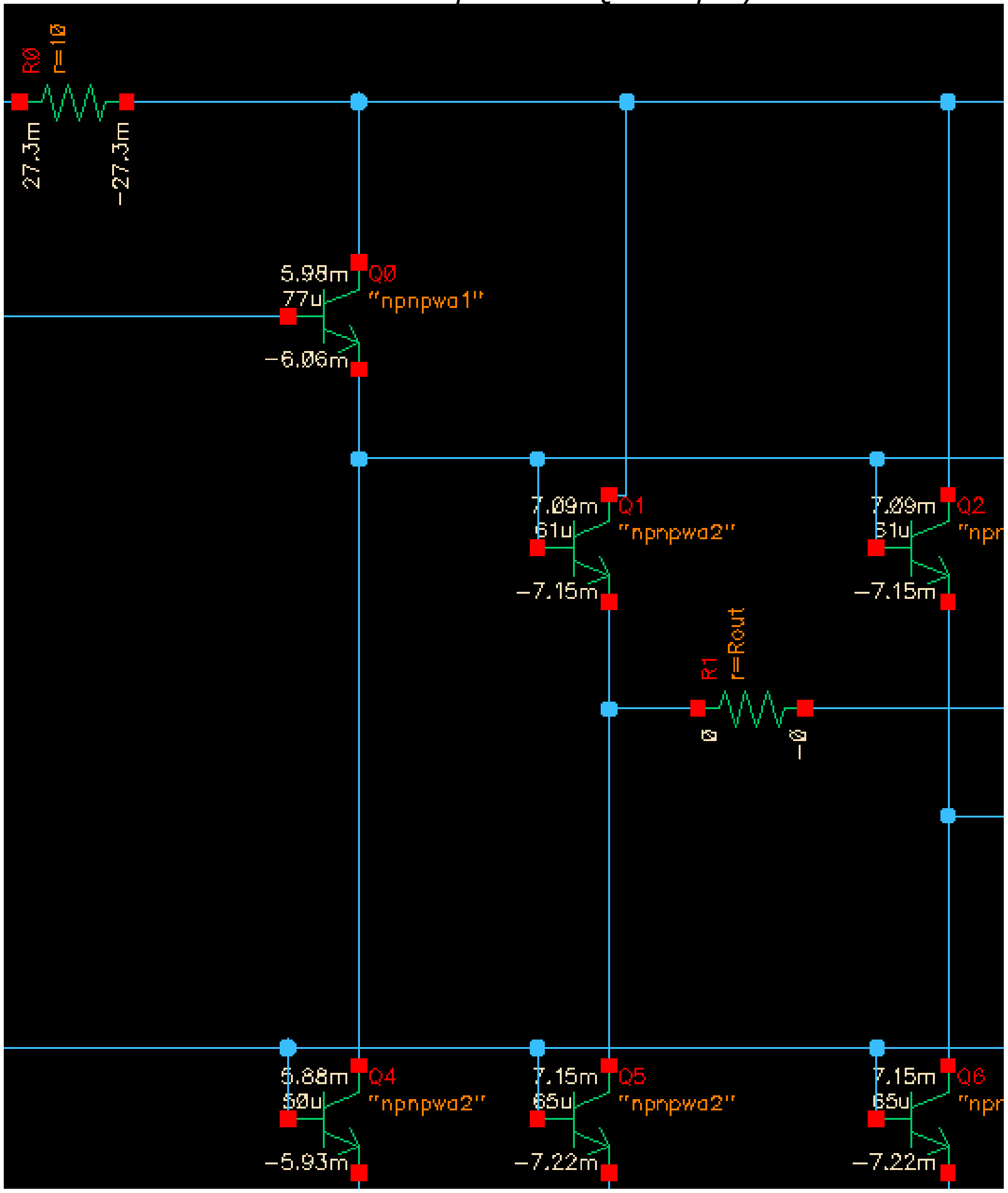
Apply To [✓ library] cell instance

none pin name net name voltage [✓ current]

[✓ DC] transient

The next figure shows the design with the DC currents annotated.

6. Click OK to clear the *Edit Component `Q0' Display* form.

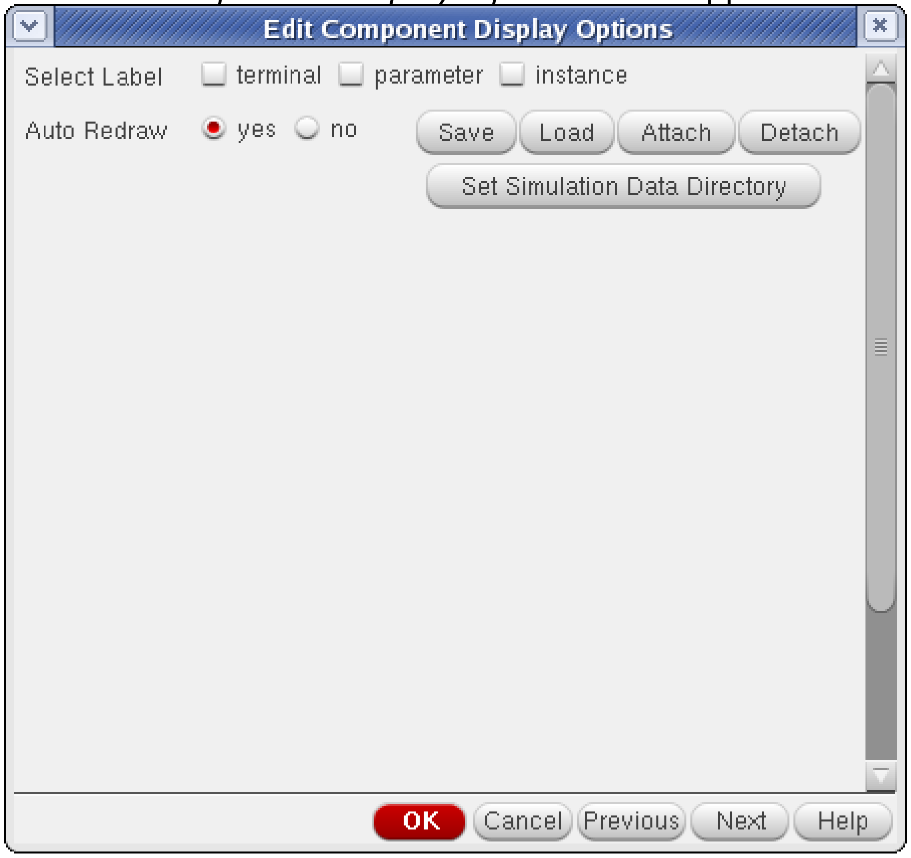


DC Current Annotation on the Cadence Virtuoso Schematic window

Note
If you do not select an instance, the library that your changes will apply to will be the library that contains the schematic (i.e. *examples* for the *power_amp* example). Since the primitives (i.e. the *res* and *npn* cells) are not in the schematic's library, you will not see any changes to the annotation if you do not first select an instance from the proper library. The Edit Component Display form enables you to control how labels are displayed for each library, cell and instance in the design.

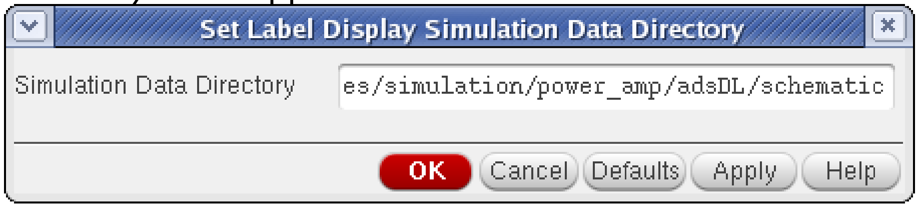
Displaying Voltages or Currents from a Previous Simulation

To display voltages or currents from a previous simulation:

1. Before displaying voltages or currents you must have annotated a DC solution to the schematic in a prior Cadence session. Follow the instructions for annotating a DC solution (see Annotating a Cellview) to a schematic if you have not already done so.
2. From the Cadence Virtuoso Schematic window, choose **Edit** > **Component Display** . The *Edit Component Display Options* form appears.

Edit Component Display Options

Select Label ☐ terminal ☐ parameter ☐ instance

Auto Redraw [✓ yes] no

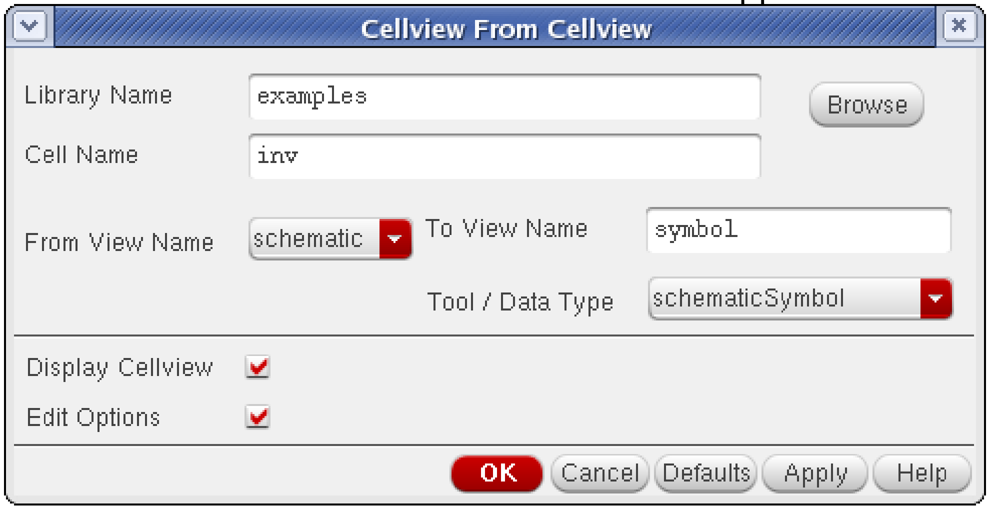
3. Click **Set Simulation Data Directory** . The *Set Label Display Simulation Data Directory* form appears.

Set Label Display Simulation Data Directory

Simulation Data Directory: es/simulation/power_amp/adsDL/schematic

4. Enter the full path to the Data Directory. This is everything up to the *psf* directory. The *psf* directory contains Cadence formatted data. The structure for the path name is,
   < *Cadence_project_dir* >/< *cell_name* >/< *tool_name* >/< *view* >
   The path for the Data Directory used in the example for DC Voltage Annotation on the Cadence Virtuoso Schematic window was,
   ~/simulation/power_amp/adsDL/schematic
   The annotation code looks in the *psf* directory.
5. Click OK in the *Set Label Display Simulation Data Directory* form.
6. Click OK in the *Edit Component Display Options* form.

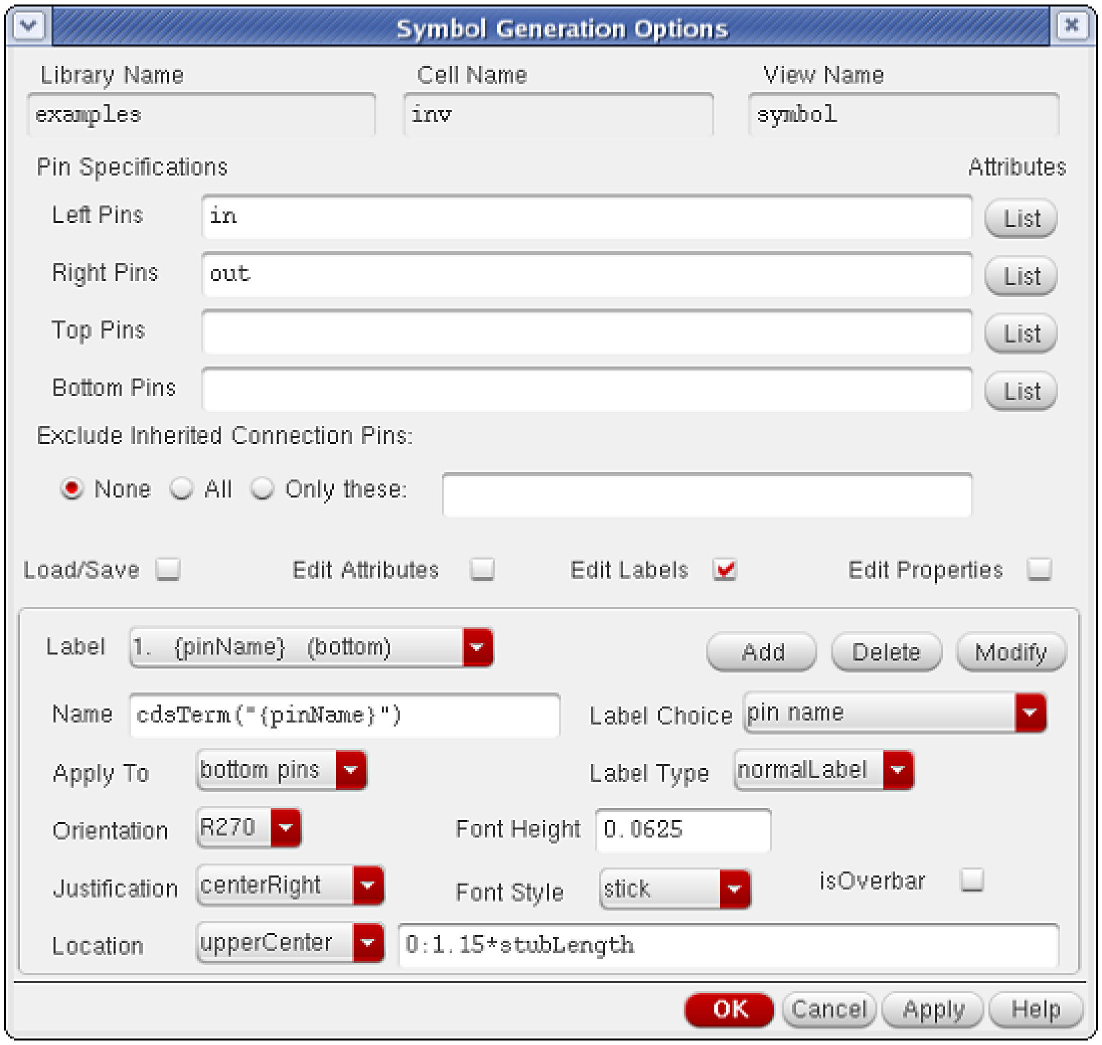
Creating Symbols for Hierarchical Subcircuits with cdsTerm

If you want to annotate a hierarchical Cadence design, you must create a Cadence symbol for the design.
To create symbols for hierarchical subcircuits using *cdsTerm* :

1. From the Cadence Virtuoso Schematic window, choose **Create > Cellview > From Cellview** . The *Cellview From Cellview* form appears.

Cellview From Cellview

Library Name: examples
Cell Name: inv
From View Name: schematic
To View Name: symbol
Tool / Data Type: schematicSymbol
Display Cellview: ✓
Edit Options: ✓

2. In the *Cellview From Cellview* form, ensure the following settings are correct:
   - From View Name - *schematic*
   - To View Name - *symbol*
   - Tool / Data Type - *Composer-Symbol*
     Click **OK**. The Symbol Generation Options form appears, assuming a symbol does not already exist.
3. In the *Symbol Generation Options* form, click the *Edit Labels* checkbox. Your form will display the *Label* options.
4. In the *Symbol Generation Options* form, select *analog pin annotate* from the *Label Choice* pull-down menu. The Name field should now display cdsTerm("(pinname)").
5. Select *all pins* from the *Apply To* drop-down menu and click **Add** . This generates a new label rule that creates a *cdsTerm* for each pin. You may alter the location if you choose. The form with all appropriate option settings is shown in the following figure.



Library Name: examples
Cell Name: inv
View Name: symbol

Pin Specifications | Attributes
Left Pins: in
Right Pins: out
Top Pins:
Bottom Pins:

Exclude Inherited Connection Pins: [✓ None] All Only these:

Load/Save ☐ Edit Attributes ☐ Edit Labels ☑ Edit Properties ☐

Label: 1. {pinName} (bottom)
Name: cdsTerm("{pinName}")
Label Choice: pin name
Apply To: bottom pins
Label Type: normalLabel
Orientation: R270
Font Height: 0.0625
Justification: centerRight
Font Style: stick
isOverbar ☐
Location: upperCenter 0:1.15*stubLength

Symbol Generation Options to create a symbol with cdsTerms on each pin

For more information on the *Symbol Generation Option* form, refer to your Cadence documentation.

Using Switch Views, Stop Views and the Hierarchy Editor

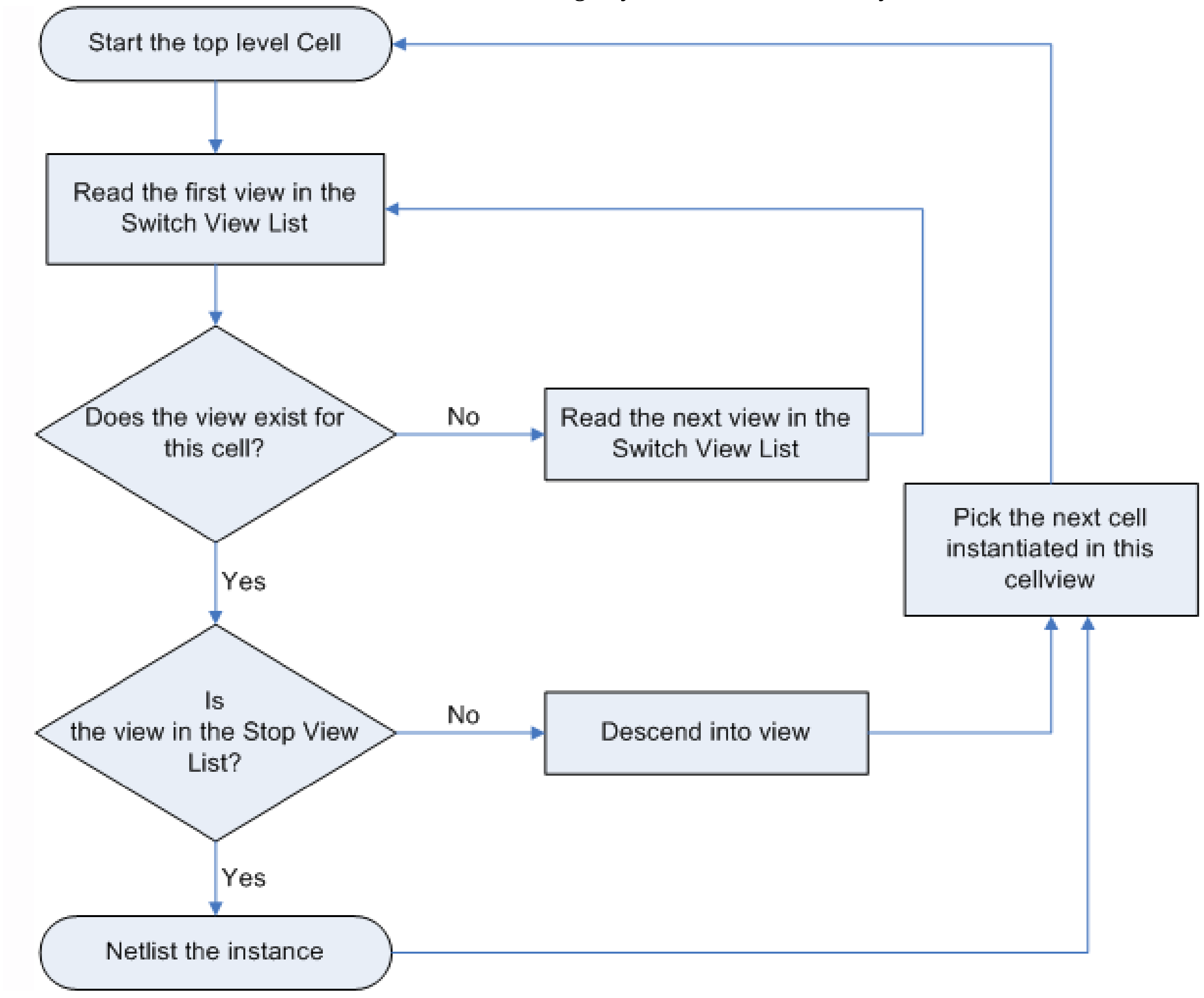
This section provides information on using *Switch Views*, *Stop Views* and the *Hierarchy Editor*. A Switch View is a list that describes the views to use and their priority. A Stop View is a list that designates when to stop moving down in hierarchy. In other words, if a view is in the Switch View List, and also in the Stop View List, it won't traverse any lower in hierarchy.

RFIC Dynamic Link supports the concept of using *Switch Views* and *Stop Views*. Dynamic Link also supports Cadence's *Hierarchy Editor* tool, which enables more detailed specification of Switch Views and Stop Views than the standard Artist forms. Switch views and Stop Views are utilized by the netlister to expand hierarchy. The Hierarchy Editor enables you to override the Switch View List and Stop View List for each instance in a schematic's hierarchy.

Note
The information provided in this section refers to using the Cadence Hierarchy Editor tool with the RFIC Dynamic Link and Advanced Design System. For more detailed information on using the Cadence Hierarchy Editor tool exclusively, refer to your *Virtuoso Analog Circuit Design Environment User Guide*. For more detailed information on expanding hierarchy in the Cadence environment, refer to the section on *How the Netlister Expands Hierarchy* in your Cadence documentation.

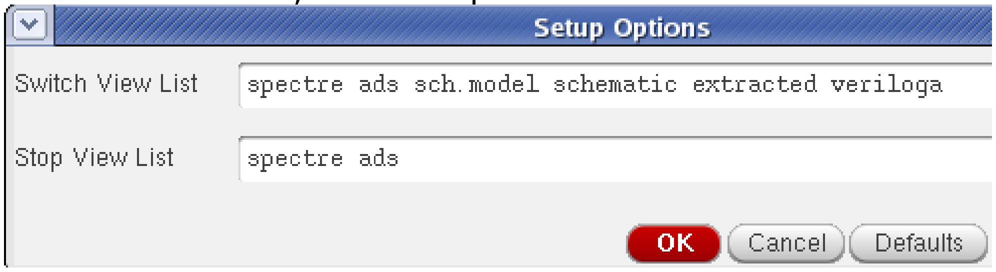
Expanding Hierarchy with the Dynamic Link Netlister

The flowchart shown in the following figure describes the general process used by the RFIC Dynamic Link netlister to move through and expand the hierarchy in Dynamic Link.


Start the top level Cell
Read the first view in the Switch View List
Does the view exist for this cell?
No
Read the next view in the Switch View List
Pick the next cell instantiated in this cellview
Yes
Is the view in the Stop View List?
No
Descend into view
Yes
Netlist the instance

Netlist Hierarchy Expansion

In the Cadence schematic window, choose the **DynamicLink > Setup Options** menu item to access the *Setup Options* form. The Setup Options form enables you to designate a Switch View List, and a Stop View List.

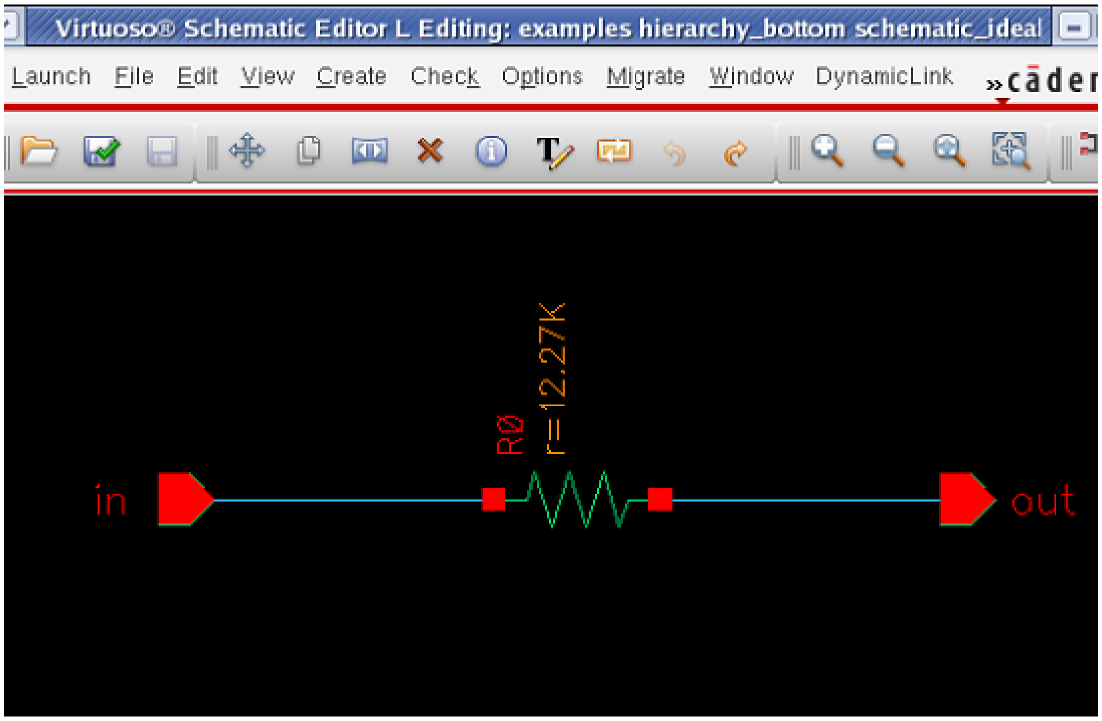

Setup Options

Switch View List	spectre ads sch.model schematic extracted veriloga
Stop View List	spectre ads

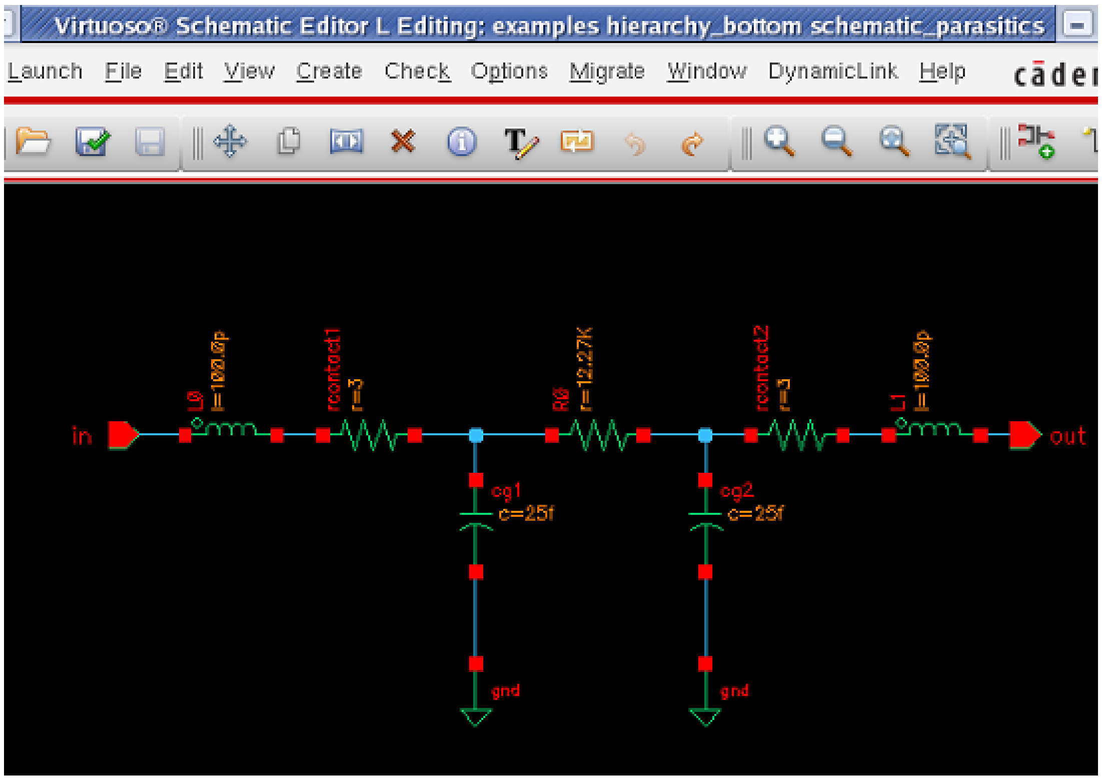
Setting a Switch View List and a Stop View List.

Each instance in a hierarchy has a master cell, that contains a number of views for that cell. Typically views are schematic, symbol, layout, extracted, etc. The Switch View List is used to enable you to designate the priority of each view for hierarchical netlisting. The Stop View is used to designate whether a view in the Switch View List should be traversed for hierarchy. A Stop View is implied to have no hierarchy. For an example of this, refer to

the design *hierarchy_bottom* in the library *examples* that is provided with the Dynamic Link installation.
In this example, the cell *hierarchy_bottom* has two alternate schematic views, *schematic_ideal* and *schematic_parasitics* (see the following two figures).

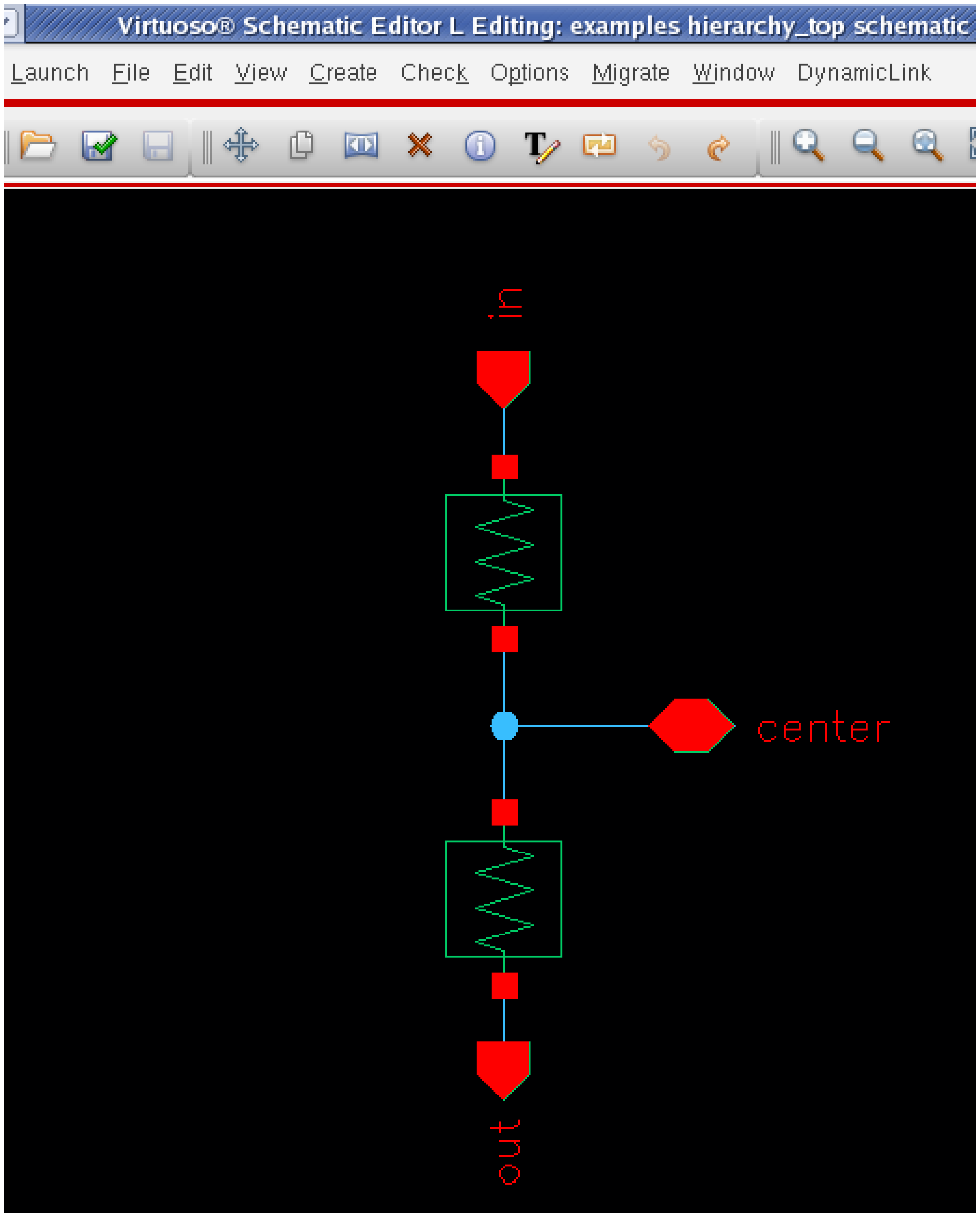

Virtuoso® Schematic Editor L Editing: examples hierarchy_bottom schematic_ideal
Launch File Edit View Create Check Options Migrate Window DynamicLink
in
R0
r=12.27K
out

Alternate schematic view *schematic_ideal* for the example cell *hierarchy_bottom*.


Virtuoso® Schematic Editor L Editing: examples hierarchy_bottom schematic_parasitics
Launch File Edit View Create Check Options Migrate Window DynamicLink Help
in
L0
l=100.0p
rcontact1
r=3
R0
r=12.27K
rcontact2
r=3
L1
l=100.0p
out
cg1
c=25f
cg2
c=25f
gnd
gnd

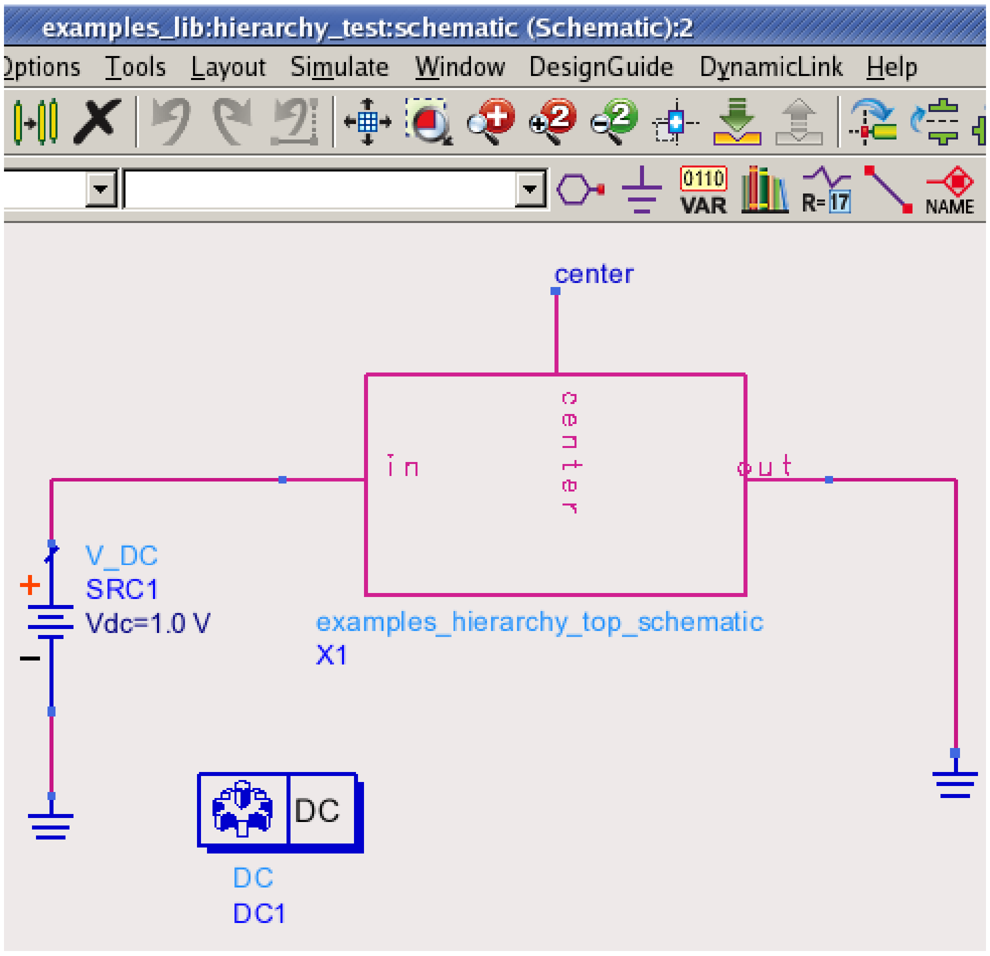
Alternate schematic view *schematic_parasitics* for the example cell *hierarchy_bottom*.

Two instances of the cell *hierarchy_bottom* have been placed in the schematic *hierarchy_top* (see the following figure) in the *examples*.



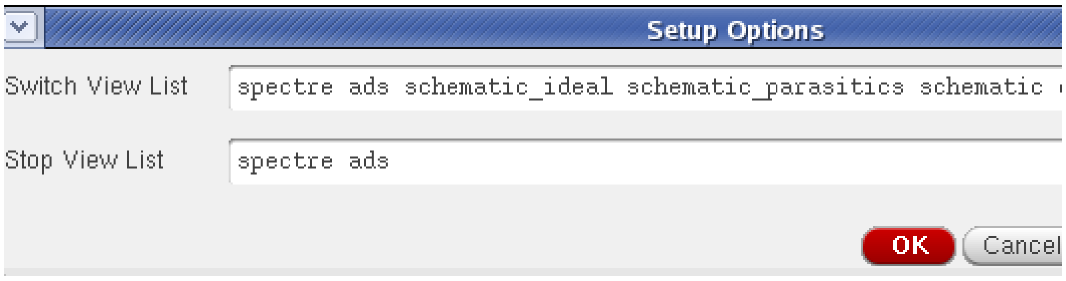
The schematic view for the example cell *hierarchy_top*

This cell is in turn placed in the ADS example workspace schematic cellview *hierarchy_test* , with the schematic view chosen as the Cadence view (see the following figure). Because there are two alternate schematics, it becomes necessary to tell the netlister which one to use when you netlist the design *hierarchy_top*. The Switch View List is used to tell the netlister which one you want to use.


examples_lib:hierarchy_test:schematic (Schematic):2
Options Tools Layout Simulate Window DesignGuide DynamicLink Help
VAR
NAME
center
in
center
out
V_DC
SRC1
Vdc=1.0 V
examples_hierarchy_top_schematic
X1
DC
DC
DC1

The cadence cell *hierarchy_top* placed in ADS

The following figure shows how the setup options have been changed to designate that *schematic_ideal* takes priority.


Setup Options

Switch View List `spectre ads schematic_ideal schematic_parasitics schematic`

Stop View List `spectre ads`

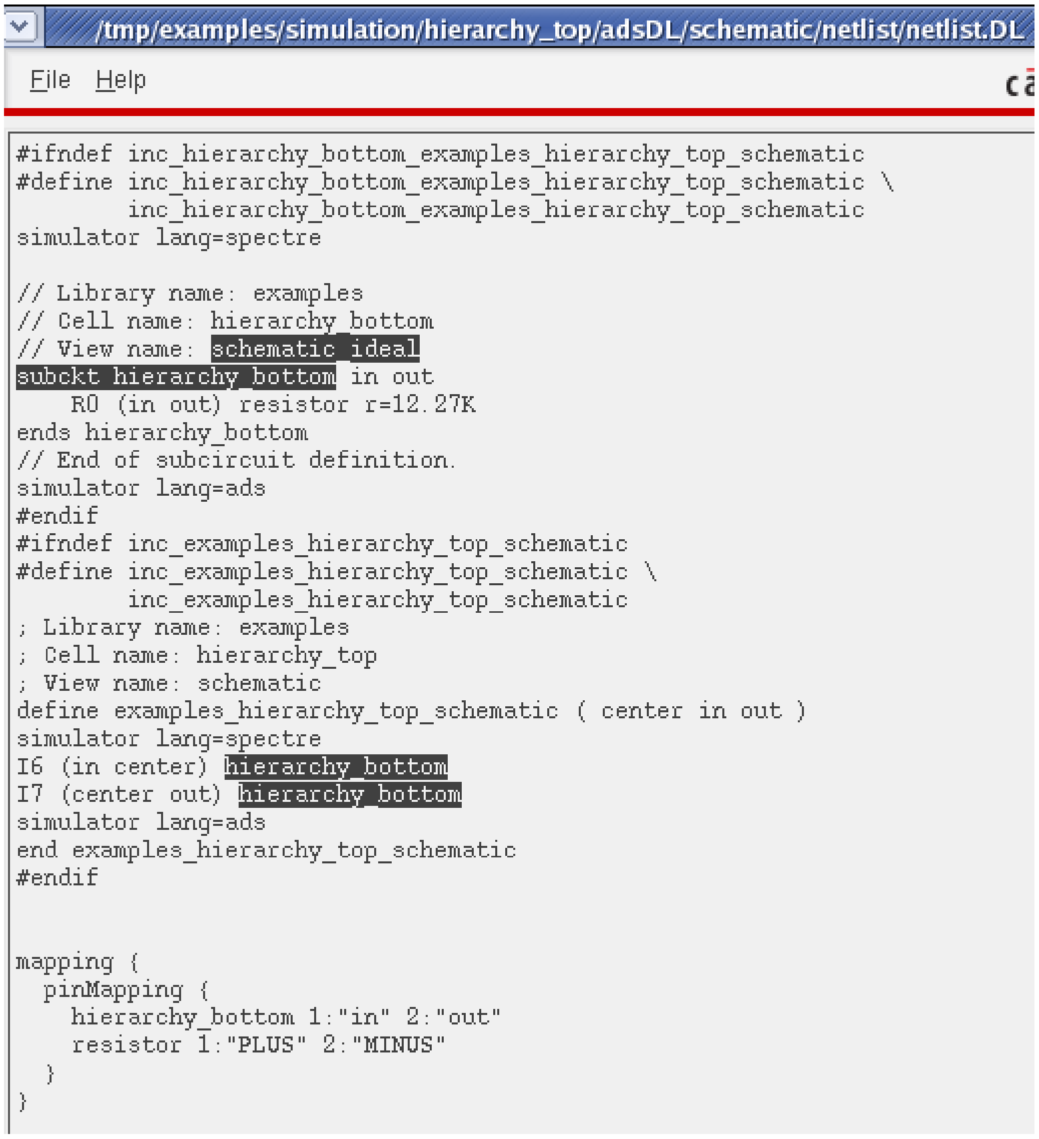
Switch View List with schematic_ideal taking priority

Referring to the Hierarchy Expansion flowchart in the figure Netlist Hierarchy Expansion, when an instance of *hierarchy_bottom* is encountered, the following occurs:

1. There is no *ads* view, so the next switch view is looked at.
2. There is no *schematic* view, so the code continues to the next switch view.
3. There is a *schematic_ideal* view. Since *schematic_ideal* is not in the Stop View List, it is opened, so that a subcircuit can be netlisted for it.
4. With *hierarchy_bottom* as the top cell, now the first instance, and only instance encountered, is the analogLib *res* component. The Switch View List is consulted.
    - Is there an *ads* view? Yes, there is.
    - Is the *ads* view a stop view? Yes, it is.

    This means that the *res* component will not be opened as a schematic with hierarchy. Instead, it is meant to be a simulator primitive. Either a built in simulator component exists, which will be used, or a subcircuit definition has been included into the simulator that defines what the component is. A single component line is output for it, and the netlister continues on.

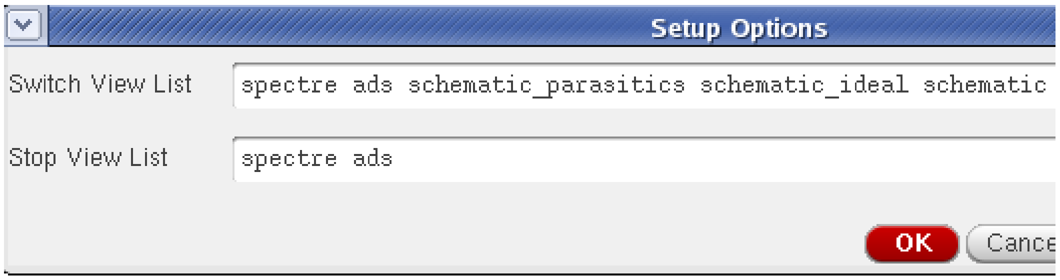
The following figure shows the resulting netlist from netlisting *hierarchy_top* with *schematic_ideal* having higher priority than *schematic_parasitics*. Note that the instances and subcircuit definition have been highlighted.


/tmp/examples/simulation/hierarchy_top/adsDL/schematic/netlist/netlist.DL
File
Help

```
#ifndef inc_hierarchy_bottom_examples_hierarchy_top_schematic
#define inc_hierarchy_bottom_examples_hierarchy_top_schematic \
        inc_hierarchy_bottom_examples_hierarchy_top_schematic
simulator lang=spectre

// Library name: examples
// Cell name: hierarchy_bottom
// View name: schematic_ideal
subckt hierarchy_bottom in out
    R0 (in out) resistor r=12.27K
ends hierarchy_bottom
// End of subcircuit definition.
simulator lang=ads
#endif
#ifndef inc_examples_hierarchy_top_schematic
#define inc_examples_hierarchy_top_schematic \
        inc_examples_hierarchy_top_schematic
; Library name: examples
; Cell name: hierarchy_top
; View name: schematic
define examples_hierarchy_top_schematic ( center in out )
simulator lang=spectre
I6 (in center) hierarchy_bottom
I7 (center out) hierarchy_bottom
simulator lang=ads
end examples_hierarchy_top_schematic
#endif


mapping {
  pinMapping {
    hierarchy_bottom 1:"in" 2:"out"
    resistor 1:"PLUS" 2:"MINUS"
  }
}
```

Netlist of the example cell *hierarchy_top* with *schematic_ideal* taking priority

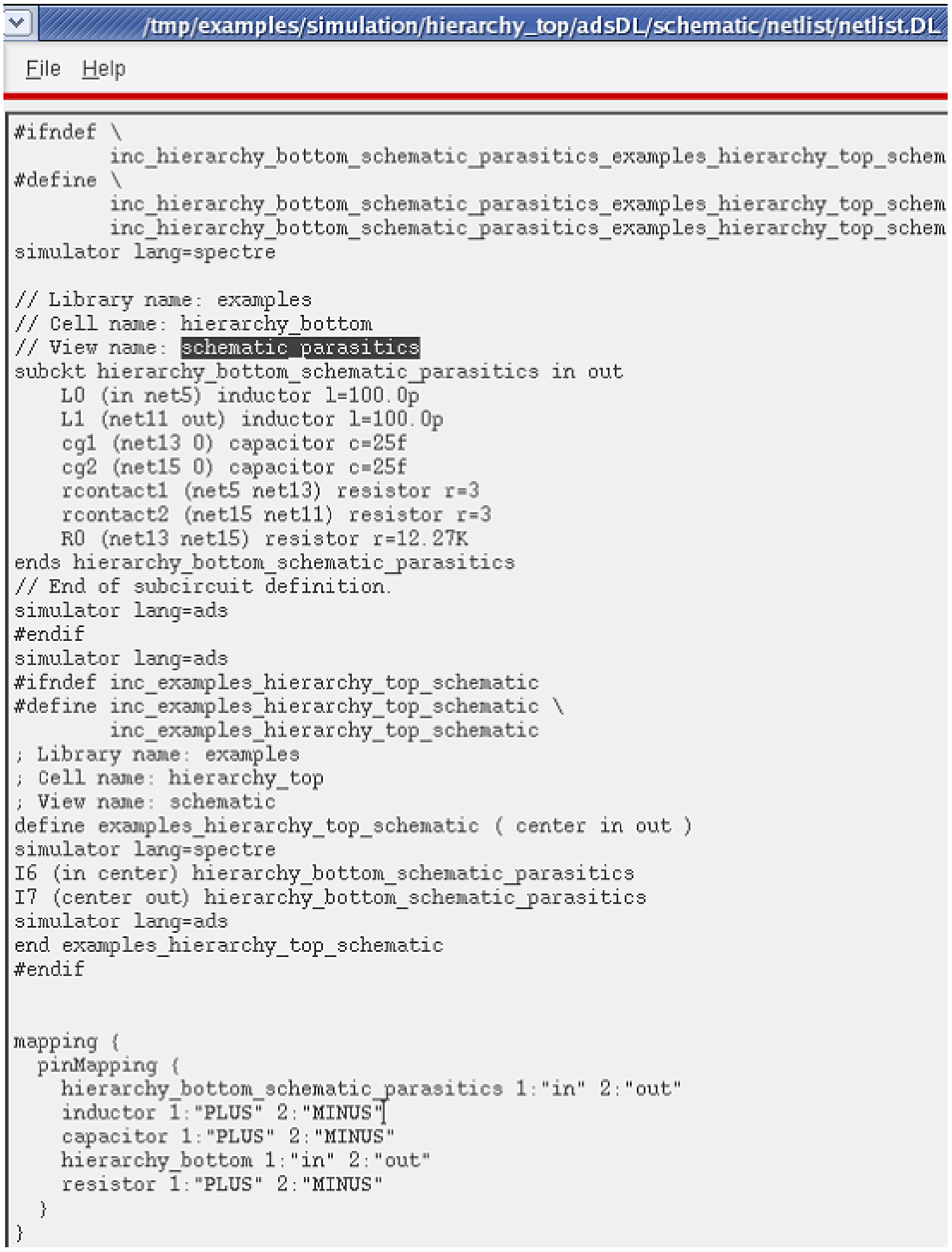
If you prefer to use the *schematic_parasitics* schematic, the *hierarchy_top* schematic does not need to be changed. Instead, in the setup options form, you change the Switch View List so that *schematic_parasitics* comes before *schematic_ideal* as shown in the following figure.



Switch View List	spectre ads schematic_parasitics schematic_ideal schematic
Stop View List	spectre ads

Switch View List with *schematic_parasitics* taking priority over *schematic_ideal*

Referring to the flowchart in the figure Netlist Hierarchy Expansion again, when an instance of *hierarchy_bottom* is encountered, the Switch View List is checked. There is no *ads* view or *schematic* view. There is a *schematic_parasitics* view, which is not a part of the Stop View List. This results in the netlister opening the *schematic_parasitics* view, which is netlisted as a subcircuit. The netlister then goes on to the next instance, without ever checking for a *schematic_ideal* view (in point of fact, it is not necessary to put *schematic_ideal* in the Switch View List, since it will not be used). The following figure shows the resulting netlist with the setup options from the previous figure.


/tmp/examples/simulation/hierarchy_top/adsDL/schematic/netlist/netlist.DL
File Help

```
#ifndef \
        inc_hierarchy_bottom_schematic_parasitics_examples_hierarchy_top_schem
#define \
        inc_hierarchy_bottom_schematic_parasitics_examples_hierarchy_top_schem
        inc_hierarchy_bottom_schematic_parasitics_examples_hierarchy_top_schem
simulator lang=spectre

// Library name: examples
// Cell name: hierarchy_bottom
// View name: schematic_parasitics
subckt hierarchy_bottom_schematic_parasitics in out
    L0 (in net5) inductor l=100.0p
    L1 (net11 out) inductor l=100.0p
    cg1 (net13 0) capacitor c=25f
    cg2 (net15 0) capacitor c=25f
    rcontact1 (net5 net13) resistor r=3
    rcontact2 (net15 net11) resistor r=3
    R0 (net13 net15) resistor r=12.27K
ends hierarchy_bottom_schematic_parasitics
// End of subcircuit definition.
simulator lang=ads
#endif
simulator lang=ads
#ifndef inc_examples_hierarchy_top_schematic
#define inc_examples_hierarchy_top_schematic \
        inc_examples_hierarchy_top_schematic
; Library name: examples
; Cell name: hierarchy_top
; View name: schematic
define examples_hierarchy_top_schematic ( center in out )
simulator lang=spectre
I6 (in center) hierarchy_bottom_schematic_parasitics
I7 (center out) hierarchy_bottom_schematic_parasitics
simulator lang=ads
end examples_hierarchy_top_schematic
#endif


mapping {
  pinMapping {
    hierarchy_bottom_schematic_parasitics 1:"in" 2:"out"
    inductor 1:"PLUS" 2:"MINUS"
    capacitor 1:"PLUS" 2:"MINUS"
    hierarchy_bottom 1:"in" 2:"out"
    resistor 1:"PLUS" 2:"MINUS"
  }
}
```

Netlist of example *hierarchy_top* with *schematic_parasitics* taking priority

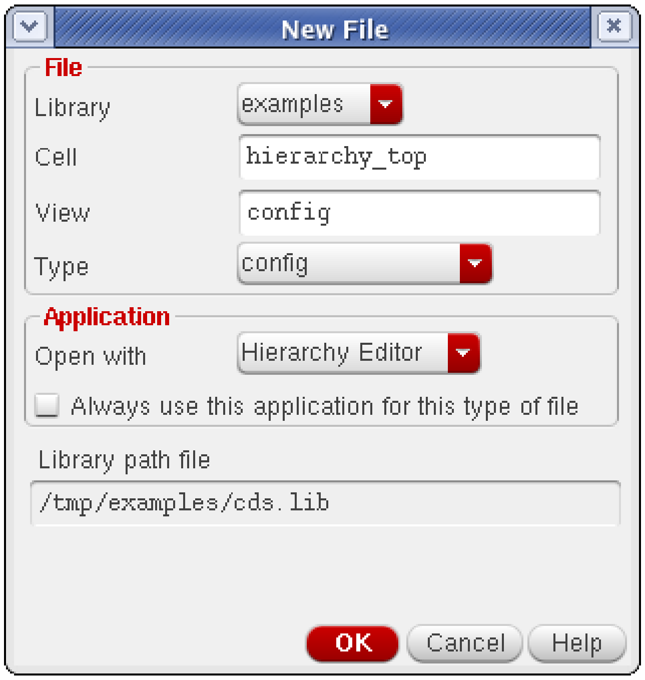
In all of the above examples, the Stop View List was set to *ads*. This is the recommended Stop View List to use for the RFIC Dynamic Link netlister. If you consult the netlist flowchart in the figure Netlist Hierarchy Expansion, you will notice that it is not necessary for the stop view to be *ads*, any view can be used to designate that a part is a primitive. In the future, the RFIC Dynamic Link netlister will be expanded, so that alternate simulation definitions can be used based on which stop view is encountered (e.g. ads vs. ads_ptolemy). At present, the netlister always uses the simulation definition defined for ads in the cell's CDF, no matter which stop view is encountered.

Using the Hierarchy Editor with RFIC Dynamic Link

The Hierarchy Editor is a stand alone Cadence tool that enables Switch Views and Stop Views to be designated for each instance in a schematic hierarchy. For information regarding the hierarchy editor in general, consult your *Cadence Hierarchy Editor User Guide*.

When the Hierarchy Editor is used, a cell view specific to that tool is generated. The view itself is actually a text file, and cannot be opened directly in anything other than the Hierarchy Editor. The default view name for the Hierarchy Editor is *config*. A *config* view has a top cell view that it points to. All other information regarding Switch Views and Stop Views is then created based on the top cell view that is being pointed to. It is not necessary for the Hierarchy Editor view to be a part of the cell that it is pointing to, although in most cases this will be the simplest way to organize things.

The following figure displays the *Create New File* form used to create a Hierarchy Editor view. In this case, a Hierarchy Editor view is being made for the example cell *hierarchy_top* (see the figure The schematic view for the example cell hierarchy_top). The goal is to set up this config view so that, during netlisting, it is possible to specify that one instance of *hierarchy_bottom* should use the *schematic_ideal* view, while the other instance uses the *schematic_parasitics* view.


New File

File

Library	examples
Cell	hierarchy_top
View	config
Type	config

Application

Open with: Hierarchy Editor

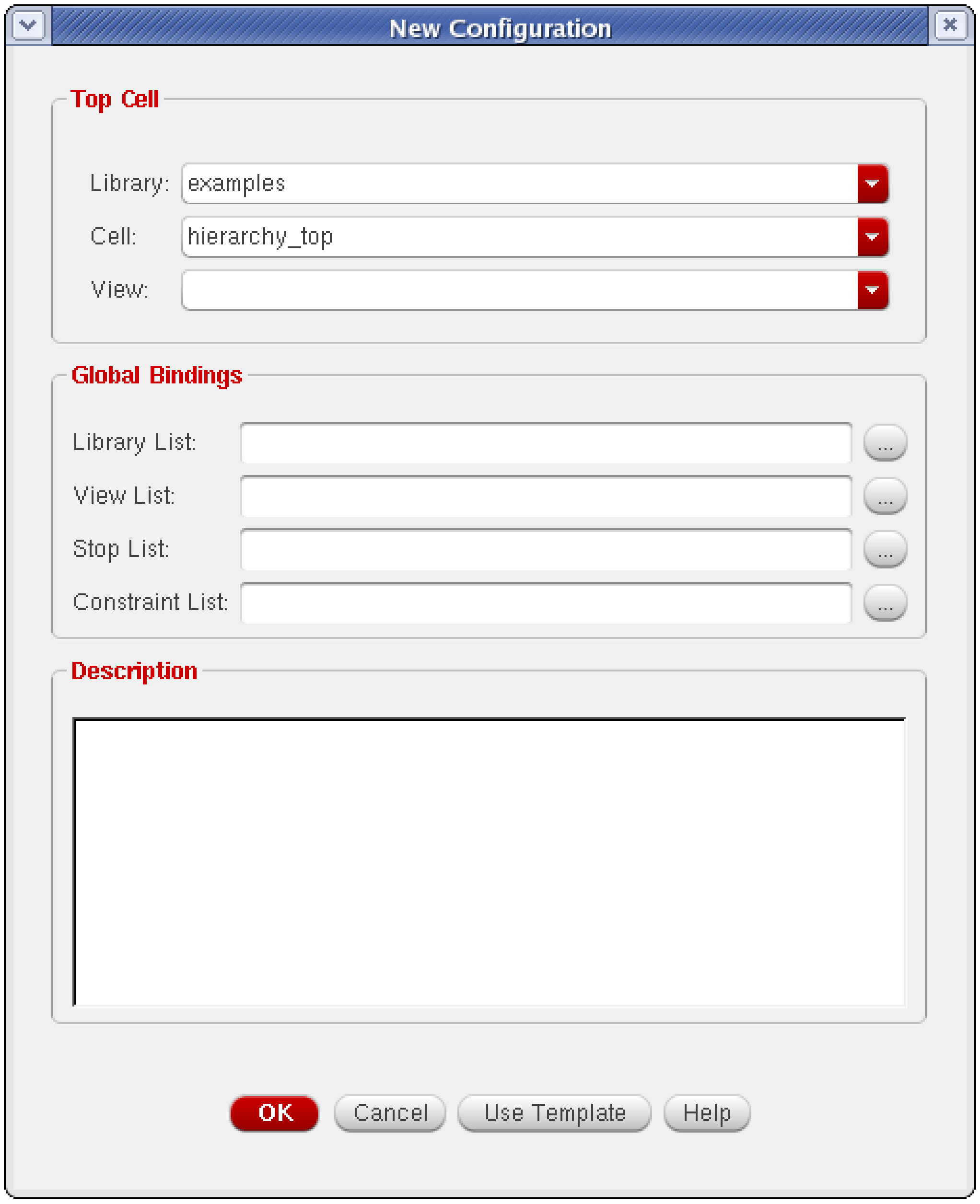
☐ Always use this application for this type of file

Library path file: /tmp/examples/cds.lib

Creating a Hierarchy-Editor view

After the new view is created, the Hierarchy Editor tool is started. For a new view, the

New Configuration form appears as shown in the following figure.


New Configuration

Top Cell

Library: examples

Cell: hierarchy_top

View:

Global Bindings

Library List:

View List:

Stop List:

Constraint List:

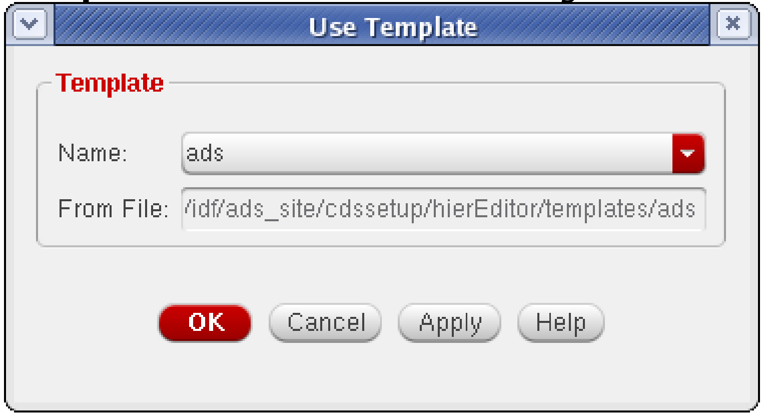
Description

New Hierarchy Editor Configuration

Notice that all fields are left blank initially, except the *Library* and *Cell* fields which were specified in the *Create New File* form shown in the figure Creating a Hierarchy-Editor view. At this point, it is necessary to designate the top cell view, as well as the Switch View List and Stop View List. The *Library List* can normally be left blank. For more information on the Library List field, refer to your *Cadence Hierarchy Editor User Guide*.

It is necessary to fill in the Top Cell view. The top cell view should be a standard CDB view. From the standpoint of RFIC Dynamic Link, this means either a view that is edited with Virtuoso-schematic, or Virtuoso-layout. If you use a layout view, it should be an extracted view of some sort (i.e. the layout will contain connectivity and instances that can be traced back to schematic equivalents). Typically, you will put in either schematic or extracted as the view name.

The Library List, View List, and Stop List can be filled in by using a template. Templates contain default settings for particular simulators. To select a template, click the **Use Template** button in the *New Configuration* form. The Use Template form appears.

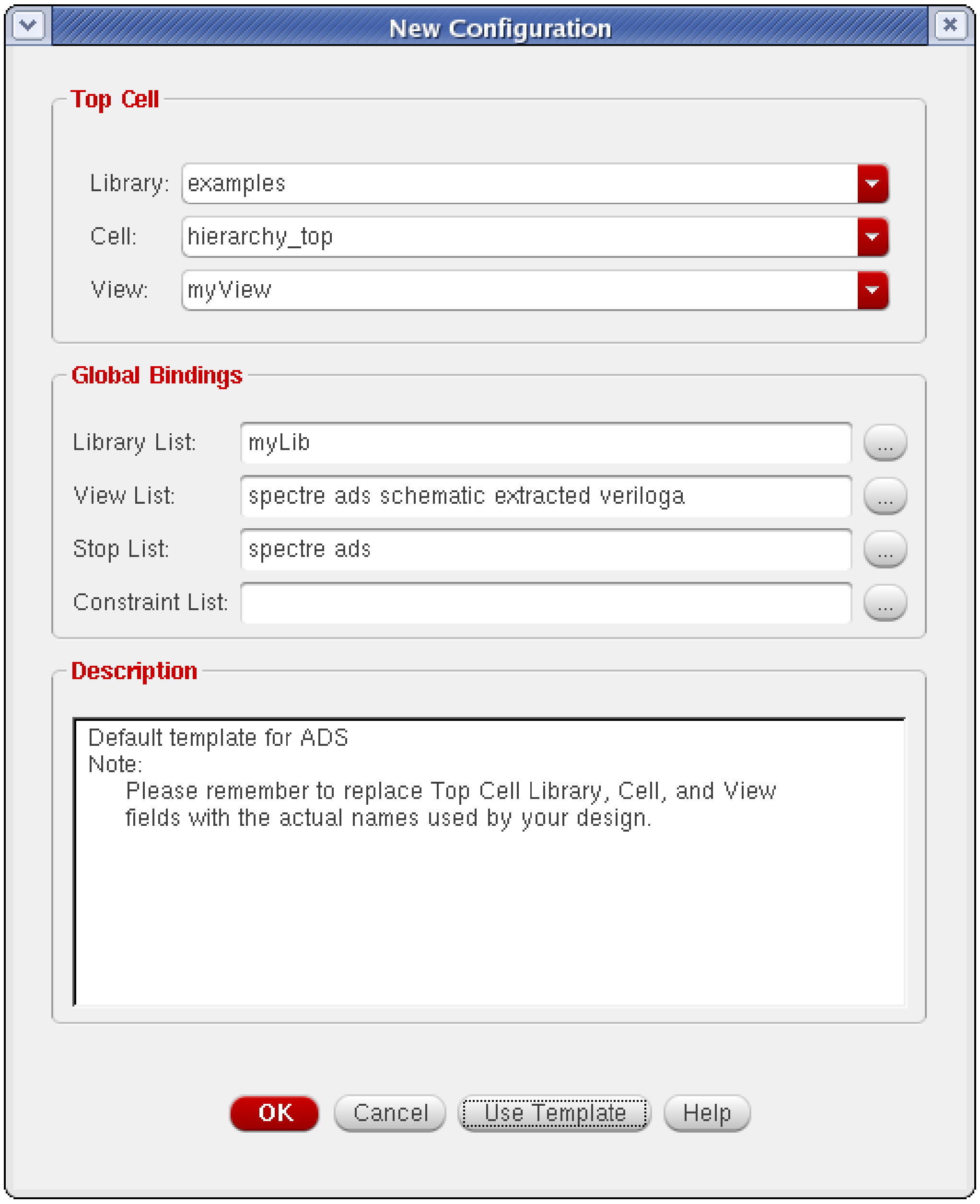

Use Template

Template

Name: ads

From File: /idf/ads_site/cdssetup/hierEditor/templates/ads

In the following figure, the ads template was selected, which has filled the View List in as *spectre ads schematic extracted veriloga*, and the Stop List as *spectre ads*. These names represent the default view names for the tools that the Dynamic Link netlister supports. During netlisting, the View List and Stop List will override the Switch View List and Stop View List that is specified using the Dynamic Link setup options dialog. Also, the Hierarchy Editor view will not consult the Dynamic Link options dialog to fill in the Switch View List or Stop View List, it is a completely separate tool, and must be set up independently.



Top Cell

Library: examples

Cell: hierarchy_top

View: myView

Global Bindings

Library List: myLib

View List: spectre ads schematic extracted veriloga

Stop List: spectre ads

Constraint List:

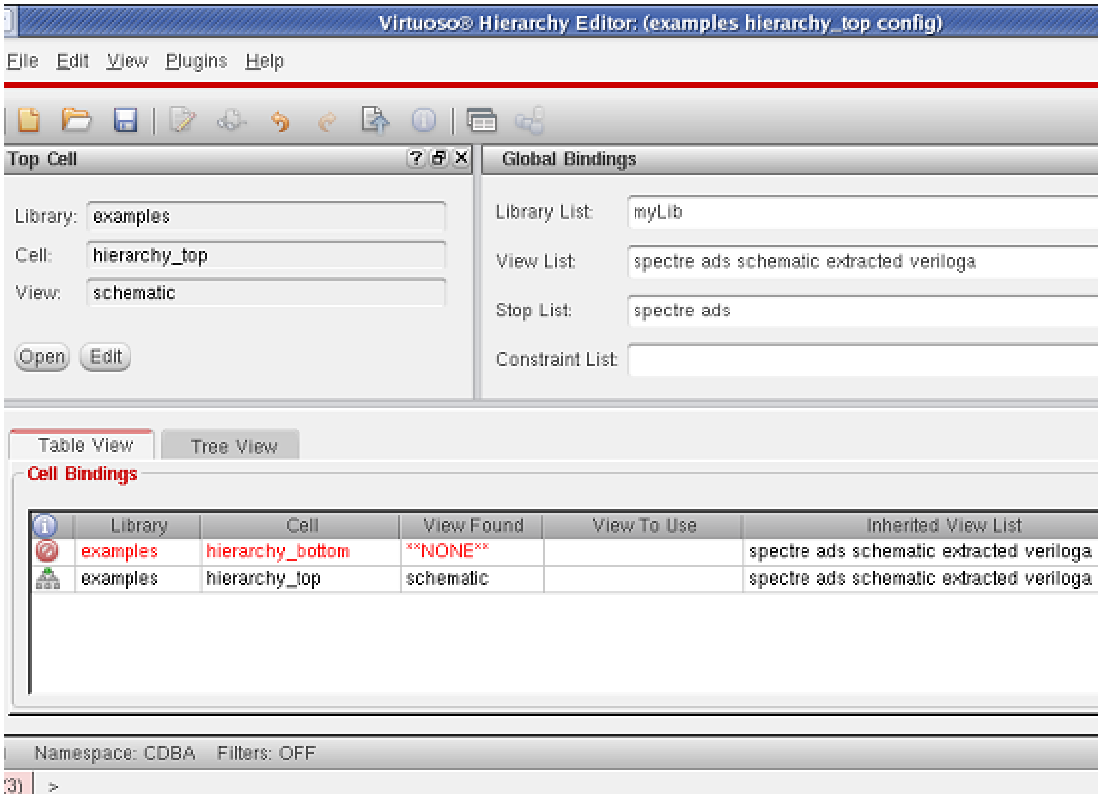
Description

Default template for ADS
Note:
    Please remember to replace Top Cell Library, Cell, and View
    fields with the actual names used by your design.

New Hierarchy Editor Configuration, using ADS template

When the new configuration is accepted, the main Hierarchy Editor window will be filled out. The main window shows the current Top Cell setting, as well as the global Library List, View List, and Stop List. In addition, the Hierarchy Editor will expand the hierarchy of the top cell, and show which views will be used for each instance within the top cell's hierarchy. When nothing has been overridden, the expansion will follow the flow chart shown in the figure Netlist Hierarchy Expansion.

The following figure shows how the example *hierarchy_top* schematic view was expanded using the global bindings. The Cell Bindings shows the two cells that were found, *hierarchy_top* and *hierarchy_bottom*. You may need to select **View** > **Update** to see the *hierarchy_bottom* Cell in the Cell Bindings list. The view found for *hierarchy_top* was schematic, this is a special case. Because it is the top cell, the library, cell, and view found will always be the same as what is specified for the top cell, regardless of the View List and Stop List. The cell *hierarchy_bottom* says that the view found was **NONE**. Because *hierarchy_bottom* has the views *schematic_ideal*, *schematic_parasitics*, and *symbol*, this is accurate. None of *hierarchy_bottom*'s views are in the global View List. If netlisting were attempted at this point, an error would result. The cell bindings give a visual indication of this.


Virtuoso® Hierarchy Editor: (examples hierarchy_top config)
File Edit View Plugins Help
Top Cell
Global Bindings
Open Edit
Table View
Tree View
Cell Bindings
Namespace: CDBA Filters: OFF

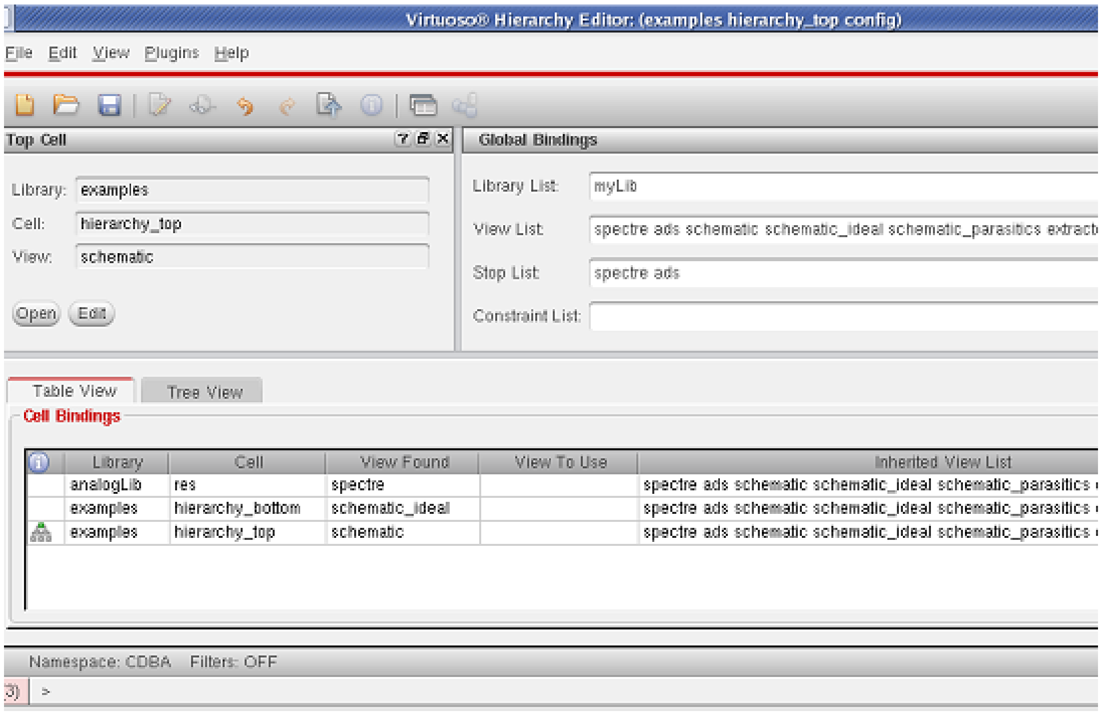
Library: examples
Cell: hierarchy_top
View: schematic

Library List: myLib
View List: spectre ads schematic extracted veriloga
Stop List: spectre ads
Constraint List:

	Library	Cell	View Found	View To Use	Inherited View List
	examples	hierarchy_bottom	**NONE**		spectre ads schematic extracted veriloga
	examples	hierarchy_top	schematic		spectre ads schematic extracted veriloga

Initial Hierarchy Editor Main Window

In the following figure, the global View List has been changed, so that it now includes *schematic_ideal* and *schematic_parasitics*. The cell bindings are now updated to reflect the new hierarchy expansion. Because *schematic_ideal* precedes *schematic_parasitics* in the global View List, the expansion now says that, for *hierarchy_bottom*, *schematic_ideal* is the view found. Also, the analogLib *res* cell has been found, because *hierarchy_bottom* was expanded. This expansion happens because *schematic_ideal* is not listed in the global Stop View List. Since *res* has an *ads* view, the expander decides that *ads* will be the view used for the *res* cell.


Virtuoso® Hierarchy Editor: (examples hierarchy_top config)
File Edit View Plugins Help
Top Cell
Open Edit
Global Bindings
Table View
Tree View
Cell Bindings

Library: examples
Cell: hierarchy_top
View: schematic

Library List: myLib
View List: spectre ads schematic schematic_ideal schematic_parasitics extract
Stop List: spectre ads
Constraint List:

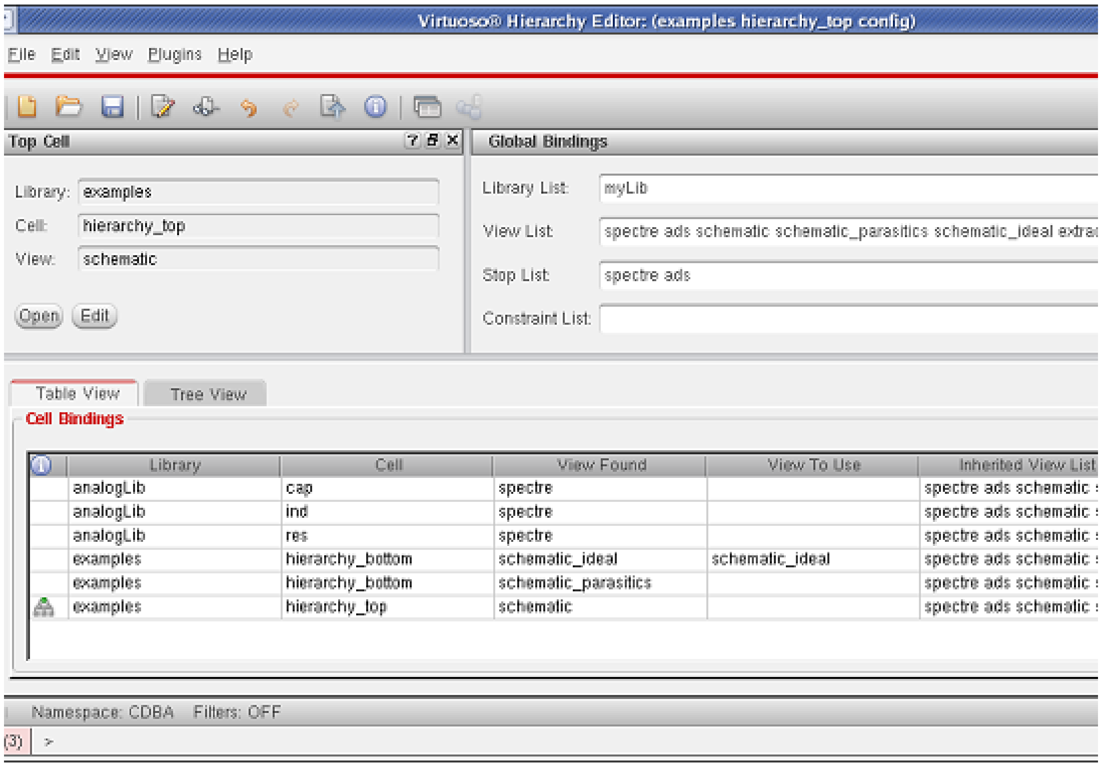
	Library	Cell	View Found	View To Use	Inherited View List
	analogLib	res	spectre		spectre ads schematic schematic_ideal schematic_parasitics
	examples	hierarchy_bottom	schematic_ideal		spectre ads schematic schematic_ideal schematic_parasitics
	examples	hierarchy_top	schematic		spectre ads schematic schematic_ideal schematic_parasitics

Namespace: CDBA Filters: OFF

Hierarchy Editor with schematic_ideal in Global View List

So far, all of this demonstrates is that the Hierarchy Editor can be used to see how expansion would occur for a specified View List and Stop List. That's nice, but it doesn't add any functionality over what was provided by the Setup Options form. What is needed is a way of overriding the View List. As it happens, the Hierarchy Editor is capable of doing just that.

In the following figure, the Switch View List has been changed, so that *schematic_parasitics* is now first in the Switch View List.


Virtuoso® Hierarchy Editor: (examples hierarchy_top config)
File Edit View Plugins Help
Top Cell
Global Bindings
Open Edit
Table View Tree View
Cell Bindings
Namespace: CDBA Filters: OFF
(3) >

Top Cell

Library:	examples
Cell:	hierarchy_top
View:	schematic

Global Bindings

Library List:	myLib
View List:	spectre ads schematic schematic_parasitics schematic_ideal extra
Stop List:	spectre ads
Constraint List:	

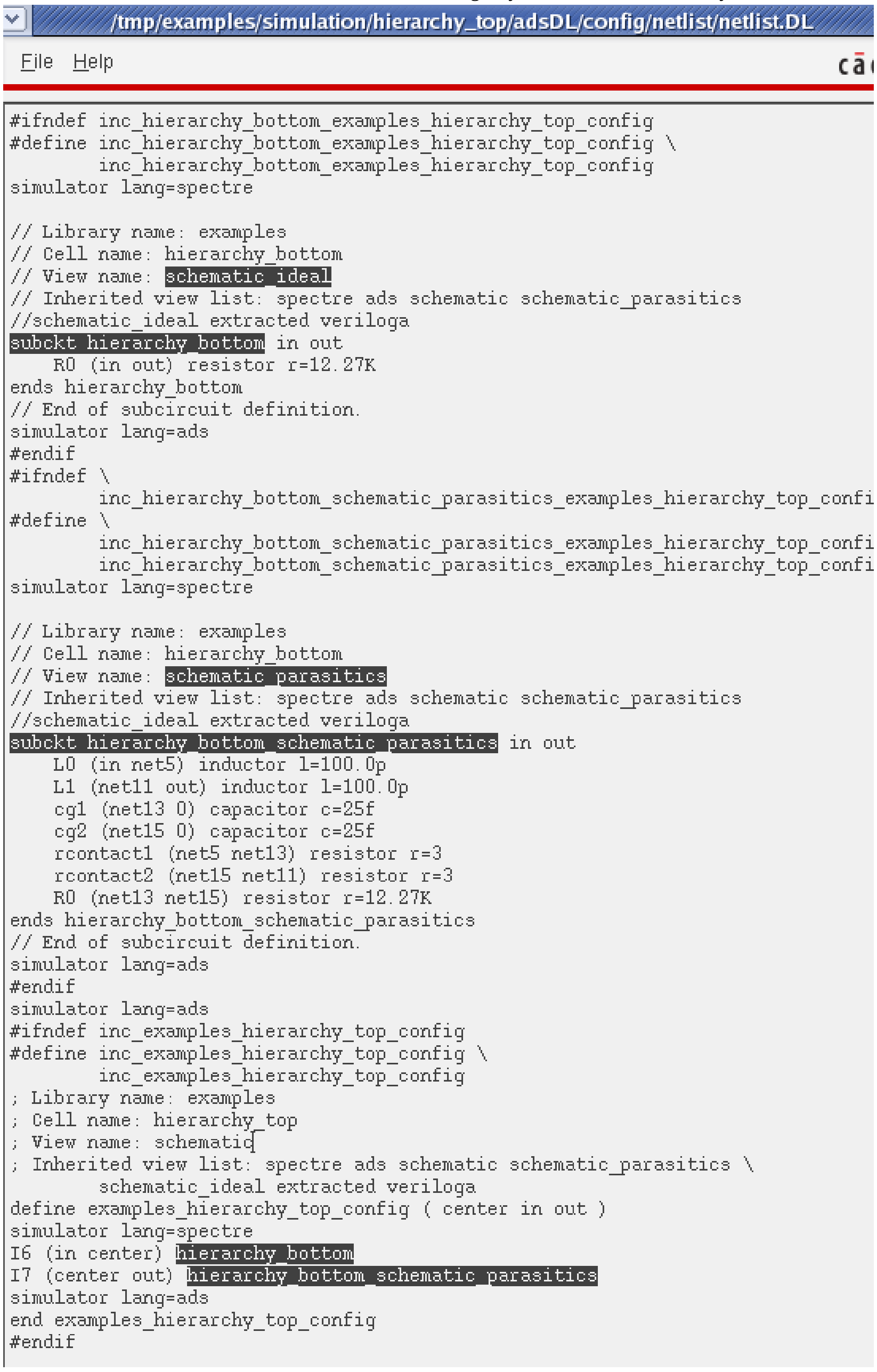
Cell Bindings

	Library	Cell	View Found	View To Use	Inherited View List
	analogLib	cap	spectre		spectre ads schematic s
	analogLib	ind	spectre		spectre ads schematic s
	analogLib	res	spectre		spectre ads schematic s
	examples	hierarchy_bottom	schematic_ideal	schematic_ideal	spectre ads schematic s
	examples	hierarchy_bottom	schematic_parasitics		spectre ads schematic s
	examples	hierarchy_top	schematic		spectre ads schematic s

Hierarchy Editor with an instance view overridden

Notice that the instance bindings table is now displayed. When *hierarchy_top* is selected, the instance table shows all of the instances in the schematic. As it happens, *hierarchy_top* contains two instances of *hierarchy_bottom*, *I6* and *I7* (see the figure The schematic view for the example cell hierarchy_top). If the global Switch View List is used, both instances will expand to use *schematic_parasitics*. In this case, we have chosen to change the default behavior. In the view to use field, *schematic_ideal* has been specified. The view found field now specifies *schematic_ideal* instead of *schematic_parasitics*. Now, when the cell *hierarchy_top* is netlisted with this configuration, it will be necessary to expand the hierarchy for both *schematic_parasitics* and *schematic_ideal*. The netlister keeps track of the proper component name for the netlist.

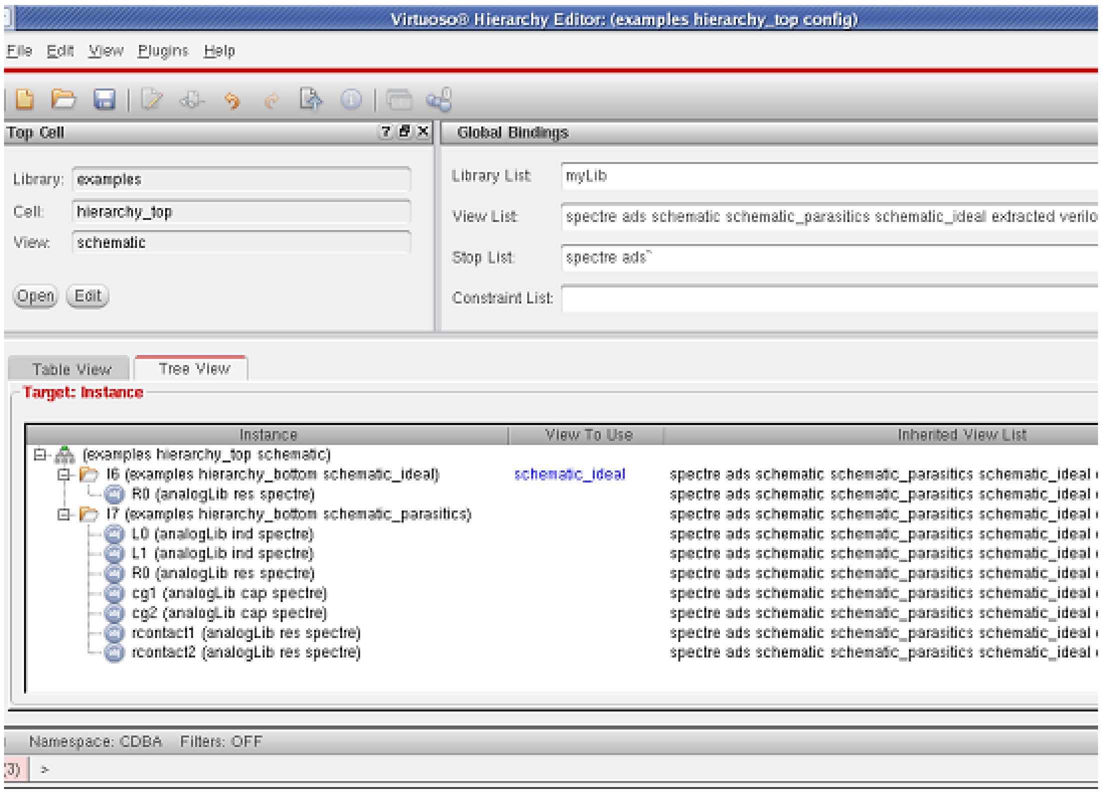
The figure below shows the resultant netlist that is created using this configuration.


/tmp/examples/simulation/hierarchy_top/adsDL/config/netlist/netlist.DL
File Help

```
#ifndef inc_hierarchy_bottom_examples_hierarchy_top_config
#define inc_hierarchy_bottom_examples_hierarchy_top_config \
        inc_hierarchy_bottom_examples_hierarchy_top_config
simulator lang=spectre

// Library name: examples
// Cell name: hierarchy_bottom
// View name: schematic_ideal
// Inherited view list: spectre ads schematic schematic_parasitics
//schematic_ideal extracted veriloga
subckt hierarchy_bottom in out
    R0 (in out) resistor r=12.27K
ends hierarchy_bottom
// End of subcircuit definition.
simulator lang=ads
#endif
#ifndef \
        inc_hierarchy_bottom_schematic_parasitics_examples_hierarchy_top_confi
#define \
        inc_hierarchy_bottom_schematic_parasitics_examples_hierarchy_top_confi
        inc_hierarchy_bottom_schematic_parasitics_examples_hierarchy_top_confi
simulator lang=spectre

// Library name: examples
// Cell name: hierarchy_bottom
// View name: schematic_parasitics
// Inherited view list: spectre ads schematic schematic_parasitics
//schematic_ideal extracted veriloga
subckt hierarchy_bottom_schematic_parasitics in out
    L0 (in net5) inductor l=100.0p
    L1 (net11 out) inductor l=100.0p
    cg1 (net13 0) capacitor c=25f
    cg2 (net15 0) capacitor c=25f
    rcontact1 (net5 net13) resistor r=3
    rcontact2 (net15 net11) resistor r=3
    R0 (net13 net15) resistor r=12.27K
ends hierarchy_bottom_schematic_parasitics
// End of subcircuit definition.
simulator lang=ads
#endif
simulator lang=ads
#ifndef inc_examples_hierarchy_top_config
#define inc_examples_hierarchy_top_config \
        inc_examples_hierarchy_top_config
; Library name: examples
; Cell name: hierarchy_top
; View name: schematic
; Inherited view list: spectre ads schematic schematic_parasitics \
        schematic_ideal extracted veriloga
define examples_hierarchy_top_config ( center in out )
simulator lang=spectre
I6 (in center) hierarchy_bottom
I7 (center out) hierarchy_bottom_schematic_parasitics
simulator lang=ads
end examples_hierarchy_top_config
#endif
```

Netlist using configuration detailed in previous figure.

An alternate way of viewing the overridden expansion is to look at the tree view, as opposed to the cell and instance binding tables. This is shown in the following figure.


Virtuoso® Hierarchy Editor: (examples hierarchy_top config)
File Edit View Plugins Help
Top Cell
Global Bindings
Open Edit
Table View
Tree View
Target: Instance

Library: examples
Cell: hierarchy_top
View: schematic

Library List: myLib
View List: spectre ads schematic schematic_parasitics schematic_ideal extracted verilo
Stop List: spectre ads
Constraint List:

Instance	View To Use	Inherited View List
(examples hierarchy_top schematic)		
I6 (examples hierarchy_bottom schematic_ideal)	schematic_ideal	spectre ads schematic schematic_parasitics schematic_ideal
R0 (analogLib res spectre)		spectre ads schematic schematic_parasitics schematic_ideal
I7 (examples hierarchy_bottom schematic_parasitics)		spectre ads schematic schematic_parasitics schematic_ideal
L0 (analogLib ind spectre)		spectre ads schematic schematic_parasitics schematic_ideal
L1 (analogLib ind spectre)		spectre ads schematic schematic_parasitics schematic_ideal
R0 (analogLib res spectre)		spectre ads schematic schematic_parasitics schematic_ideal
cg1 (analogLib cap spectre)		spectre ads schematic schematic_parasitics schematic_ideal
cg2 (analogLib cap spectre)		spectre ads schematic schematic_parasitics schematic_ideal
rcontact1 (analogLib res spectre)		spectre ads schematic schematic_parasitics schematic_ideal
rcontact2 (analogLib res spectre)		spectre ads schematic schematic_parasitics schematic_ideal

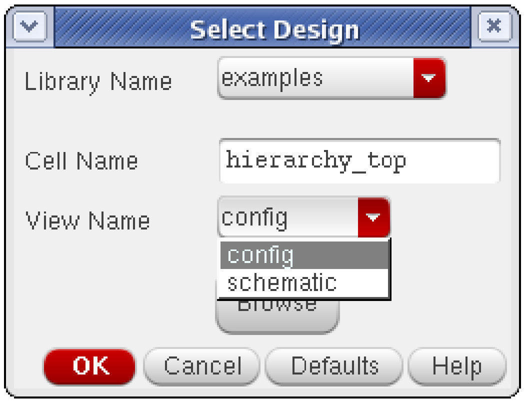
Namespace: CDBA Filters: OFF

Tree view of hierarchy_top configuration with I6 set to use schematic_ideal.

The Hierarchy Editor makes it possible to override the hierarchy expansion on any instance. It is also possible to override the global Switch View List on any instance. This allows you to change the expansion options for one tree of a hierarchy, while leaving all other trees intact.

Placing the config view in ADS

In order for a Hierarchy Editor configuration to be used during netlisting, it is necessary to place the Hierarchy Editor view in the ADS schematic. This is done by selecting the **DynamicLink > Instance > Add Instance of Cellview** menu option in ADS. When the *Select Design* dialog appears, set the view name to the Hierarchy Editor view. In the *hierarchy_top* example, this is done by selecting the view name *config* (see the following figure).



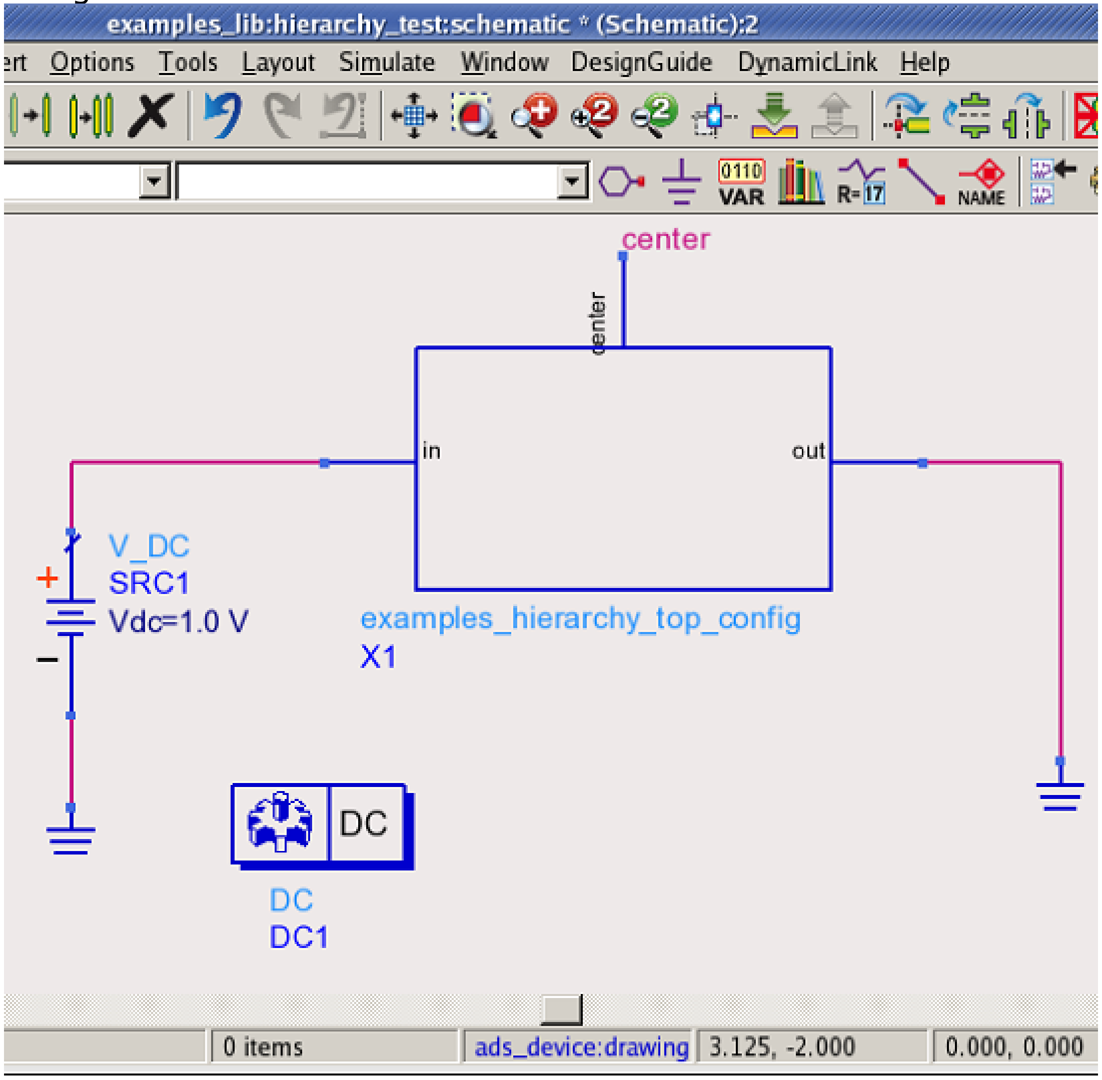
Library Name: examples

Cell Name: hierarchy_top

View Name: config

Selecting the Hierarchy Editor view for placement in ADS.

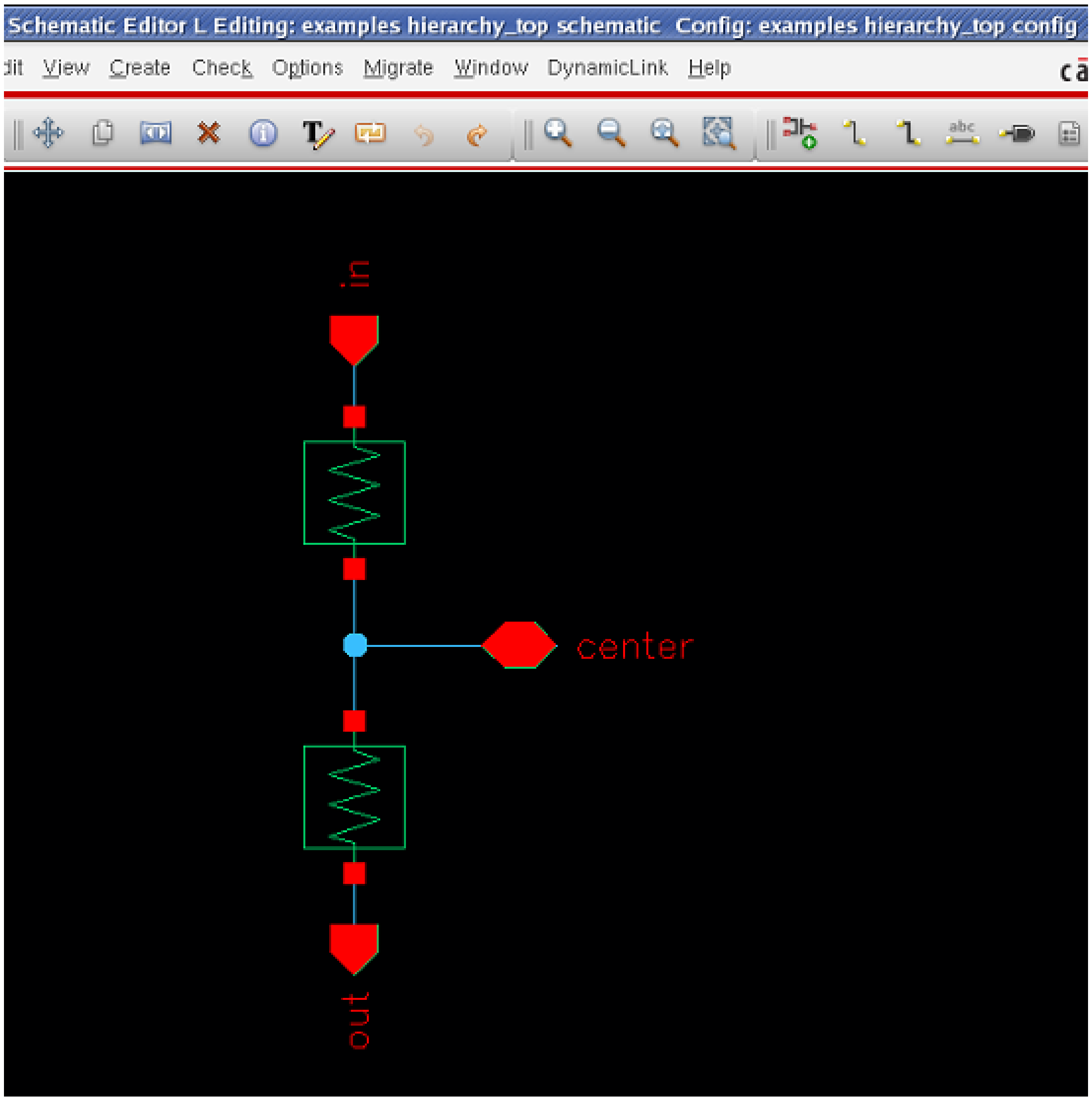
A new symbol is generated for the config view, and you are then be able to place it. In the *hierarchy_top* example, the symbol graphics are inherited from the *hierarchy_top* cell. This results in a symbol that looks identical to the *hierarchy_top* schematic symbol for ADS. However, this symbol is now linked to the Hierarchy Editor view, as opposed to being linked to the schematic view.


examples_lib:hierarchy_test:schematic * (Schematic):2
ert Options Tools Layout Simulate Window DesignGuide DynamicLink Help
VAR
NAME
center
center
in
out
V_DC
SRC1
Vdc=1.0 V
examples_hierarchy_top_config
X1
DC
DC
DC1
0 items
ads_device:drawing
3.125, -2.000
0.000, 0.000

Hierarchy Editor test bench with config view placed

The previous figure shows the new test bench where the config view is used instead of the schematic view. If the Cadence instance is selected, and you descend into it's hierarchy, the top cell view for the configuration will be opened. The configuration being used will be indicated in the title of the schematic/layout window that is opened. The following figure shows the *hierarchy_top* schematic that will be opened. Note that the title indicates that the Config in use is *examples hierarchy_top config*. This is important, as it means that hierarchy expansion will obey that particular configuration.

Note also that the top cell view does not need to remain constant. Thus, if you wish to do a simulation where the top cell schematic is used, and then do another simulation wherein an extracted view is used, you do not need to make multiple ADS symbols and then swap them. You can make a single Hierarchy Editor configuration, and swap the top cell view name. Once the test bench is set up, you do not need to modify anything in ADS to change your simulation. For the *hierarchy_top* example, a simulation would result in the netlist shown in the figure Netlist using configuration detailed in figure 10-15., based on the configuration shown in the figure Hierarchy Editor with an instance view overridden.


Schematic Editor L Editing: examples hierarchy_top schematic Config: examples hierarchy_top config
dit View Create Check Options Migrate Window DynamicLink Help
ca
in
center
out

Schematic window attached to a Hierarchy Editor configuration