cādence®

Virtuoso AMS Designer Environment Tutorials

Product Version 6.1.6
March 2014

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

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

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

Contents

1

Understanding AMS Designer Simulator Use Models

The Virtuoso® AMS Designer simulator is a single executable for language-based mixed-signal simulation. You can use the AMS Designer simulator to design and verify large and complex mixed-signal SoCs (systems on chips) and multichip designs. The two primary use models for the AMS Designer simulator are:

- AMS Designer Virtuoso use model

  For schematic-based, analog-centric designs, run the AMS Designer simulator from the Virtuoso Analog Design Environment (ADE) using the OSS netlister and `irun`.

- AMS Designer Incisive use model

  For digital-centric design verification, run the AMS Designer simulator from the command line using `irun`. This use model takes advantage of the power of the `amsd` block. For more information using `irun` and the `amsd` block for AMS simulation, see the *Virtuoso AMS Designer Simulator User Guide*. For tutorials that focus on this use model, see *Virtuoso AMS Designer Simulator Tutorials*.

Both use models feature the simulation front end (SFE) parser, which is the same parser that the Spectre circuit simulator uses.

The tutorials in this document focus on the AMS Designer Virtuoso use model (sometimes abbreviated as AVUM).

Important

Before running these tutorials, verify that your AMS Designer installation is set up and working. See also “Before You Begin” on page 13.

See also the following topics:

- Benefits of Using the OSS Netlister on page 8
- Important Considerations when Using the OSS Netlister on page 9

Benefits of Using the OSS Netlister

Using the OSS netlister means

- You do not need to convert Spectre PDKs to use the AMS simulator (because the OSS netlister uses Spectre CDF and simInfo, which is compatible with the AMS Designer simulator)
- You do not have to re-netlist the parts of your design that you have not changed (because the OSS netlister is an incremental netlister)
- You do not have to have writable master libraries and you do not have to use explicit or implicit TMP directories (because the OSS netlister does not write to the Cadence "5x" library/cell/view structure)
- You can run standalone simulations and debug more simply (because the OSS netlister writes a single netlist to one location)
- The compilation step is faster (because the OSS netlister does not use the "5x" structure)
- You can use `irun`, which is consistent with the digital use model such that you can share design information
- You can have VHDL modules in your design
    - Your VHDL modules must be at the leaf level only
    - You cannot use instance binding for VHDL modules

      For information about instance binding, see the *Cadence Hierarchy Editor User Guide*.

Important Considerations when Using the OSS Netlister

See the following topics for important considerations when using the OSS netlister:

- Netlisting Text Views on page 9
- Using View Lists and Stop Lists on page 10
- Netlisting the Symbol View for an Instance as Analog or Digital on page 10
- Netlisting Inherited Connections on page 11
- Updating Text Views that Do Not Have Design Database Information on page 11
- Selecting irun for Simulation on page 12

Netlisting Text Views

To netlist a text view, you must import the text view into the Cadence "5x" library/cell/view structure, or the text view must contain Virtuoso database files (`.oa`). You can satisfy this requirement using any of the following methods:

Note: If you have Verilog or VHDL text files that you do not want to netlist, you can specify them by clicking *Library Files/Directories* on the *Main* tab in the AMS Options form.

- Import Verilog text files using the Verilog In utility.

  For more information about using Verilog In, see the *Verilog In for Virtuoso Design Environment User Guide and Reference*.

- Import VHDL text files using the VHDL In utility.

  For more information about using VHDL In, see the *VHDL In for Virtuoso Design Environment User Guide and Reference*.

- In the schematic editor, choose *Create – Cellview – From Cellview*.
- In the command interpreter window (CIW), choose *File – New – Cellview*.
- In the Library Manager window, open the cell for editing and then save it.

Note: If you were previously using the cellview-based netlister and you do not know which cells do not have a Virtuoso database file (`.oa`), or if you have many views that do not have a database file, see “Updating Text Views that Do Not Have Design Database Information” on page 11.

Using View Lists and Stop Lists

Cadence software uses configuration view lists and stop lists to determine how to netlist design cells. Each cell in your design must have an associated schematic or simulator primitive. You specify valid views, in order of preference, in your view list. The netlister netlists cells that are leaf nodes in the design hierarchy according to the views you specify in the stop list. You specify the views that correspond to the most detailed simulation descriptions for a cell in the stop list. For each instance view in your design, the netlister stops the design hierarchy expansion process when it encounters a matching view in the stop list and writes that information to the netlist.

In the Cadence Hierarchy Editor, you can specify global bindings as well as view binding at the instance level. You can specify instance-level view binding if you want to netlist using a different view for a particular instance of a cell.

Note: For more information about configuration views, see "Understanding Configurations" in the *Virtuoso® AMS Designer Environment User Guide*. For more information about view lists and stop lists, see the *Cadence Hierarchy Editor User Guide*.

Netlisting the Symbol View for an Instance as Analog or Digital

By default, if you bind an instance to a symbol view in your configuration, the netlister netlists that instance as digital. To netlist such an instance instead as an analog primitive using the Spectre CDF simulation information, do one of the following:

- In the Virtuoso® Hierarchy Editor:

  a. Open your configuration.

  b. Replace `symbol` with `spectre` in the *View List* and *Stop List* fields.

- In ADE:

  a. Choose *Simulation – Options – AMS Simulator*,

  The AMS Options form appears.

  b. Select the *Netlister* tab.

  c. In the *Netlist using spectre CDF simInfo* field, type `symbol`.

Netlisting Inherited Connections

An inherited connection is a net expression associated with either a signal or a terminal. You use inherited connections to override specific global names in your design. You can use this feature for a design that has more than one power supply.

To implement separate power supplies (such as analog and digital, or +3 Volts and +5 Volts), do the following:

1. Assign net expressions to those global signals whose default values you might want to override.
2. Use `netSet` properties to specify new values for those signals.

Note: For more information about inherited connections and `netSet` properties, see “Inherited Connections” in the *Virtuoso Schematic Editor L User Guide*.

Updating Text Views that Do Not Have Design Database Information

The OSS-based AMS netlister does not support designs with text-only views; that is, text views that do not contain a Virtuoso® database file (`.oa`). You can create the `.oa` database for such views by doing the following:

1. In ADE, choose *Tools – Update Text Views*.

   The Update Text Views form appears.
2. Select *Create database for text views*.
3. In the *Text views* group box, specify the text view or views for which you want to create the Virtuoso database.
4. Click *OK*.

Selecting irun for Simulation

You can use `irun` for single-step compilation, elaboration, and simulation of your mixed-signal designs. AMS Designer relies on the OSS-based netlister to netlist designs for `irun` simulation. For OSS-based netlisting, you must run AMS from the Virtuoso® Analog Design Environment (ADE) and do the following:

1. Choose *Setup – Simulator*.

   The Choosing Simulator/Directory/Host form appears.

2. Select *ams* from the *Simulator* drop-down combo box.
3. Click *OK*.
4. Choose *Simulation – Netlist and Run Options*.

   The Netlist and Run Options form appears.

5. In the *NETLIST AND RUN MODE* section, select *OSS-based netlister with irun*.
6. Click *OK*.

Note: For more information about the OSS netlister, see "Using the OSS Netlister" in the *Virtuoso AMS Designer Environment User Guide*. For more information about `irun` simulation, see “Using irun for AMS Simulation” in the *Virtuoso AMS Designer Simulator User Guide*.

2

Before You Begin

Note: ADE is an abbreviation for the Virtuoso® Analog Design Environment. HED is an abbreviation for the Virtuoso Hierarchy Editor. AMS-Spectre stands for the AMS Designer simulator with the Spectre solver. AMS-UltraSim stands for the AMS Designer simulator with the UltraSim solver.

You can run these tutorials using the AMS Designer simulator along with Cadence® Virtuoso® software—such as the Virtuoso Analog Design Environment (ADE) and the Virtuoso Hierarchy Editor—from the IC 6.1.2 ISR 14 release or later. Your MMSIM installation must be version 7.0 or later.

Make sure your paths and environment variables are set up to use the correct releases. For example:

```
setenv CDSHOME /cds/tools/IC612
setenv AMSHOME /cds/tools/ius82
set path= ( $AMSHOME/tools/dfII/bin $AMSHOME/tools/bin $path)
set path = ( $CDSHOME/tools/bin $CDSHOME/tools/dfII/bin
$CDSHOME/tools/dfII/pvt/bin $path)
```

You can download the tutorial files from the installation hierarchy:

```
your_install_dir/tools/dfII/samples/tutorials/AMS/
```

To download all tutorials, do the following:

1. Create a tutorial directory in your local area. For example:

   ```
   mkdir myAmsTutorials
   ```

2. Copy the tutorial files from the installation hierarchy. For example:

   ```
   cp -r $CDSHOME/tools/dfII/samples/tutorials/AMS/* myAmsTutorials
   ```

   The system copies all the tutorial files from `samples/tutorials/AMS` into `myAmsTutorials`.

3. Change to your local tutorials directory. For example:

   ```
   cd myAmsTutorials
   ```

4. Decompress each tutorial file. For example, to decompress the `AMSDInADE.tar.gz` file, do the following:

```
gunzip AMSDInADE.tar.gz
tar xf AMSDInADE.tar
```

To download only a particular tutorial, do the following:

1. Change to the directory where you want to download the tutorial. For example:

```
cd myAmsTutorials
```

2. Copy the tutorial file from the installation hierarchy. For example:

```
cp -r $CDSHOME/tools/dfII/samples/tutorials/AMS/AMSDInADE.tar.gz .
```

The system copies the `AMSDInADE.tar.gz` file into the `myAmsTutorials` directory.

3. Decompress the archive file:

```
gunzip AMSDInADE.tar.gz
tar xf AMSDInADE.tar
```

You are ready to begin.

The following tutorials are available at `$CDSHOME/tools/dfII/samples/tutorials/AMS/`:

File	**Documentation**
AMSDInADE.tar.gz	“Building an AMS Test Case in ADE” on page 17
MATLABCosimulation.tar.gz	Coming soon... For now, see *Cosimulation Using the Virtuoso® AMS Designer Simulator and The MathWorks MATLAB®/Simulink®*
`AMSS_Envelope_ADE`	*Coming soon...*
`AMSSpectreTurbo`	*Coming soon...*
MigrateFromCBNToOSSN.tar.gz	“Migrating to the OSS Netlister” on page 129
MigrateFromVerimixToAMSDinADE.tar.gz	“Migrating to AMS Designer” on page 69
MultiPowerDis.tar.gz	“Using Digital Disciplines for Multiple Power Supply Design” on page 157
MultiPwerInhConn.tar.gz	“Using Inherited Connections for Multiple Power Supply Design” on page 137

3

Building an AMS Test Case in ADE

You can take an existing schematic-based analog design and some digital (Verilog, in this example) text modules and build an AMS test case in the Virtuoso® Analog Design Environment (ADE) and run a full-chip AMS simulation.

Note: Begin this tutorial in the `AMSDInADE` subdirectory. This tutorial examples takes about 30 minutes to complete, not including the simulation run time for the design.

You can run the AMS Designer simulator from ADE or from the Virtuoso® Hierarchy Editor (HED). You can also run the AMS Designer simulator using a command-line interface. Being familiar with using other circuit simulators (such as Spectre, UltraSim, and SpectreVerilog) in ADE can help you understand this tutorial.

This tutorial demonstrates how you can build an AMS test case in ADE and HED. The basic steps are as follows:

1. Start with an existing analog schematic-based design that you previously simulated using Spectre or UltraSim (or SpectreVerilog or UltraSimVerilog).
2. Import the digital modules—either Verilog or VHDL—into the Virtuoso design environment (we will demonstrate how to do this).
3. Complete the mixed-signal design in the Virtuoso Schematic Editor.
4. Configure the design in the Virtuoso Hierarchy Editor.
5. Set up options and customize connect rules in ADE.
6. Simulate the design using AMS in ADE.

Important

Before starting this tutorial, see “Before You Begin” on page 13.

See the following topics for details:

- The Tutorial Example on page 19
- Using Verilog In to Import the Verilog Module into ADE on page 20

The Tutorial Example

This tutorial example is a mixed-signal 160 MHz PLL circuit that consists of both schematic and Verilog language design units. The schematic contains the following analog components: a VCO, a phase frequency detector (PFD), a charge pump, and a loop filter. The two digital frequency dividers are RTL-level Verilog modules.

The key files and directories are:

`cds.lib`	Defines the associated libraries for designs, Cadence-shipped standard libraries, Fab foundry libraries, and so on
`artist_states`	Location of simulator setup information for ADE
`gpdk090`	90nm process design kit (PDK)
`models`	Device model files in Spectre format
`amsPLL`	Library of PLL blocks for the schematic database
`dig_source`	Location of the two behavioral Verilog frequency dividers
`clean_up`	Script for cleaning up intermediate files

The key signals are:

`pll_160MHZ_sim.I3.vCNTL`	VCO's control voltage signal
`pll_160MHZ_sim.CLK_REF`	25 MHz clock reference
`pll_160MHZ_sim.CLK_160MHZ`	160 MHz PLL output
`pll_160MHZ_sim.I3.VCO_CLK`	VCO's output voltage signal

Using Verilog In to Import the Verilog Module into ADE

You can use Verilog In to import a Verilog® module into a Cadence library and create a symbol for that module for use in the Virtuoso® schematic editor. We will import the two Verilog behavioral frequency dividers from the `dig_source` directory.

See the following topics for more information:

- Viewing the Verilog Behavioral Files for the Frequency Dividers on page 20
- Importing the Verilog Files into the Virtuoso Design Environment on page 21
- Viewing the New symbol and verilog Views for the Divider Cells on page 25

If you do not need to create a symbol for your module, you can use one of three other methods to import Verilog modules. See "Importing Verilog Modules without Creating a Symbol" on page 26 for more information.

Viewing the Verilog Behavioral Files for the Frequency Dividers

To view the two Verilog behavioral frequency dividers in the `dig_source` directory, do the following:

1. Change to the directory that contains the two behavioral Verilog frequency dividers:

   ```
   cd dig_source
   ```

2. Type `ls` to view the files:

   ```
   ls
   ```

   You will see the following two files:

`PLL_160MHZ_MDIV.v`	Divider with factor 64, for VCO's output clock
`PLL_160MHZ_PDIV.v`	Divider with factor 5, for input reference clock

3. Type more `*.v` to view their contents.

Importing the Verilog Files into the Virtuoso Design Environment

To import these files into the Virtuoso design environment (that is, into the lib/cell/view structure of Cadence libraries), do the following:

1. Start Cadence software:

   ```
   virtuoso &
   ```

   Note: If you get a warning message such as the following in your CIW when you start Cadence software:

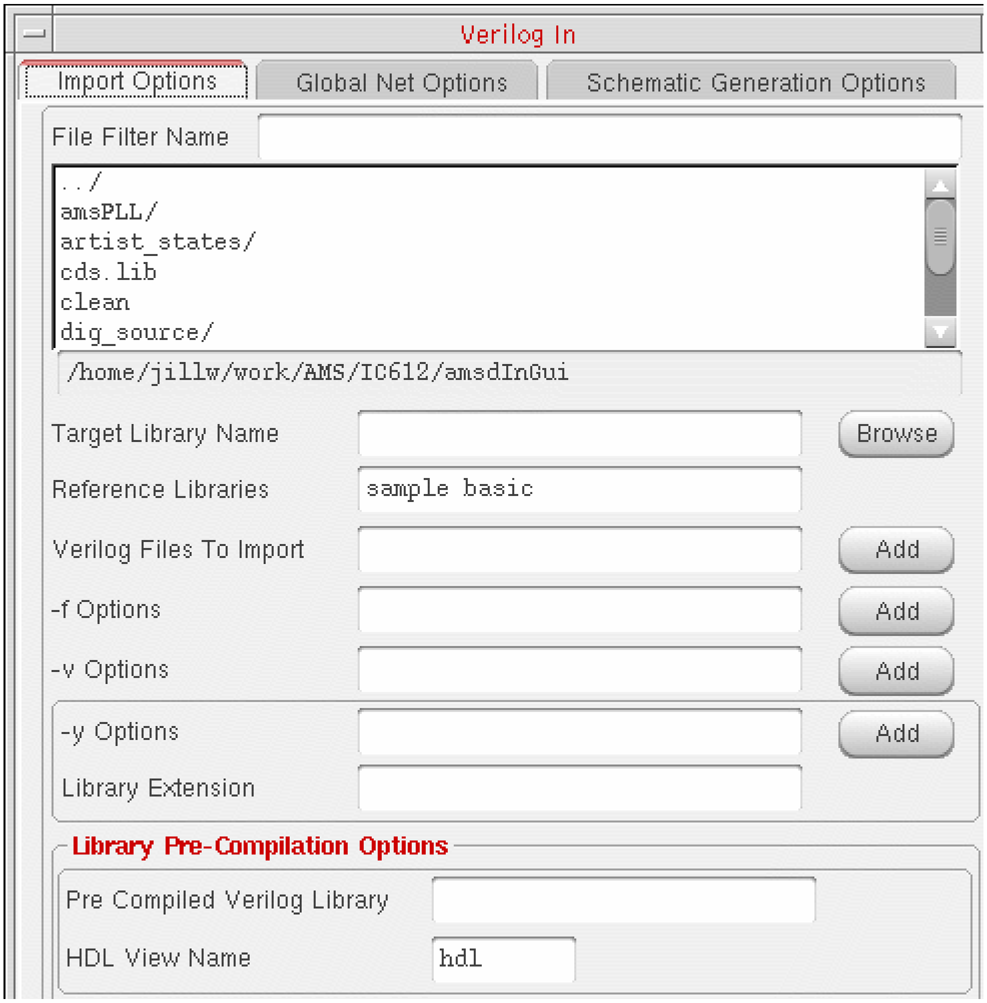
   ```
   *WARNING* envLoadFile: file .cdsenv does not exist, or is not readable.
   ```

   you can add `;;` to the beginning of the following command in the `.cdsinit` file so that it does not execute:

   ```
   envLoadFile(".cdsenv")
   ```

2. In the CIW, choose *File – Import – Verilog*.

The Verilog In form appears.



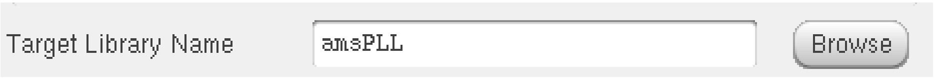
```
../
amsPLL/
artist_states/
cds.lib
clean
dig_source/
```

/home/jillw/work/AMS/IC612/amsdInGui

Reference Libraries: sample basic

HDL View Name: hdl

3. In the *Target Library Name* field, type `amsPLL`.

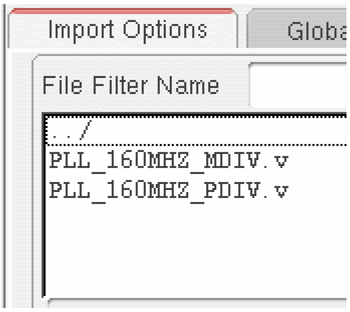


Target Library Name: amsPLL

Note: You can also use the *Browse* button to open the Library Browser form and select *amsPLL* from the *Library* column on that form.

4. In the *File Filter Name* scrolling list box, double-click *dig_source*.

The two Verilog behavioral files in that directory appear on the form.



File Filter Name

```
../
PLL_160MHZ_MDIV.v
PLL_160MHZ_PDIV.v
```

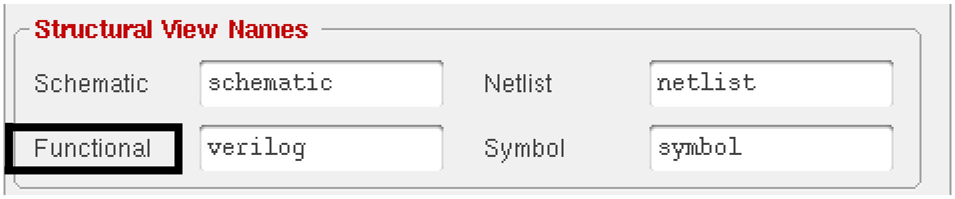
5. For each of the two files in this list, do the following:

   a. Select the file name.

   b. To the right of the *Verilog Files to Import* field, click *Add*.

      The full path to the file appears in the field. When more than one file name appears in the field, a space separates one file name from the next.

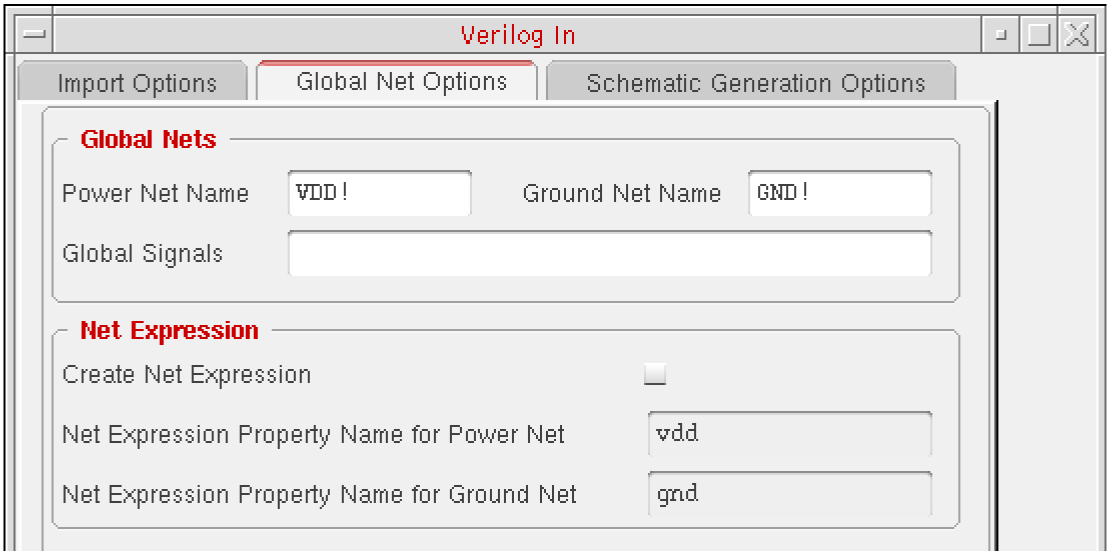
6. In the *Structural View Names* group box, type `verilog` in the *Functional* field.



Structural View Names

Schematic	schematic	Netlist	netlist
Functional	verilog	Symbol	symbol

7. Select the *Global Net Options* tab.



Global Nets

Power Net Name	VDD!	Ground Net Name	GND!
Global Signals			

Net Expression

Create Net Expression	□
Net Expression Property Name for Power Net	vdd
Net Expression Property Name for Ground Net	gnd

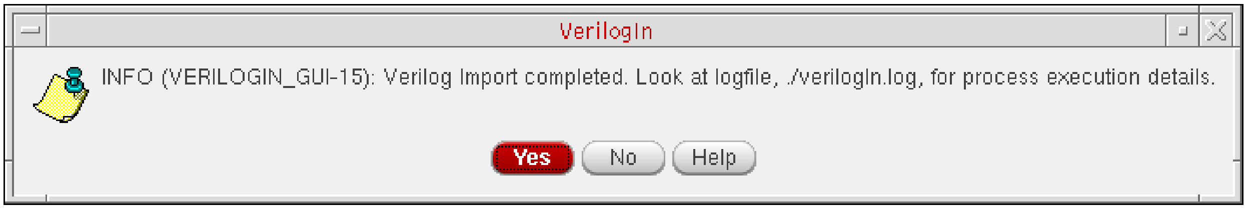
8. In the *Power Net Name* field, change *VDD!* to `VDD1!`.

   This tutorial example has nets `VDD` and `VSS`, and neither is a global net, so to avoid a name conflict (with our `VDD` net), we change the global power net name here to `VDD1!`.

9. Click *OK*.

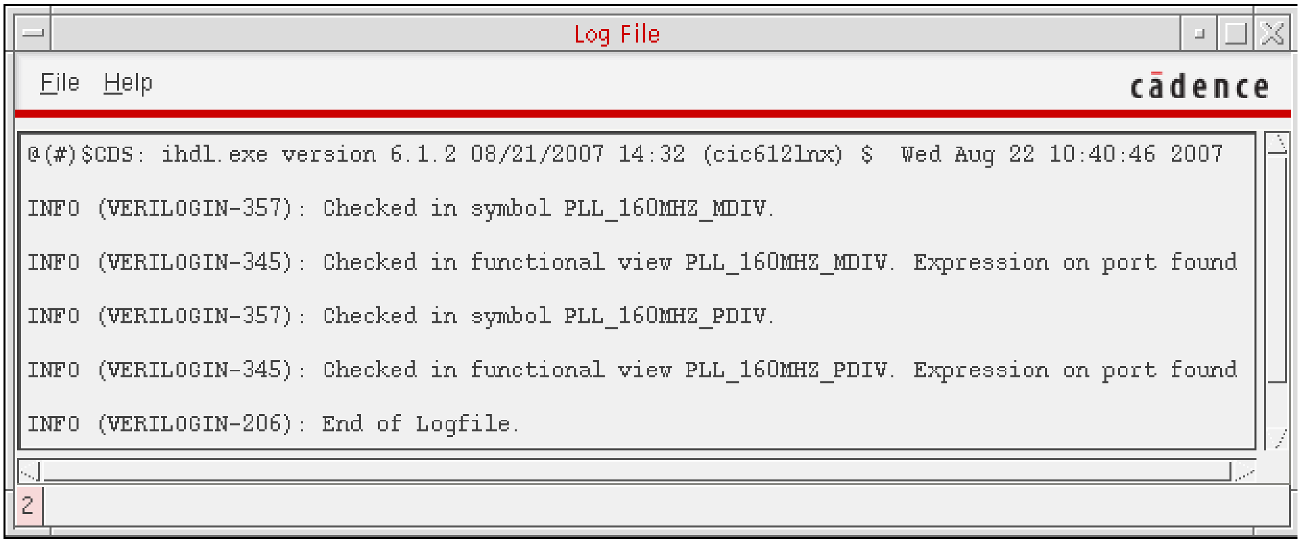
   Informational messages appear in the output area of the CIW.

   A prompt appears.



INFO (VERILOGIN_GUI-15): Verilog Import completed. Look at logfile, ./verilogIn.log, for process execution details.

10. To view the log file, click *Yes*. (If you do not care to see the log file, you can click *No*.)



```
@(#)$CDS: ihdl.exe version 6.1.2 08/21/2007 14:32 (cic612lnx) $  Wed Aug 22 10:40:46 2007
INFO (VERILOGIN-357): Checked in symbol PLL_160MHZ_MDIV.
INFO (VERILOGIN-345): Checked in functional view PLL_160MHZ_MDIV. Expression on port found
INFO (VERILOGIN-357): Checked in symbol PLL_160MHZ_PDIV.
INFO (VERILOGIN-345): Checked in functional view PLL_160MHZ_PDIV. Expression on port found
INFO (VERILOGIN-206): End of Logfile.
```

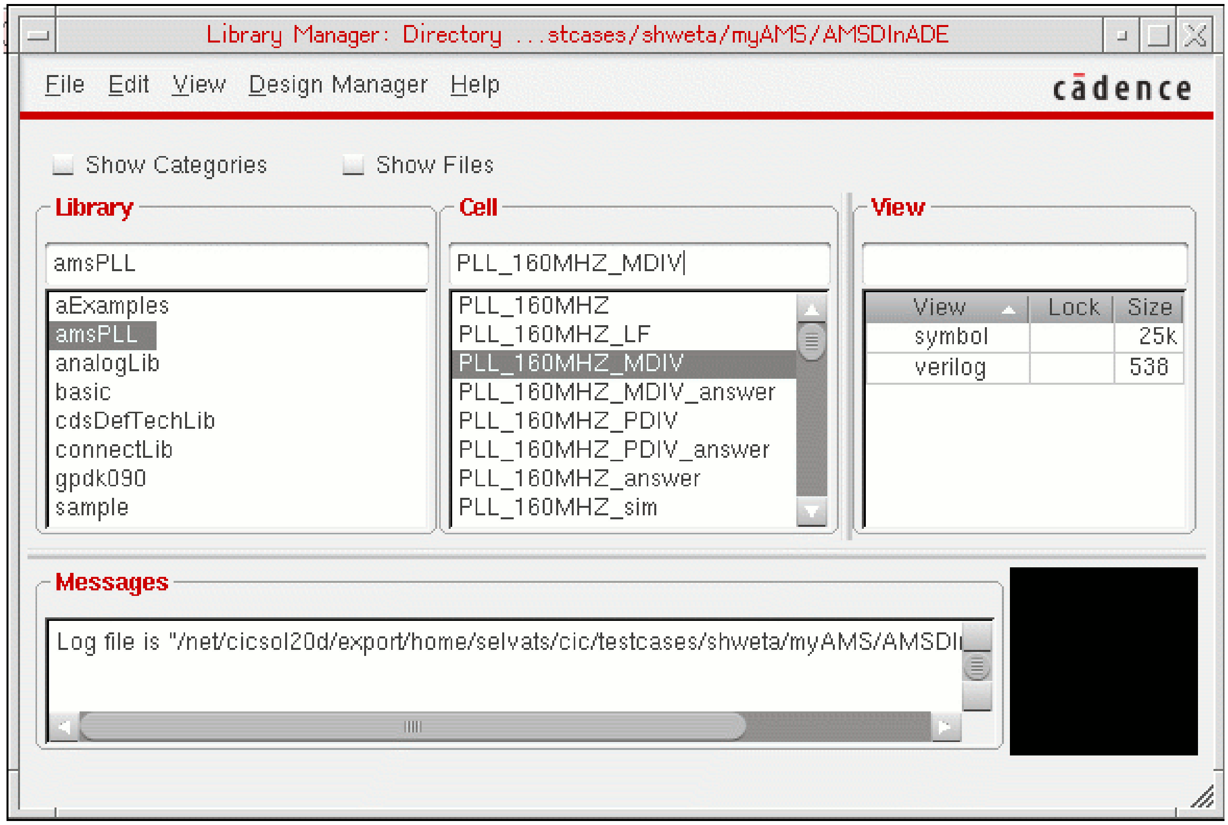
11. Choose *File – Close Window* to close the log file viewing window.

Viewing the New symbol and verilog Views for the Divider Cells

You can view the new `symbol` and `verilog` views for these two new cells (`PLL_160MHZ_MDIV` and `PLL_160MHZ_PDIV`) in the `amsPLL` library as follows:

1. In the CIW, choose *Tools – Library Manager*.
2. In the *Library* column, select *amsPLL*.
3. In the *Cell* column for each new cell, select the new cell name (*PLL_160MHZ_MDIV* and *PLL_160MHZ_PDIV*).

   In the *View* column for each new cell you select, both *symbol* and *verilog* views appear.



Library: amsPLL

Cell: PLL_160MHZ_MDIV

View	Lock	Size
symbol		25k
verilog		538

```
Log file is "/net/cicsol20d/export/home/selvats/cic/testcases/shweta/myAMS/AMSDI
```

Note: You can ignore the *PLL_160MHZ_MDIV_answer* and *PLL_160MHZ_PDIV_answer* cells which we provided with this tutorial for your reference and convenience only.

Importing Verilog Modules without Creating a Symbol

You can use the following methods to import a Verilog module for simulation:

- Use `-y`/`-v` in the analog design environment (ADE) to specify the Verilog text module.

  Note: This method produces results similar to the original SpectreVerilog simulator. For information about how to specify these options, see "Specifying Library Files and Directories for the Compiler" in the *Virtuoso AMS Designer Environment User Guide*. See also "Choose Simulation – Options – AMS Simulator" in the "Running a Simulation" chapter of the *Virtuoso Analog Design Environment L User Guide*.

- Specify the Verilog source file in the Virtuoso® hierarchy editor.

- Use `ncvlog -use5x` to compile the Verilog module into a lib/cell/view Cadence library structure.

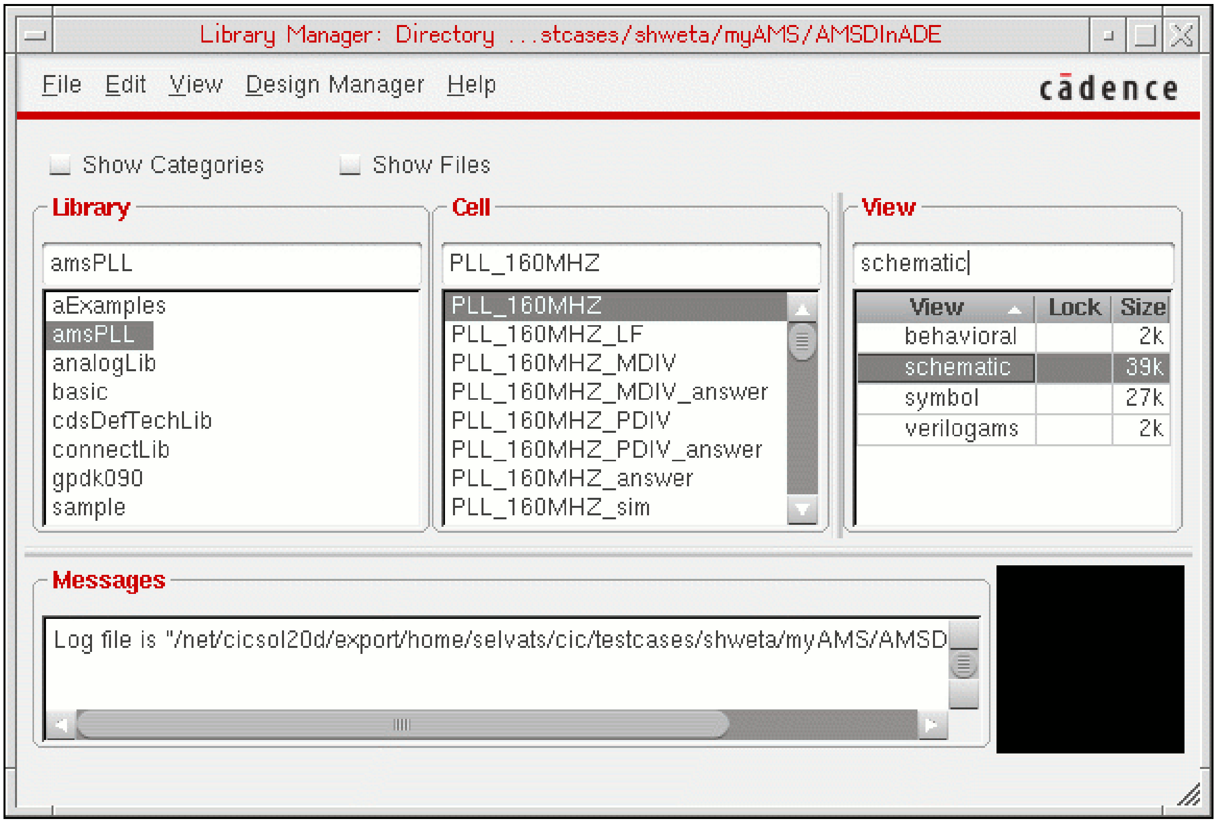
  See "ncvlog Command Syntax and Options" in the *Virtuoso AMS Designer Simulator User Guide* for information about the `-use5x` option.

When you use any of these methods, you must create the symbol yourself. Alternatively, you can use Verilog In to create a symbol for you. See "Using Verilog In to Import the Verilog Module into ADE" on page 20 for more information.

Adding Divider Block Instances to the Tutorial Schematic

To add component instances for the Verilog behavioral divider cells to the tutorial schematic, do the following:

1. In the Library Manager, select *amsPLL, PLL_160MHZ, schematic*.


Library Manager: Directory ...stcases/shweta/myAMS/AMSDInADE
File
Edit
View
Design Manager
Help
cādence
Messages

□ Show Categories □ Show Files

Library

amsPLL

- aExamples
- amsPLL
- analogLib
- basic
- cdsDefTechLib
- connectLib
- gpdk090
- sample

Cell

PLL_160MHZ

- PLL_160MHZ
- PLL_160MHZ_LF
- PLL_160MHZ_MDIV
- PLL_160MHZ_MDIV_answer
- PLL_160MHZ_PDIV
- PLL_160MHZ_PDIV_answer
- PLL_160MHZ_answer
- PLL_160MHZ_sim

View

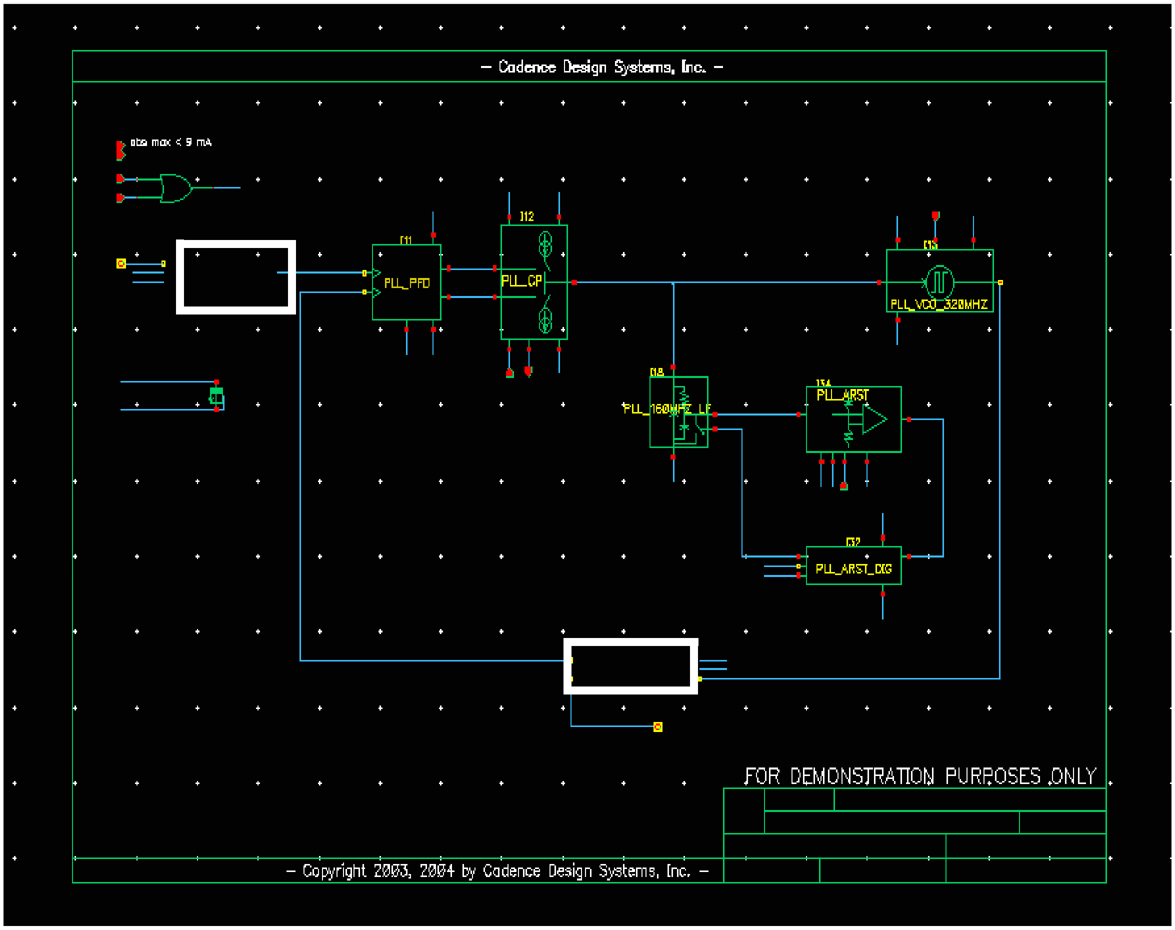
schematic

View	Lock	Size
behavioral		2k
schematic		39k
symbol		27k
verilogams		2k

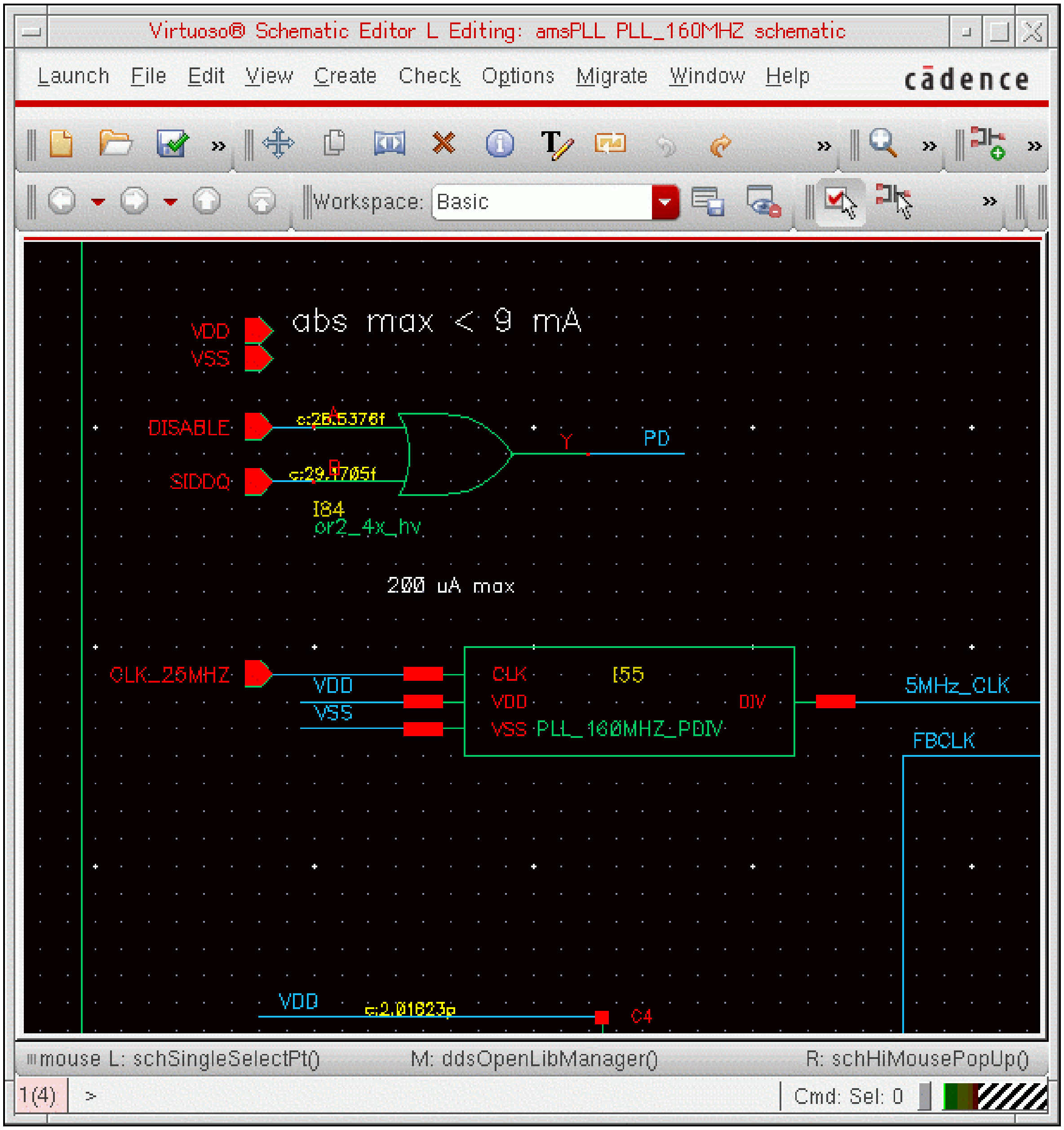
```
Log file is "/net/cicsol20d/export/home/selvats/cic/testcases/shweta/myAMS/AMSD
```

2. Choose *File – Open*.

The tutorial schematic appears in a Schematic Editor window. There are vacancies on the schematic where you will place the new symbols.


– Cadence Design Systems, Inc. –
abs max < 9 mA
I11
PLL_PFD
I12
PLL_CP
I13
PLL_VCO_320MHZ
I18
PLL_160MHZ_LF
I34
PLL_ARST
I32
PLL_ARST_DIG
FOR DEMONSTRATION PURPOSES ONLY
– Copyright 2003, 2004 by Cadence Design Systems, Inc. –

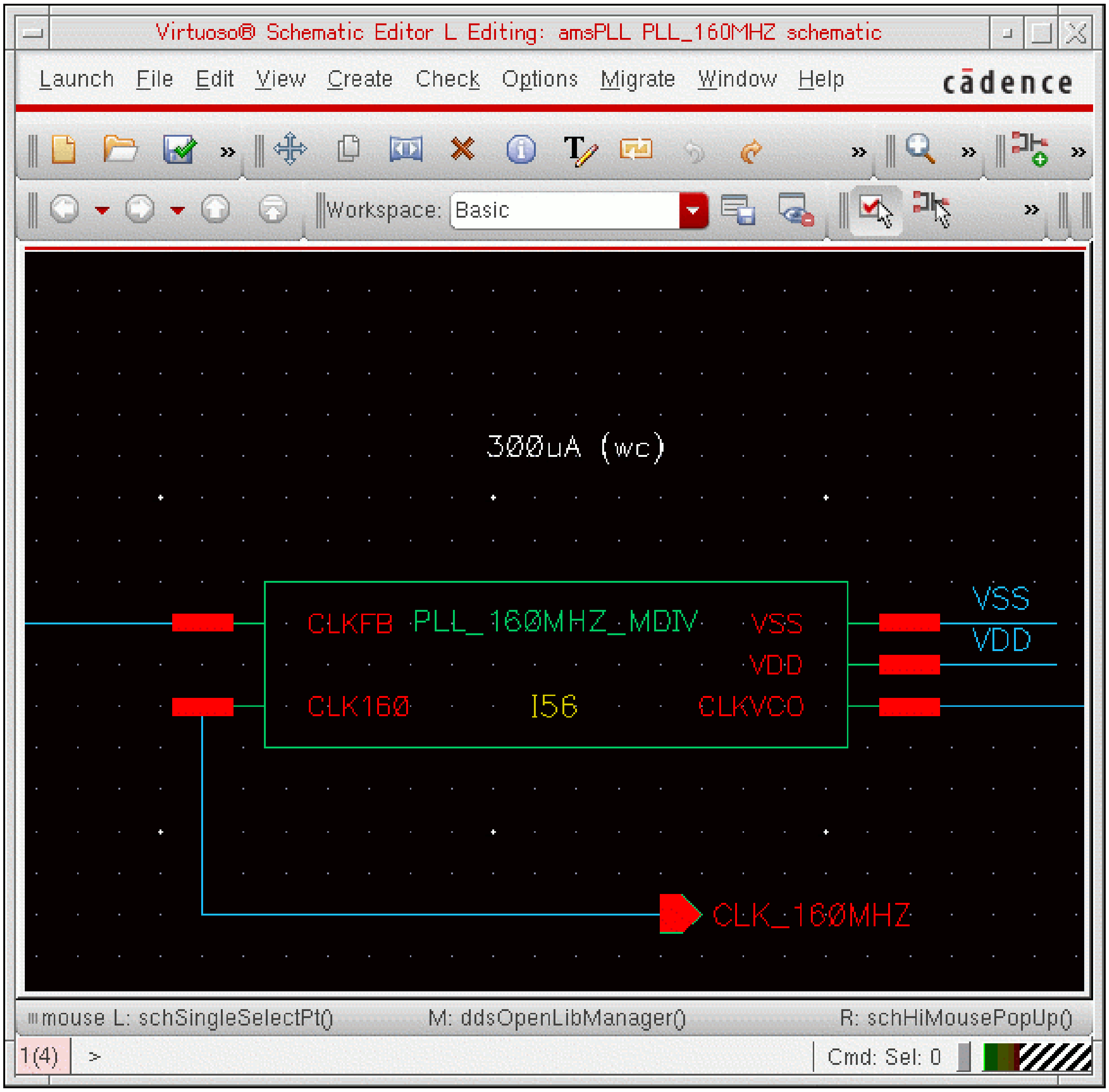
3. Place an instance of *PLL_160MHZ_PDIV* in the upper left vacancy and connect the nets as follows:



Tip

If you want instructions for how to select and place these parts on the schematic, see “Selecting and Placing Divider Cell Instances” on page 31.

4. Place an instance of *PLL_160MHZ_MDIV* in the lower vacancy and connect the nets as follows:


Virtuoso® Schematic Editor L Editing: amsPLL PLL_160MHZ schematic
Launch File Edit View Create Check Options Migrate Window Help
cādence
Workspace: Basic
300uA (wc)
CLKFB
PLL_160MHZ_MDIV
VSS
VDD
CLK160
I56
CLKVCO
VSS
VDD
CLK_160MHZ
mouse L: schSingleSelectPt()
M: ddsOpenLibManager()
R: schHiMousePopUp()
1(4)
>
Cmd: Sel: 0

Note: Remember to type `r` twice to rotate the part to its correct orientation.

5. Check and save the schematic.

Note: We provided the `PLL_160MHZ_answer` schematic with this tutorial for your reference and convenience.

Selecting and Placing Divider Cell Instances

To select and place divider cell instances, do the following:

1. In the schematic editor, choose *Create – Instance* (or type `i`).

   The Add Instance form appears.

2. Click *Browse*.

   The Library Browser window appears.

3. In the *Library* column, select *amsPLL*.
4. Click *Filters*.

   The View Filter By form appears.

5. In the *Cell Filter* field, type `PLL_160MHZ*` to narrow the search field and click *OK*.

   Only cell names beginning with this string appear in the *Cell* column.

6. In the Library Browser window, select *PLL_160MHZ_PDIV*.

   As you drag your mouse over the schematic, you can see the outline of the symbol.

7. Click to place the instance on the schematic.
8. Select *PLL_160MHZ_MDIV*.

   As you drag your mouse over the schematic, you can see the outline of the symbol. You will need to change the orientation of this instance before placing it.

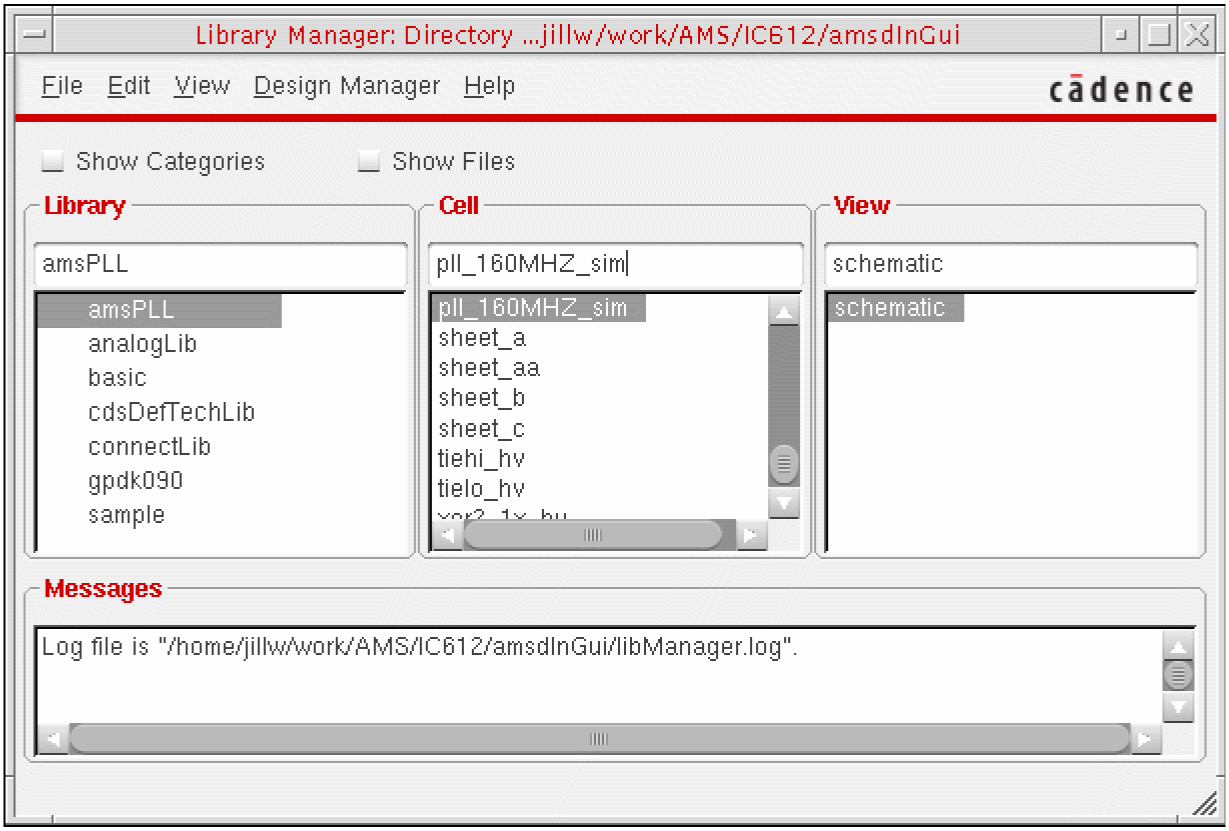
9. Type `r` twice to rotate the instance into the correct orientation.
10. Click to place the instance on the schematic.
11. Press *Esc* when finished.

For more information, see “Adding Instances” in the “Creating Schematics” chapter of the *Virtuoso Schematic Editor L User Guide*.

Viewing the Testbench for the PLL Design

To view the testbench schematic for the tutorial example (a PLL design), do the following:

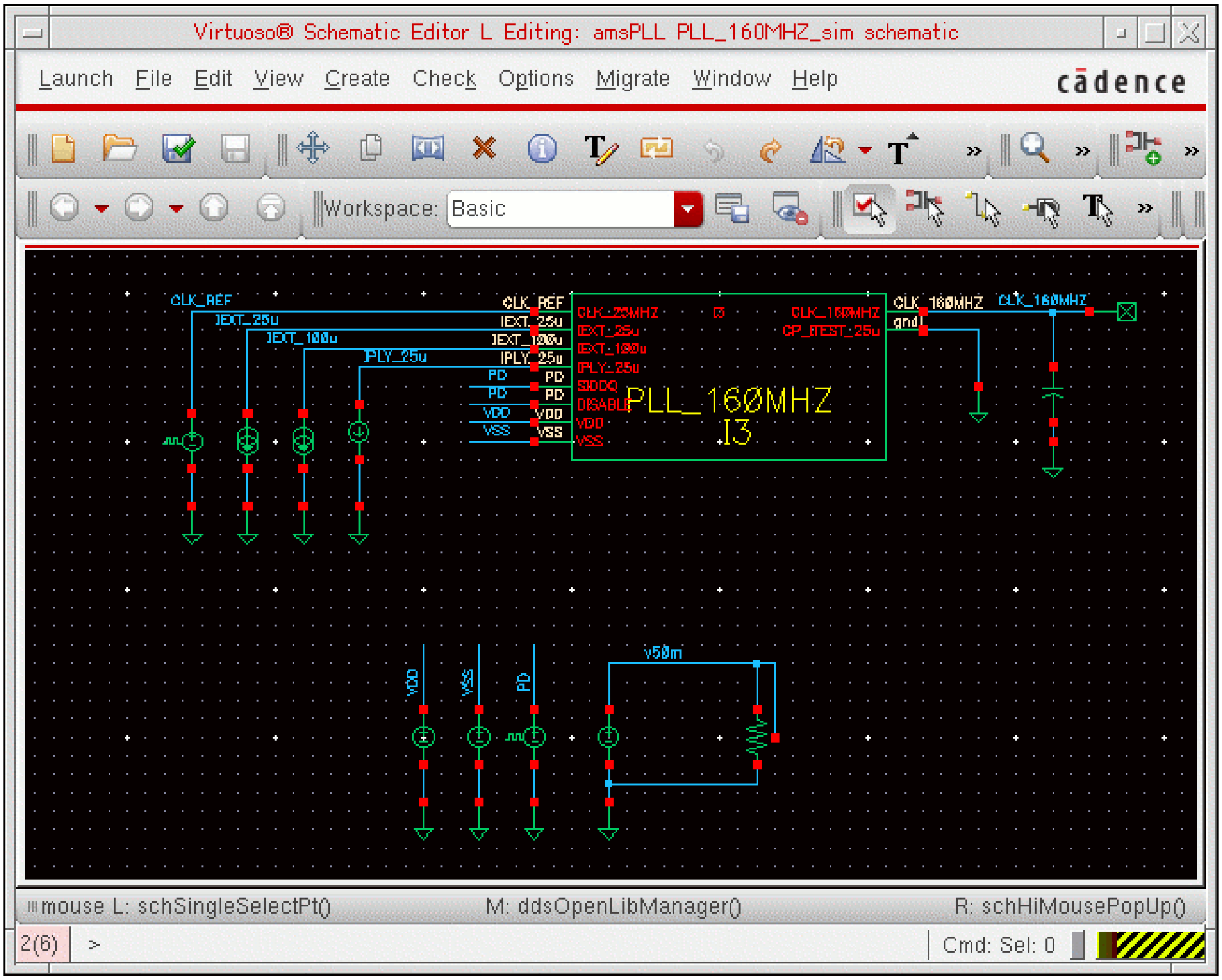
1. In the Library Manager, select *amsPLL, pll_160MHZ_sim, schematic*.



Log file is "/home/jillw/work/AMS/IC612/amsdlnGui/libManager.log".

2. Choose *File – Open*.

The testbench schematic appears in a Schematic Editor window.



Workspace: Basic

mouse L: schSingleSelectPt() M: ddsOpenLibManager() R: schHiMousePopUp()

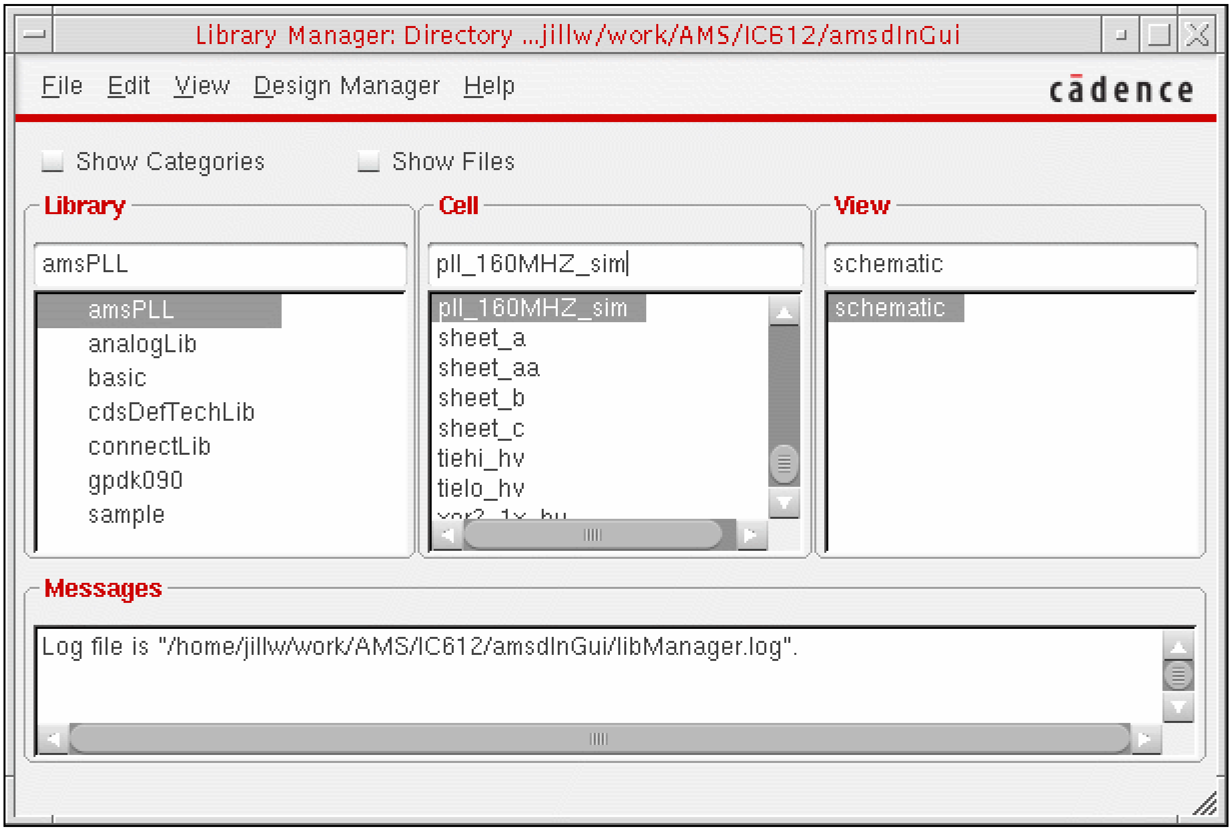
Cmd: Sel: 0

Notice that the power supply (*VDD*) is 2.5 Volts.

Creating a Configuration View for AMS Simulation

To create a configuration view for the testbench schematic for AMS simulation, do the following:

1. In the Library Manager, select *amsPLL, pll_160MHZ_sim*.


Library Manager: Directory ...jillw/work/AMS/IC612/amsdlnGui
File
Edit
View
Design Manager
Help
cādence

□ Show Categories □ Show Files

Library: amsPLL

- amsPLL
- analogLib
- basic
- cdsDefTechLib
- connectLib
- gpdk090
- sample

Cell: pll_160MHZ_sim

- pll_160MHZ_sim
- sheet_a
- sheet_aa
- sheet_b
- sheet_c
- tiehi_hv
- tielo_hv

View: schematic

- schematic

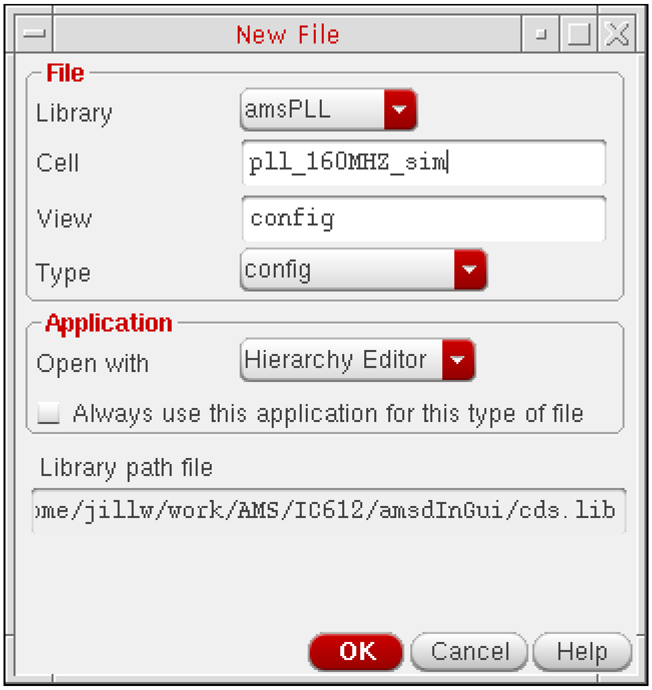
Messages

```
Log file is "/home/jillw/work/AMS/IC612/amsdlnGui/libManager.log".
```

2. Choose *File – New – Cell View*.

   The New File form appears.

3. In the *Type* field, select *config*.



New File

File

Library: amsPLL

Cell: pll_160MHZ_sim

View: config

Type: config

Application

Open with: Hierarchy Editor

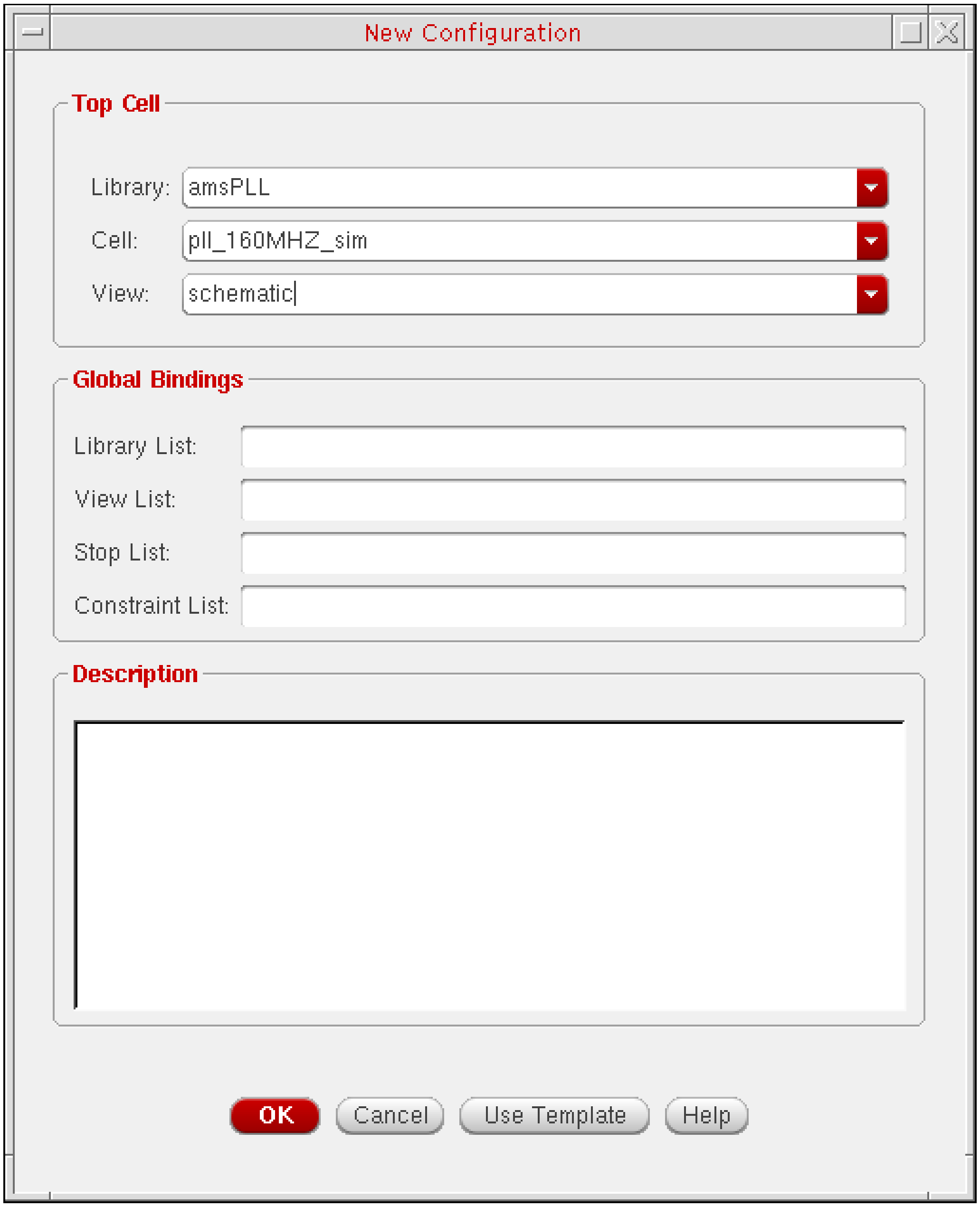
□ Always use this application for this type of file

Library path file: ome/jillw/work/AMS/IC612/amsdInGui/cds.lib

4. Click *OK*.

The New Configuration form appears.

5. In the *View* field in the *Top Cell* group box, select *schematic*.



New Configuration

Top Cell

Library: amsPLL

Cell: pll_160MHZ_sim

View: schematic

Global Bindings

Library List:

View List:

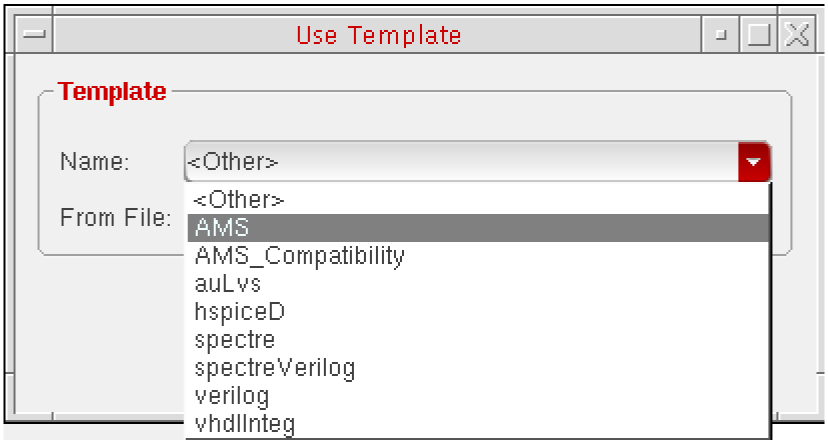
Stop List:

Constraint List:

Description

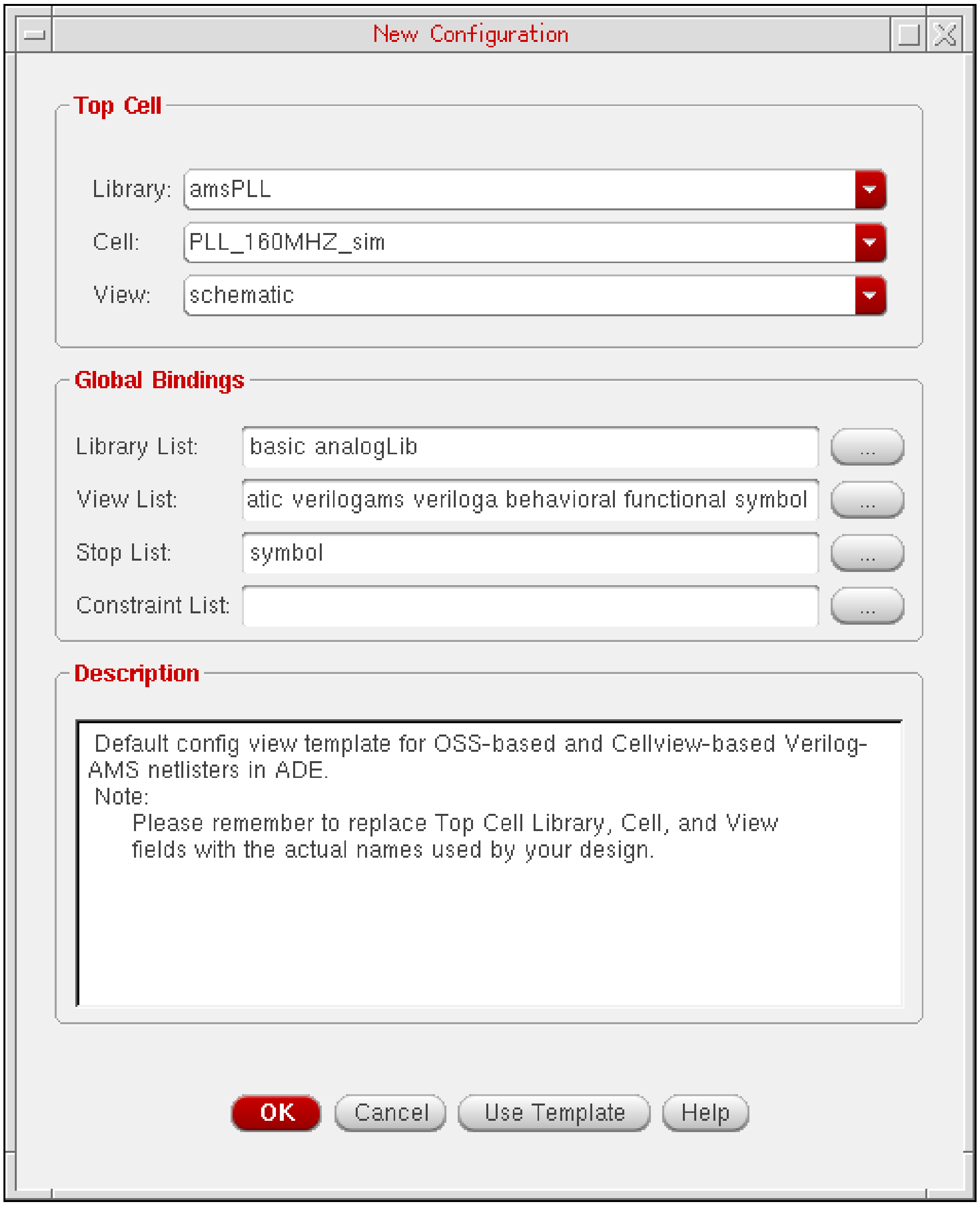
6. Click *Use Template*.

7. In the *Name* field in the *Template* group box, select *AMS*.



8. On the Use Template form, click *OK*.

Bindings appropriate for AMS simulation appear in the fields of the *Global Bindings* group box. For more information about global bindings, see "Global Bindings Section" in the "Cadence Hierarchy Editor Overview."



New Configuration

Top Cell

Library: amsPLL
Cell: PLL_160MHZ_sim
View: schematic

Global Bindings

Library List: basic analogLib
View List: atic verilogams veriloga behavioral functional symbol
Stop List: symbol
Constraint List:

Description

Default config view template for OSS-based and Cellview-based Verilog-AMS netlisters in ADE.
Note:
Please remember to replace Top Cell Library, Cell, and View fields with the actual names used by your design.

In this tutorial design, most cells are schematic cells. Only `PLL_160MHZ_MDIV` and `PLL_160MHZ_PDIV` are Verilog cells. Therefore, to get the cell bindings you want, you need to change the view list so that `verilog` and `schematic` appear at the beginning of the list.

9. In the *View List*, click to place the edit cursor and change the view list so that `verilog` and `schematic` appear at the beginning of the list.

   ```
   verilog schematic ...
   ```

10. Click *OK*.
11. In the Virtuoso® Hierarchy Editor window, choose *File – Save (Needed)*.

    The `config` view is open in the hierarchy editor and the `schematic` view is open in the schematic editor.

Setting Up the Simulation in the Analog Design Environment

To set up the simulation in the analog design environment (ADE), do the following:

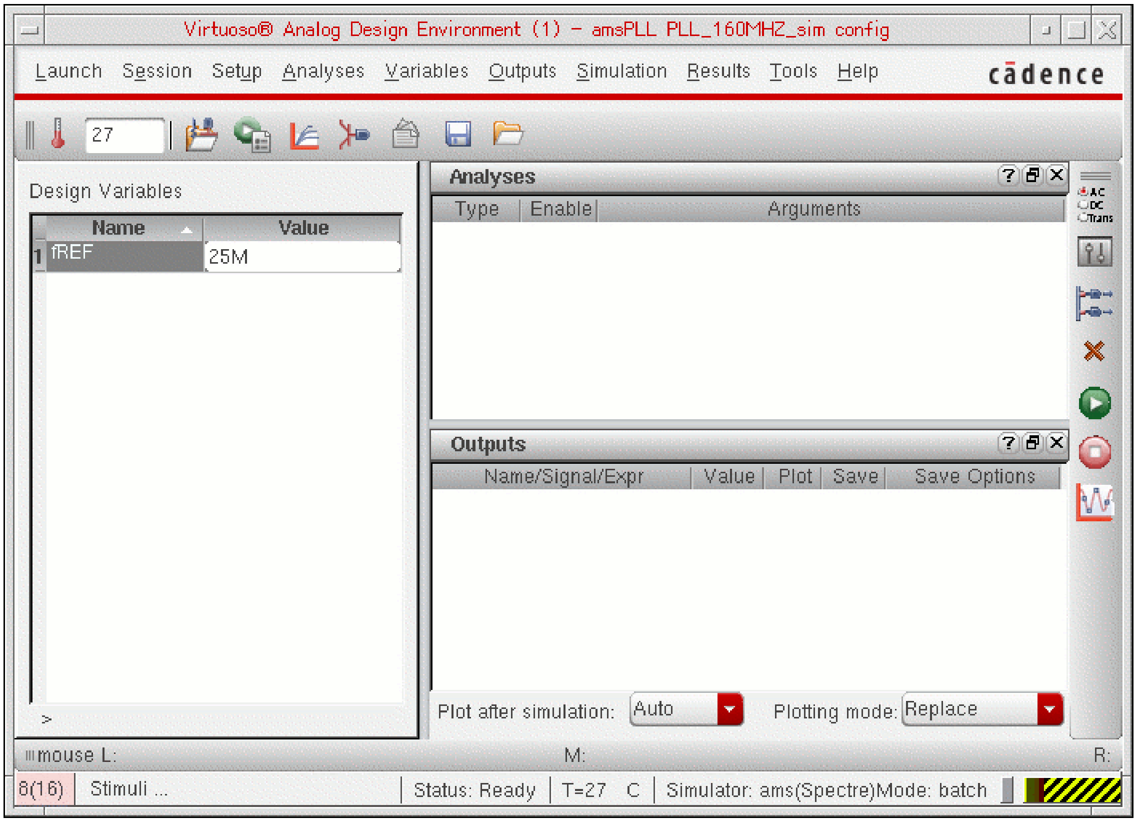
- Launching ADE on page 39
- Selecting the AMS Designer Simulator with the Spectre Solver on page 41
- Specifying the Transient Analysis on page 42
- Specifying the Model Libraries on page 43
- Customizing Connect Rules on page 45

Launching ADE

To launch ADE and specify the `config` view, do the following:

1. In the Schematic Editor window for *amsPLL pll_160MHZ_sim schematic*, choose *Launch – ADE L*.

The Virtuoso® Analog Design Environment window appears.



Virtuoso® Analog Design Environment (1) - amsPLL PLL_160MHZ_sim config
Launch Session Setup Analyses Variables Outputs Simulation Results Tools Help
cādence
27
AC
DC
Trans
mouse L:
M:
R:
8(16)
Stimuli ...

Design Variables

	Name	Value
1	fREF	25M

Analyses

Type	Enable	Arguments

Outputs

Name/Signal/Expr	Value	Plot	Save	Save Options

Plot after simulation: Auto Plotting mode: Replace

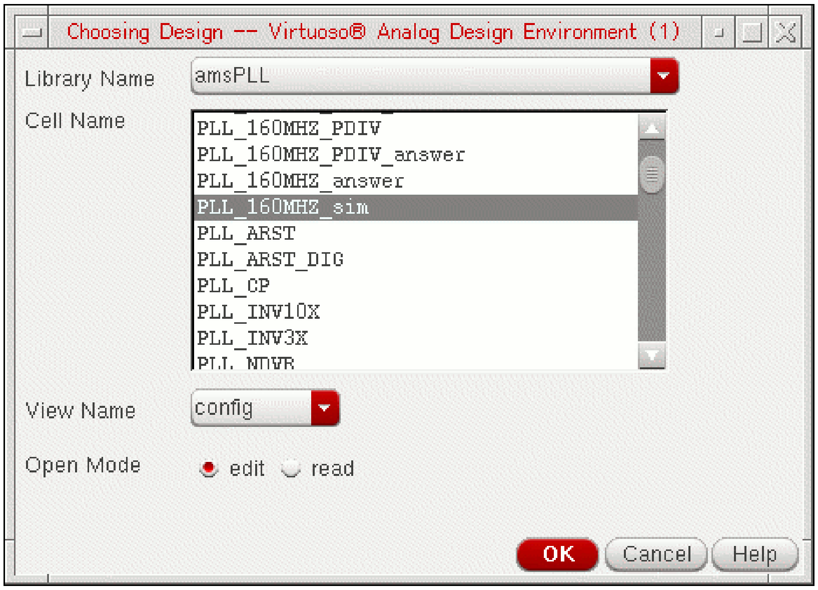
Status: Ready | T=27 C | Simulator: ams(Spectre)Mode: batch

In order to select the AMS simulator in ADE, you must choose the `config` view first.

2. Choose *Setup – Design*.

   The Choosing Design form appears.

3. In the *View Name* field, select *config*.



Library Name: amsPLL

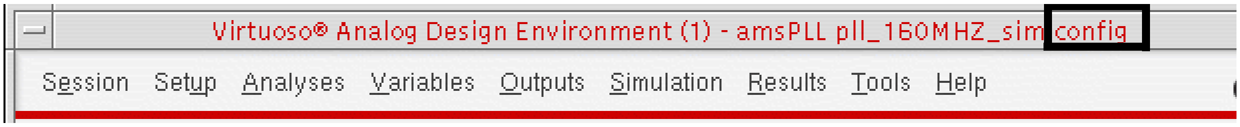
Cell Name:
PLL_160MHZ_PDIV
PLL_160MHZ_PDIV_answer
PLL_160MHZ_answer
PLL_160MHZ_sim
PLL_ARST
PLL_ARST_DIG
PLL_CP
PLL_INV10X
PLL_INV3X
PLL_NDVR

View Name: config

Open Mode: ● edit ○ read

4. Click *OK*.

config appears in the title banner of the ADE window.



Virtuoso® Analog Design Environment (1) - amsPLL pll_160MHZ_sim config
Session
Setup
Analyses
Variables
Outputs
Simulation
Results
Tools
Help

Now you can select the AMS simulator.

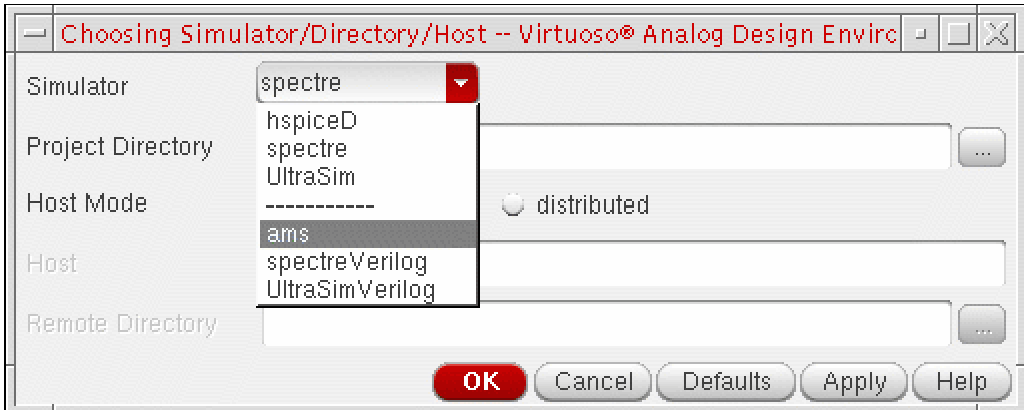
Selecting the AMS Designer Simulator with the Spectre Solver

To select the AMS Designer simulator with the Spectre solver, do the following:

1. Choose *Setup – Simulator/Directory/Host*.

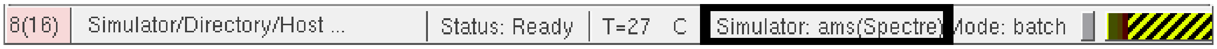
The Choosing Simulator form appears.

2. In the *Simulator* field, select *ams*.



3. Click *OK*.

   ams appears in the window header after *Simulator*. The currently configured analog solver appears next to *ams* in parentheses. *Spectre* is the default analog solver. We will use the Spectre solver for this part of the tutorial.



Note: You can change the solver on the Choose Solver form by choosing *Simulation – Solver*.

Specifying the Transient Analysis

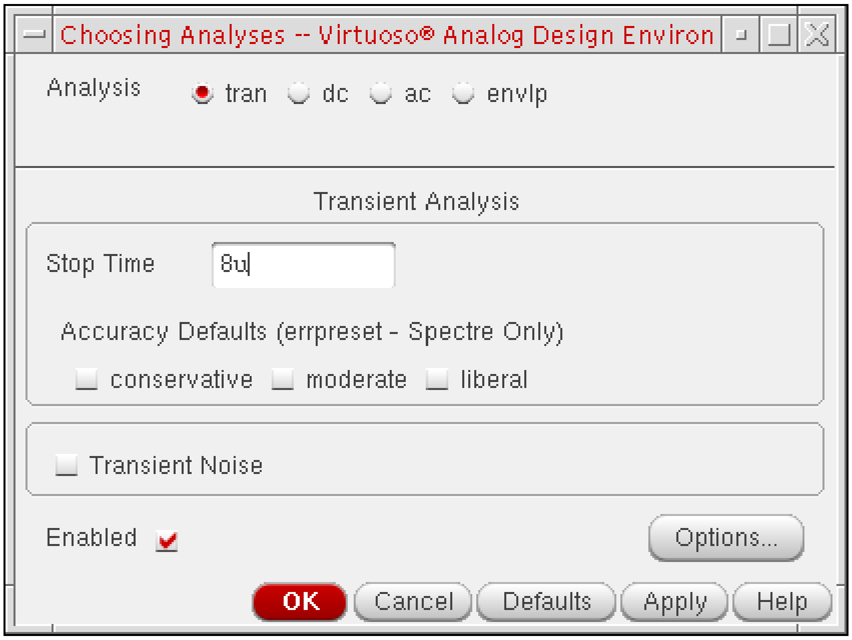
To specify the transient analysis for this tutorial, do the following:

1. Choose *Analyses – Choose*.

   The Choosing Analyses form appears. The default *Analysis* selection is *tran*.

2. In the *Stop Time* field, type *8u*.

A check mark appears in the *Enabled* check box.



Choosing Analyses -- Virtuoso® Analog Design Environ
Transient Analysis
Accuracy Defaults (errpreset - Spectre Only)
Options...
OK
Cancel
Defaults
Apply
Help

Analysis ● tran ○ dc ○ ac ○ envlp

Stop Time 8u

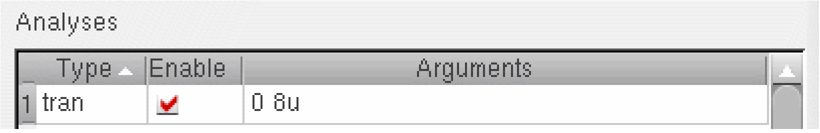
□ conservative □ moderate □ liberal

□ Transient Noise

Enabled ☑

3. Click *OK*.

   The transient analysis setup appears in the *Analyses* area in the ADE window.



Analyses

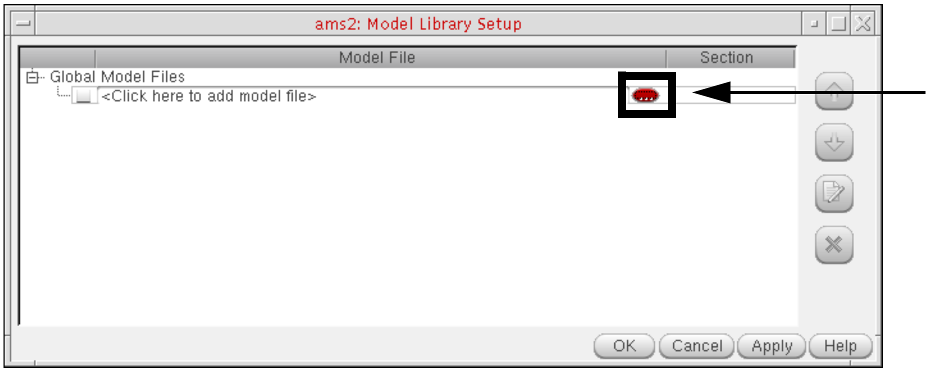
	Type	Enable	Arguments
1	tran	✓	0 8u

Specifying the Model Libraries

To specify the model libraries for this tutorial, do the following:

1. Choose *Setup – Model Libraries*.

The Model Library Setup form appears.

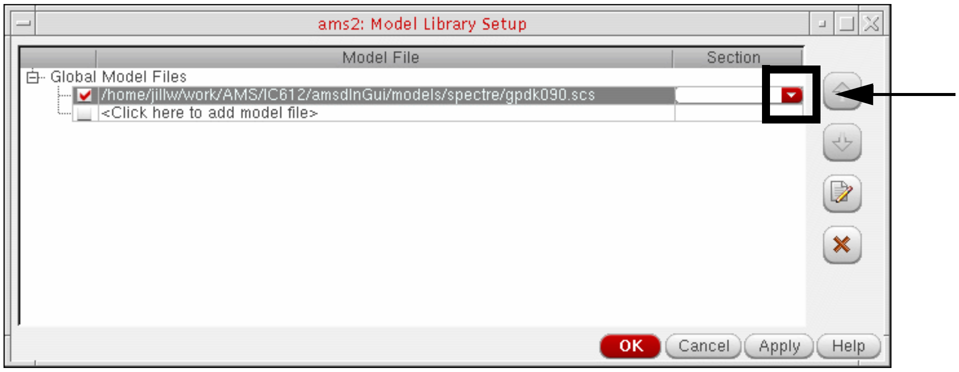


Browse button

2. Click the browse button.

3. On the Choose form that appears, navigate to and double-click `models/spectre/gpdk090.scs`.

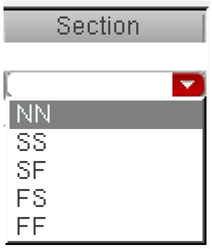
4. To select a particular section, click once in the Section column so that the down arrow appears.



☑ /home/jillw/work/AMS/IC612/amsdlnGui/models/spectre/gpdk090.scs

Down arrow

5. On the drop-down list that appears in the *Section* column, select *NN*.



6. On the Model Library Setup form, click *OK*.

Customizing Connect Rules

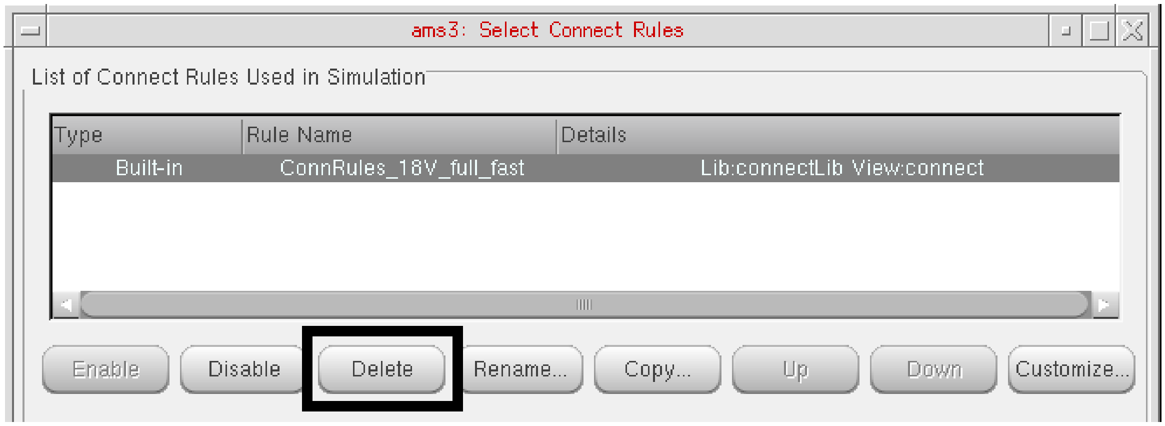
The power supply for the PLL design in this tutorial is 2.5 Volts. There are no built-in rules for this voltage, so you need to customize a set of connect rules for 2.5 Volts. We will select and customize the connect rules for a 3-Volt supply.

To customize connect rules for this tutorial, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Setup – Connect Rules*.

   The Select Connect Rules form appears. *ConnRules_5V_full* appears in the *List of Connect Rules Used in Simulation* table by default.

2. Select *ConnRules_5V_full* and click *Delete*.


ams3: Select Connect Rules
List of Connect Rules Used in Simulation

Type	Rule Name	Details
Built-in	ConnRules_18V_full_fast	Lib:connectLib View:connect

Enable
Disable
Delete
Rename...
Copy...
Up
Down
Customize...

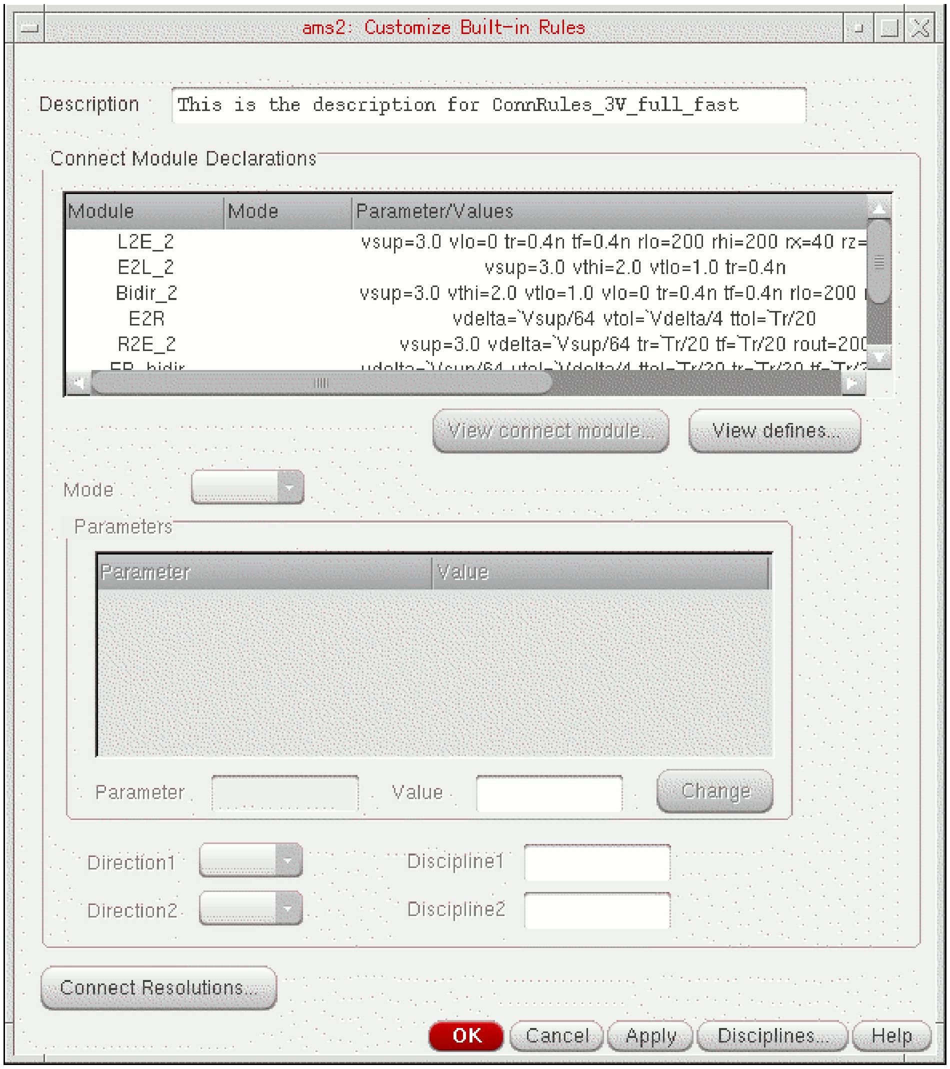
   This action removes the default connect rules so that you can specify your own set of customized connect rules for this tutorial.

3. Using the drop-down combo box in the *Rules Name* field of the *Built-in rules* group box, select *connectLib.ConnRules_18V_full_fast*.

   The `_full` rules are the most accurate. Cadence recommends the `_full_fast` rules for use with AMS Designer. You can customize the built-in rules you select for the supply value of your design. In this case, the PLL is a 2.5-Volt design.

4. Click *Customize*.

The Customize Built-in Rules form appears.

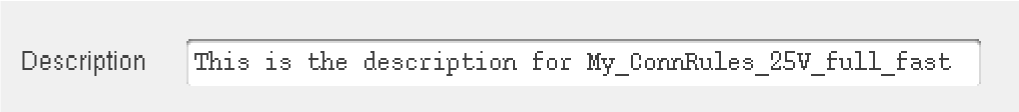


Description: `This is the description for ConnRules_3V_full_fast`

Module	Mode	Parameter/Values
L2E_2		vsup=3.0 vlo=0 tr=0.4n tf=0.4n rlo=200 rhi=200 rx=40 rz=
E2L_2		vsup=3.0 vthi=2.0 vtlo=1.0 tr=0.4n
Bidir_2		vsup=3.0 vthi=2.0 vtlo=1.0 vlo=0 tr=0.4n tf=0.4n rlo=200
E2R		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20
R2E_2		vsup=3.0 vdelta=`Vsup/64 tr=`Tr/20 tf=`Tr/20 rout=20

5. In the *Description* field, click to place the edit cursor and change the description to

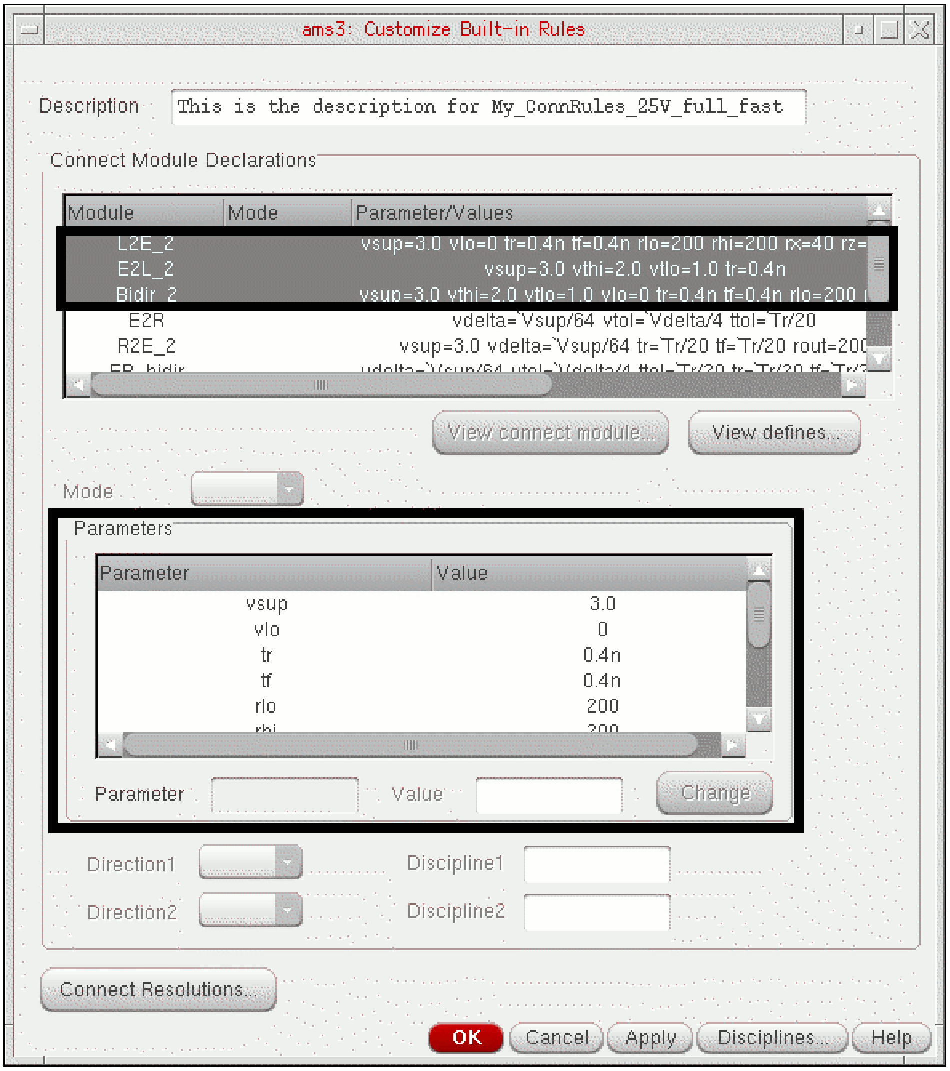
`This is the description for My_ConnRules_25V_full_fast`



Description: `This is the description for My_ConnRules_25V_full_fast`

6. In the *Connect Module Declarations* table, highlight the top three lines containing information for modules *E2L_2*, *L2E_2*, and *Bidir_2*.

   The parameters for these modules appear in the *Parameters* group box.



Description: This is the description for My_ConnRules_25V_full_fast

Connect Module Declarations

Module	Mode	Parameter/Values
L2E_2		vsup=3.0 vlo=0 tr=0.4n tf=0.4n rlo=200 rhi=200 rx=40 rz=
E2L_2		vsup=3.0 vthi=2.0 vtlo=1.0 tr=0.4n
Bidir_2		vsup=3.0 vthi=2.0 vtlo=1.0 vlo=0 tr=0.4n tf=0.4n rlo=200
E2R		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20
R2E_2		vsup=3.0 vdelta=`Vsup/64 tr=`Tr/20 tf=`Tr/20 rout=200
ER_bidir		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20 tr=`Tr/20 tf=`Tr/2

Parameters

Parameter	Value
vsup	3.0
vlo	0
tr	0.4n
tf	0.4n
rlo	200
rhi	200

Parameter: Value:

7. Change these values as follows:

Parameter	*Value*	Change it to
vsup	*1.8*	`2.5`
vthi	*1.2*	`1.7`
vtlo	*0.6*	`0.8`

a. Select the parameter you want to change.

Note: For *vthi* and *vtlo*, use the scroll bar to scroll down to the bottom of the list.

Its name appears in the *Parameter* field. Its value appears in the *Value* field.

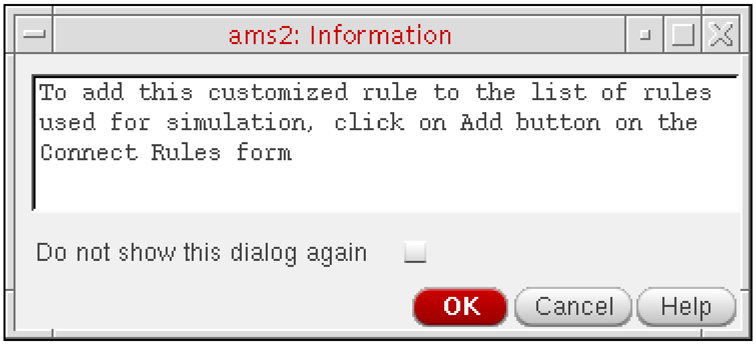
b. In the *Value* field, click and drag the mouse to highlight the value for editing.

c. Type the new value.

d. Click *Change*.

8. Click *OK*.

An Information prompt appears.

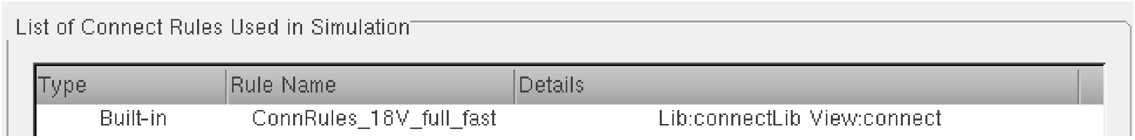


```
To add this customized rule to the list of rules
used for simulation, click on Add button on the
Connect Rules form
```

Do not show this dialog again □

9. Click *OK* on the Information prompt.

10. On the Select Connect Rules form, click *Add*.

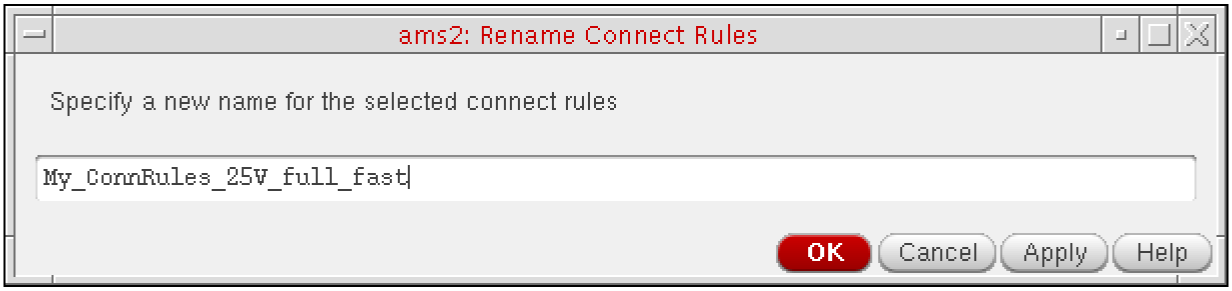

List of Connect Rules Used in Simulation

Type	Rule Name	Details
Built-in	ConnRules_18V_full_fast	Lib:connectLib View:connect

Modified built-in appears in the *Type* column.

11. Select the *Modified built-in* rule and click *Rename*.

12. On the Rename Connect Rules form, edit the name of the rule to be `My_ConnRules_25V_full_fast`.



ams2: Rename Connect Rules

Specify a new name for the selected connect rules

My_ConnRules_25V_full_fast

13. Click *OK*.

The modified name appears in the *Rules Name (Cell)* column in the *List of Connect Rules Used in Simulation*.

14. On the Select Connect Rules form, click *OK*.

The connect rules you specify on the Select Connect Rules form apply to the whole design.

Note: You might want to have several connect rules in the same design. You can set disciplines on a net, cell, instance, or library, and you can specify several connect rules accordingly.

Selecting Outputs for Plotting

To select the outputs you want to plot, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Outputs – To Be Plotted – Select on Schematic*.

   The schematic window appears in the foreground.

2. On the top-level schematic, select *CLK_REF* and *CLK_160MHZ*.
3. Choose *Edit – Hierarchy – Descend Edit* and select instance *I3*.
4. On the Descend form, click *OK*.

   The *PLL_160MHZ* schematic appears.

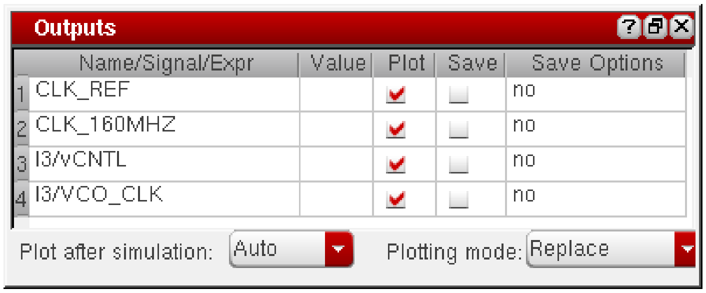
5. Select *vCNTL* and *VCO_CLK*.

Tip

You can zoom in to see the net labels by typing `]` (shortcut) two or three times.

6. Press *Esc*.

The signals you selected appear in the *Outputs* table in the ADE window.



Outputs

	Name/Signal/Expr	Value	Plot	Save	Save Options
1	CLK_REF		☑	□	no
2	CLK_160MHZ		☑	□	no
3	I3/vCNTL		☑	□	no
4	I3/VCO_CLK		☑	□	no

Plot after simulation: Auto

Plotting mode: Replace

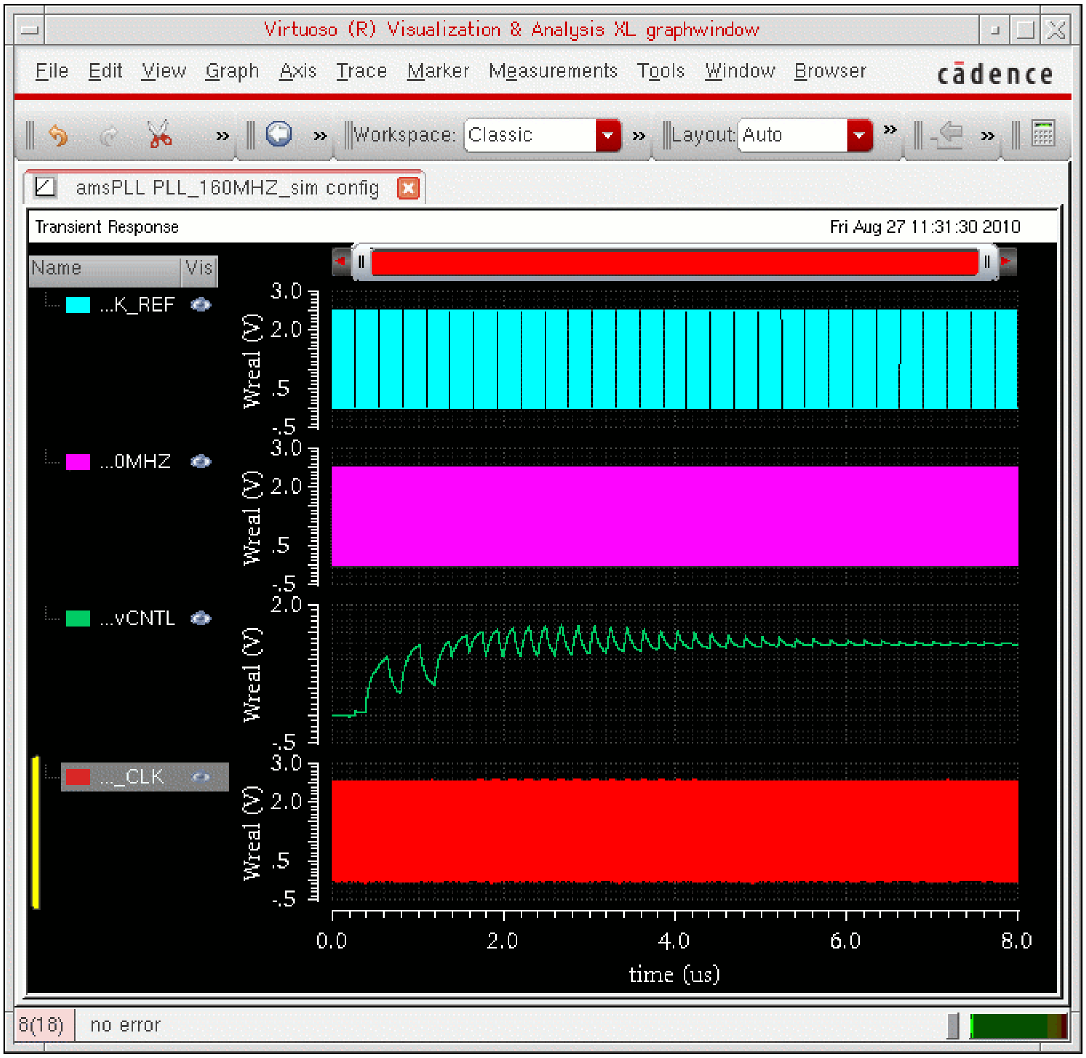
Specifying the OSS-Based Netlister and irun

To specify the OSS-based netlister and `irun` for simulation, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Simulation – Netlist and Run Options*.

   The Netlist and Run Options form appears.

2. In the *NETLIST AND RUN MODE* section, select *OSS-based netlister with irun*.
3. In the *RUN OPTIONS* section, turn on *Clean existing snapshot and pak files*.



Note: For information about the other run options available on this form, see "Specifying Run Options" in the *Virtuoso AMS Designer Environment User Guide*.

4. Click *OK*.

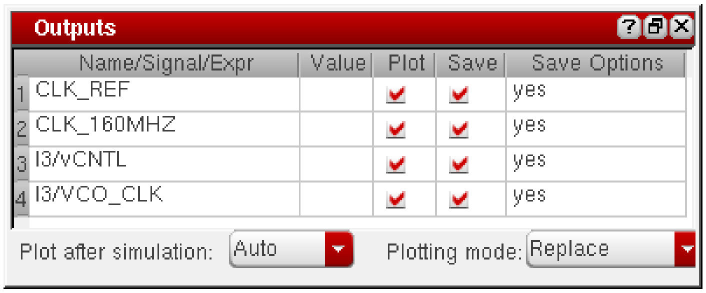
The default simulation mode is *Batch*, which means that the waveform viewer appears after the simulation finishes. If you select *Interactive* instead, the SimVision environment appears and you control the simulation interactively. You can use Tcl commands for debugging.

Creating and Displaying the Netlist

To create and display the netlist, do the following:

1. In the Analog Design Environment window, choose *Simulation – Netlist – Create*.

   Running Netlist appears in the lower right corner of the ADE window during netlisting. Success messages appear in the CIW. Check marks appear in the *Save* check boxes for all outputs in the *Outputs* table in the ADE window.

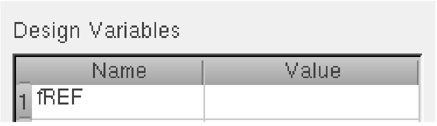
   

   Outputs

	Name/Signal/Expr	Value	Plot	Save	Save Options
1	CLK_REF		✓	✓	yes
2	CLK_160MHZ		✓	✓	yes
3	I3/vCNTL		✓	✓	yes
4	I3/VCO_CLK		✓	✓	yes

   Plot after simulation: Auto   Plotting mode: Replace

   fREF appears in the *Design Variables* table in the ADE window.

   Design Variables

	Name	Value
1	fREF	

   Note: You can also find the design variables in your design by choosing *Variables – Copy from Cellview*.

2. In the ADE window, choose *Simulation – Netlist – Display*.

   The netlist appears in a viewing window.

   The OSS netlister concatenates the various netlist and control files (such as `netlist.vams`, `cds_globals.vams`, `amsControl.scs`, and `amsControl.tcl`) for viewing purposes only.

   You can scroll through the file to see that the `fREF` variable does not show a value yet:

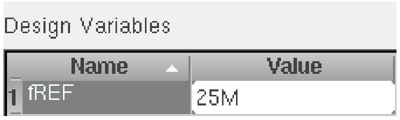
   ```
   module cds_globals;
   ...
   // Design Variables
      dynamicparam real fREF = "** unset **";
   ```

   Note: You can view, and even edit, the actual netlist file in

   `runDirectory``/simulation/PLL_160MHZ_sim/ams/config/netlist/netlist.vams`

If you edit this file, you must run your AMS Designer simulation using the `runSimulation` script so that the Virtuoso® environment does not overwrite the `netlist.vams` file, thus overwriting your changes.

3. In the netlist viewing window, choose *File – Close Window*.
4. In the *Design Variables* table in the ADE window, click in the *Value* field and type `25M`.


Design Variables

	Name	Value
1	fREF	25M

Now, if you re-netlist, you will see this value for `fREF`.

5. In the Analog Design Environment window, choose *Simulation – Netlist – Create*.
6. Choose *Simulation – Netlist – Display*.

   The netlist appears in a viewing window. The `fREF` variable shows the value you typed:

```
module cds_globals;
...
// Design Variables
   dynamicparam real fREF = 25M;
```

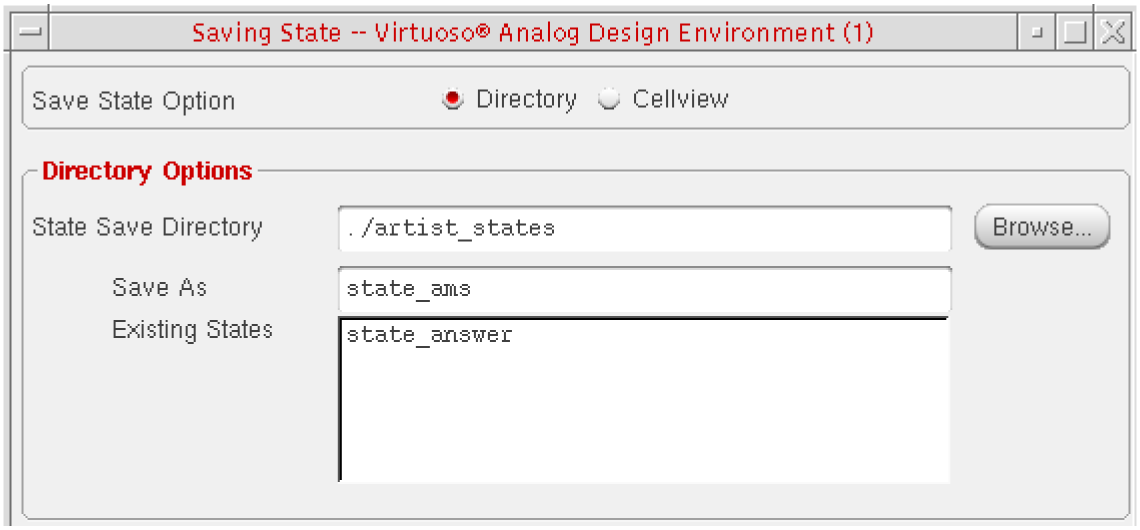
7. In the netlist viewing window, choose *File – Close Window*.

Saving the State

To save the state, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Session – Save State*.
2. In the *State Save Directory* field, type or browse to `./artist_states`.

3. In the *Save As* field, type `state_ams`.



Saving State -- Virtuoso® Analog Design Environment (1)

Save State Option ● Directory ○ Cellview

Directory Options

State Save Directory ./artist_states

Save As state_ams

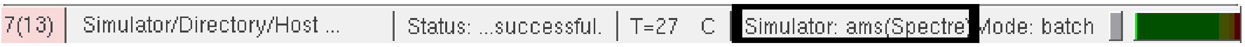
Existing States state_answer

Note: You can ignore the *state_answer* state which we provided with this tutorial for your reference and convenience only.

4. Click *OK*.

Running the AMS Simulator with the Spectre Solver

The currently configured simulator (*ams*) and solver (*Spectre*) appears in the status bar of the Virtuoso® Analog Design Environment window (see "Selecting the AMS Designer Simulator with the Spectre Solver" on page 41). The simulation mode is *batch*.

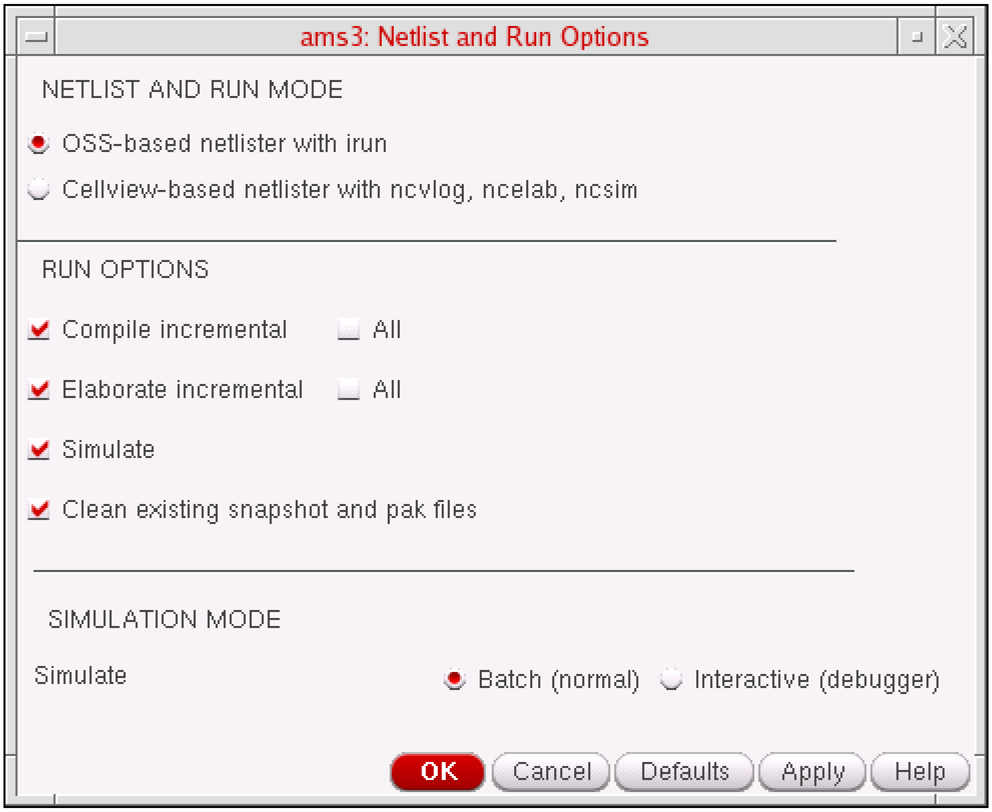


7(13) | Simulator/Directory/Host ... | Status: ...successful. | T=27 C | Simulator: ams(Spectre) Mode: batch

To run the AMS Designer simulator with the Spectre solver in batch mode, do the following:

1. (Optional) To view netlister and run mode options, choose *Simulation – Netlist and Run Options*.

The Netlist and Run Options form appears.



ams3: Netlist and Run Options

NETLIST AND RUN MODE

● OSS-based netlister with irun
○ Cellview-based netlister with ncvlog, ncelab, ncsim

RUN OPTIONS

☑ Compile incremental □ All
☑ Elaborate incremental □ All
☑ Simulate
☑ Clean existing snapshot and pak files

SIMULATION MODE

Simulate ● Batch (normal) ○ Interactive (debugger)

a. Notice that the simulation mode is *Batch*.

b. Click *OK* or *Cancel* when finished.

Important

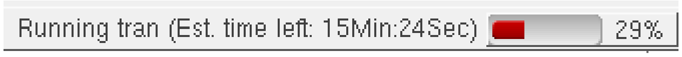
Because this next step can take a very long time (about two hours), you might consider either shortening the run time or skipping the simulation altogether.

2. Click the run button.

Job status appears on the status bar in the ADE window.

Status: Running...

Estimated run time appears in the lower right corner.



Running tran (Est. time left: 15Min:24Sec) 29%

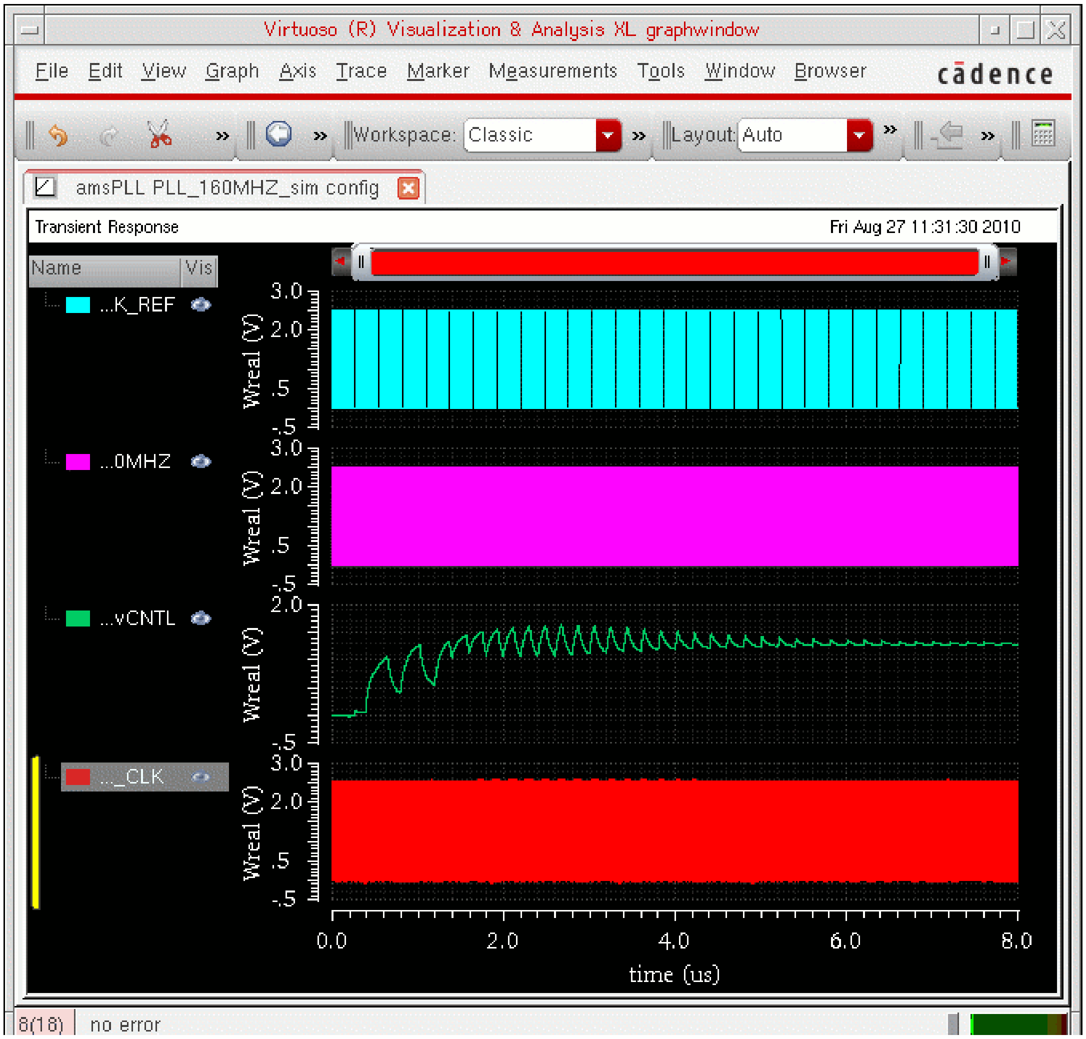
The simulation takes about two hours using the Spectre solver.

Tip

You can experiment by using Spectre Turbo instead. See the *Virtuoso Spectre Circuit Simulator User Guide* for information about Spectre Turbo. You can turn on and set up Spectre Turbo by choosing *Setup – Turbo* in ADE.

When the simulation finishes, the selected outputs appear in a waveform window.

3. In the waveform window, choose *Axis – Strips*.



Virtuoso (R) Visualization & Analysis XL graphwindow
File Edit View Graph Axis Trace Marker Measurements Tools Window Browser
cādence
Name
Vis
...K_REF
...0MHZ
...vCNTL
..._CLK
Wreal (V)
3.0
2.0
.5
-.5
0.0
2.0
4.0
6.0
8.0
time (us)

Workspace: Classic

Layout: Auto

amsPLL PLL_160MHZ_sim config

Transient Response

Fri Aug 27 11:31:30 2010

8(18) no error

You can see that the VCO control voltage signal reaches the stable value at about 7 us and the whole PLL system obtains the lock state.

4. When you are finished viewing waveforms, choose *File – Close*.

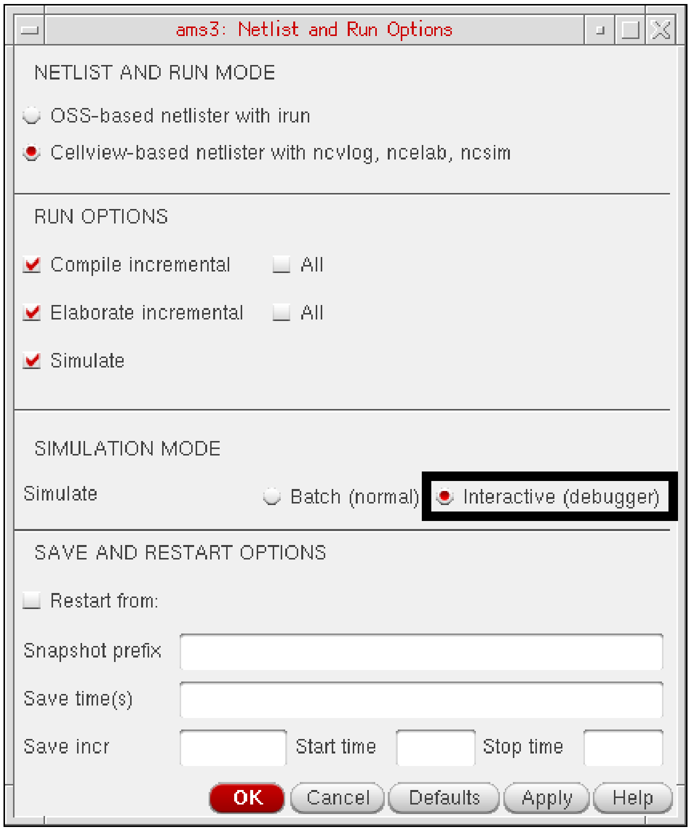
Running AMS-Spectre in Interactive Mode

To run the AMS Designer simulator with the Spectre solver in interactive mode, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Simulation – Netlist and Run Options*.

   The Netlist and Run Options form appears.

2. In the *SIMULATION MODE* section, select *Interactive*.



ams3: Netlist and Run Options

NETLIST AND RUN MODE

○ OSS-based netlister with irun

● Cellview-based netlister with ncvlog, ncelab, ncsim

RUN OPTIONS

☑ Compile incremental □ All

☑ Elaborate incremental □ All

☑ Simulate

SIMULATION MODE

Simulate ○ Batch (normal) ● Interactive (debugger)

SAVE AND RESTART OPTIONS

□ Restart from:

Snapshot prefix

Save time(s)

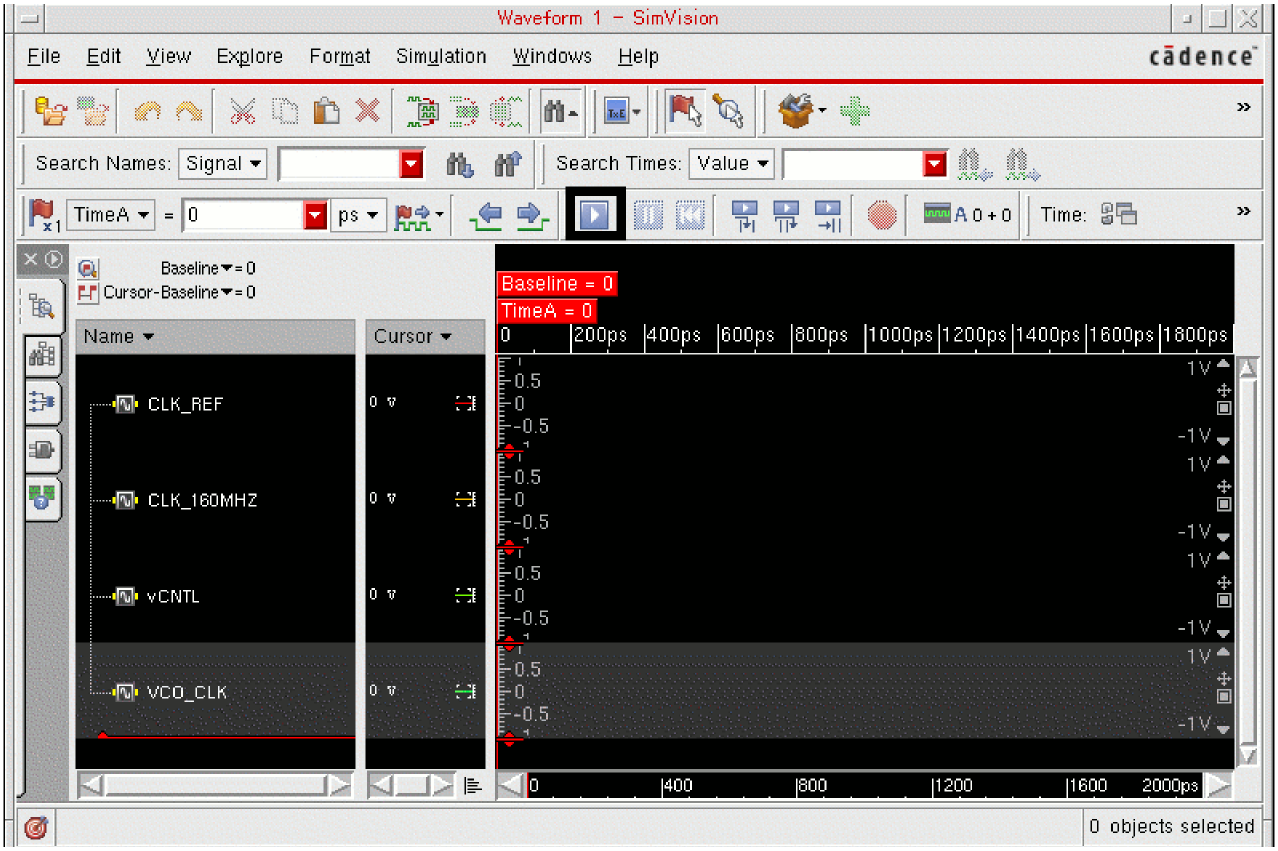
Save incr Start time Stop time

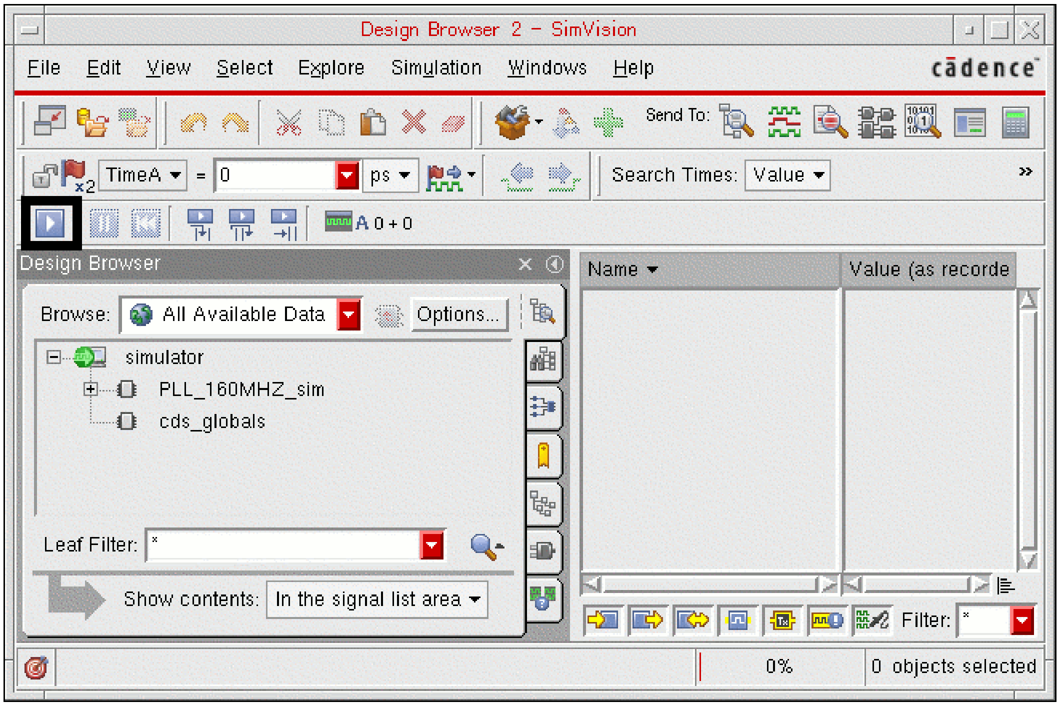
3. Click *OK*.

4. Click the run button.

   ▶

The SimVision Console, Waveform, and Design Browser windows appear.



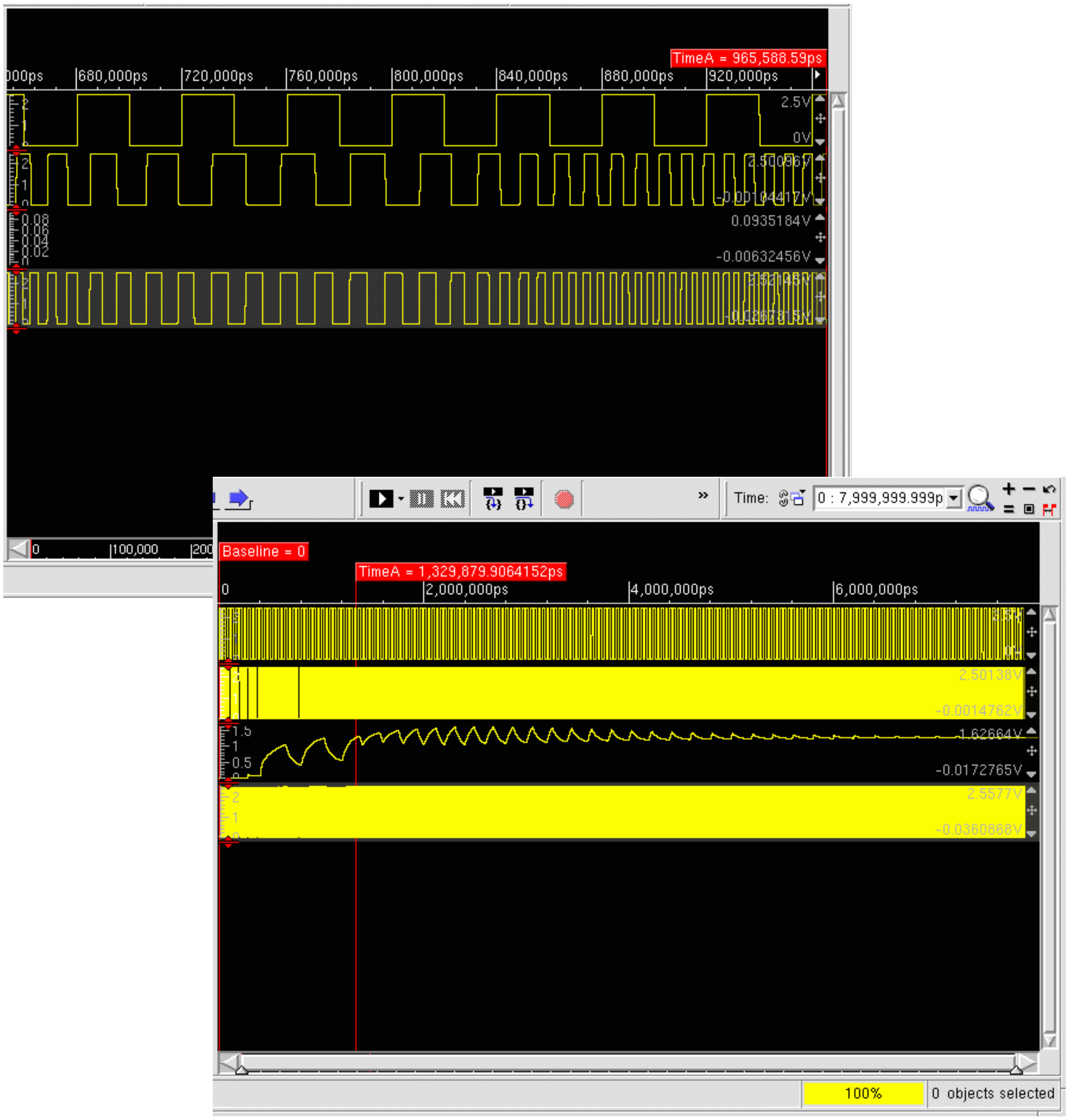

Design Browser 2 – SimVision
File Edit View Select Explore Simulation Windows Help
cādence
Send To:
TimeA = 0 ps
Search Times: Value
A 0 + 0
Design Browser
Browse: All Available Data
Options...
simulator
PLL_160MHZ_sim
cds_globals
Leaf Filter: *
Show contents: In the signal list area
Name
Value (as recorde
Filter: *
0%
0 objects selected

5. In any of the SimVision windows, press the play button to start the simulation.

   The waveforms update progressively until the simulation completes. You can observe them in detail by zooming in on the X and Y axes:

   a. In the SimVision Waveform window, choose *View – Zoom – Full X*.

   b. Choose *View – Zoom – Full Y*.



6. When the simulation finishes and you are finished viewing waveforms, choose *File – Exit SimVision* (in any SimVision window).

Running the AMS Simulator with the UltraSim Solver

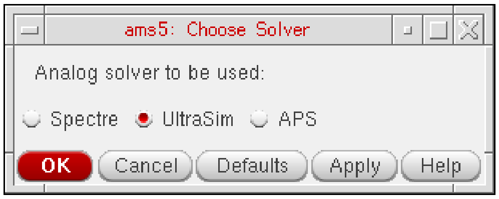
To run the AMS Designer simulator with the UltraSim solver in interactive mode, do the following:

Change solvers by doing the following:

1. In the Virtuoso® Analog Design Environment window, choose *Simulation – Solver*.

   The Choose Solver form appears.

2. Select *UltraSim*.



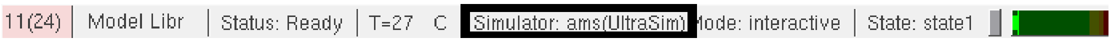
ams5: Choose Solver

Analog solver to be used:

○ Spectre ● UltraSim ○ APS

3. Click *OK*.

   ams(UltraSim) appears on the status bar of the ADE window to indicate that the next time you start a simulation, the AMS Designer simulator will use the UltraSim solver. The simulation mode is still *interactive* from the previous run.



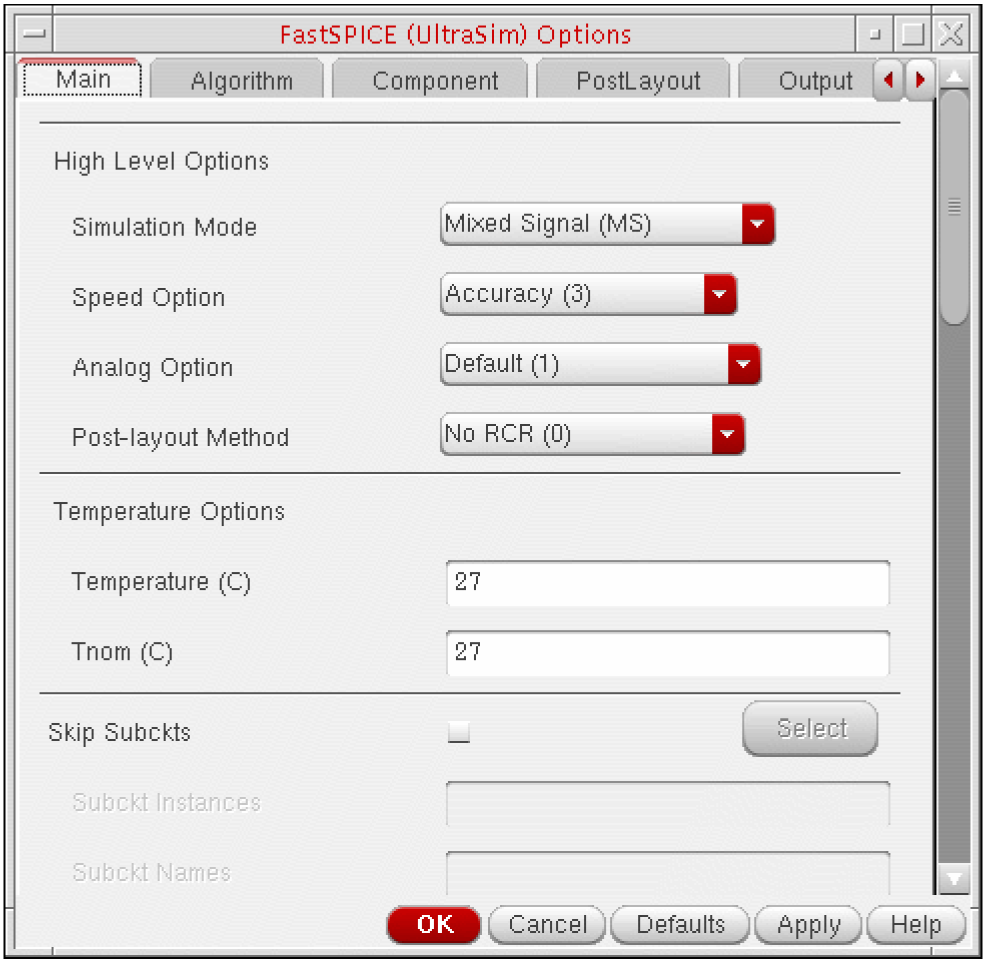
11(24) | Model Libr | Status: Ready | T=27 C | Simulator: ams(UltraSim) Mode: interactive | State: state1

Change the speed option for the UltraSim solver by doing the following:

1. In the ADE window, choose *Simulation – Options – FastSPICE(UltraSim)*.

   The FastSPICE (UltraSim) Options tabbed window appears.

2. In the *Speed Option* field (on the *Main* tab), select *Accuracy (3)*.



High Level Options

Simulation Mode: Mixed Signal (MS)

Speed Option: Accuracy (3)

Analog Option: Default (1)

Post-layout Method: No RCR (0)

Temperature Options

Temperature (C): 27

Tnom (C): 27

Skip Subckts □

Subckt Instances

Subckt Names

3. Click *OK*.

4. Click the run button.

   The SimVision Console, Waveform, and Design Browser windows appear.

5. In any of the SimVision windows, press the play button to start the simulation.

The waveforms update progressively until the simulation completes. You can observe them in detail by zooming in on the X and Y axes:

a. In the SimVision Waveform window, choose *View – Zoom – Full X*.

b. Choose *View – Zoom – Full Y*.

The simulation using the UltraSim solver takes about ten minutes. Compare this simulation time to the simulation time of about two hours using the Spectre solver. The AMS Designer simuator with the UltraSim solver achieves almost a 15x speed-up over AMS Designer with the Spectre solver. Generally, AMS Designer with the UltraSim solver performs better for complex analog/mixed-signal circuits.

6. When you are finished viewing waveforms, choose *File – Exit SimVision*.

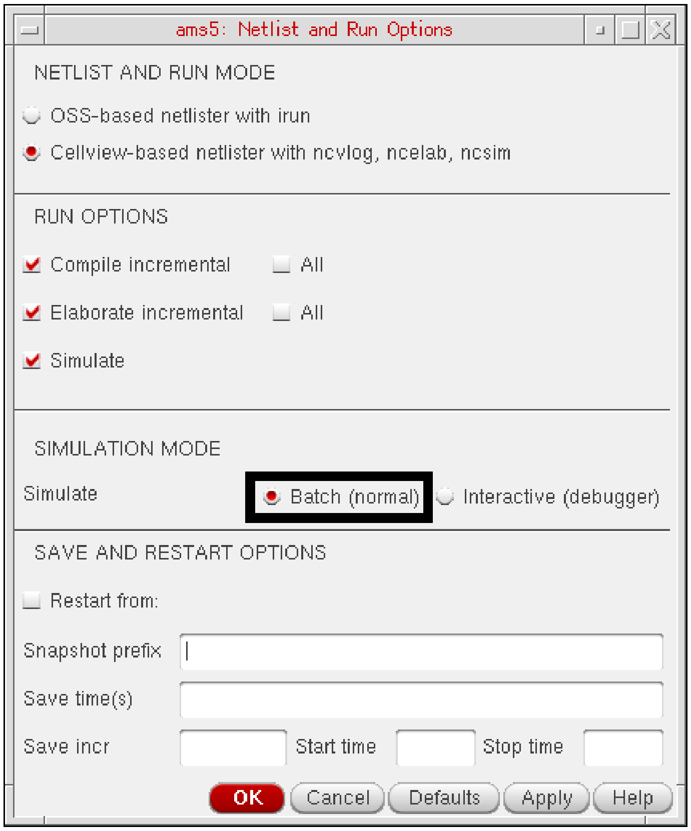
Running AMS-UltraSim in Batch Mode

To run the AMS Designer simulator with the UltraSim solver in batch mode, do the following:

1. In the Virtuoso$^{®}$ Analog Design Environment window, choose *Simulation – Netlist and Run Options*.

   The Netlist and Run Options form appears.

2. In the *SIMULATION MODE* section, select *Batch*.



ams5: Netlist and Run Options

NETLIST AND RUN MODE

○ OSS-based netlister with irun

● Cellview-based netlister with ncvlog, ncelab, ncsim

RUN OPTIONS

☑ Compile incremental ☐ All

☑ Elaborate incremental ☐ All

☑ Simulate

SIMULATION MODE

Simulate ● Batch (normal) ○ Interactive (debugger)

SAVE AND RESTART OPTIONS

☐ Restart from:

Snapshot prefix

Save time(s)

Save incr Start time Stop time

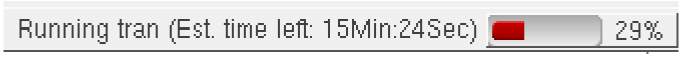
3. Click *OK*.

4. Click the run button.

   Job status appears on the status bar in the ADE window.

Status: Running...

Estimated run time appears in the lower right corner.

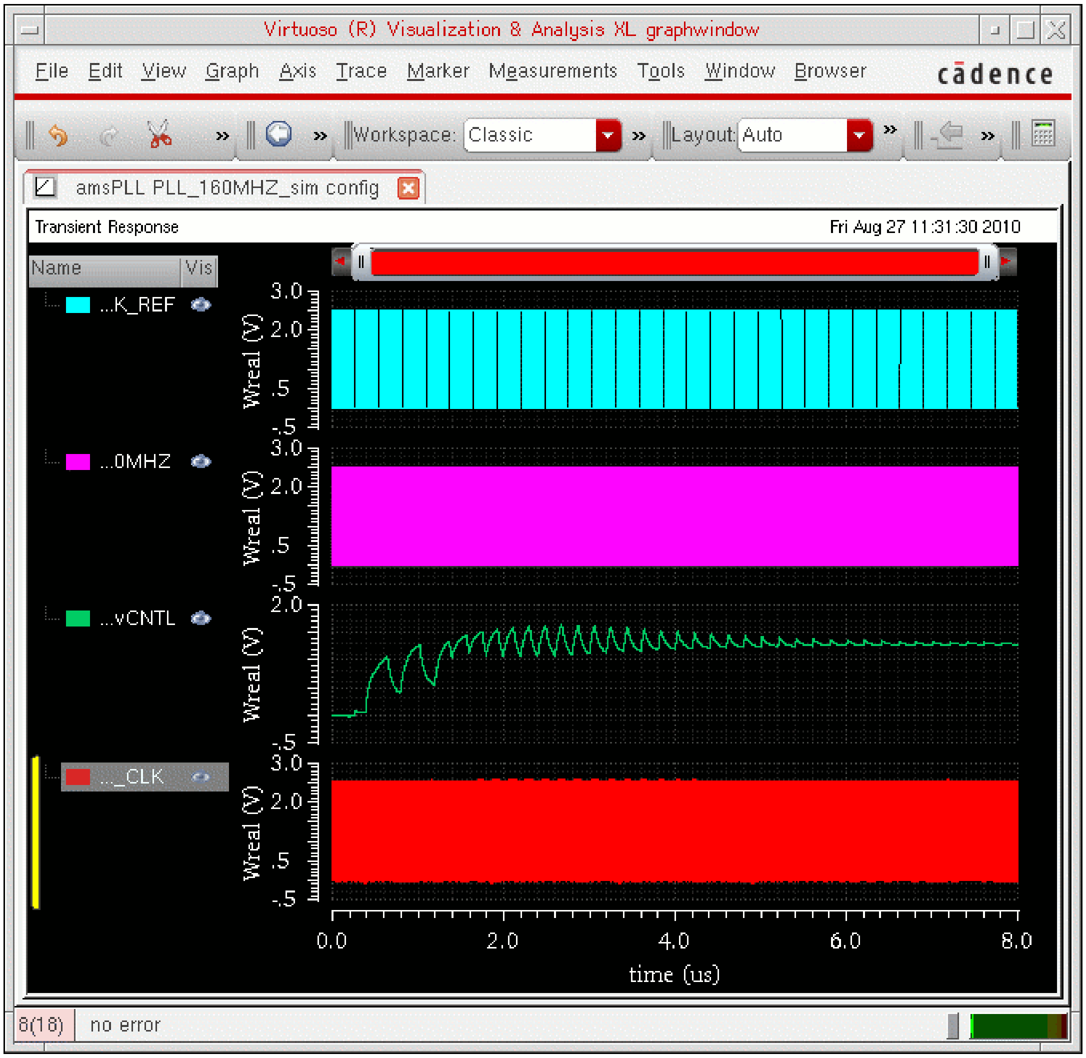


Running tran (Est. time left: 15Min:24Sec) 29%

The simulation takes about two hours using the Spectre solver.

When the simulation finishes, the selected outputs appear in a waveform window.

5. In the waveform window, choose *Axis – Strips*.



You can see that the VCO control voltage signal reaches the stable value at about 7 us and the whole PLL system obtains the lock state.

6. When you are finished viewing waveforms, choose *File – Close*.

4

Migrating to AMS Designer

Before the AMS Designer simulator, the major mixed-signal solutions on the market were SpectreVerilog and UltraSimVerilog. The AMS Designer simulator provides faster simulation speed, increased capacity, and enhanced features for handling more complicated mixed-signal designs with new technologies. More and more chip designers use AMS Designer for mixed-signal simulation and verification.

This migration example shows you how you would simulate a PLL design using UltraSimVerilog, SpectreVerilog, and, finally, AMS Designer using the same `config` view. You will learn about the OSS netlister, the `ncverilog` flow, using `-v` and `-y` command-line options, compiled Verilog-A, fastcross, and other AMS Designer features.

For details about this tutorial example, see “The Migration Example” on page 90.

Important

Before starting this tutorial, see “Before You Begin” on page 13.

To begin, do the following in the `MigrateFromVerimixToAMSDinADE` directory:

1. Source the setup file:

   ```
   source SETUP
   ```

   The `SETUP` file sets the `TUT_DIR` environment variable to your current directory:

   ```
   setenv TUT_DIR `pwd`
   ```

2. Start Cadence software:

   ```
   virtuoso &
   ```

See the following topics for further information:

- Simulating with UltraSimVerilog on page 70
- Simulating with SpectreVerilog on page 93
- Using the AMS Designer Simulator on page 99

Simulating with UltraSimVerilog

This part of the tutorial consists of the following actions related to simulating the migration example with UltraSimVerilog:

- Opening the Configuration View for the PLL Testbench on page 71
- Descending into the PLL Schematic on page 73
- Starting the Analog Design Environment (ADE) on page 74
- Changing the Simulator to UltraSimVerilog on page 76
- Loading the State File for UltraSimVerilog on page 78
- Verifying Model Libraries on page 80
- Viewing UltraSim Simulator Options on page 81
- Enabling Mixed-Signal Options on page 84
- Netlisting, Simulating, and Viewing Results on page 85
- Viewing Interface Elements on the Schematic on page 88

Opening the Configuration View for the PLL Testbench

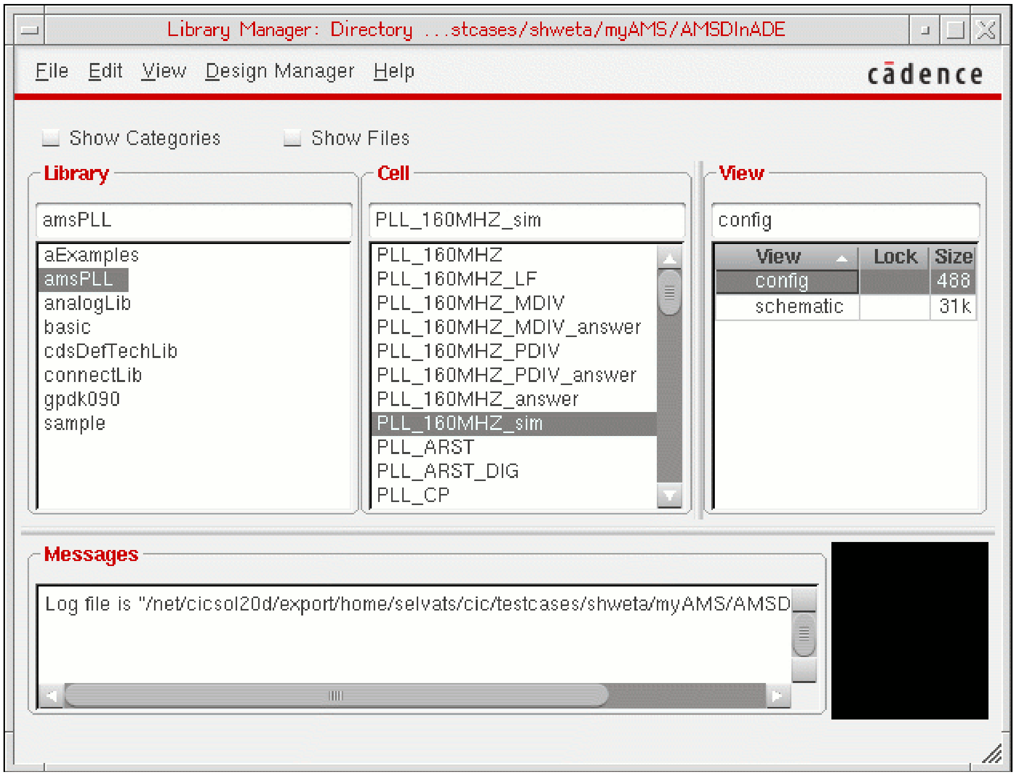
To open the config view for the PLL testbench, do the following:

1. In the command interpreter window (CIW), choose *Tools – Library Manager*.

   The Library Manager window appears.

2. Select the following:

Library	**Cell**	**View**
amsPLL	*pll_160MHz_sim*	*config*



Library Manager: Directory ...stcases/shweta/myAMS/AMSDInADE

□ Show Categories □ Show Files

Library: amsPLL

aExamples
amsPLL
analogLib
basic
cdsDefTechLib
connectLib
gpdk090
sample

Cell: PLL_160MHZ_sim

PLL_160MHZ
PLL_160MHZ_LF
PLL_160MHZ_MDIV
PLL_160MHZ_MDIV_answer
PLL_160MHZ_PDIV
PLL_160MHZ_PDIV_answer
PLL_160MHZ_answer
PLL_160MHZ_sim
PLL_ARST
PLL_ARST_DIG
PLL_CP

View: config

View	Lock	Size
config		488
schematic		31k

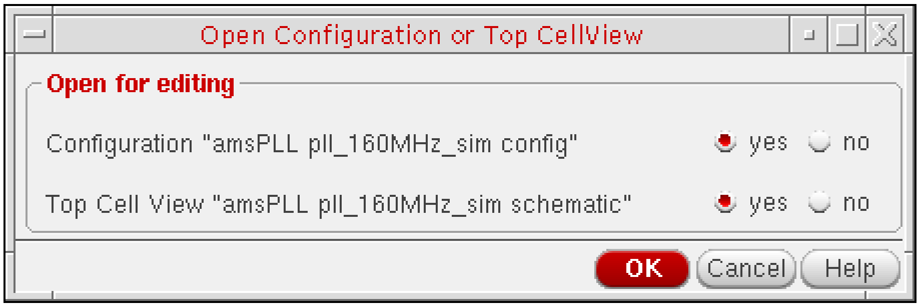
Messages

```
Log file is "/net/cicsol20d/export/home/selvats/cic/testcases/shweta/myAMS/AMSD
```

3. Choose *File – Open*.

   The Open Configuration or Top Cellview form appears.

4. In the *Open for editing* group box, select *yes* for *Configuration* (*yes* is already marked for *Top Cell View*).



Open Configuration or Top CellView

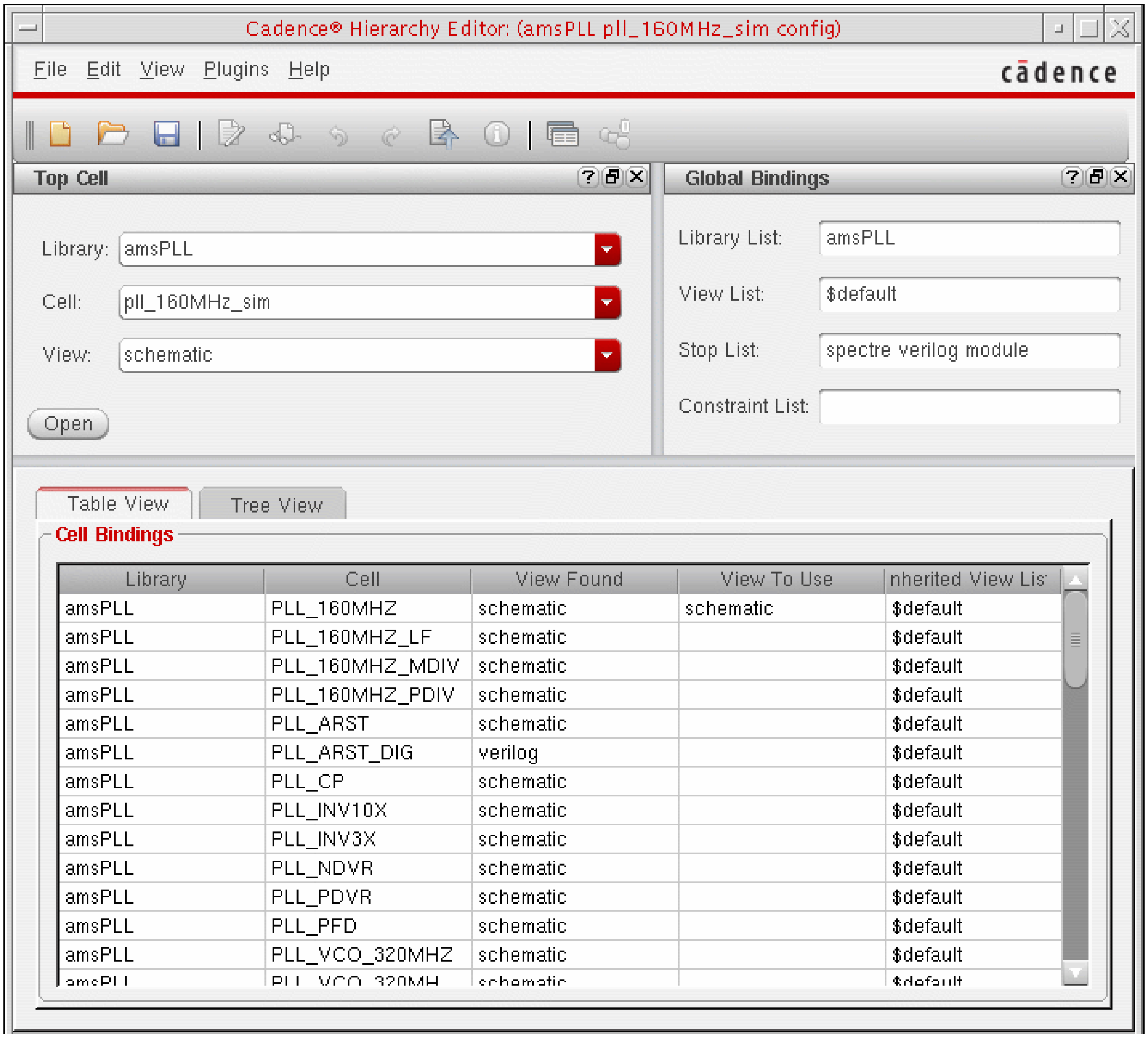
Open for editing

Configuration "amsPLL pll_160MHz_sim config" ● yes ○ no

Top Cell View "amsPLL pll_160MHz_sim schematic" ● yes ○ no

5. Click *OK*.

The testbench schematic appears in a Virtuoso® Schematic Editing window and the Cadence hierarchy editor window appears.


Cadence® Hierarchy Editor: (amsPLL pll_160MHz_sim config)
File Edit View Plugins Help
cādence
Top Cell
Global Bindings
Open
Table View
Tree View
Cell Bindings

Library: amsPLL

Cell: pll_160MHz_sim

View: schematic

Library List: amsPLL

View List: $default

Stop List: spectre verilog module

Constraint List:

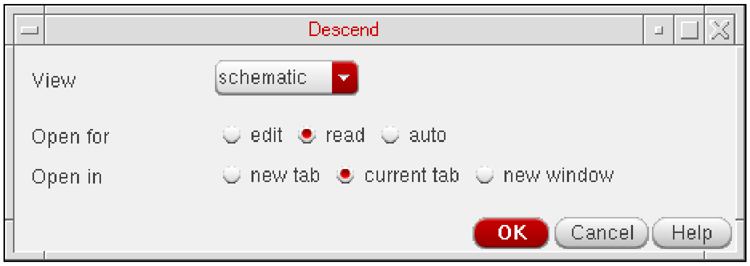
Library	Cell	View Found	View To Use	nherited View Lis
amsPLL	PLL_160MHZ	schematic	schematic	$default
amsPLL	PLL_160MHZ_LF	schematic		$default
amsPLL	PLL_160MHZ_MDIV	schematic		$default
amsPLL	PLL_160MHZ_PDIV	schematic		$default
amsPLL	PLL_ARST	schematic		$default
amsPLL	PLL_ARST_DIG	verilog		$default
amsPLL	PLL_CP	schematic		$default
amsPLL	PLL_INV10X	schematic		$default
amsPLL	PLL_INV3X	schematic		$default
amsPLL	PLL_NDVR	schematic		$default
amsPLL	PLL_PDVR	schematic		$default
amsPLL	PLL_PFD	schematic		$default
amsPLL	PLL_VCO_320MHZ	schematic		$default
amsPLL	PLL_VCO_320MH	schematic		$default

Descending into the PLL Schematic

To descend into the PLL schematic, do the following:

1. In the Schematic Editing window, select *I3* and press *e*.

The Descend form appears. *schematic* appears in the *View* field.



Descend

View: schematic

Open for: ○ edit ● read ○ auto

Open in: ○ new tab ● current tab ○ new window

2. Click *OK*.

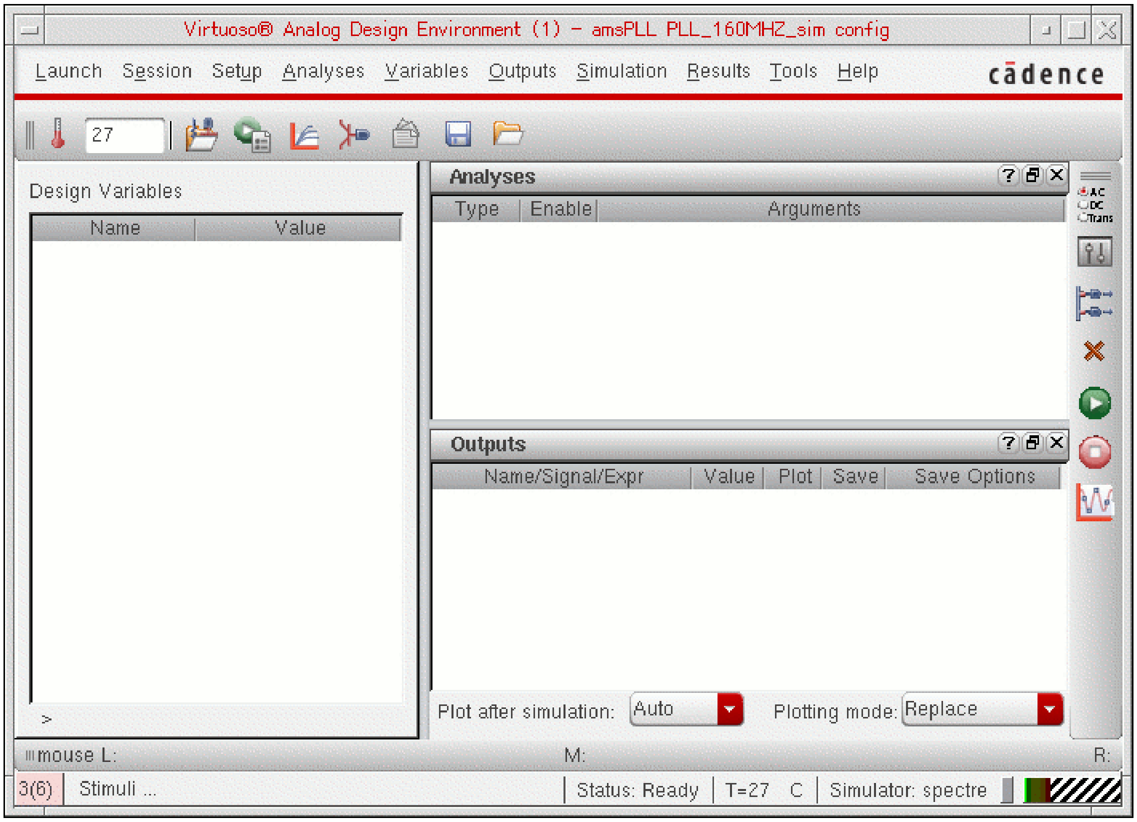
The contents of the *I3* instance appear in the Virtuoso® Schematic Reading window.

Starting the Analog Design Environment (ADE)

To start the Virtuoso® Analog Design Environment, do the following:

➤ In the Schematic Reading window, choose *Launch – ADE L*.

The Virtuoso® Analog Design Environment session window appears. *Simulator: spectre* appears on the status bar (just under the menu bar).



Plot after simulation: Auto

Plotting mode: Replace

Status: Ready | T=27 C | Simulator: spectre

The top-level schematic reappears in the schematic editing window.

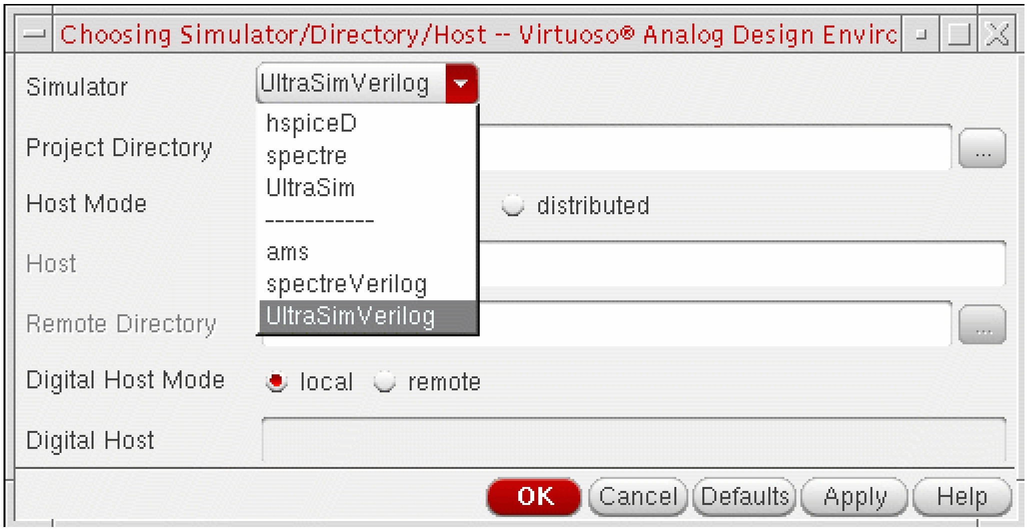
Changing the Simulator to UltraSimVerilog

To change the simulator to UltraSimVerilog, do the following:

1. In the Virtuoso® Analog Design Environment session window, choose *Setup – Simulator/Directory/Host*.

   The Choosing Simulator/Directory/Host form appears.

2. In the *Simulator* drop-down combo box, select *UltraSimVerilog*.



Choosing Simulator/Directory/Host -- Virtuoso® Analog Design Envirc

Simulator: UltraSimVerilog

Project Directory

Host Mode: ○ distributed

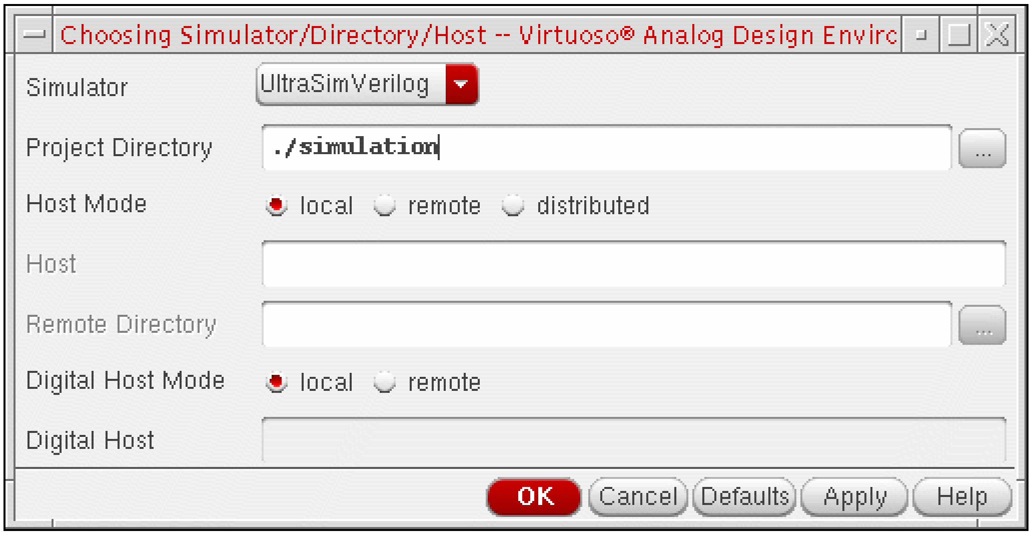
Host

Remote Directory

Digital Host Mode: ● local ○ remote

Digital Host

The *Digital Host Mode* and *Digital Host* selections appear at the bottom of the form.



Choosing Simulator/Directory/Host -- Virtuoso® Analog Design Envirc

Simulator: UltraSimVerilog

Project Directory: ./simulation

Host Mode: ● local ○ remote ○ distributed

Host

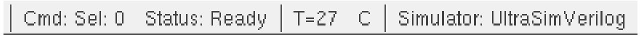
Remote Directory

Digital Host Mode: ● local ○ remote

Digital Host

3. Click *OK*.

Simulator: UltraSimVerilog appears on the status bar in the Virtuoso® Analog Design Environment session window.



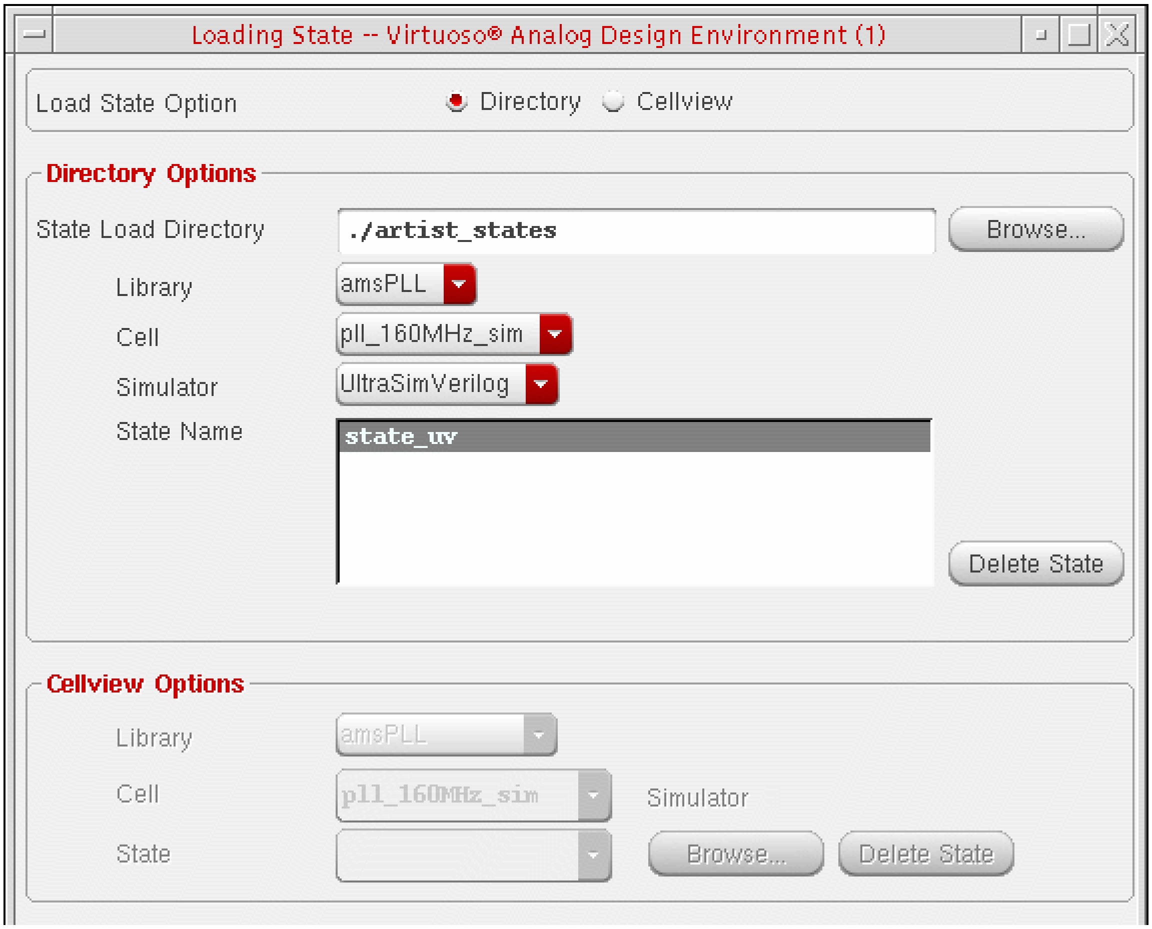
Cmd: Sel: 0 Status: Ready | T=27 C | Simulator: UltraSimVerilog

Loading the State File for UltraSimVerilog

To load the state file for UltraSimVerilog, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Session – Load State*.

   The Loading State form appears.



Loading State -- Virtuoso® Analog Design Environment (1)

Load State Option ● Directory ○ Cellview

Directory Options

State Load Directory ./artist_states

Library amsPLL

Cell pll_160MHz_sim

Simulator UltraSimVerilog

State Name state_uv

Cellview Options

Library amsPLL

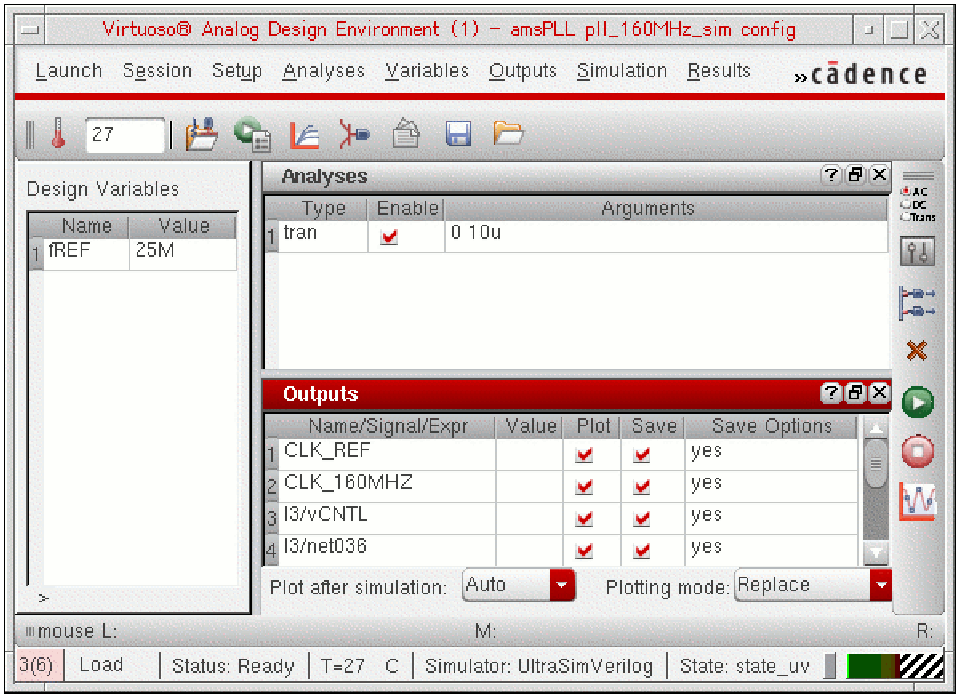
Cell pll_160MHz_sim

Simulator

State

2. In the *State Name* box, select *state_uv*.
3. Click *OK*.

The state settings appear in the Virtuoso® Analog Design Environment session window, such as *tran … 10u* in the *Analyses* area and nodes to plot in the *Outputs* area.



Design Variables

	Name	Value
1	fREF	25M

Analyses

	Type	Enable	Arguments
1	tran	✓	0 10u

Outputs

	Name/Signal/Expr	Value	Plot	Save	Save Options
1	CLK_REF		✓	✓	yes
2	CLK_160MHZ		✓	✓	yes
3	I3/vCNTL		✓	✓	yes
4	I3/net036		✓	✓	yes

Plot after simulation: Auto   Plotting mode: Replace

mouse L: M: R:

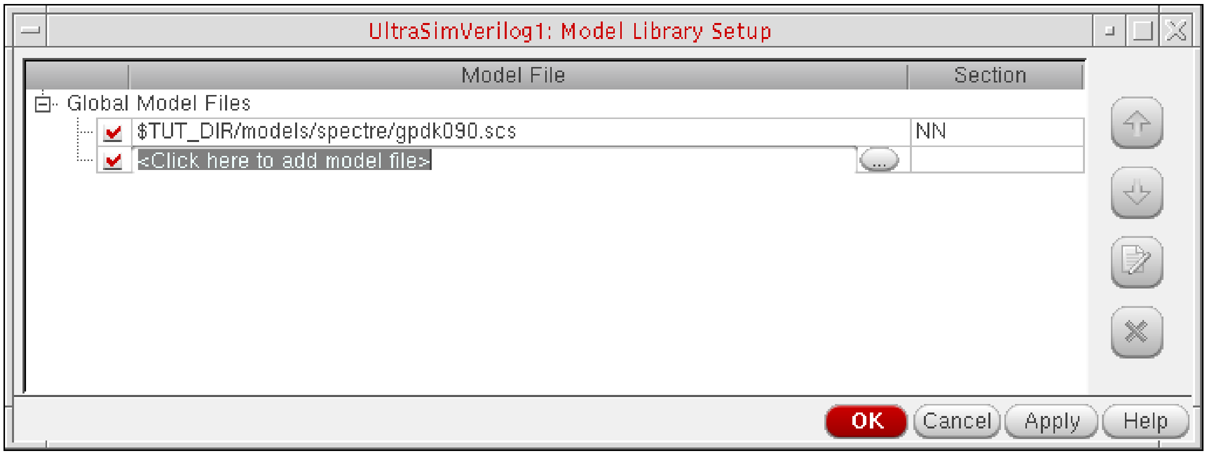
3(6) | Load | Status: Ready | T=27 C | Simulator: UltraSimVerilog | State: state_uv

Verifying Model Libraries

To verify model libraries, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Setup – Model Libraries*.

   The `$TUT_DIR/models/spectre/gpdk090.scs` model library appears on the Model Library Setup form. *NN* appears in the *Section* column.



UltraSimVerilog1: Model Library Setup

Model File	Section
Global Model Files	
☑ $TUT_DIR/models/spectre/gpdk090.scs	NN
☑ <Click here to add model file>	

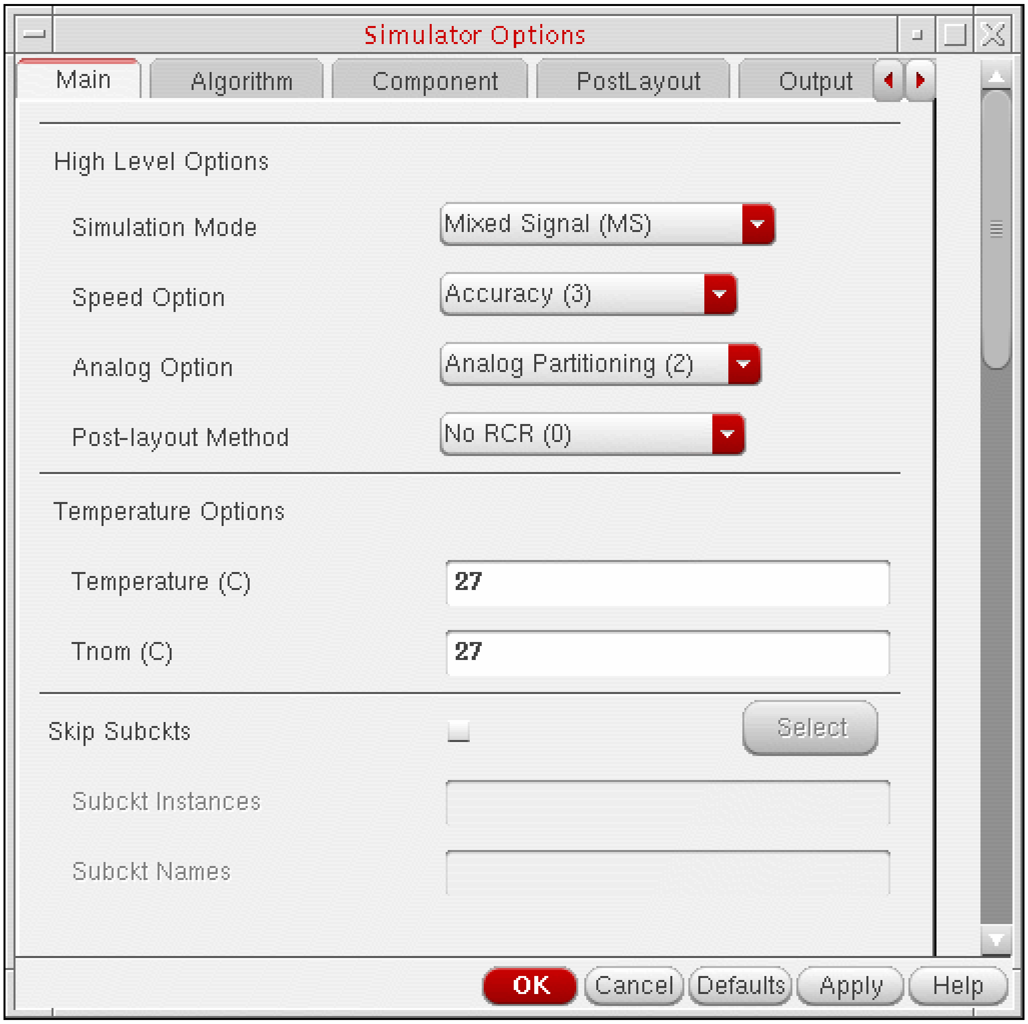
2. When you are finished viewing the model library setup, click *Cancel* to close the form.

Viewing UltraSim Simulator Options

To view UltraSim simulator options, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Simulation – Options – Analog*.

   The Simulator Options form appears. For information about the UltraSim simulation options that appear on this form, see the *Virtuoso UltraSim Simulator User Guide*.



Simulator Options

High Level Options

Simulation Mode: Mixed Signal (MS)

Speed Option: Accuracy (3)

Analog Option: Analog Partitioning (2)

Post-layout Method: No RCR (0)

Temperature Options

Temperature (C): 27

Tnom (C): 27

Skip Subckts □

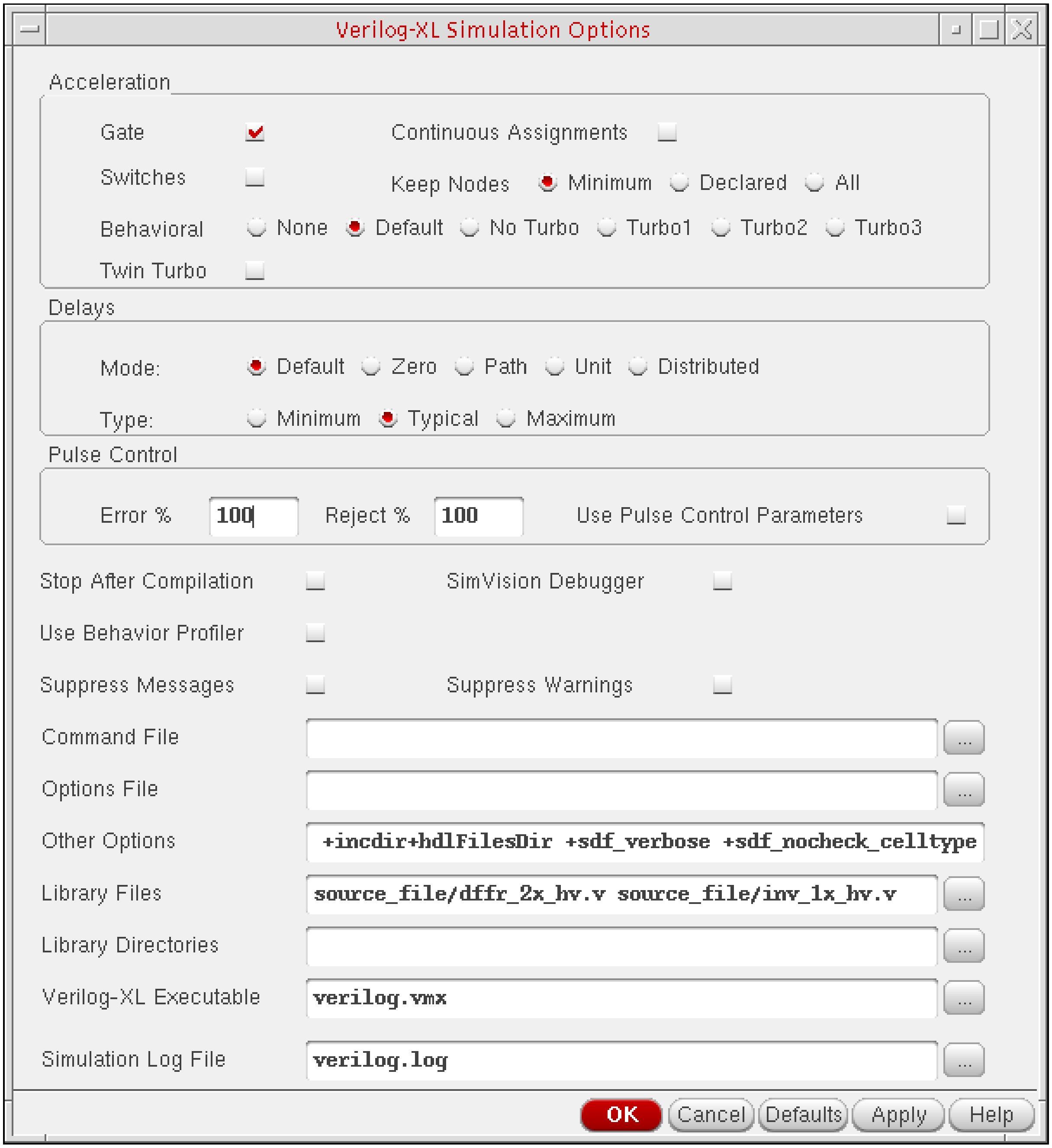
Subckt Instances

Subckt Names

2. When you are finished viewing the analog simulation options, click *Cancel* to close the form.

3. In the Virtuoso® Analog Design Environment window, choose *Simulation – Options – Digital*.

   The Verilog-XL Simulation Options form appears.



Verilog-XL Simulation Options

Acceleration

Gate ☑ Continuous Assignments □

Switches □ Keep Nodes ● Minimum ○ Declared ○ All

Behavioral ○ None ● Default ○ No Turbo ○ Turbo1 ○ Turbo2 ○ Turbo3

Twin Turbo □

Delays

Mode: ● Default ○ Zero ○ Path ○ Unit ○ Distributed

Type: ○ Minimum ● Typical ○ Maximum

Pulse Control

Error % 100 Reject % 100 Use Pulse Control Parameters □

Stop After Compilation □ SimVision Debugger □

Use Behavior Profiler □

Suppress Messages □ Suppress Warnings □

Command File	
Options File	
Other Options	`+incdir+hdlFilesDir +sdf_verbose +sdf_nocheck_celltype`
Library Files	`source_file/dffr_2x_hv.v source_file/inv_1x_hv.v`
Library Directories	
Verilog-XL Executable	`verilog.vmx`
Simulation Log File	`verilog.log`

Note: The string `source_file/dffr_2x_hv.v soure_file/inv_1x_hv.v` appears in the *Library Files* field. If you open the file `amsPLL/dffnr_2x_hv/`

`module/verilog.v`, you will see calls to the modules contained in these files:

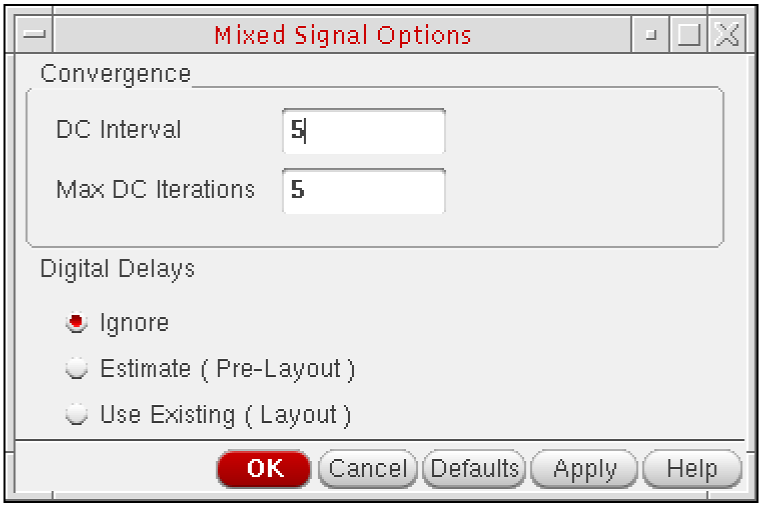
```
//Verilog HDL for "amsPLL", "dffnr_2x_hv" "module"
`timescale 1ps/1ps

module dffnr_2x_hv ( Q, QN, D, CKN, RN );
  output QN;
  input RN;
  input D;
  input CKN;
  output Q;

  dffr_2x_hv I1 ( Q, QN, D, net18, RN);
  inv_1x_hv I2 ( net18, CKN);
endmodule
```

You must use the `-v` option to include these module files in the design. The same is true for the AMS `ncverilog` flow.

4. When you are finished viewing Verilog-XL simulation options, click *Cancel* to close the form.

5. In the Virtuoso® Analog Design Environment window, choose *Simulation – Options – Mixed Signal*.

   The Mixed Signal Options form appears.



Mixed Signal Options

Convergence

DC Interval: 5

Max DC Iterations: 5

Digital Delays

● Ignore
○ Estimate (Pre-Layout)
○ Use Existing (Layout)

The *DC Interval* and *Max DC Iterations* settings are both *5*.

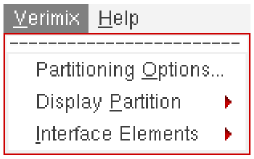
6. When you are finished viewing mixed-signal options, click *Cancel* to close the form.

Enabling Mixed-Signal Options

To enable mixed-signal options, do the following:

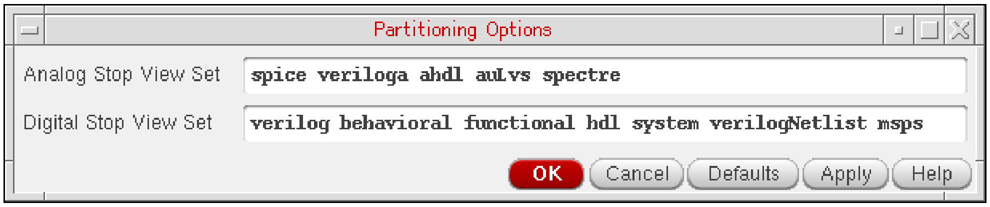
1. In the schematic window, choose *Launch – Mixed Signal Options – Verimix.*

   The *Verimix* menu appears on the menu banner.



2. Choose *Verimix – Partitioning Options*.

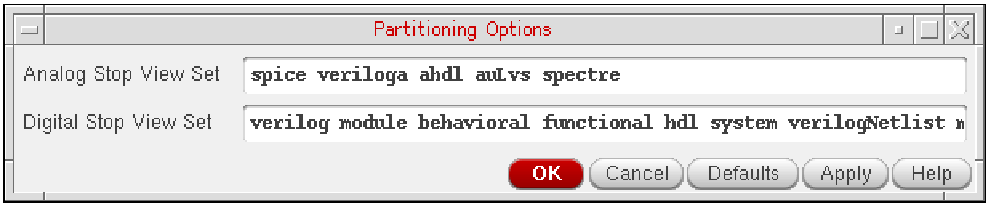
   The Partitioning Options form appears.



Analog Stop View Set: spice veriloga ahdl auLvs spectre

Digital Stop View Set: verilog behavioral functional hdl system verilogNetlist msps

3. In the *Digital Stop View Set* field, type `module` between *verilog* and *behavioral*.



Analog Stop View Set: spice veriloga ahdl auLvs spectre

Digital Stop View Set: verilog module behavioral functional hdl system verilogNetlist m

4. Click *OK*.

Netlisting, Simulating, and Viewing Results

To netlist and simulate using UltraSimVerilog, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Simulation – Netlist and Run*.

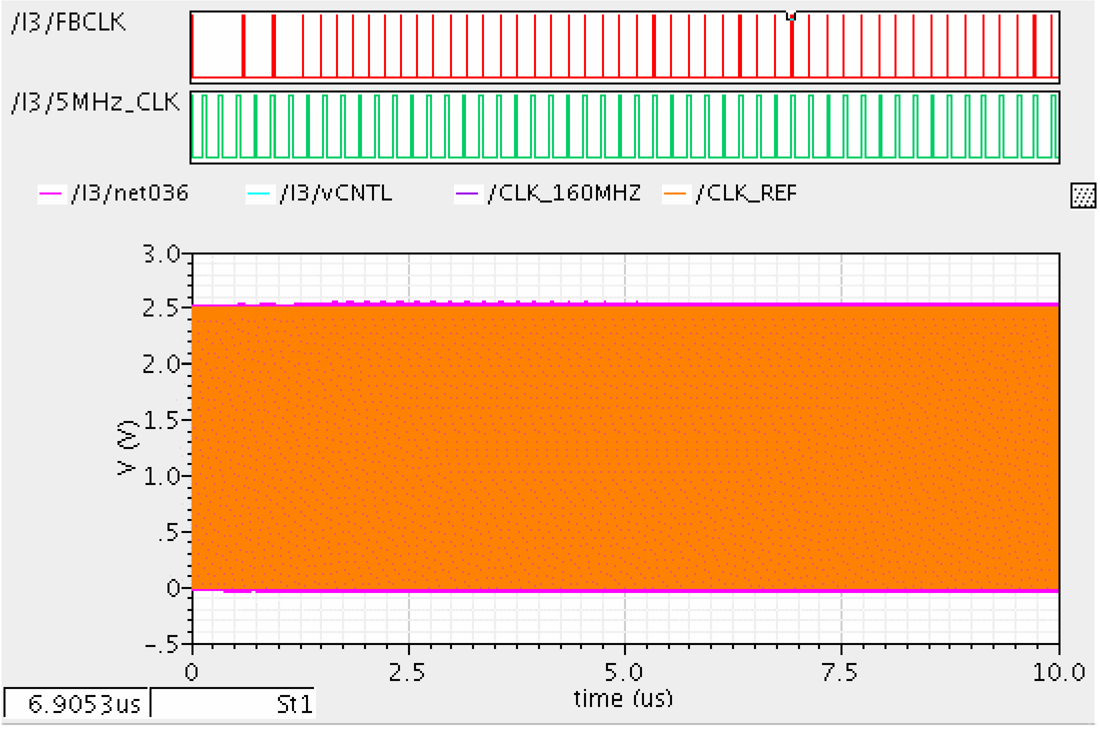
Status appears in the upper left corner of the window. Simulation output information appears in the `ultrasim.out` and `verilog.log` files. Each of these files appears in its own window during simulation. The simulation time appears at the end of the `ultrasim.out` file:

```
*** End-time: Tue Aug 21 14:24:21 2007
*** CPU time usage: 0:06:26 (386.290 sec), real time usage: 0:06:59 (419.940 sec)
*** Max. Memory Usage: 42.7857 MB.
```

Note: This simulation ran for 8 minutes 58 seconds on our Solaris machine with a 1.6 G CPU.

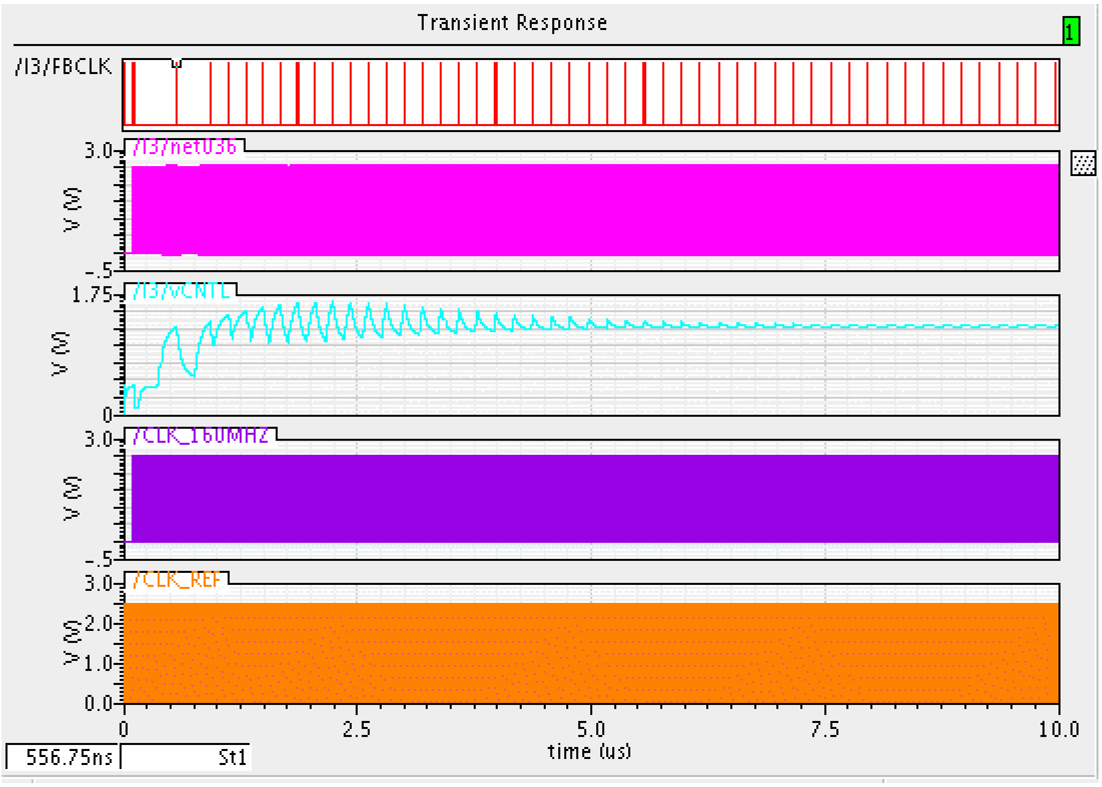
You can close each window by choosing *File – Close Window*.

When the simulation finishes, a graph window appears.



2. In the graph window, choose *Axis – Strips*.

   Your window looks like this:



   The *vCNTL* signal (third graph up from the bottom) oscillates at first and then gradually becomes a flat line.

3. When you are finished viewing results, choose *File – Close* to close the graph window.

Viewing Interface Elements on the Schematic

To view interface elements (IE) on the schematic, do the following:

1. In the schematic window, descend into *I3* :

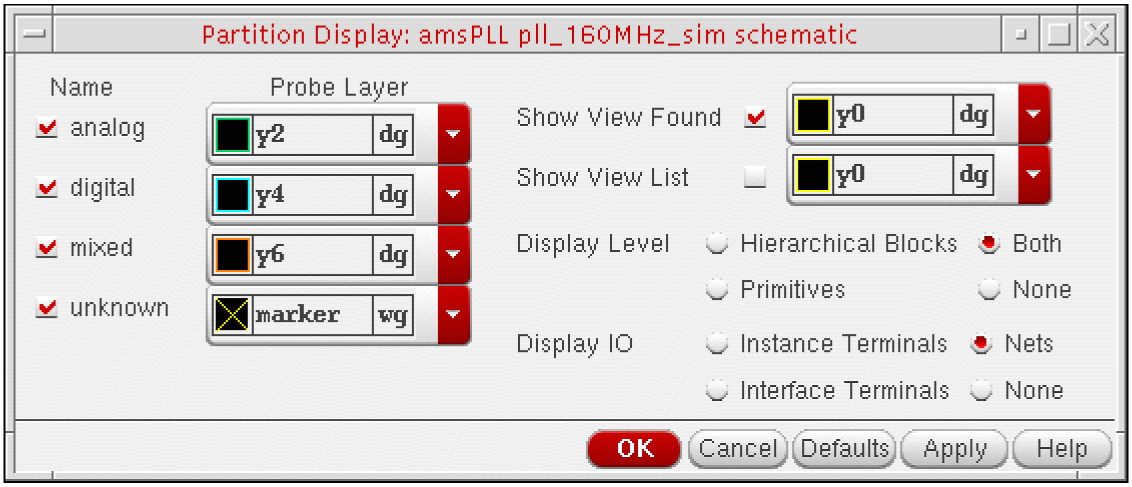
   a. Select I3.

   b. Type `e`.

   c. Click *OK*.

2. Choose *Verimix – Display Partition – Interactive*.

   The Partition Display form appears. Different colors represent different natures of blocks in the design, such as *analog*, *digital*, and *mixed*.



Partition Display: amsPLL pll_160MHz_sim schematic

Name	Probe Layer
☑ analog	y2 dg
☑ digital	y4 dg
☑ mixed	y6 dg
☑ unknown	marker wg

Show View Found ☑ y0 dg

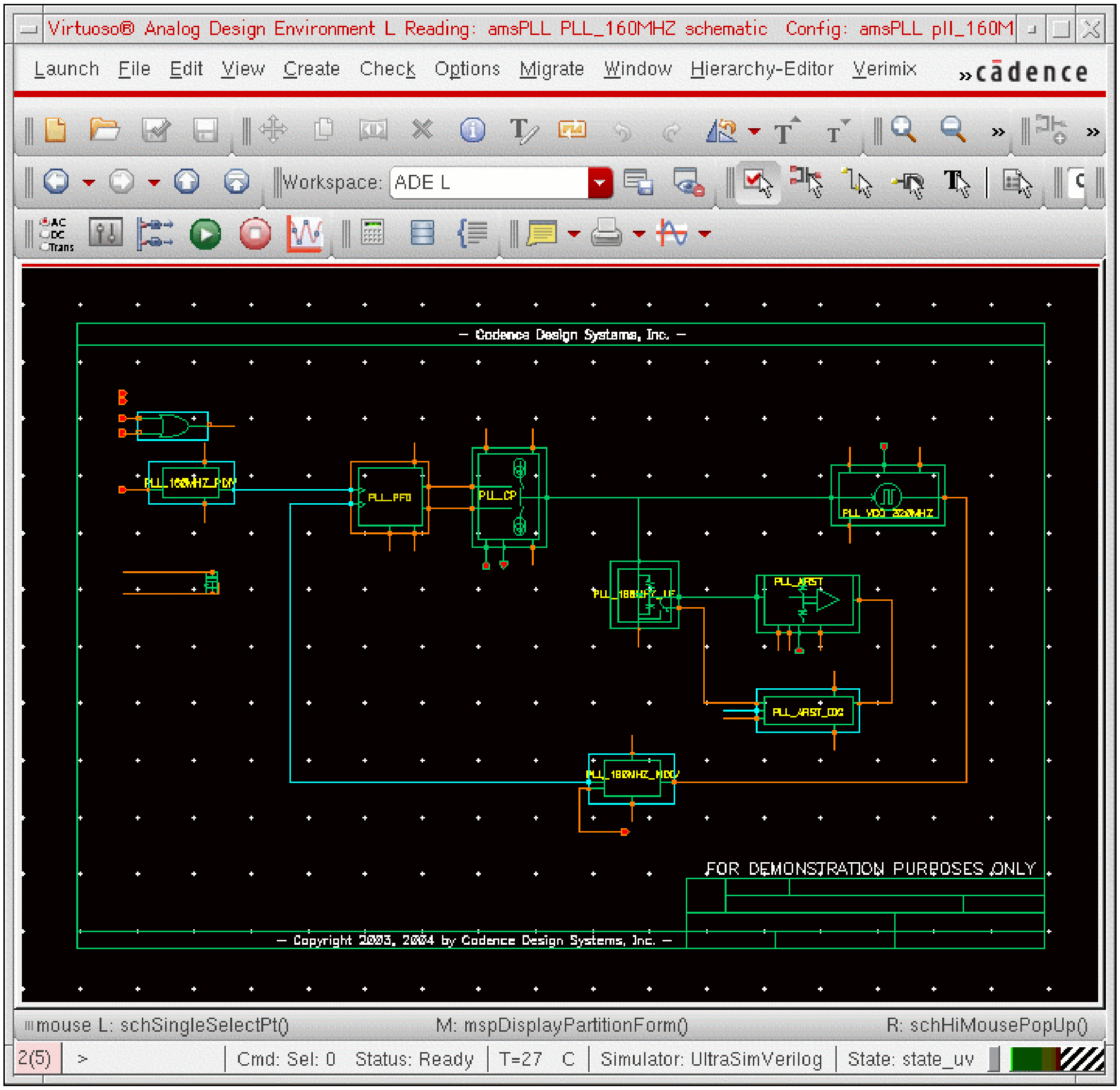
Show View List □ y0 dg

Display Level ○ Hierarchical Blocks ● Both ○ Primitives ○ None

Display IO ○ Instance Terminals ● Nets ○ Interface Terminals ○ None

3. Click *OK*.

On the schematic, mixed-signal items appear in orange.


Virtuoso® Analog Design Environment L Reading: amsPLL PLL_160MHZ schematic Config: amsPLL pll_160M
Launch File Edit View Create Check Options Migrate Window Hierarchy-Editor Verimix
cadence
Workspace: ADE L
AC
DC
Trans
– Codence Design Systems, Inc. –
PLL_160MHZ_PDIV
PLL_PFD
PLL_CP
PLL_VCO_320MHZ
PLL_160MHZ_LF
PLL_ARST
PLL_ARST_DIG
PLL_160MHZ_MDIV
FOR DEMONSTRATION PURPOSES ONLY
– Copyright 2003, 2004 by Codence Design Systems, Inc. –

mouse L: schSingleSelectPt() M: mspDisplayPartitionForm() R: schHiMousePopUp()

2(5) > Cmd: Sel: 0 Status: Ready T=27 C Simulator: UltraSimVerilog State: state_uv

In general, you will not see IEs.

4. You can choose *Verimix – Interface Elements – Library* and *Verimix – Interface Elements – Default Options* to review the setup for IEs.

   When you migrate to AMS Designer, you will notice that this setup is different.

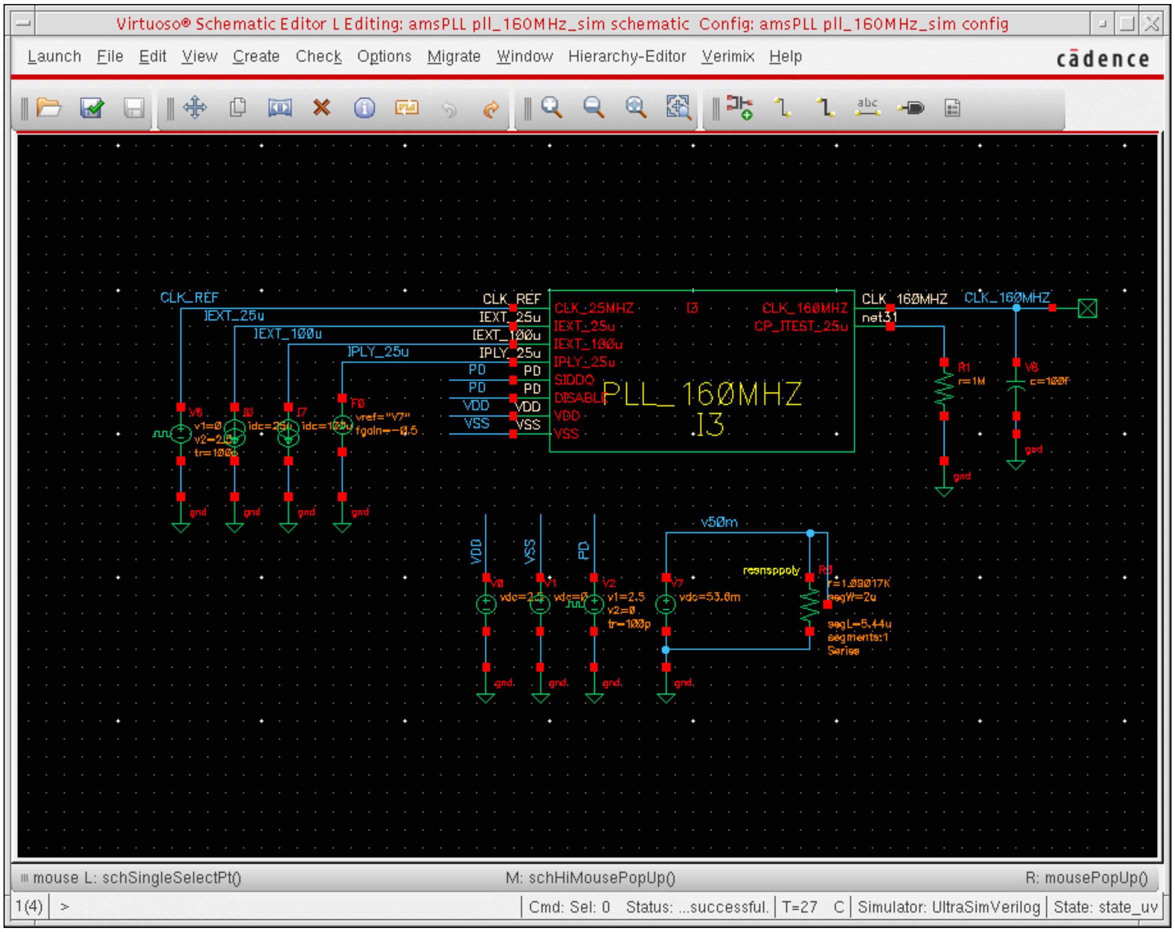
5. Click *Cancel* when you are finished viewing this setup.

The Migration Example

This migration example is a PLL design that has a 25 MHz input signal, a 160 MHz output signal, 305 MOSFETs, 97 resistors, 35 capacitors, and more than 30 behavioral modules.

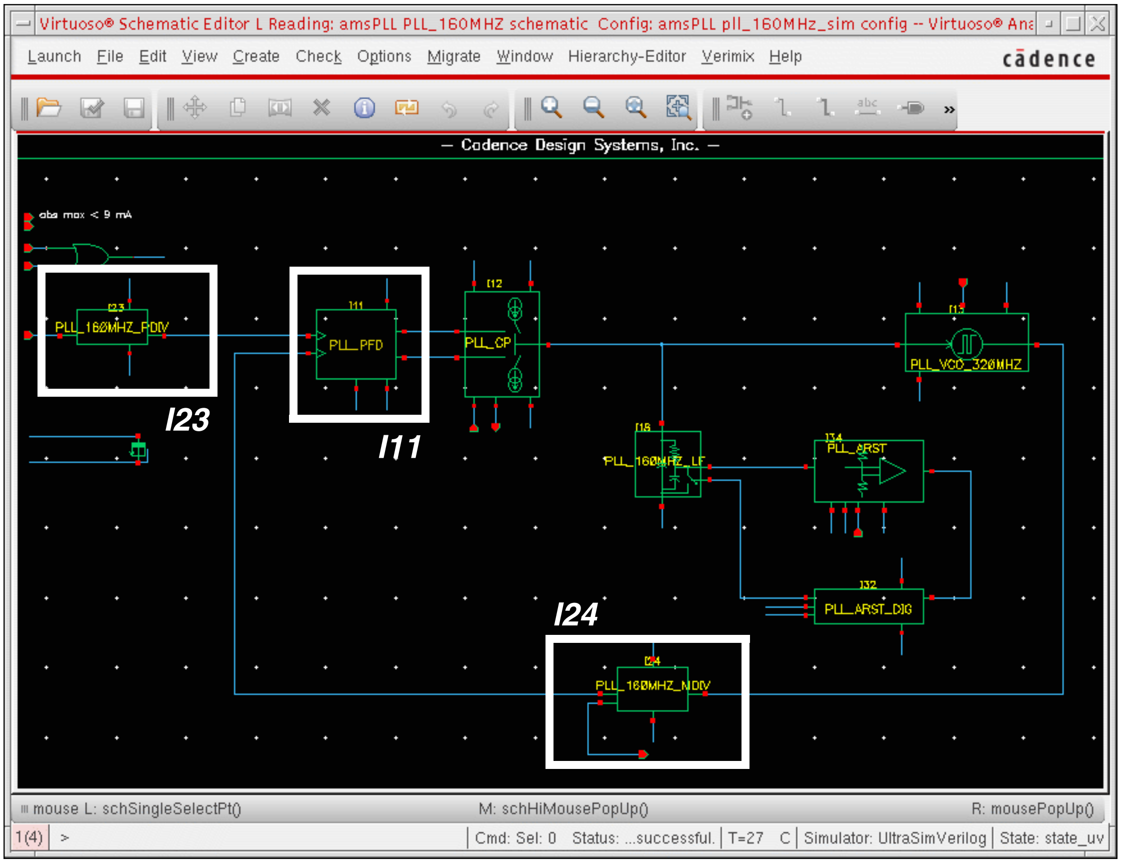
Here is the testbench schematic for the migration example (a PLL design).

Figure 4-1 Testbench Schematic for PLL Design



Inside the *I3* instance (*PLL_160MHZ*), the *I23* instance (*PLL_160MHZ_PDIV*) outputs a 5 MHz reference signal for the loop. The *I24* instance (*PLL_160MHZ_MDIV*) outputs a 160 MHz signal and a 5 MHz feedback signal for the *PLL_FPD* instance (*I11*).

Figure 4-2 Inside the I3 Instance

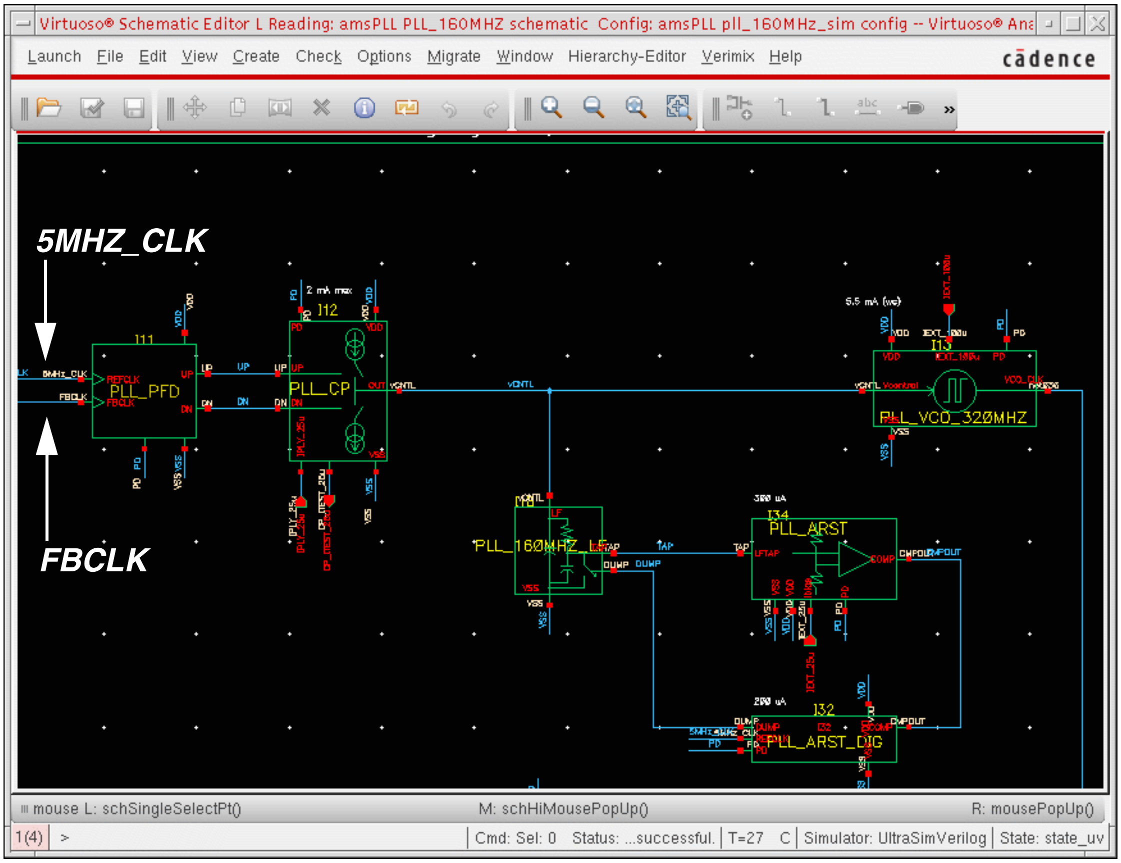

Virtuoso® Schematic Editor L Reading: amsPLL PLL_160MHZ schematic Config: amsPLL pll_160MHz_sim config -- Virtuoso® Ana
Launch File Edit View Create Check Options Migrate Window Hierarchy-Editor Verimix Help
cādence
— Cadence Design Systems, Inc. —
abs max < 9 mA
I23
PLL_160MHZ_PDIV
I11
PLL_PFD
I12
PLL_CP
I13
PLL_VCO_320MHZ
I16
PLL_160MHZ_LF
I34
PLL_ARST
I32
PLL_ARST_DIG
I24
PLL_160MHZ_MDIV
I23
I11
I24

mouse L: schSingleSelectPt()   M: schHiMousePopUp()   R: mousePopUp()

1(4) > Cmd: Sel: 0 Status: ...successful. T=27 C Simulator: UltraSimVerilog State: state_uv

When the two *PD* input signals to *I3* (see Figure 4-1 on page 90) are out of sync, the PFD (*PLL_PFD*) generates corrective pulses (*UP*, *DN*) to adjust the charge pump output voltage (*vCNTL*) which controls the frequency of the VCO (*PLL_VCO_320MHZ*).

Figure 4-3 PFD Corrective Pulses, Charge Pump Output, VCO Input



Virtuoso® Schematic Editor L Reading: amsPLL PLL_160MHZ schematic Config: amsPLL pll_160MHz_sim config -- Virtuoso® Ana

mouse L: schSingleSelectPt() M: schHiMousePopUp() R: mousePopUp()

1(4) > Cmd: Sel: 0 Status: ...successful. T=27 C Simulator: UltraSimVerilog State: state_uv

Whenever the PLL is locked, the *FBCLK* and *5MHZ_CLK* signals are in phase and the VCO control signal (*vCNTL*) is stable.

Simulating with SpectreVerilog

This part of the tutorial consists of the following actions related to simulating the migration example with SpectreVerilog:

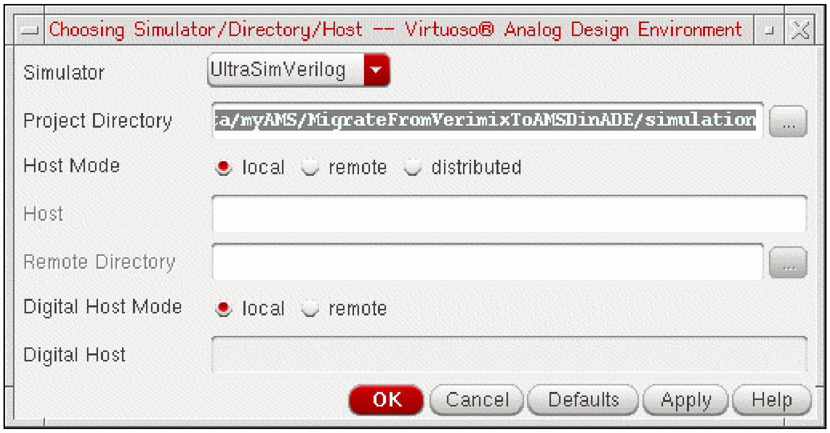
- Changing the Simulator to SpectreVerilog on page 94
- Loading the State File for UltraSimVerilog on page 78
- Netlisting and Running on page 97

Changing the Simulator to SpectreVerilog

To change the simulator to SpectreVerilog, do the following:

1. In the Virtuoso® Analog Design Environment session window, choose *Setup – Simulator/Directory/Host*.

   The Choosing Simulator/Directory/Host form appears.



Choosing Simulator/Directory/Host -- Virtuoso® Analog Design Environment

Simulator: UltraSimVerilog

Project Directory: :a/myAMS/MigrateFromVerimixToAMSDinADE/simulation

Host Mode: ● local ○ remote ○ distributed

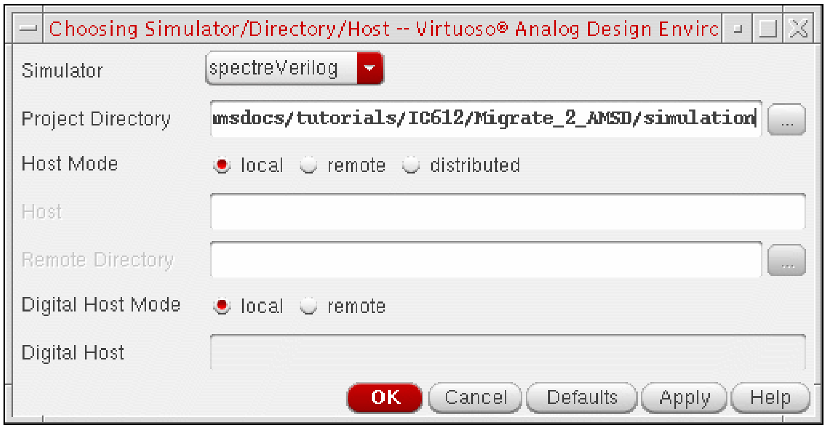
Host:

Remote Directory:

Digital Host Mode: ● local ○ remote

Digital Host:

2. In the *Simulator* drop-down combo box, select *spectreVerilog*.



Choosing Simulator/Directory/Host -- Virtuoso® Analog Design Envirc

Simulator: spectreVerilog

Project Directory: msdocs/tutorials/IC612/Migrate_2_AMSD/simulation

Host Mode: ● local ○ remote ○ distributed

Host:

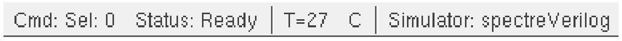
Remote Directory:

Digital Host Mode: ● local ○ remote

Digital Host:

3. Click *OK*.

Simulator: spectreVerilog appears on the status bar in the Virtuoso® Analog Design Environment session window.



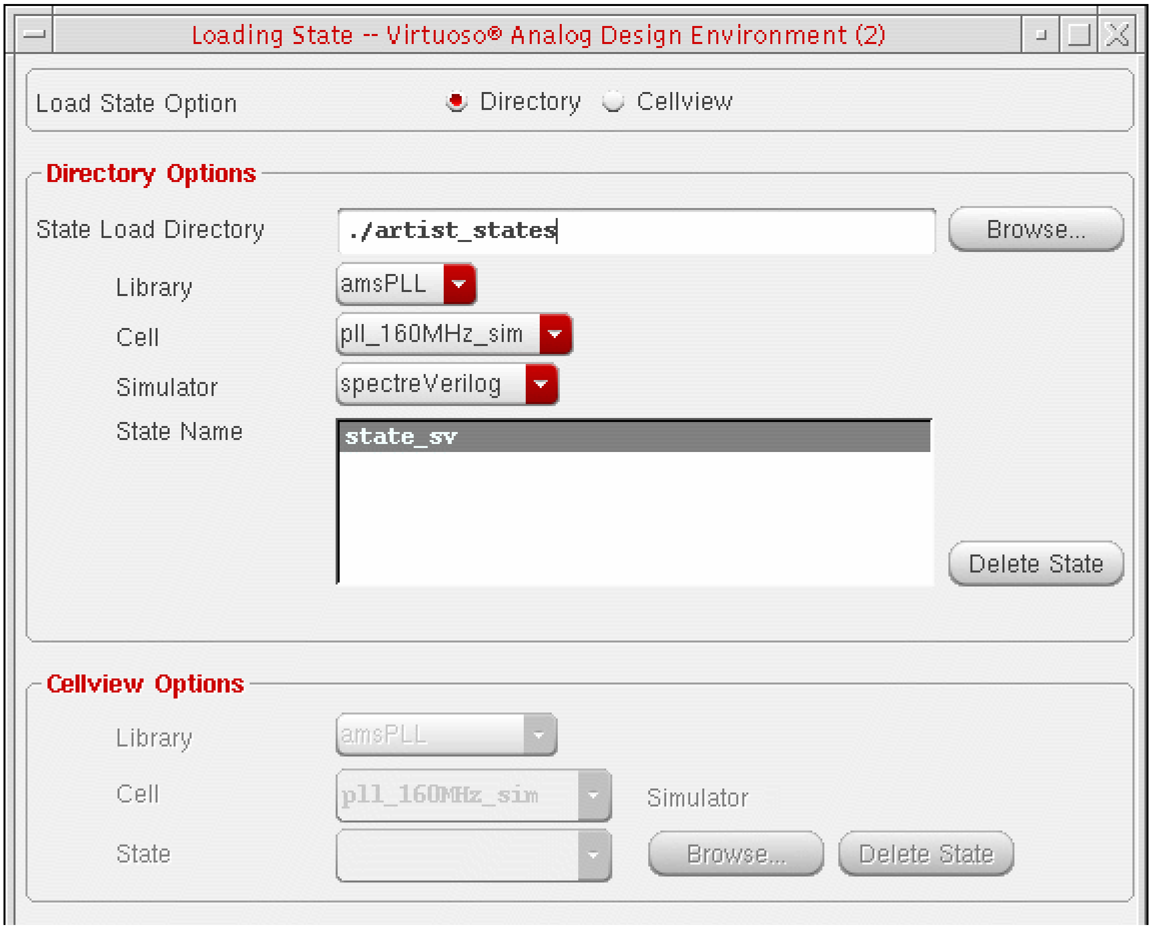
Cmd: Sel: 0   Status: Ready | T=27 C | Simulator: spectreVerilog

Loading the State File for SpectreVerilog

To load the state file for SpectreVerilog, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Session – Load State*.

   The Loading State form appears.



Loading State -- Virtuoso® Analog Design Environment (2)

Load State Option ● Directory ○ Cellview

Directory Options

State Load Directory: ./artist_states

Library: amsPLL

Cell: pll_160MHz_sim

Simulator: spectreVerilog

State Name: state_sv

Cellview Options

Library: amsPLL

Cell: pll_160MHz_sim

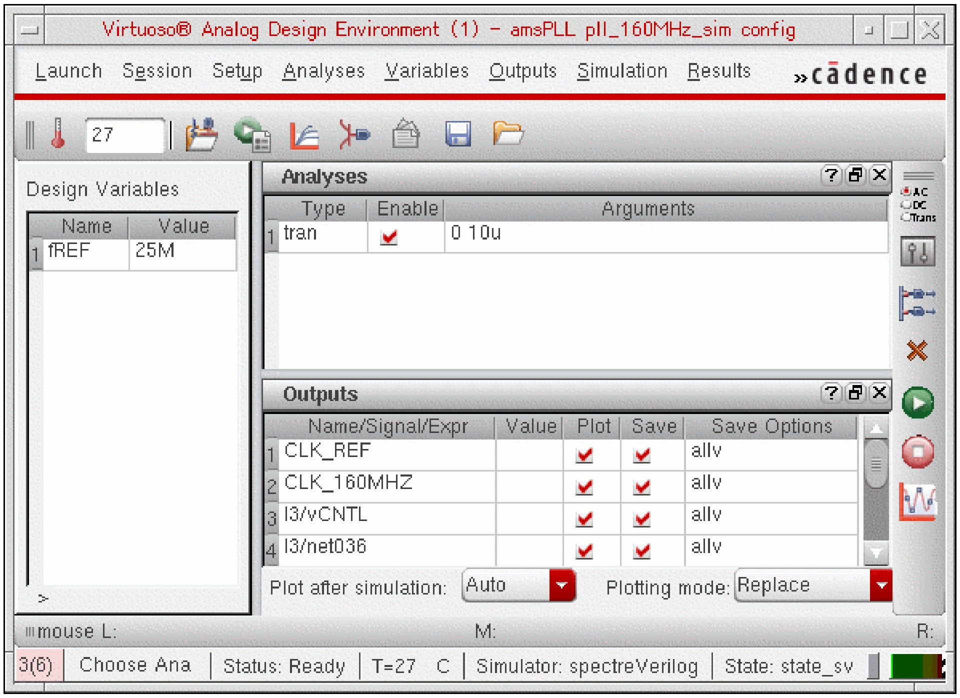
Simulator

State

2. In the *State Name* box, select *state_sv*.

3. Click *OK*.

The state settings appear in the Virtuoso® Analog Design Environment session window, such as *tran ... 10u* in the *Analyses* area and nodes to plot in the *Outputs* area.


Virtuoso® Analog Design Environment (1) - amsPLL pll_160MHz_sim config
Launch Session Setup Analyses Variables Outputs Simulation Results
»cādence
AC
DC
Trans

27

Design Variables

	Name	Value
1	fREF	25M

Analyses

	Type	Enable	Arguments
1	tran	✓	0 10u

Outputs

	Name/Signal/Expr	Value	Plot	Save	Save Options
1	CLK_REF		✓	✓	allv
2	CLK_160MHZ		✓	✓	allv
3	I3/vCNTL		✓	✓	allv
4	I3/net036		✓	✓	allv

Plot after simulation: Auto   Plotting mode: Replace

mouse L: M: R:

3(6) | Choose Ana | Status: Ready | T=27 C | Simulator: spectreVerilog | State: state_sv

Netlisting and Running

To netlist and run using SpectreVerilog, do the following:

➤ In the Virtuoso® Analog Design Environment window, choose *Simulation – Netlist and Run*.

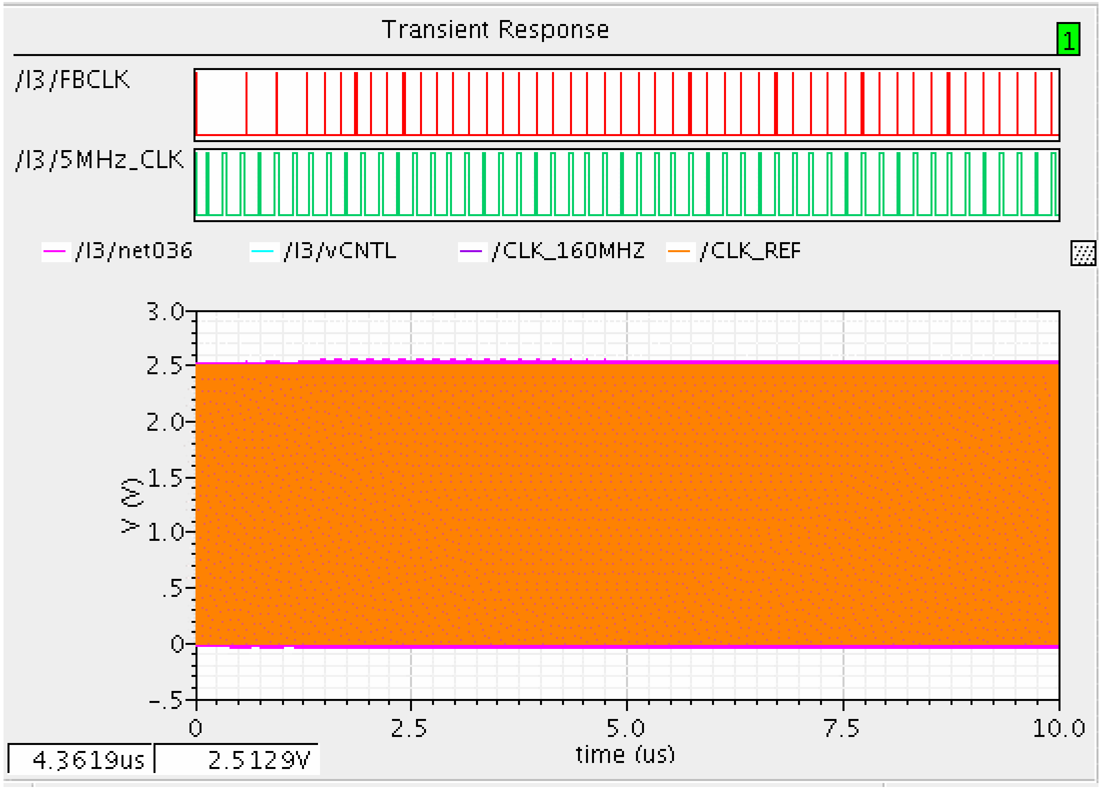
Status appears in the upper left corner of the window. Simulation output information appears in the `spectre.out` and `verilog.log` files. Each of these files appears in its own window during simulation. The simulation time appears at the end of the `spectre.out` file:

```
Time used: CPU = 2.96 ks (49m 19.0s), elapsed = 3.08 ks (51m 21.0s), util. = 96%.
```

Note: This simulation ran for 60 minutes on our Solaris machine with a 1.6 G CPU.

You can close each window by choosing *File – Close Window*.

When the simulation finishes, a graph window appears.



When you are finished viewing results, choose *File – Close* to close the graph window.

Using the AMS Designer Simulator

Virtuoso$^{®}$ AMS Designer has many advantages over the SpectreVerilog and UltraSimVerilog mixed-signal solutions:

- Better performance (33% for this particular example)
- More powerful digital solver
- Powerful connect rules (CRs)
- Flexible discipline definitions
- Bidirectional CR support
- More language support (Verilog-AMS, VHDL-AMS, SystemVerilog, SystemC)

If you use the AMS Designer simulator in the Virtuoso Analog Design Environment (ADE), there are two netlisters:

- The cell-based netlister, which is the original netlister for AMS, requires `ams` simInfo (which contains information such as a parameter list and how to netlist each component) when it generates the individual `netlist.vams` netlist files in the library/cell/view directory structure. For more information, see "Netlisting" in the *Virtuoso AMS Environment User Guide*.
- The open simulation system (OSS) netlister is available in IC 5.1.41 USR4 and later. You can use this netlister when you migrate from SpectreVerilog or UltraSimVerilog to AMS Designer. The OSS netlister uses existing `spectre` views, as do UltraSim and UltraSimVerilog. The OSS netlister generates a single netlist file (`netlist.vams`) that includes all the modules that need to be compiled. (The final netlist is also one file for Spectre, UltraSim, SpectreVerilog, and UltraSimVerilog.) The OSS netlister works the same way for the AMS Designer simulator as it does for the Spectre and UltraSim simulators.

If you are using the OSS netlister, you can use the same `config` view to run AMS Designer.

This tutorial illustrates how to use the OSS netlister and `irun` so that you can benefit from the many advantages of AMS Designer. See the following topics for more information:

- Changing the Simulator to AMS Designer on page 101
- Loading the State File for AMS Designer on page 102
- Selecting and Customizing Connect Rules for AMS Designer on page 104
- Setting Netlister and Run Modes on page 110

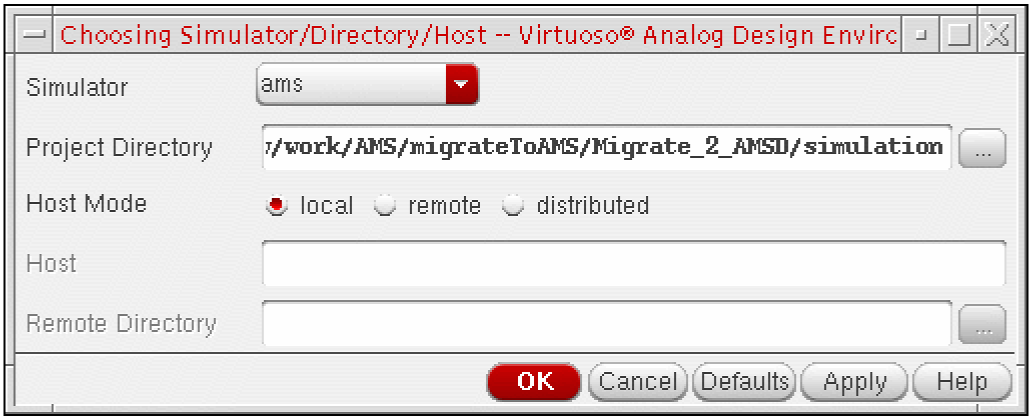
Changing the Simulator to AMS Designer

To change the simulator to AMS Designer, do the following:

1. In the Virtuoso® Analog Design Environment session window, choose *Setup – Simulator/Directory/Host*.

   The Choosing Simulator/Directory/Host form appears.

2. In the *Simulator* drop-down combo box, select *ams*.



Choosing Simulator/Directory/Host -- Virtuoso® Analog Design Envirc

Simulator: ams

Project Directory: /work/AMS/migrateToAMS/Migrate_2_AMSD/simulation

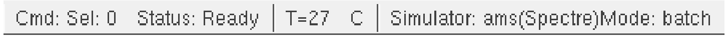
Host Mode: ● local ○ remote ○ distributed

Host

Remote Directory

3. Click *OK*.

   Simulator: ams appears on the status bar in the Virtuoso® Analog Design Environment session window. The name of the analog solver (*Spectre* or *UltraSim*) appears in parentheses after *ams*.



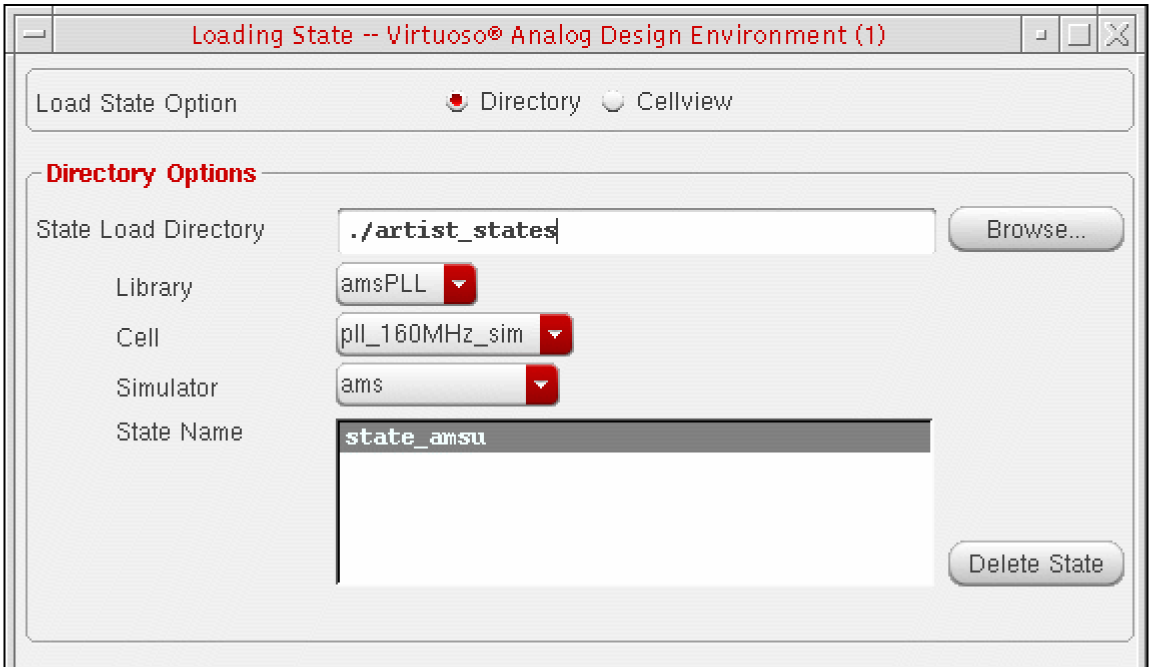
Cmd: Sel: 0 Status: Ready | T=27 C | Simulator: ams(Spectre)Mode: batch

Loading the State File for AMS Designer

To load the state file for AMS Designer, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Session – Load State*.

   The Loading State form appears.



Loading State -- Virtuoso® Analog Design Environment (1)

Load State Option ● Directory ○ Cellview

Directory Options

State Load Directory: ./artist_states

Library: amsPLL

Cell: pll_160MHz_sim

Simulator: ams

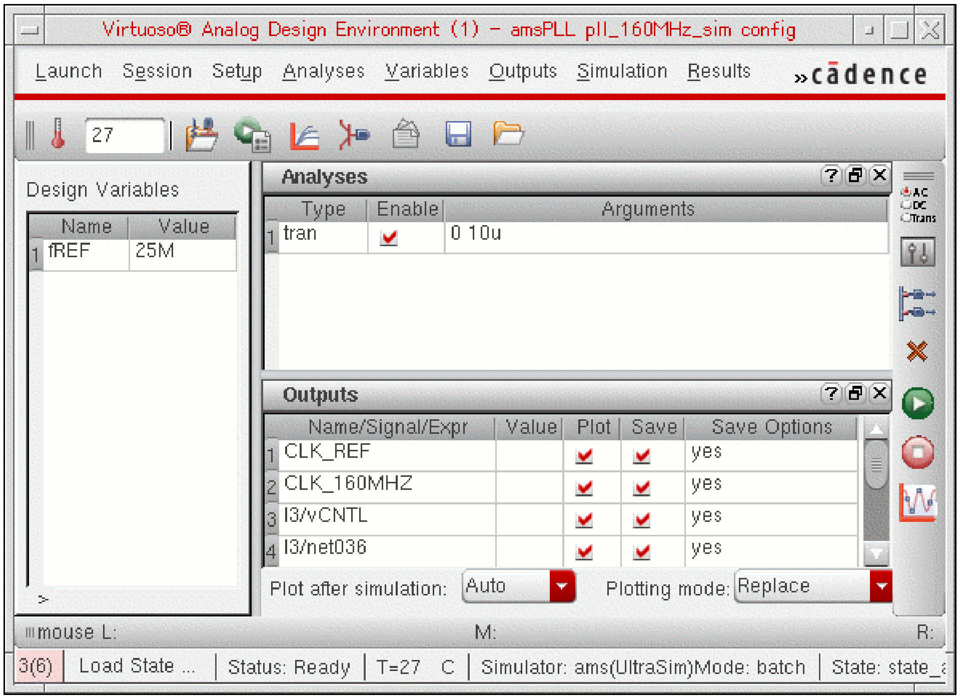
State Name: state_amsu

2. In the *State Name* area, select *state_amsu*.

   This state uses the UltraSim solver.

3. Click *OK*.

The state settings appear in the Virtuoso® Analog Design Environment session window, such as *tran … 10u* in the *Analyses* area and nodes to plot in the *Outputs* area. *Simulator: ams(UltraSim)* appears onthe status bar in the ADE window.



Design Variables

	Name	Value
1	fREF	25M

Analyses

	Type	Enable	Arguments
1	tran	✓	0 10u

Outputs

	Name/Signal/Expr	Value	Plot	Save	Save Options
1	CLK_REF		✓	✓	yes
2	CLK_160MHZ		✓	✓	yes
3	I3/vCNTL		✓	✓	yes
4	I3/net036		✓	✓	yes

Plot after simulation: Auto   Plotting mode: Replace

mouse L: M: R:

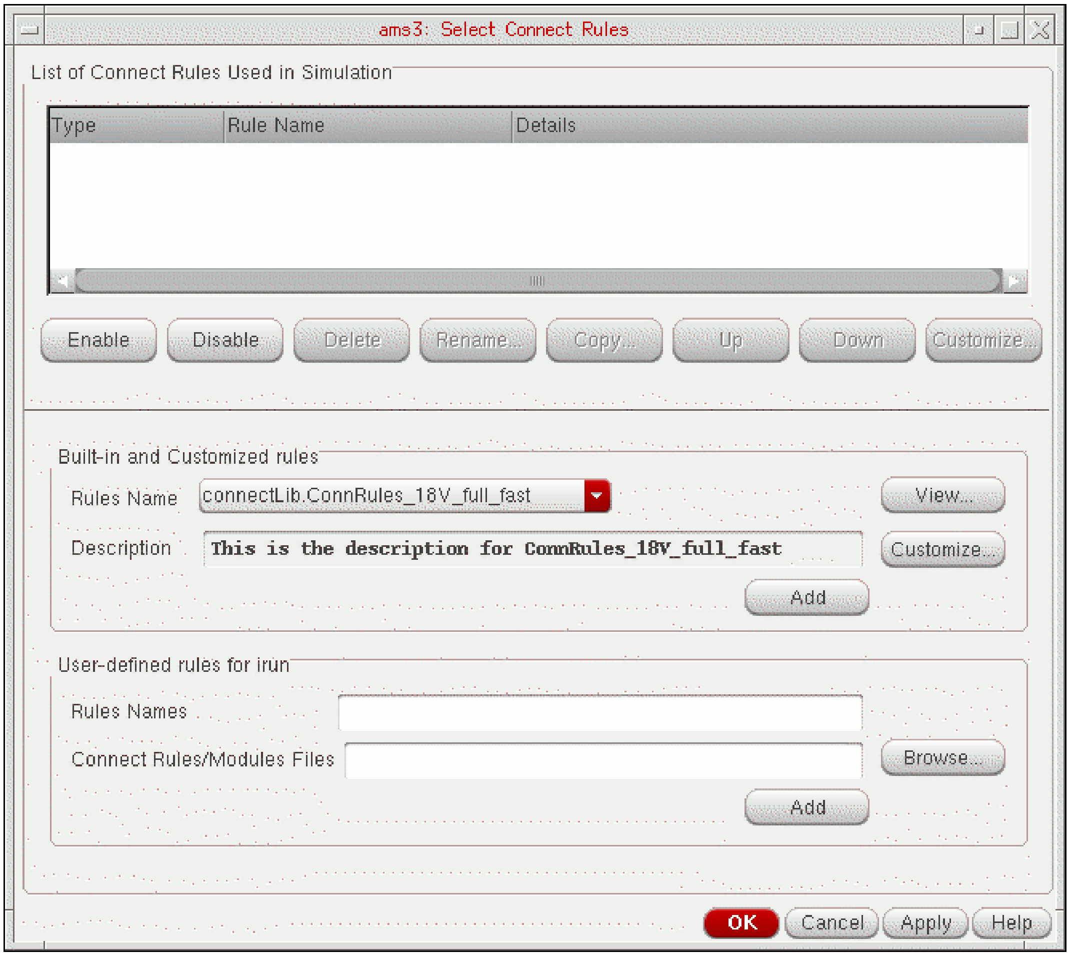
3(6) | Load State ... | Status: Ready | T=27 C | Simulator: ams(UltraSim)Mode: batch | State: state_

Selecting and Customizing Connect Rules for AMS Designer

To specify and customize connect rules for the AMS Designer simulator, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Setup – Connect Rules*.

   The Select Connect Rules form appears.


ams3: Select Connect Rules

List of Connect Rules Used in Simulation

Type	Rule Name	Details

Built-in and Customized rules

Rules Name: connectLib.ConnRules_18V_full_fast

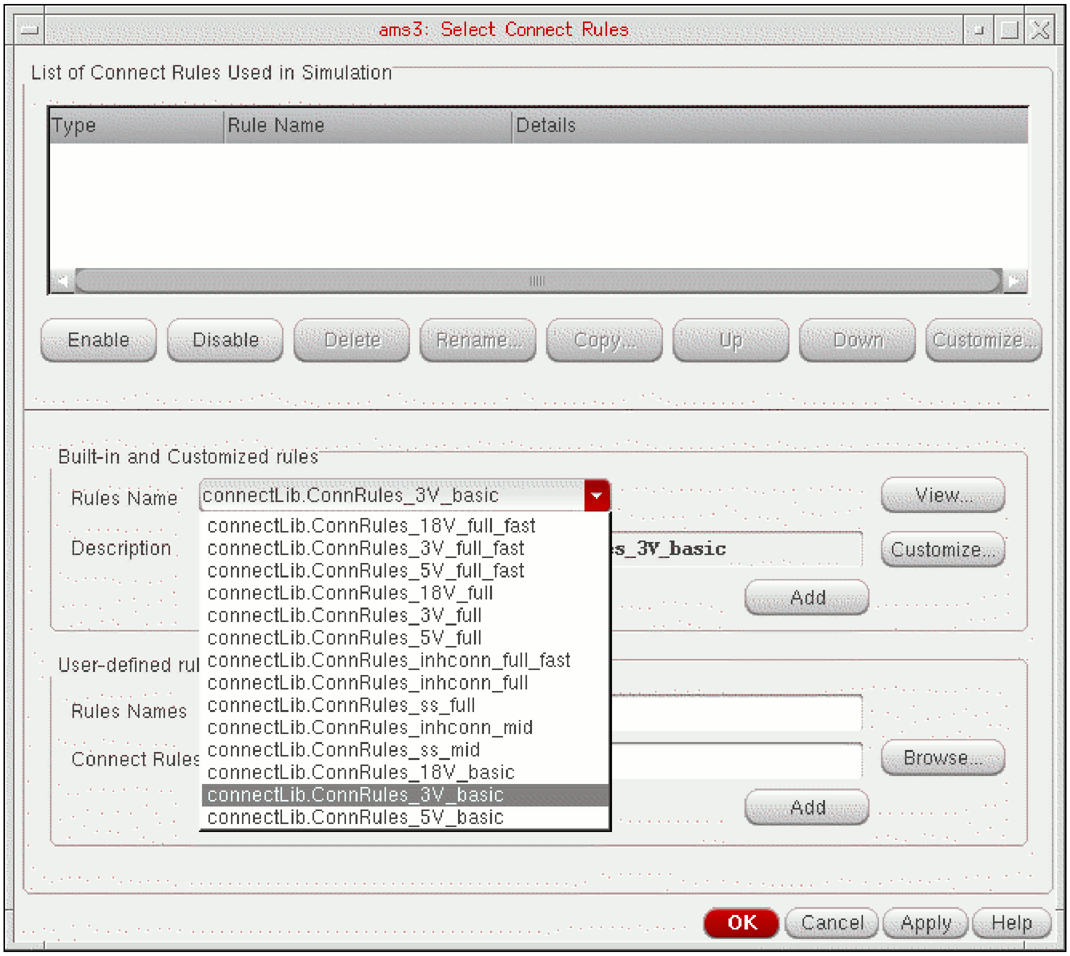
Description: This is the description for ConnRules_18V_full_fast

User-defined rules for irun

Rules Names:

Connect Rules/Modules Files:

2. In the *Rules Name* drop-down combo box, select *connectLib.ConnRules_3V_basic*.


ams3: Select Connect Rules
List of Connect Rules Used in Simulation
Type
Rule Name
Details
Enable
Disable
Delete
Rename...
Copy...
Up
Down
Customize...
Built-in and Customized rules
Rules Name
Description
View...
Customize...
Add
User-defined rul
Rules Names
Connect Rules
Browse...
Add
connectLib.ConnRules_18V_full_fast
connectLib.ConnRules_3V_full_fast
connectLib.ConnRules_5V_full_fast
connectLib.ConnRules_18V_full
connectLib.ConnRules_3V_full
connectLib.ConnRules_5V_full
connectLib.ConnRules_inhconn_full_fast
connectLib.ConnRules_inhconn_full
connectLib.ConnRules_ss_full
connectLib.ConnRules_inhconn_mid
connectLib.ConnRules_ss_mid
connectLib.ConnRules_18V_basic
connectLib.ConnRules_3V_basic
connectLib.ConnRules_5V_basic
OK
Cancel
Apply
Help

Rules Name: connectLib.ConnRules_3V_basic

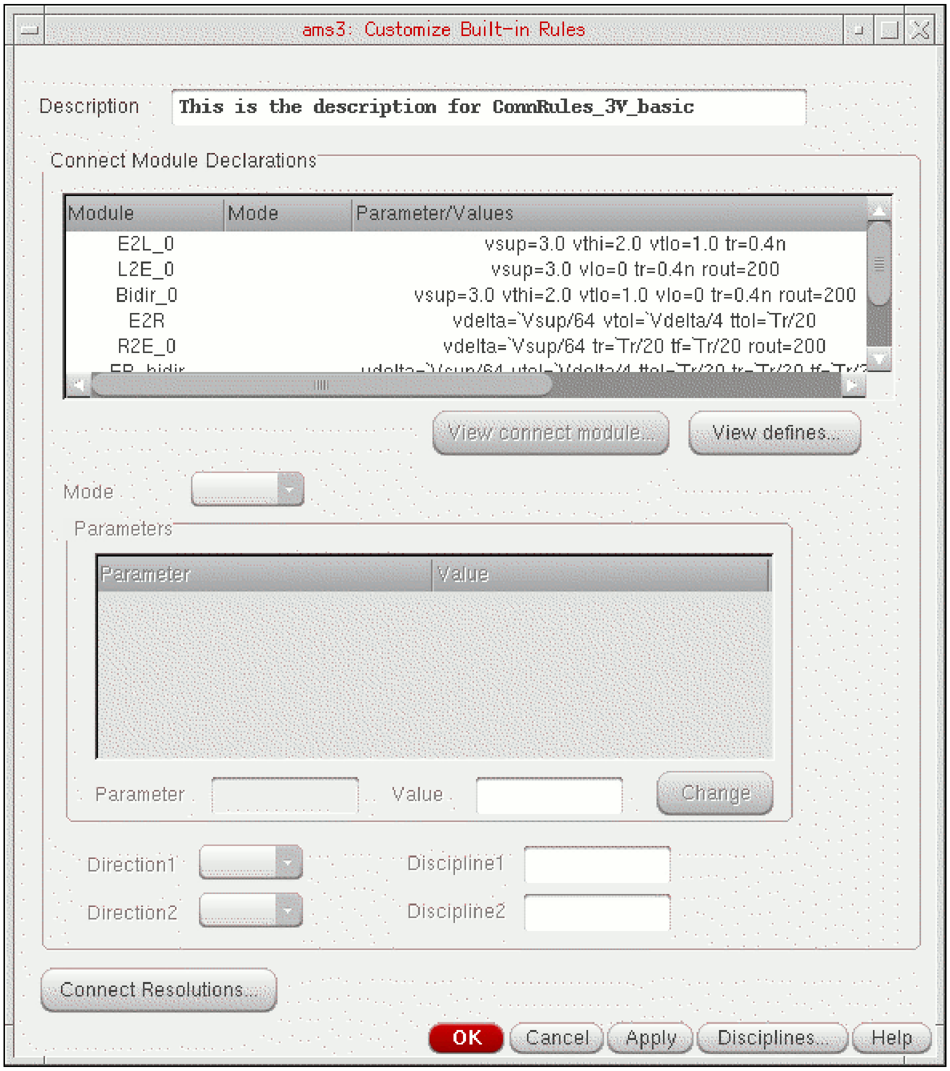
Description: s_3V_basic

3. (Optional) To view the contents of the connect rule, click *View*.

   The connect rule file appears in a window. When you are finished viewing the file, you can choose *File – Close Window*.

4. To customize this connect rule, click *Customize*.

The Customize Built-in Rules form appears.



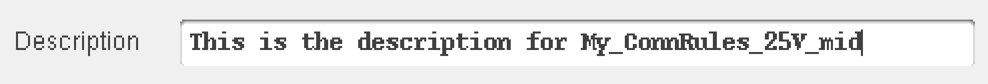
Description: This is the description for ConnRules_3V_basic

Connect Module Declarations

Module	Mode	Parameter/Values
E2L_0		vsup=3.0 vthi=2.0 vtlo=1.0 tr=0.4n
L2E_0		vsup=3.0 vlo=0 tr=0.4n rout=200
Bidir_0		vsup=3.0 vthi=2.0 vtlo=1.0 vlo=0 tr=0.4n rout=200
E2R		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20
R2E_0		vdelta=`Vsup/64 tr=`Tr/20 tf=`Tr/20 rout=200
ER_bidir		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20 tr=`Tr/20 tf=`Tr/2

5. In the *Description* field, change the name of the rule to `My_ConnRules_25V_mid`:

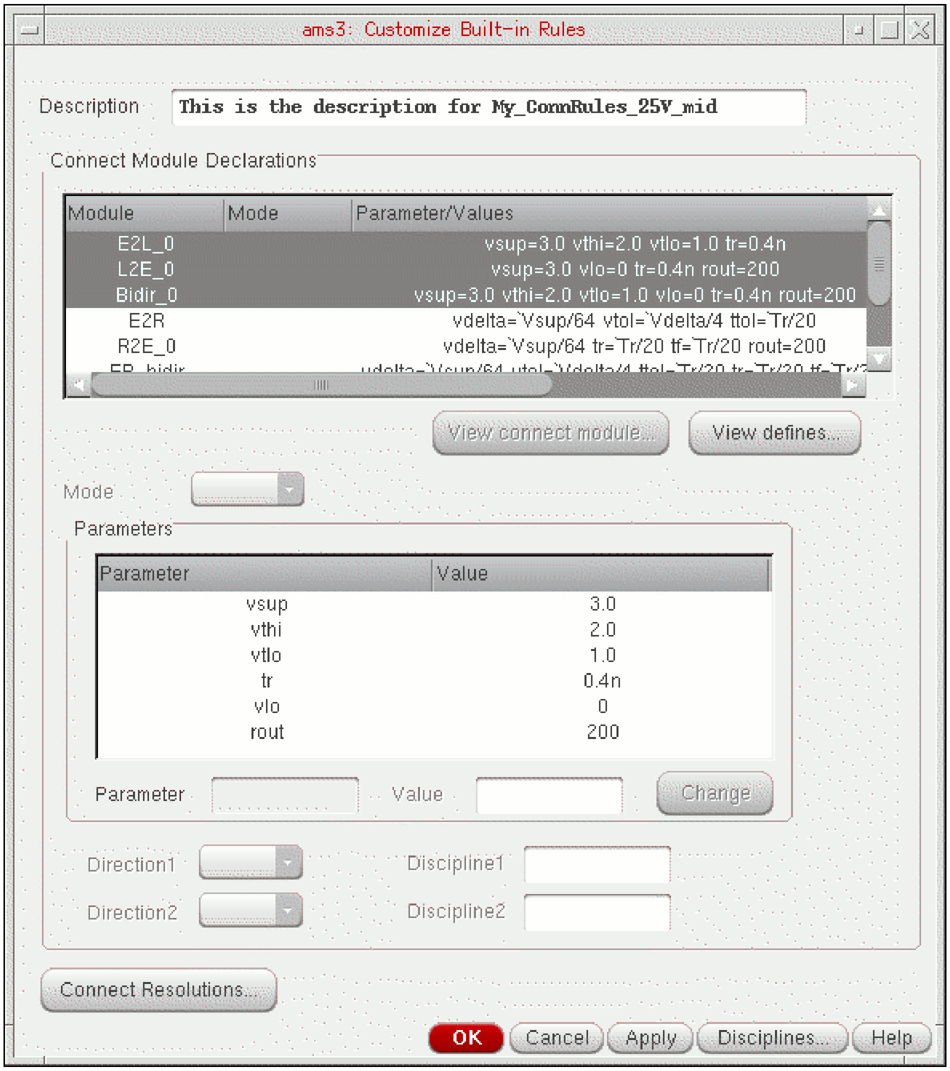
```
This is the description for My_ConnRules_25V_mid
```



Description: This is the description for My_ConnRules_25V_mid

6. In the *Connect Module Declarations* group box, hilight the top three lines containing information for modules *E2L_0*, *L2E_0*, and *Bidir_0*.

   The shared parameters appear in the *Parameters* group box.



Description: This is the description for My_ConnRules_25V_mid

Connect Module Declarations

Module	Mode	Parameter/Values
E2L_0		vsup=3.0 vthi=2.0 vtlo=1.0 tr=0.4n
L2E_0		vsup=3.0 vlo=0 tr=0.4n rout=200
Bidir_0		vsup=3.0 vthi=2.0 vtlo=1.0 vlo=0 tr=0.4n rout=200
E2R		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20
R2E_0		vdelta=`Vsup/64 tr=`Tr/20 tf=`Tr/20 rout=200
ER_bidir		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20 tr=`Tr/20 tf=`Tr/2

Parameters

Parameter	Value
vsup	3.0
vthi	2.0
vtlo	1.0
tr	0.4n
vlo	0
rout	200

7. Change these values as follows:

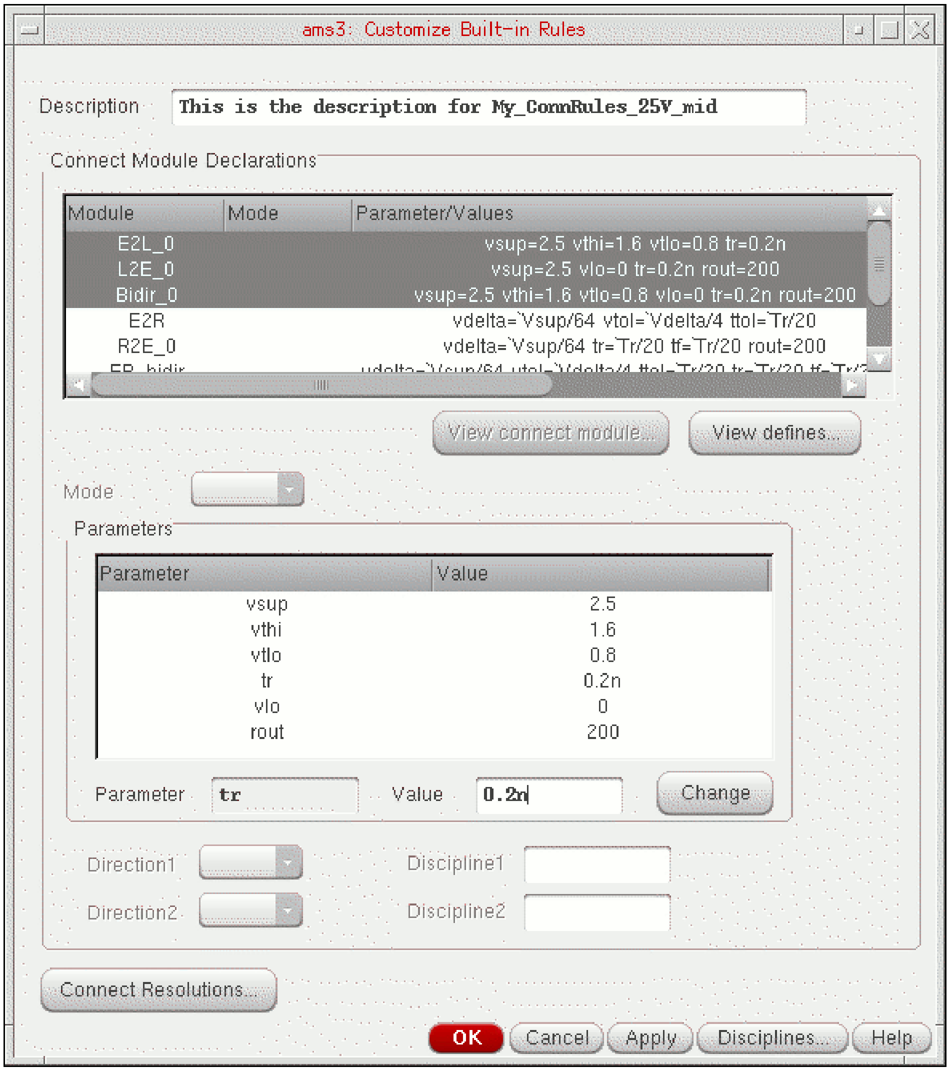
Parameter	*Value*	Change it to
vsup	*3.0*	`2.5`
vthi	*2.0*	`1.6`

Parameter	*Value*	Change it to
vtlo	*1.0*	`0.8`
tr	*0.4n*	`0.2n`

a. Select the parameter you want to change.

b. In the *Value* field, change its value.

c. Click *Change*.



Description: `This is the description for My_ConnRules_25V_mid`

Connect Module Declarations

Module	Mode	Parameter/Values
E2L_0		vsup=2.5 vthi=1.6 vtlo=0.8 tr=0.2n
L2E_0		vsup=2.5 vlo=0 tr=0.2n rout=200
Bidir_0		vsup=2.5 vthi=1.6 vtlo=0.8 vlo=0 tr=0.2n rout=200
E2R		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20
R2E_0		vdelta=`Vsup/64 tr=`Tr/20 tf=`Tr/20 rout=200
ER_bidir		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20 tr=`Tr/20 tf=`Tr/2

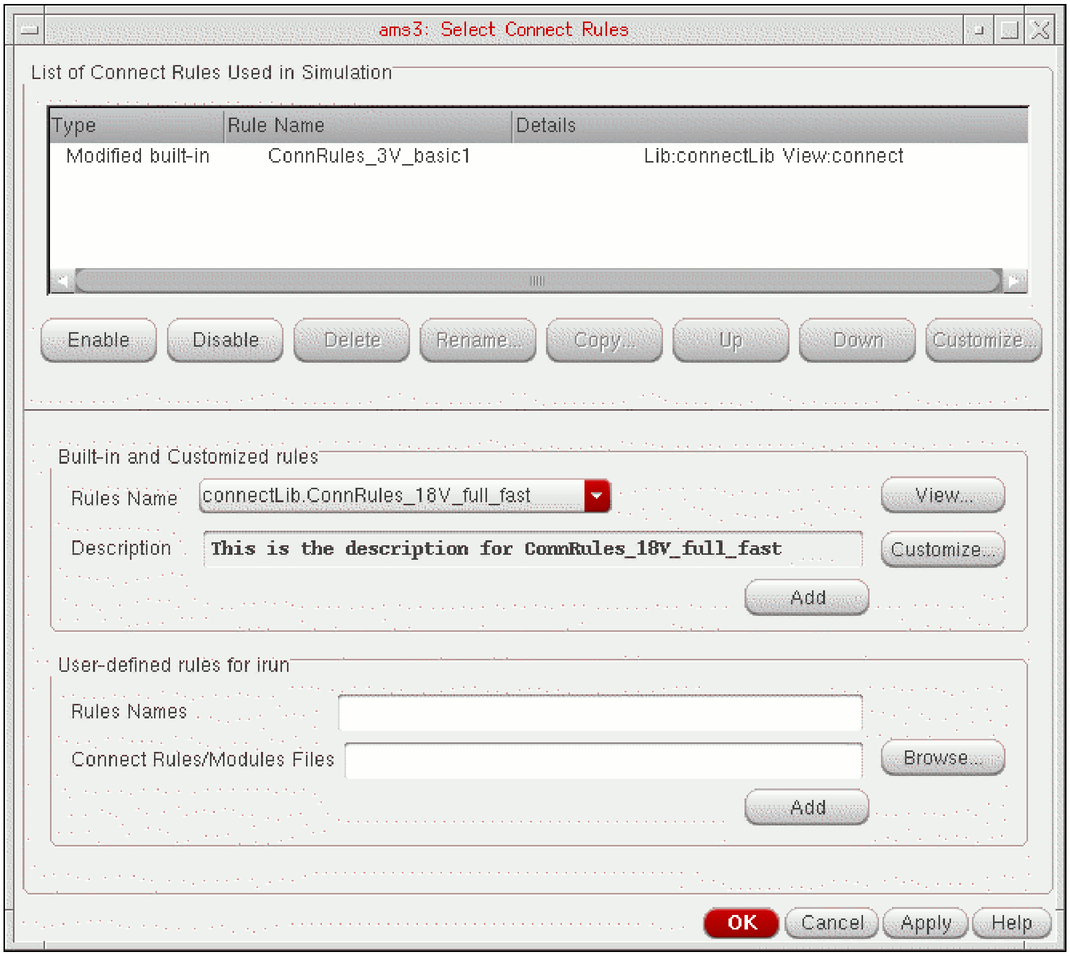
Parameters

Parameter	Value
vsup	2.5
vthi	1.6
vtlo	0.8
tr	0.2n
vlo	0
rout	200

Parameter: `tr` Value: `0.2n`

8. Click *OK*.

9. On the Select Connect Rules form, click *Add*.


ams3: Select Connect Rules
List of Connect Rules Used in Simulation
Type
Rule Name
Details
Enable
Disable
Delete
Rename...
Copy...
Up
Down
Customize...
Built-in and Customized rules
View...
Customize...
Add
User-defined rules for irun
Rules Names
Connect Rules/Modules Files
Browse...
Add
OK
Cancel
Apply
Help

Type	Rule Name	Details
Modified built-in	ConnRules_3V_basic1	Lib:connectLib View:connect

Rules Name: connectLib.ConnRules_18V_full_fast

Description: This is the description for ConnRules_18V_full_fast

Modified built-in appears in the *Type* column.

10. Click *OK*.

The connect rules you specify on the Select Connect Rules form apply to the whole design.

Note: You might want to have several connect rules in the same design. You can set disciplines on a net, cell, instance, or library, and you can specify several connect rules accordingly.

Setting Netlister and Run Modes

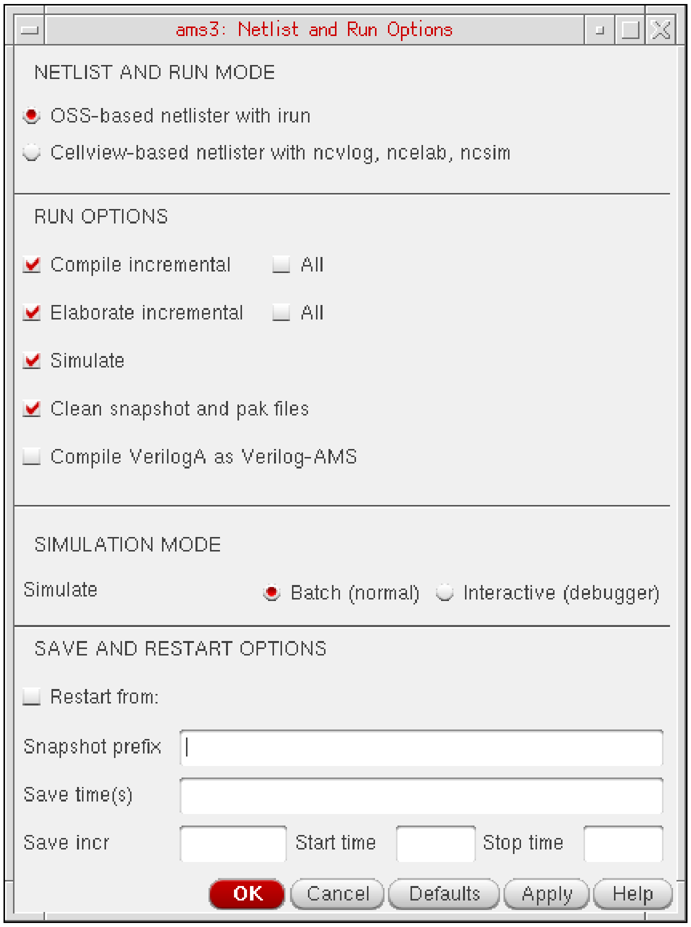
To set netlister and run modes, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Simulation – Netlist and Run Options*.

   The Netlister and Run Options form appears.

2. For *Netlister Mode*, select *OSS-based*.

   The *ncverilog* radio button appears as a *Run Mode* choice.



ams3: Netlist and Run Options

NETLIST AND RUN MODE

● OSS-based netlister with irun
○ Cellview-based netlister with ncvlog, ncelab, ncsim

RUN OPTIONS

☑ Compile incremental □ All
☑ Elaborate incremental □ All
☑ Simulate
☑ Clean snapshot and pak files
□ Compile VerilogA as Verilog-AMS

SIMULATION MODE

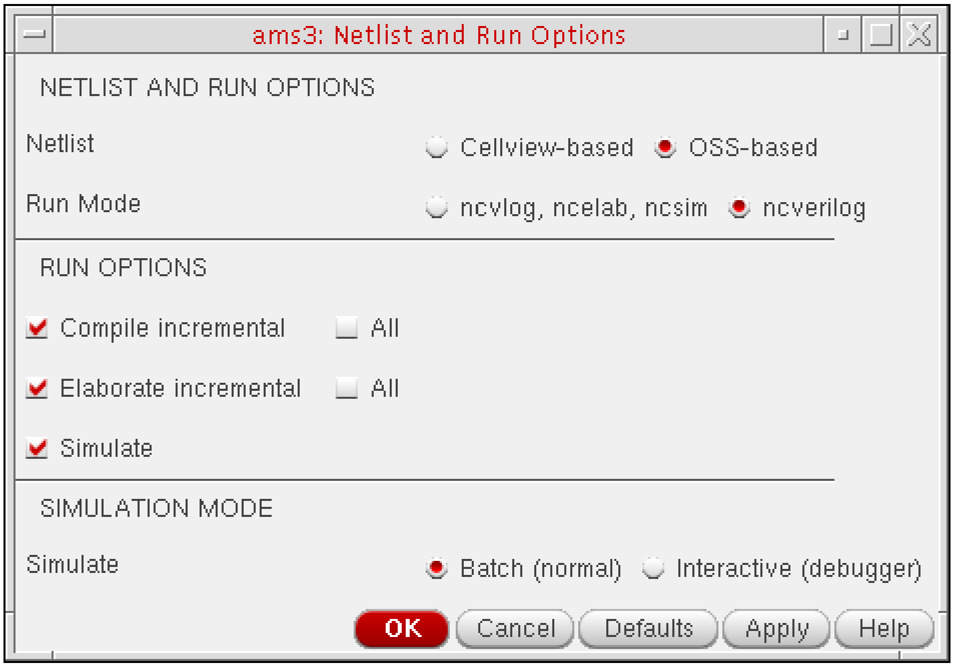
Simulate ● Batch (normal) ○ Interactive (debugger)

SAVE AND RESTART OPTIONS

□ Restart from:

Snapshot prefix
Save time(s)
Save incr Start time Stop time

3. For *Run Mode*, select *ncverilog*.



ams3: Netlist and Run Options

NETLIST AND RUN OPTIONS

Netlist ○ Cellview-based ● OSS-based

Run Mode ○ ncvlog, ncelab, ncsim ● ncverilog

RUN OPTIONS

☑ Compile incremental □ All

☑ Elaborate incremental □ All

☑ Simulate

SIMULATION MODE

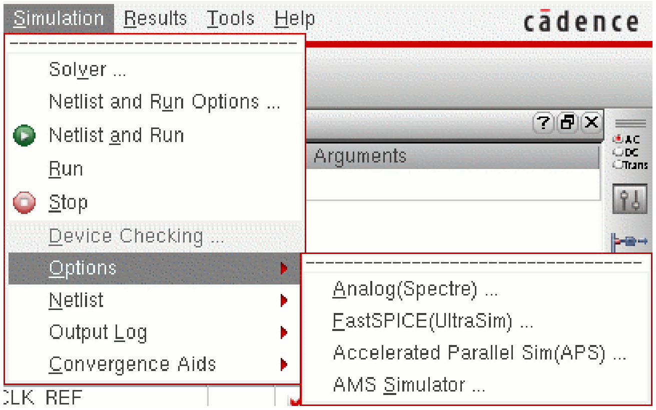
Simulate ● Batch (normal) ○ Interactive (debugger)

4. Click *OK*.

   You are ready to simulate.

Viewing Options

As you proceed through this next set of steps, you will notice several choices on the *Simulation – Options* menu in the Virtuoso® Analog Design Environment window:



You will not change any of these options during this example, but you will view some of the forms.

See

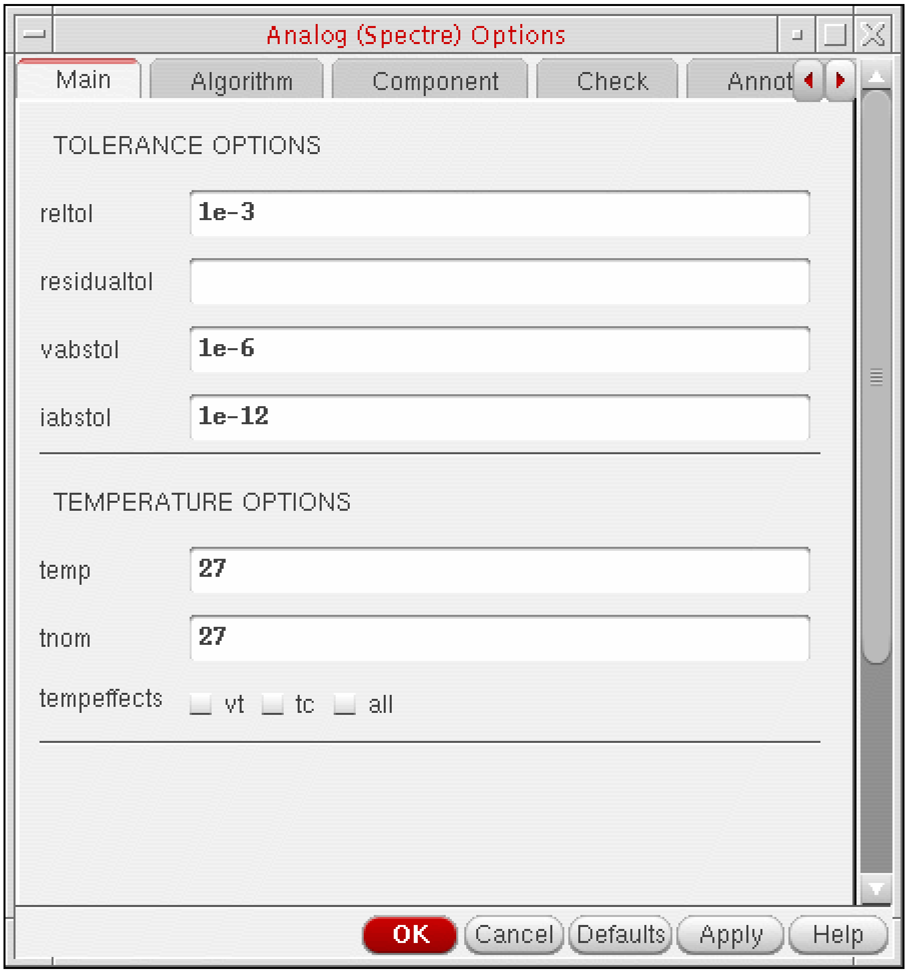
- Viewing Analog (Spectre) Options on page 113
- Viewing FastSPICE (UltraSim) Options on page 114
- Viewing AMS Options on page 115

Viewing Analog (Spectre) Options

To view Analog (Spectre) options, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Simulation – Options – Analog(Spectre)*.

   The Analog (Spectre) Options form appears.

   Analog (Spectre) Options

   TOLERANCE OPTIONS

reltol	1e-3
residualtol	
vabstol	1e-6
iabstol	1e-12

   TEMPERATURE OPTIONS

temp	27
tnom	27
tempeffects	□ vt □ tc □ all

   For information about the options you can set on this form, see the *Virtuoso Spectre Circuit Simulator User Guide*.

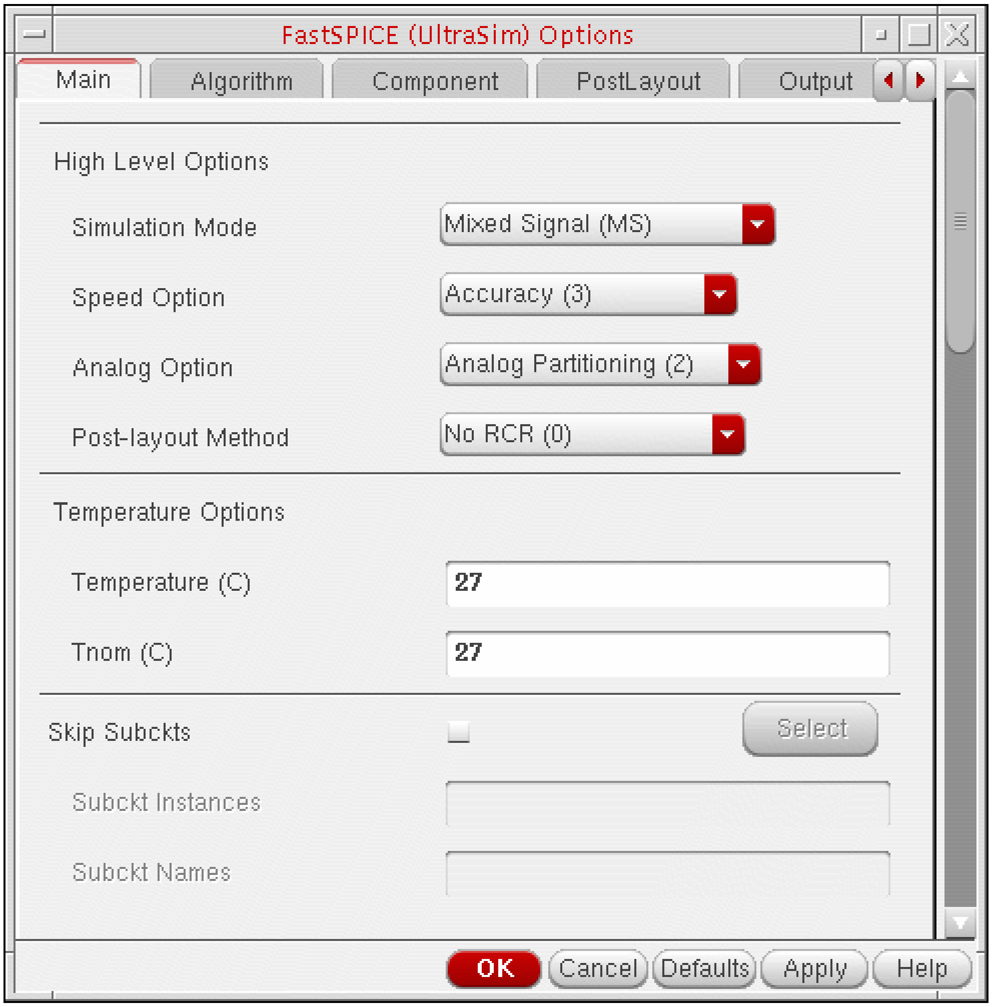
2. When you are finished viewing Spectre options, click *Cancel* to close the form.

Viewing FastSPICE (UltraSim) Options

To view FastSPICE (UltraSim) options, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Simulation – Options – FastSPICE(UltraSim)*.

   The FastSPICE (UltraSim) Options form appears.



FastSPICE (UltraSim) Options

High Level Options

Simulation Mode: Mixed Signal (MS)

Speed Option: Accuracy (3)

Analog Option: Analog Partitioning (2)

Post-layout Method: No RCR (0)

Temperature Options

Temperature (C): 27

Tnom (C): 27

Skip Subckts □

Subckt Instances

Subckt Names

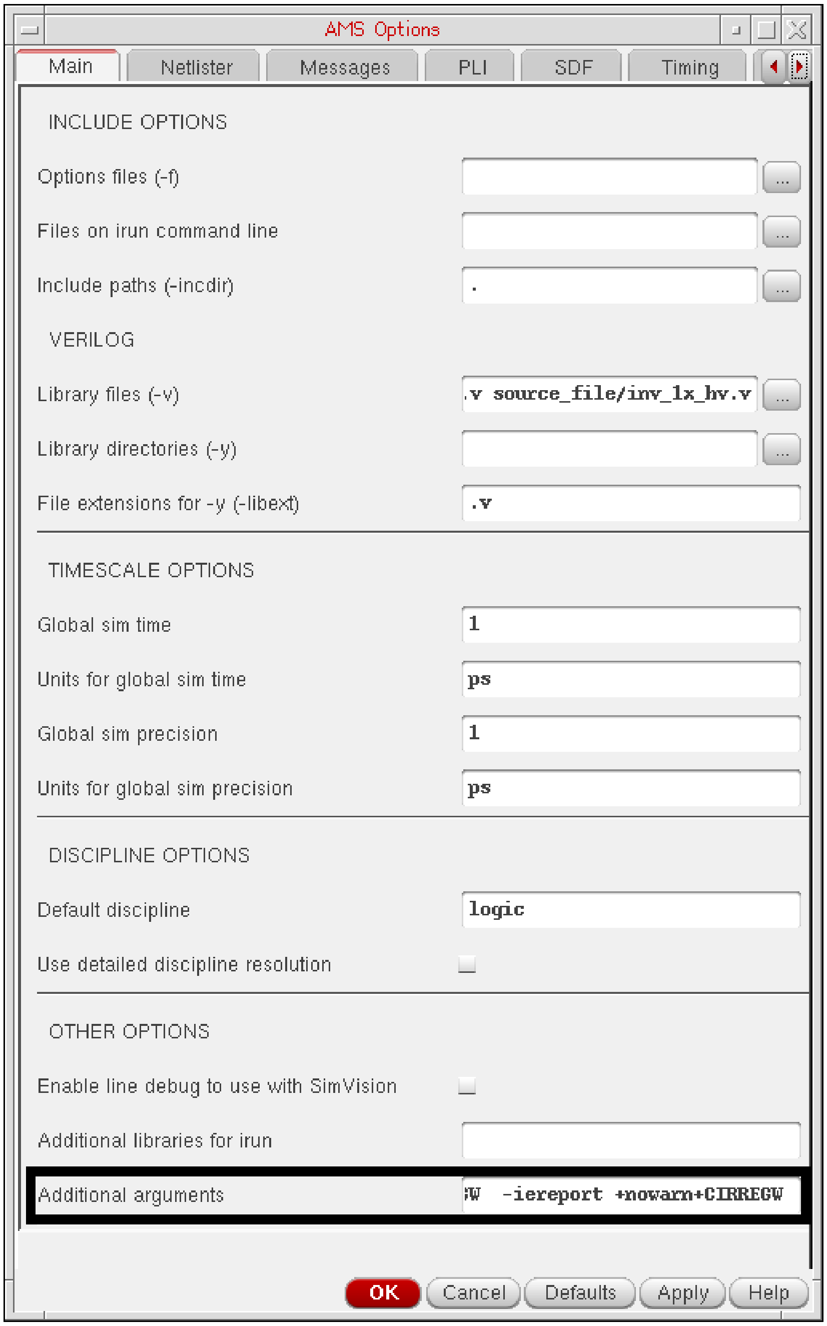
   For information about the options you can set on this form, see the *Virtuoso UltraSim Simulator User Guide*.

2. When you are finished viewing FastSPICE options, click *Cancel* to close the form.

Viewing AMS Options

To view AMS simulation options, do the following:

1. In the Virtuoso® Analog Design Environment window, choose *Simulation – Options – AMS Simulator*.



INCLUDE OPTIONS

Options files (-f)

Files on irun command line

Include paths (-incdir) .

VERILOG

Library files (-v) .v source_file/inv_1x_hv.v

Library directories (-y)

File extensions for -y (-libext) .v

TIMESCALE OPTIONS

Global sim time 1

Units for global sim time ps

Global sim precision 1

Units for global sim precision ps

DISCIPLINE OPTIONS

Default discipline logic

Use detailed discipline resolution □

OTHER OPTIONS

Enable line debug to use with SimVision □

Additional libraries for irun

Additional arguments ;W -iereport +nowarn+CIRREGW

2. Scroll down to the bottom of this form to see that `-iereport` appears in the *Additional arguments* field.

   When you specify the `-iereport` option, the elaborator generates an interface element (IE) report. The IE report appears at the top of the simulation log file. This report contains information about each IE the software inserted into the design, such as its name, net, discipline, and so on.

3. When you are finished viewing options, click *Cancel* to close the form.

Netlisting and Running

To netlist and run, do the following:

➤ In the Virtuoso® Analog Design Environment window, choose *Simulation – Netlist and Run*.

Status appears in the upper left corner of the window. Simulation output information appears in the `ncverilog.log` file. The simulation time appears at the end of the file:

```
Time Usage:
  Total user time: 0:04:16 (256.980 sec), system time: 0:00:01 (1.100 sec),
real time: 0:04:26 (266.600 sec)
```

Note: This simulation ran for 5 minutes 47 seconds on our Solaris machine with a 1.6 G CPU.

Because *-iereport* appears in the *Additional arguments* field on the Main tab of the AMS tabbed window (see "Viewing AMS Options" on page 115), an IE report appears at the top of the log file. That report might look something like this:

```
----------IE report -------------

Automatically inserted instance: pll_160MHz_sim.I3.I11.I15.net18__E2L__logic
(merged):
        connectmodule name: E2L,
        inserted across signal: net18
        and ports of discipline: logic
        Sensitivity infomation:
                No Sensitivity info
        Discipline of Port (Ain): electrical, Analog port
        Discipline of Port (Dout): logic, Digital port
        Drivers of port Dout: No drivers
        Loads of port Dout: No loads
```

As you scroll down past UltraSim version and build time information, you will notice messages related to the compiled C flow for Verilog-A. Those messages might look something like this:

```
File read: .../Migrate_2_AMSD/models/spectre/resd_va.va
Created directory amsControl.ahdlSimDB/ (775)
Created directory
        amsControl.ahdlSimDB/
2143_migrateToAMS_Migrate_2_AMSD_models_spectre_resd_va.va.ahdlcmi/
        (775)
Created directory
        amsControl.ahdlSimDB/
2143_migrateToAMS_Migrate_2_AMSD_models_spectre_resd_va.va.ahdlcmi/
Linux2.4.21-37.ELsmp+gcc/
        (775)
Compiling ahdlcmi module library.
Finished compilation in 2 s (elapsed).
Installed compiled interface for resd_va.
```

You can use the compiled C flow to boost performance particularly when you are using Verilog-A to model bsources or CMOS devices such as MOSFETs, resistors, and capacitors. See “Using the Compiled C Code Flow” in the *Cadence Verilog-A Language Reference* for more information.

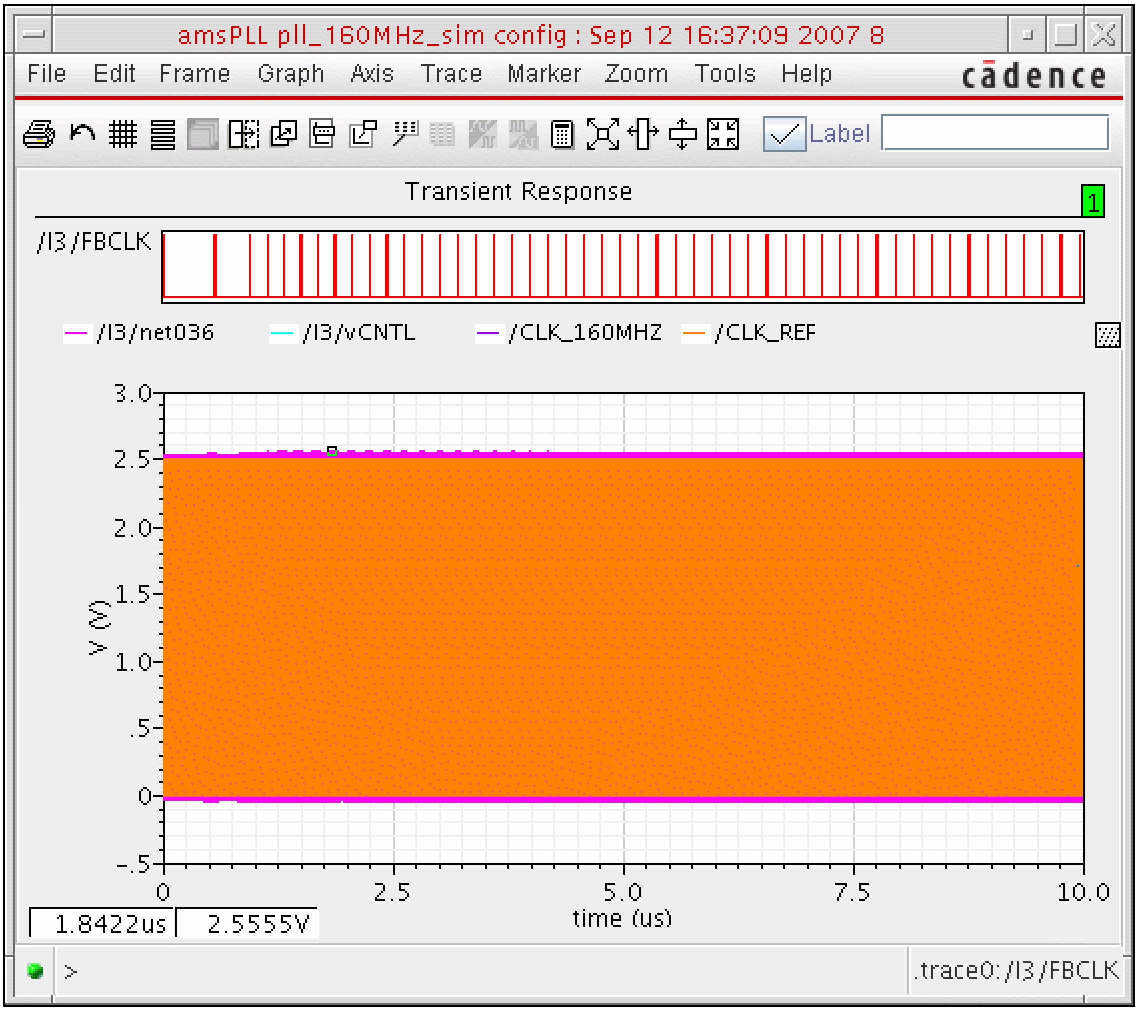
The IUS 5.83 release supports a feature called FastCross that speeds up the simulation by reducing the number of global time steps. Information about the total global time steps appears near the end of the log file:

```
Total Accepts: 1.452 M
```

You can close each window by choosing *File – Close Window*.

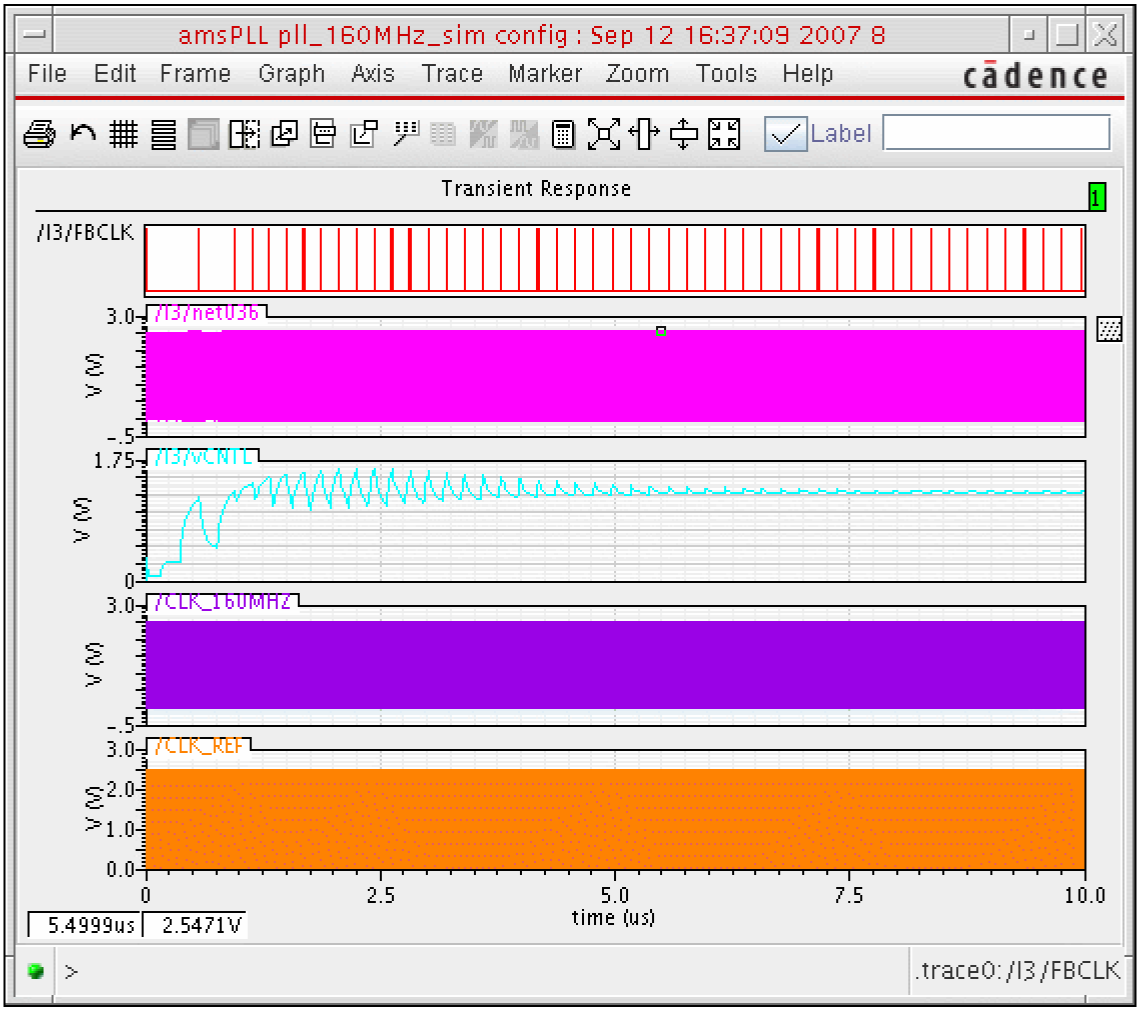
Viewing Waveforms

When the simulation finishes, a graph window appears.



You can do the following to compare the display with your UltraSimVerilog results to verify that they are the same.

1. In the graph window, choose *Axis – Strips*.



2. Compare the resulting waveform display with your UltraSimVerilog results.

3. When you are finished viewing results, choose *File – Close* to close the graph window.

Displaying Partitions

You can also verify that the partitions for AMS Designer are the same as those for your UltraSimVerilog (mixed-signal) simulation:

1. In the schematic window, descend into *I3* :

   a. Select I3.

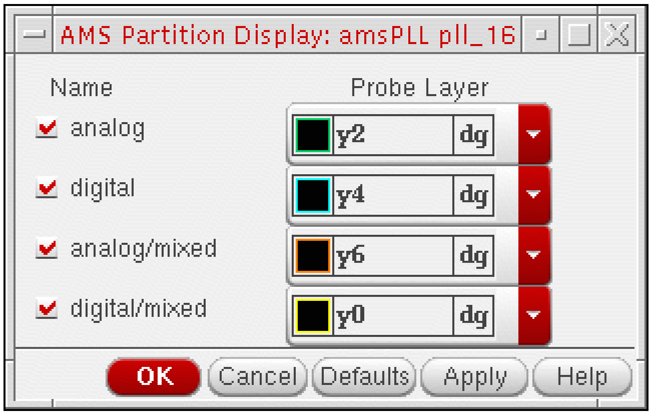
   b. Type `e`.

   c. Click *OK*.

2. In the schematic window, choose *AMS – Display Partition – Initialize*.

3. In the schematic window, choose *AMS – Display Partition – Interactive*.

   The AMS Partition Display form appears.



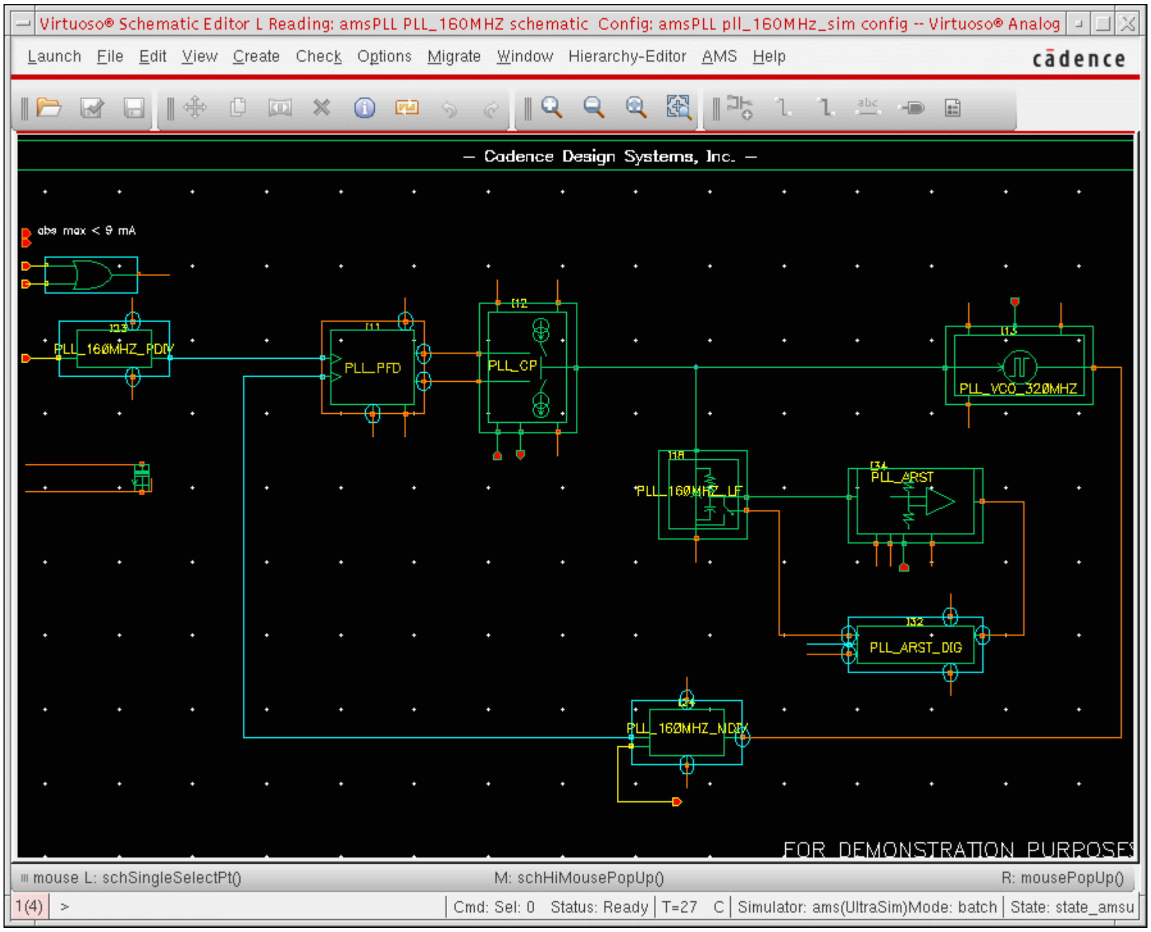
AMS Partition Display: amsPLL pll_16

Name	Probe Layer
☑ analog	y2 dg
☑ digital	y4 dg
☑ analog/mixed	y6 dg
☑ digital/mixed	y0 dg

   You will notice that this form is slightly different from the one for the UltraSimVerilog case (see Viewing Interface Elements on the Schematic on page 88).

4. Click *OK*.

On the schematic, mixed-signal items appear in orange and yellow.



Virtuoso® Schematic Editor L Reading: amsPLL PLL_160MHZ schematic Config: amsPLL pll_160MHz_sim config -- Virtuoso® Analog

Launch File Edit View Create Check Options Migrate Window Hierarchy-Editor AMS Help
cādence
– Cadence Design Systems, Inc. –
abs max < 9 mA
I13
PLL_160MHZ_PDIV
I11
PLL_PFD
I12
PLL_CP
I15
PLL_VCO_320MHZ
I18
PLL_160MHZ_LF
I34
PLL_ARST
I32
PLL_ARST_DIG
I14
PLL_160MHZ_NDIV
FOR DEMONSTRATION PURPOSES

mouse L: schSingleSelectPt() M: schHiMousePopUp() R: mousePopUp()

1(4) > Cmd: Sel: 0 Status: Ready T=27 C Simulator: ams(UltraSim)Mode: batch State: state_amsu

You can compare this display with the one you saw for UltraSimVerilog. Both examples have the same partitions. You can see the set up for connect rules and disciplines both from the *AMS* menu in the schematic window and by choosing *Setup – Connect Rules* in the Virtuoso® Analog Design Environment window.

Understanding Connect Rules and Disciplines in AMS Designer

The AMS Designer simulator uses disciplines, connect modules, and connect rules in place of A2D and D2A interface elements. A discipline denotes an object as analog or digital (with, for example, an `electrical` or `logic` discipline). When you connect objects of different disciplines, connect rules determine which connect modules to insert between the objects. The inserted connect modules convert signals to values that are appropriate for each discipline. You can modify connect rule parameters such as supply voltage and rise time in your connect modules to tailor conversion of your design.

Note: For more information about disciplines, connect rules, and connect modules, see “Mixed-Signal Aspects of Verilog-AMS” in the *Cadence Verilog-AMS Language Reference*.

Cadence provides sample connect rules in the following directory:

```
$AMSHOME/tools/affirma_ams/etc/connect_lib
```

The sample connect rules (CRs) here are built in and ready for use in the Virtuoso® Analog Design Environment (ADE). Built-in CRs work for a certain set of voltage supplies only (such as 1.8V, 3V, and 5V). You can modify the parameters to customize a built-in CR for your design needs. Advanced designers can write customized CRs and include them in the simulation.

For this example, the voltage supply is 2.5 V. We can customize the 3V built-in CR to fit our simulation.

Note: Cadence provides full-fast, full, mid, and basic built-in CRs. You can speed up the simulation for complicated designs using the full-fast CRs. For this example, we do not need a bidirectional CR; we can choose a simpler CR.

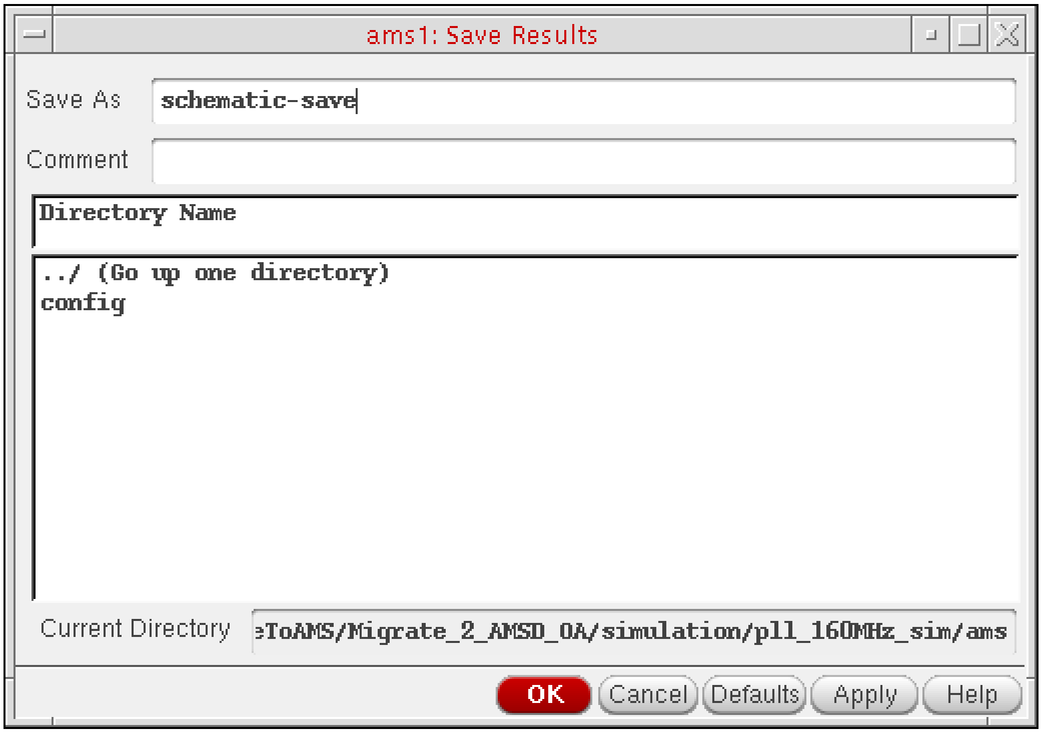
Simulating the Design Using the Spectre Solver

The AMS Designer simulator has two analog solvers: UltraSim and Spectre. To simulate the example design using the Spectre solver, you can do the following:

1. Save the results from the previous simulation as follows:

   a. In the Virtuoso® Analog Design Environment window, choose *Results – Save*.

      The Save Results form appears.



ams1: Save Results

Save As: schematic-save

Comment:

Directory Name

```
../ (Go up one directory)
config
```

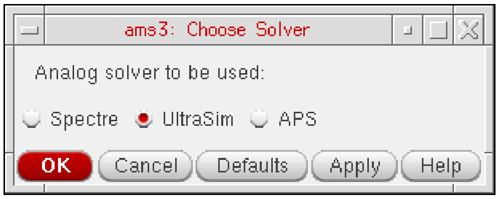
Current Directory: eToAMS/Migrate_2_AMSD_OA/simulation/pll_160MHz_sim/ams

   b. In the *Save As* field, type a name for your results.

   c. Click *OK*.

2. Choose *Simulation – Solver*.

The Choose Solver form appears.

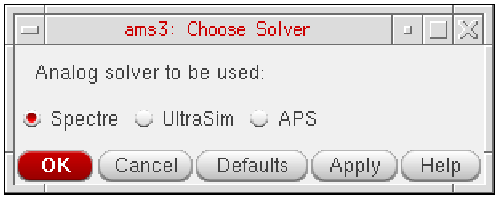


ams3: Choose Solver

Analog solver to be used:

○ Spectre ● UltraSim ○ APS

3. Select *Spectre* as the *Analog solver to be used*:



ams3: Choose Solver

Analog solver to be used:

● Spectre ○ UltraSim ○ APS

4. Click *OK*.
5. In the Virtuoso® Analog Design Environment window, choose *Simulation – Netlist and Run*.

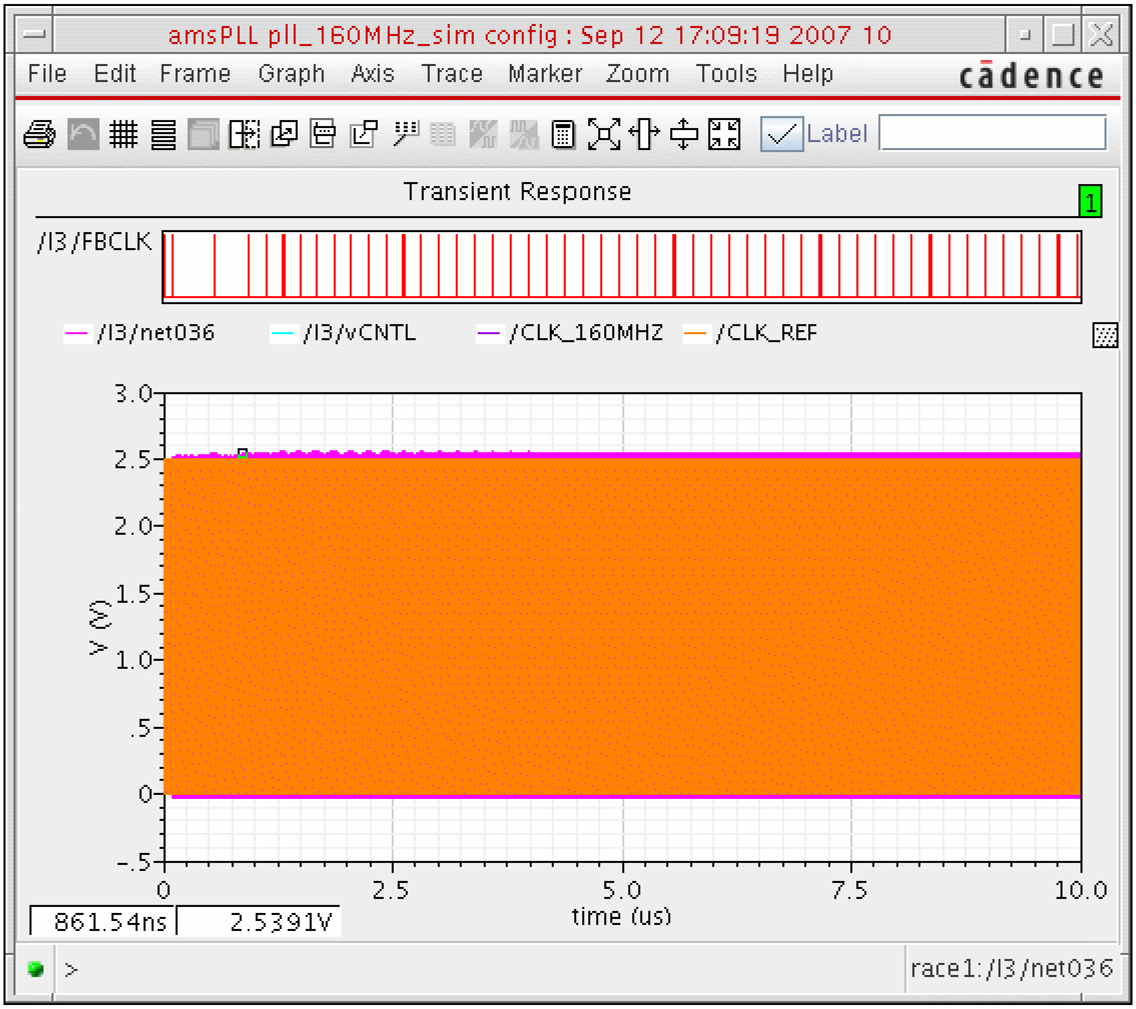
   Status appears in the upper left corner of the window. Simulation output information appears in the `ncverilog.log` file. The simulation time appears at the end of the file.

   ```
   Total time required for tran analysis `tran' was 3.11407 ks (51m  54.1s).
   ```

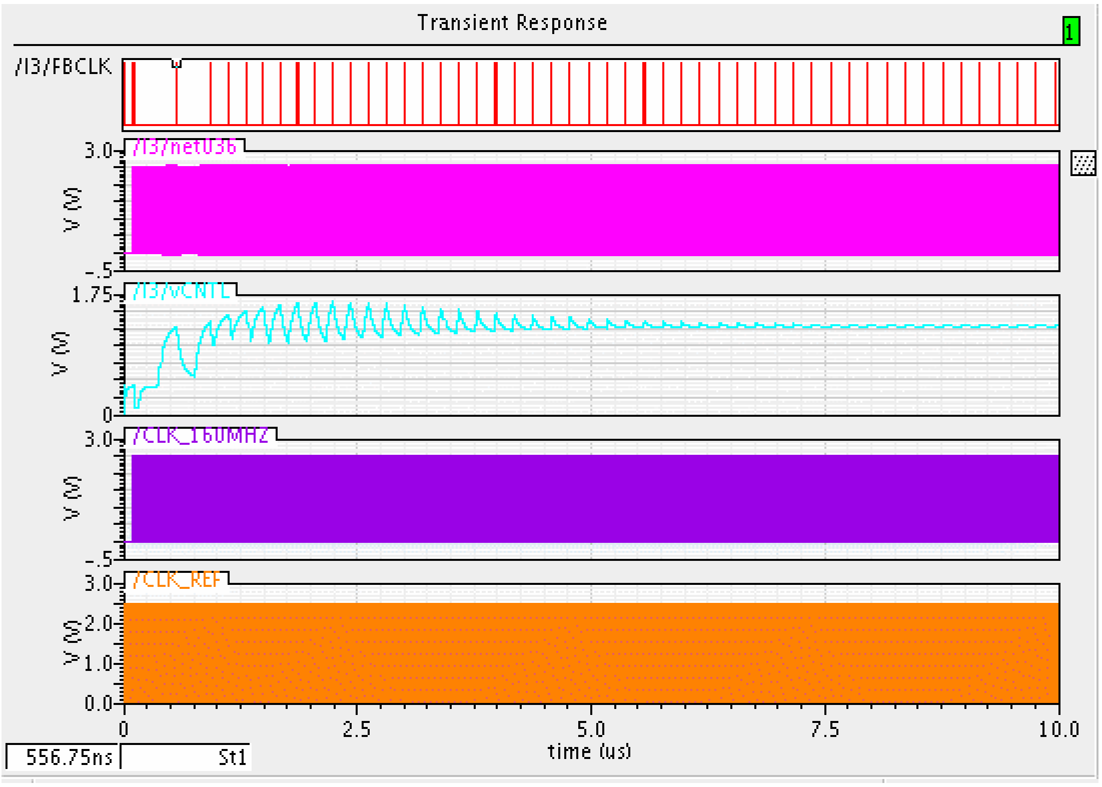
   You can compare these results to those from the simulation using AMS Designer with the UltraSim solver.

   Note: This simulation ran for 60 minutes on our Solaris machine with a 1.6 G CPU.

When the simulation finishes, a graph window appears.



6. In the graph window, choose *Axis – Strips*.



7. When you are finished viewing results, choose *File – Close* to close the graph window.

5

Migrating to the OSS Netlister

To support the `irun` command and to overcome obstacles of the cellview-based netlister, Cadence provides the OSS netlister. This tutorial example demonstrates how to migrate from using the cellview-based netlister to using the OSS netlister.

The AMS cellview-based netlister (the original netlister for AMS Designer) translates schematic cellviews into Verilog®-AMS netlists, for one cell at a time. The output of a successful netlisting run is one or more files named `verilog.vams`, each containing a valid Verilog-AMS module that corresponds to a schematic cellview, in the lib/cell/view for the cell. The cellview-based netlister requires that you have `ams` simInfo in your PDK library.

Cadence's Open Simulation System (OSS) netlister creates a single netlist of the entire design hierarchy in the netlist directory. The OSS netlister uses `spectre` simInfo. You do not need to add `ams` simInfo or convert PDKs (as you do if you use the cellview-based netlister). The Spectre and UltraSim circuit simulators use this netlister.

For more information about these netlisters, see "Netlisting" in the *Virtuoso® AMS Designer Environment User Guide*.

Important

> You must be using the following releases of Cadence software to run this tutorial example: IC 5.1.41 ISR#117 or later for ADE, and IUS 8.1 or later for the AMS Designer simulator.

For information about the tutorial design, including key directories, see "The Migration Example" on page 130.

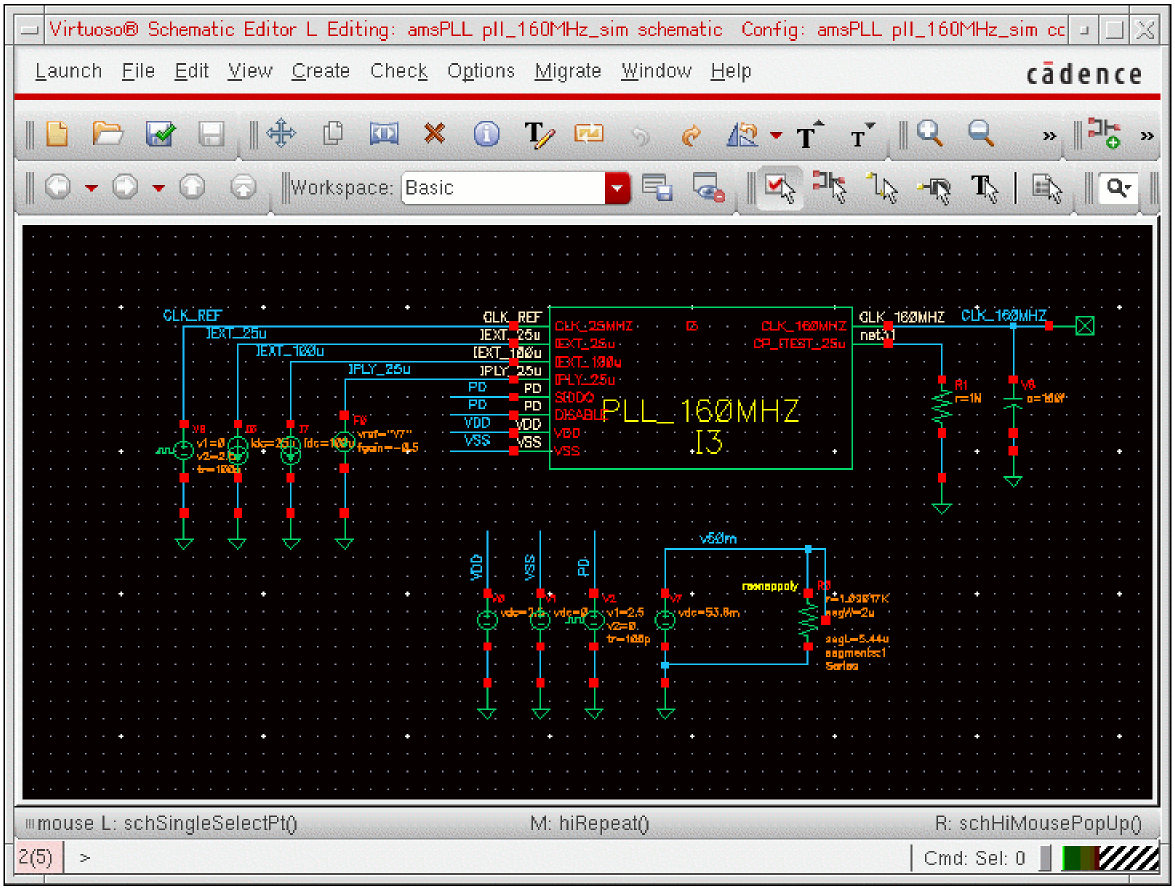
See the following topics for further information:

- Getting Started on page 131
- Selecting and Using the Cellview-Based Netlister on page 132
- Selecting and Using the OSS Netlister on page 134

See also "Important Considerations when Using the OSS Netlister" on page 9.

The Migration Example

This migration example consists of a schematic PLL design that contains Verilog language design units. The schematic contains the following analog components: a VCO, a phase frequency detector (PFD), a charge pump, and a loop filter. The two digital frequency dividers are RTL Verilog and VHDL modules.



Key directories for this tutorial example are:

Directory	Content
`gpdk090`	90 nm process design kit (PDK)
`models`	Model files in Spectre syntax
`amsPLL`	Library that contains the PLL blocks for the schematic

Directory	Content
`dig_source`	Directory containing the two behavioral Verilog and VHDL frequency dividers

Getting Started

Important

> Before starting this tutorial, see "Before You Begin" on page 13.

To begin, do the following in the `MigrateFromCBNToOSSN` directory:

1. Source the setup file:

   ```
   source SETUP
   ```

   The `SETUP` file sets the `TUT_DIR` environment variable to your current directory:

   ```
   setenv TUT_DIR `pwd`
   ```

2. Start Cadence software:

   ```
   virtuoso &
   ```

Selecting and Using the Cellview-Based Netlister

1. In the Virtuoso® command interpreter window (CIW), choose *Tools – ADE L – Simulation*.

   The Virtuoso Analog Design Environment window appears.

2. In ADE, choose *Setup – Design*.

   The Choosing Design form appears.

3. Select the following:

Field	**Selection**
Library Name	*amsPLL*
Cell Name	*pll_160MHZ_sim*
View Name	*config*

4. Click *OK*.
5. In ADE, choose *Setup – Simulator*.

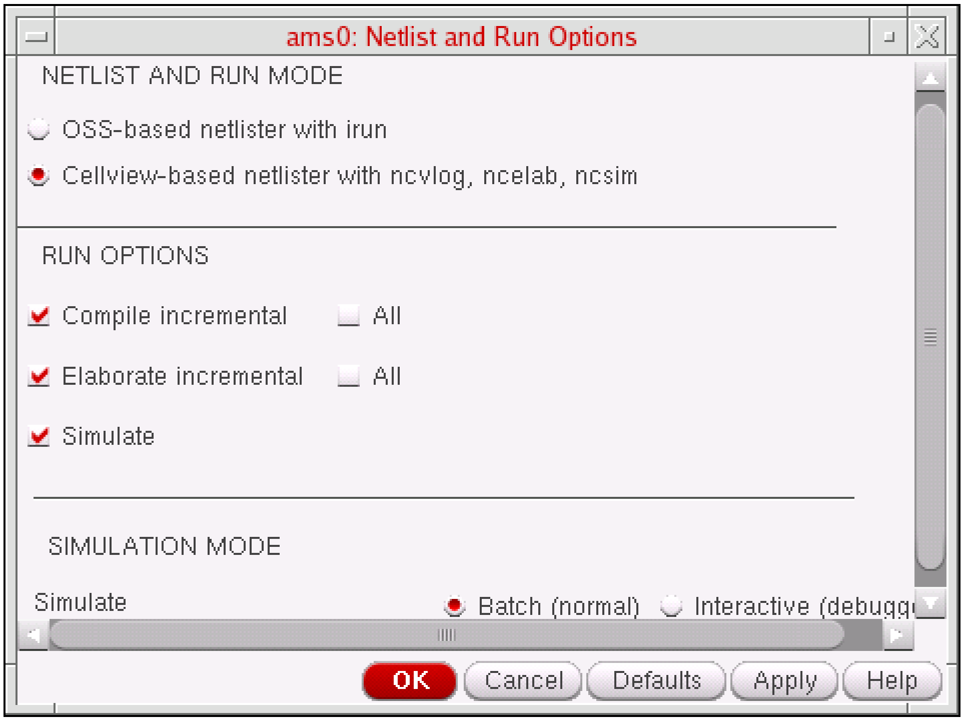
   The Choosing Simulator form appears.

6. Use the *Simulator* drop-down combo box to select *ams*.
7. Click *OK*.
8. In ADE, choose *Session – Load State*.

   The Loading State form appears.

9. In the *State Name* area, select *state_CB*.
10. Click *OK*.
11. In ADE, choose *Simulation – Netlist and Run Options*.

The Netlist and Run Options form appears.



ams0: Netlist and Run Options

NETLIST AND RUN MODE

○ OSS-based netlister with irun

● Cellview-based netlister with ncvlog, ncelab, ncsim

RUN OPTIONS

☑ Compile incremental □ All

☑ Elaborate incremental □ All

☑ Simulate

SIMULATION MODE

Simulate ● Batch (normal) ○ Interactive (debuggi

12. In the *NETLIST AND RUN MODE* section, select *Cellview-based netlister with ncvlog, ncelab, ncsim*.

The other choice is for the *OSS-based netlister with irun*. We will demonstrate this choice in “Selecting and Using the OSS Netlister” on page 134.

13. Click *OK*.

14. To run the simulation, either choose *Simulation – Netlist and Run*, or click the green Netlist and Run button.

After the simulation finishes, results appear in a waveform window.

15. When you are finished viewing results, close the waveform window.

Selecting and Using the OSS Netlister

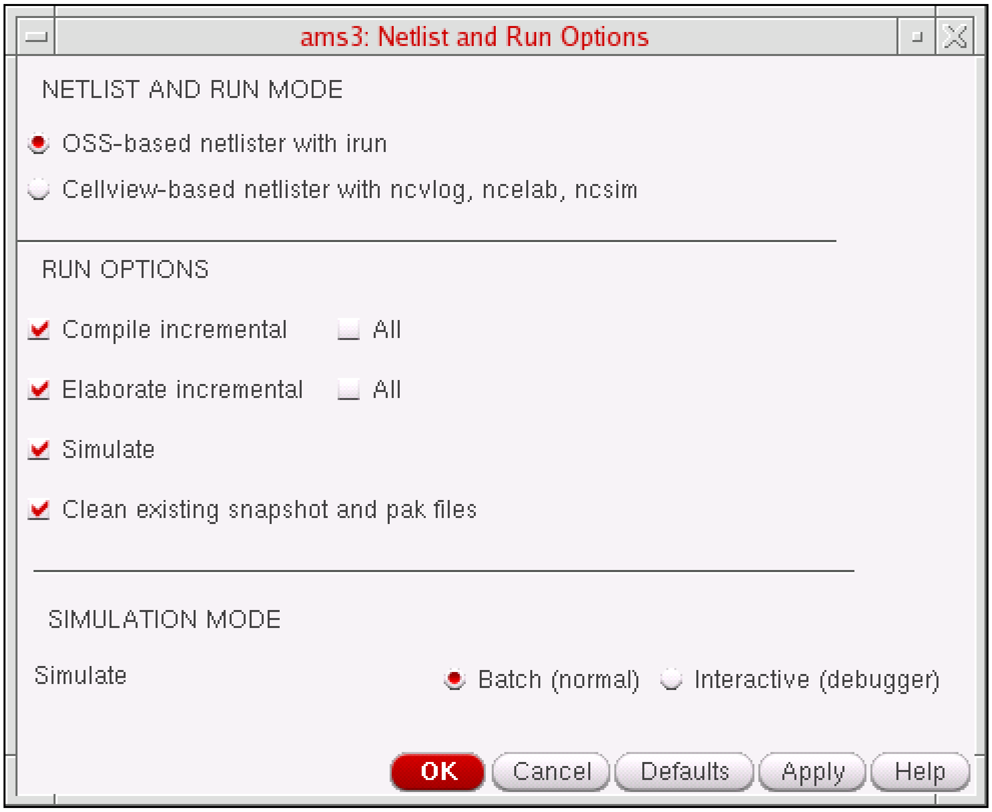
To select and use the OSS netlister, do the following:

1. In ADE, choose *Session – Load State*.

   The Loading State form appears.

2. In the *State Name* area, select *state_OSS*.
3. Click *OK*.
4. In ADE, choose *Simulation – Netlist and Run Options*.

   The Netlist and Run Options form appears.



ams3: Netlist and Run Options

NETLIST AND RUN MODE

● OSS-based netlister with irun
○ Cellview-based netlister with ncvlog, ncelab, ncsim

RUN OPTIONS

☑ Compile incremental □ All
☑ Elaborate incremental □ All
☑ Simulate
☑ Clean existing snapshot and pak files

SIMULATION MODE

Simulate ● Batch (normal) ○ Interactive (debugger)

5. In the *NETLIST AND RUN MODE* section, select *OSS-based netlister with irun*.

   The other choice is for the *Cellview-based netlister with ncvlog, ncelab, ncsim*. To see how to run the tutorial using this choice, see "Selecting and Using the Cellview-Based Netlister" on page 132.

6. Click *OK*.

7. In ADE, choose *Simulation – Netlist – Create*.

An error appears in the CIW indicating

```
ERROR (191) : AMS netlisting has failed.
```

You can scroll up to see the problem:

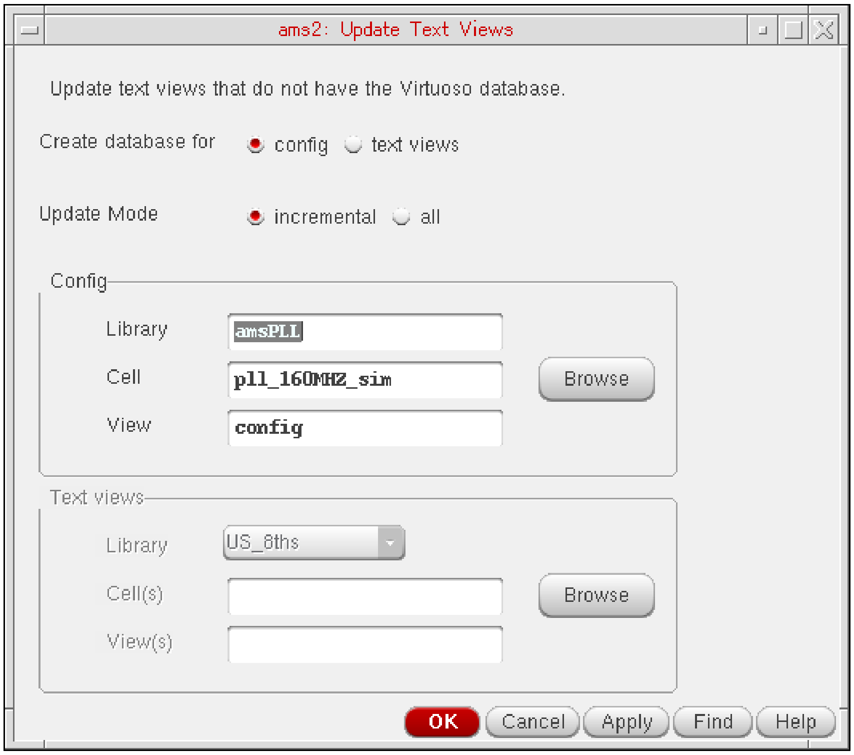
```
ERROR (OSSHNL-116): Unable to descend into any of the views defined in the view
list, 'vhdl spectre spice verilog verilogams behavioral functional system
schematic veriloga vhdl vhdlams symbol', for the instance 'INVP1' in cell
'PLL_VCO_320MHZ'. Either add one of these views to the library 'invLib', cell
'inv_VHDL' or modify the view list to contain an existing view.
```

The `inv_VHDL` cell in the `invLib` library has a VHDL text view (`vhdl`), but this text view does not have a Virtuoso® database.

To create the Virtuoso database for text views in the design, do the following:

1. In ADE, choose *Tools – Update Text Views*.

The Update Text Views form appears.



ams2: Update Text Views

Update text views that do not have the Virtuoso database.

Create database for ● config ○ text views

Update Mode ● incremental ○ all

Config

Library: amsPLL

Cell: pll_160MHZ_sim

View: config

Text views

Library: US_8ths

Cell(s):

View(s):

The choices for *Create database for* are:

- ❑ *config* — takes the lib-cell-view of the config, walks through the design, and creates a Virtuoso database for all text views that do not have one. This is the default choice.
- ❑ *text views* — takes the specified text view (such as `verilogams` or `vhdl`) and creates a Virtuoso database for it.

2. Click *OK*.

The program walks through the design and creates a Virtuoso database for all text views that do not have one.

3. In ADE, choose *Simulation – Netlist – Create*.

The program creates a netlist successfully.

To simulate using the OSS netlist, do the following:

1. In ADE, either choose *Simulation – Netlist and Run*, or click the green Netlist and Run button.

After the simulation finishes, results appear in a waveform window.

2. When you are finished viewing results, close the waveform window.

6

Using Inherited Connections for Multiple Power Supply Design

When creating a mixed-signal design with multiple power supplies, you need to share certain information between the analog and digital circuitry, such as

- What power value represents logic `1`?
- What voltage threshold triggers conversion of an analog signal from logic `0` to logic `1` and from logic `1` to logic `0`?

Using the Virtuoso® Schematic Editor, you can add inherited connection attributes and CDF `netSet` properties to the schematic to create special global signals and to override their names selectively in a design hierarchy. The elaborator automatically inserts inherited connection connect modules that have appropriate power supply values. (See also "Adding netSet Properties to Create an Inherited Connection" in the *Virtuoso Schematic Editor L User Guide*.)

Important

Before starting this tutorial, see "Before You Begin" on page 13.

See the following topics for tutorial details:

- The Tutorial Example on page 139
- Adding netSet Properties on page 141
- Verifying the Setup in ADE on page 144
- Using SimVision to Browse the Design Source on page 148
- Using SimVision to Investigate AICMs on page 151
- Using SimVision to Verify Simulation Results on page 153

See also "Designing with Multiple Power Supplies" in the *Virtuoso® AMS Designer Simulator User Guide*.

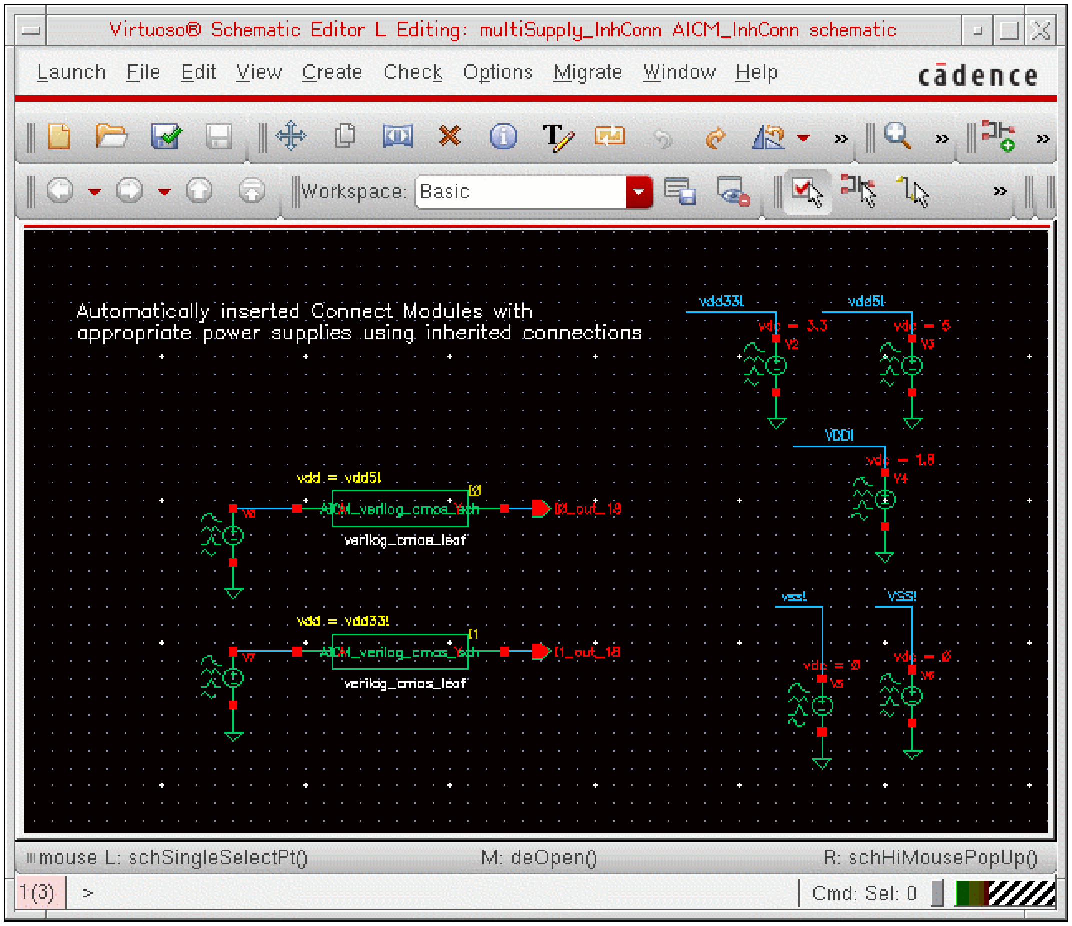
The Tutorial Example

This tutorial example features a simple schematic buffer design that has three power supplies in a mixed-signal circuit. The three power supplies are `VDD!=1.8V`, `vdd33!=3.3V`, and `vdd5!=5V`. `VDD!=1.8V` is the global power supply, instance `I0` requires `vdd5!=5V`, and instance I1 requires `vdd33!=3.3V`.

To open the schematic for the testbench, do the following in the `MultiPwerInhConn` directory:

1. In the CIW, choose *File – Open*.
2. On the Open File form, specify *multiSupply_InhConn*, *AICM_InhConn*, *schematic*.
3. Click *OK*.

   The testbench schematic appears in a Virtuoso® Schematic Editor window:


Virtuoso® Schematic Editor L Editing: multiSupply_InhConn AICM_InhConn schematic
Launch File Edit View Create Check Options Migrate Window Help
cādence
Workspace: Basic

Automatically inserted Connect Modules with appropriate power supplies using inherited connections

mouse L: schSingleSelectPt() M: deOpen() R: schHiMousePopUp()

1(3) > Cmd: Sel: 0

Instances `I0` and `I1` have `netSet` properties. The global value `vdd5!=5V` overrides the `vdd` net expression in `I0`, and the global value `vdd33!=3.3V` overrides the net expression in `I1`.

Adding netSet Properties

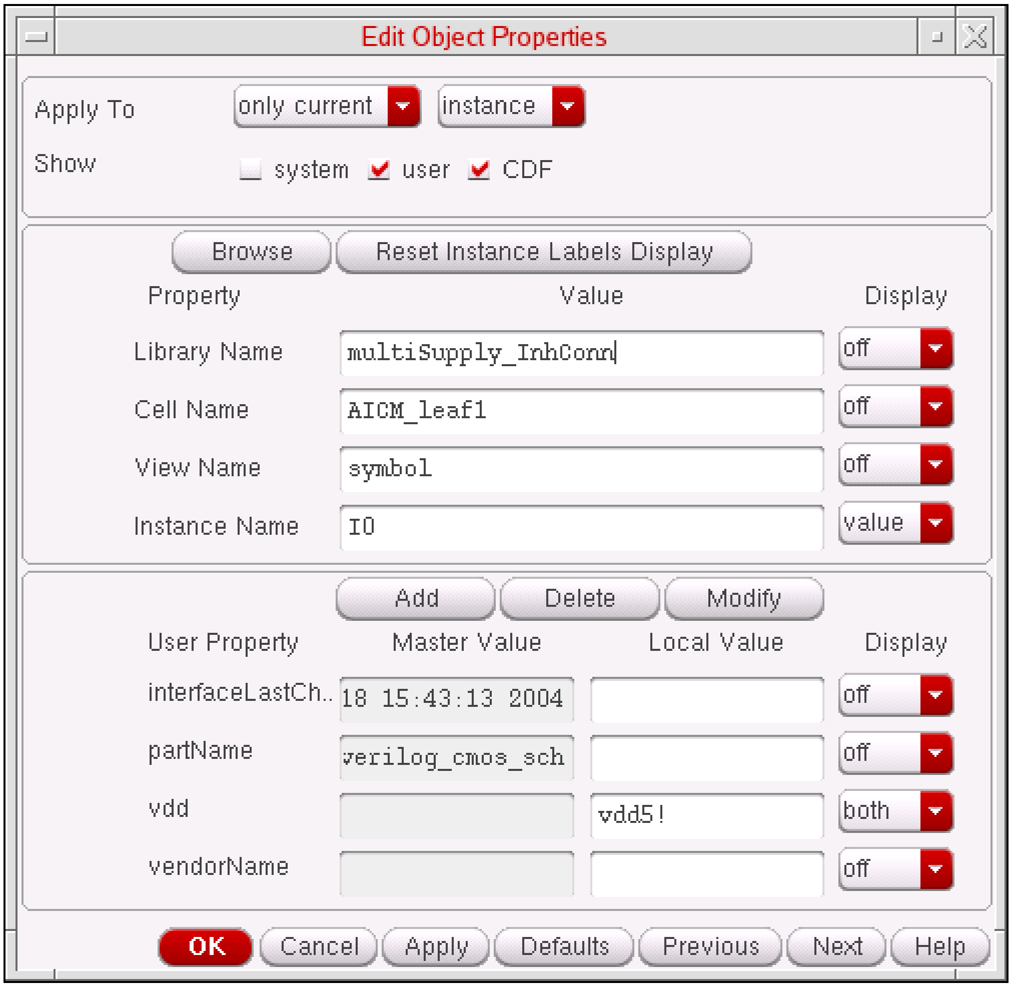
Note: We have already added `netSet` properties to the tutorial example.

To add a `netSet` property, do the following:

1. Select the instance (for example, `I0` or `I1` or `I0.I28`).
2. Type `q`.

   The Edit Object Properties form appears.

   Note: Typing `q` is a shortcut for choosing *Edit – Properties – Objects*.



Edit Object Properties

Apply To: only current, instance

Show: □ system ☑ user ☑ CDF

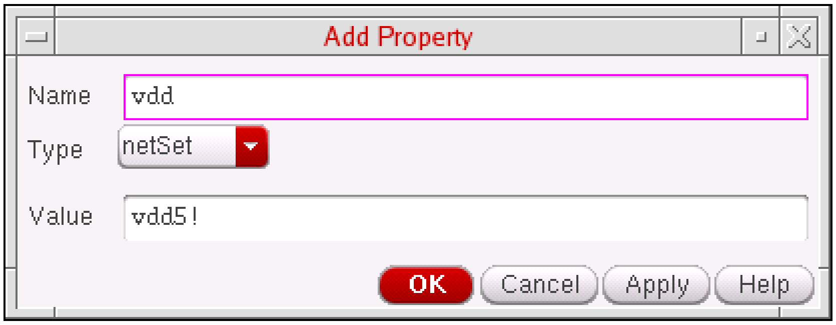
Property	Value	Display
Library Name	multiSupply_InhConn	off
Cell Name	AICM_leaf1	off
View Name	symbol	off
Instance Name	I0	value

User Property	Master Value	Local Value	Display
interfaceLastCh..	18 15:43:13 2004		off
partName	verilog_cmos_sch		off
vdd		vdd5!	both
vendorName			off

3. Click `Add`.

   The Add Properties form appears.

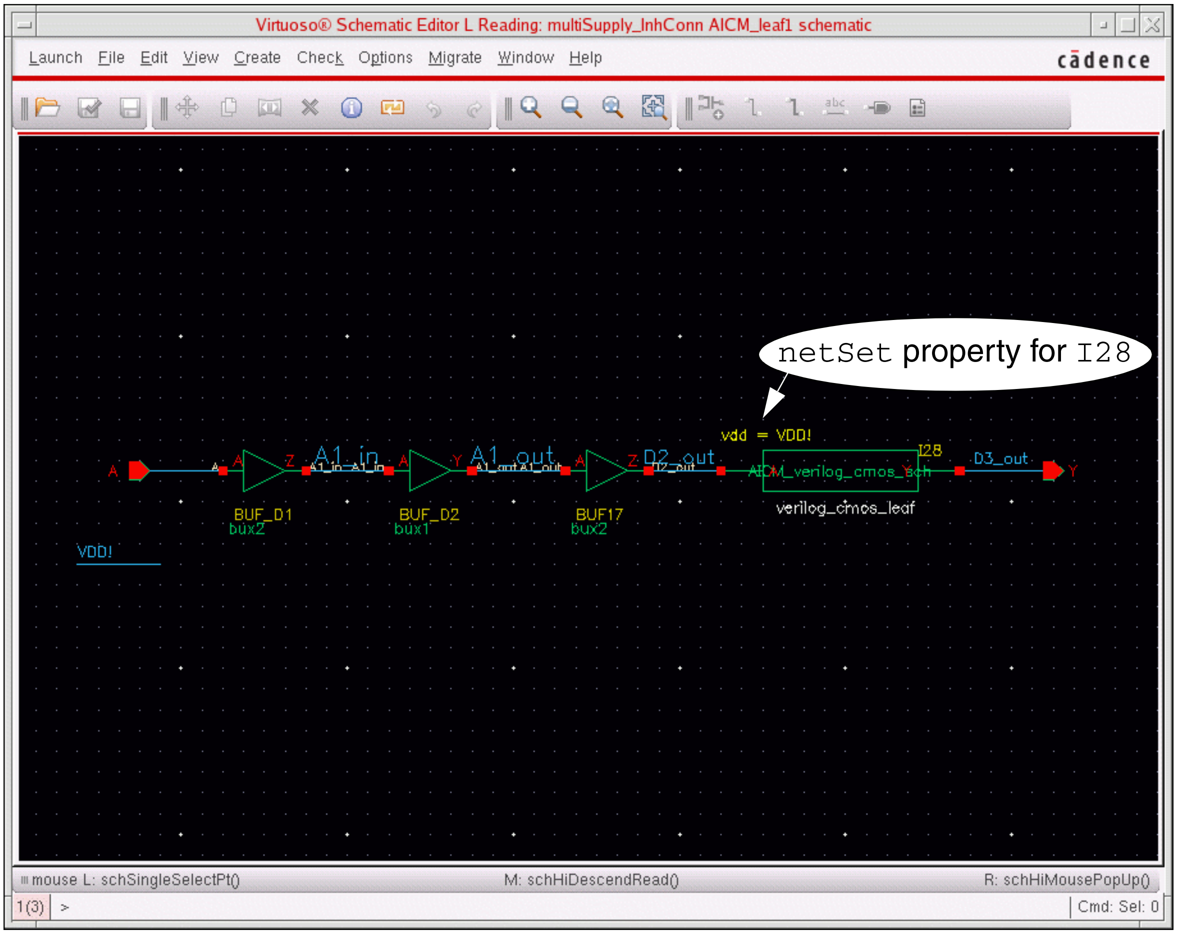
4. Using the *Type* drop-down combo box, select *netSet*.



Name: vdd
Type: netSet
Value: vdd5!

5. In the *Name* field, type the name of the `netSet` property, such as `vdd`.
6. In the *Value* field, type a value for the `netSet` property, such as `vdd5!`.
7. Click *OK*.

Because the `vdd` net expression in `I0` takes on the global value `vdd5!=5V`, all automatically-inserted connect modules (AICMs) at this level take on this value as well. Notice, however, that instance `I0.I128` has a `netSet` property such that `VDD!=1.8V`, so AICMs in `I28` inherit the value 1.8V for `vdd`.



`netSet` property for `I28`

Verifying the Setup in ADE

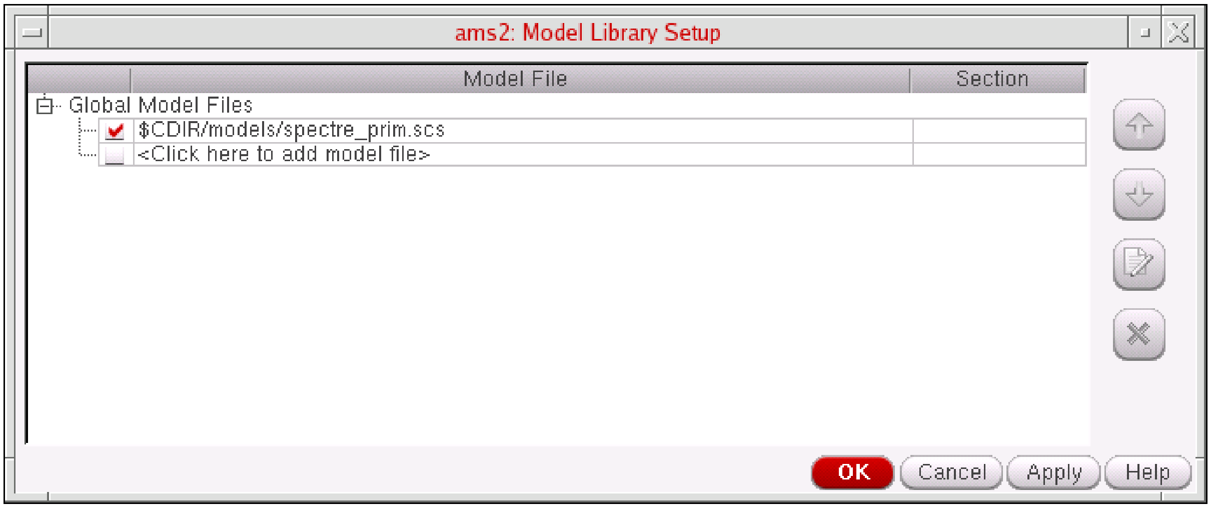
To verify the setup in ADE, do the following:

1. In the schematic editor, choose *Launch – ADE L*.
2. In ADE, choose *Setup – Design*.
3. From the *View Name* drop-down combo box on the Choosing Design form, select *config*.
4. Click *OK*.
5. In ADE, choose *Session – Load State*.
6. In the *State Name* area on the Loading State form, select *state_ams*.
7. Click *OK*.
8. In ADE, choose *Simulation – Netlist and Run Options*.

   Verify that the "NETLIST AND RUN MODE" is *OSS-based netlister with irun*. Verify that the "SIMULATION MODE" is *Interactive (debugger)*.

   We will use SimVision to help explain how the design netlist containing inherited connections works with the connect rules and connect modules.
9. Click *OK* or *Cancel*.
10. In ADE, choose *Setup – Model Libraries*.

    Verify that *$CDIR/models/spectre_prim.scs* is in the *Global Model Files* list. Optionally, click the Edit/View icon to view the device model file contents.

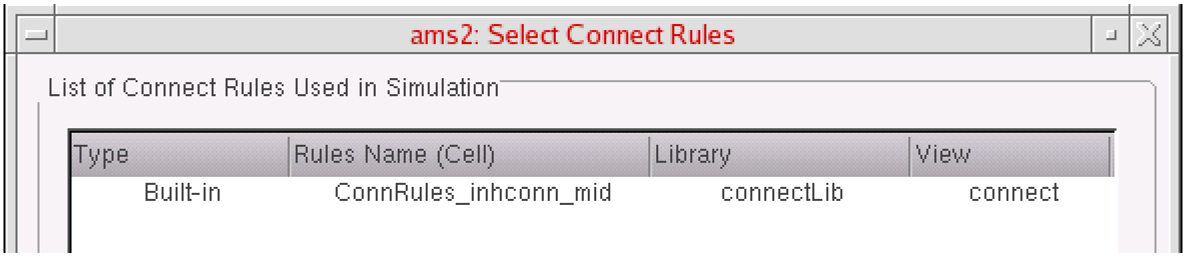


Model File	Section
Global Model Files	
☑ $CDIR/models/spectre_prim.scs	
□ <Click here to add model file>	

11. Click *OK* or *Cancel*.

12. In ADE, choose *Setup – Connect Rules*.

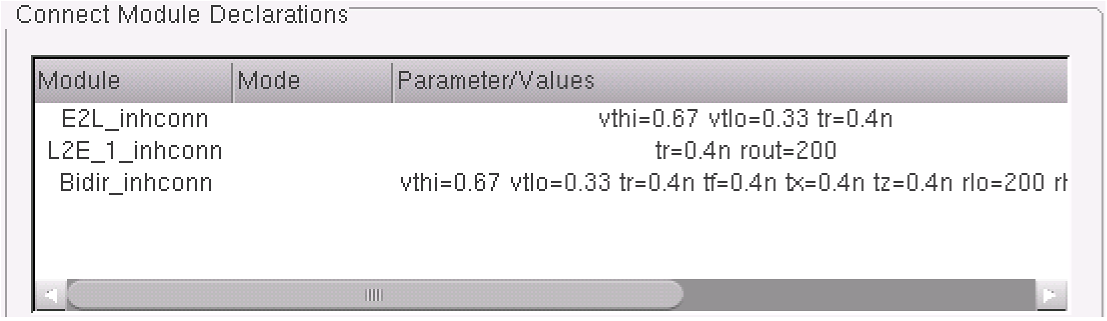
Verify that *Built-in ConnRules_inhconn_mid* appears in the *List of Connect Rules Used in Simulation*.



List of Connect Rules Used in Simulation

Type	Rules Name (Cell)	Library	View
Built-in	ConnRules_inhconn_mid	connectLib	connect

13. In the *List of Connect Rules Used in Simulation*, select *Built-in ConnRules_inhconn_mid* and click *Customize*.

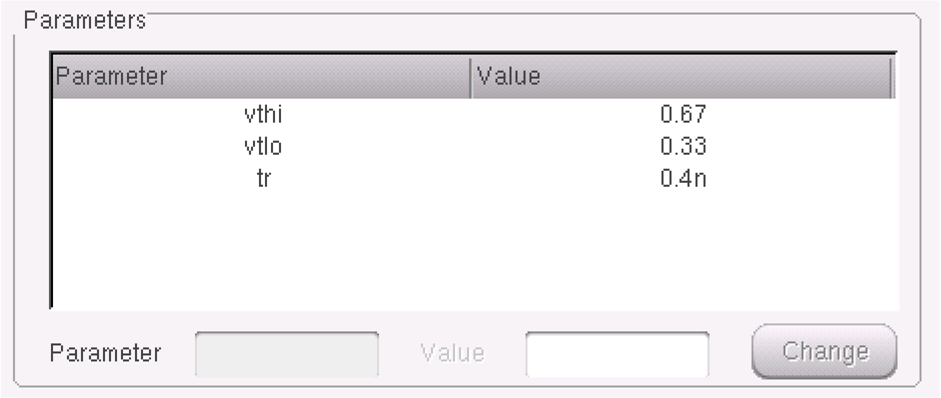


Connect Module Declarations

Module	Mode	Parameter/Values
E2L_inhconn		vthi=0.67 vtlo=0.33 tr=0.4n
L2E_1_inhconn		tr=0.4n rout=200
Bidir_inhconn		vthi=0.67 vtlo=0.33 tr=0.4n tf=0.4n tx=0.4n tz=0.4n rlo=200 rl

14. In the *Connect Module Declarations* list box, select *E2L_inhconn*.

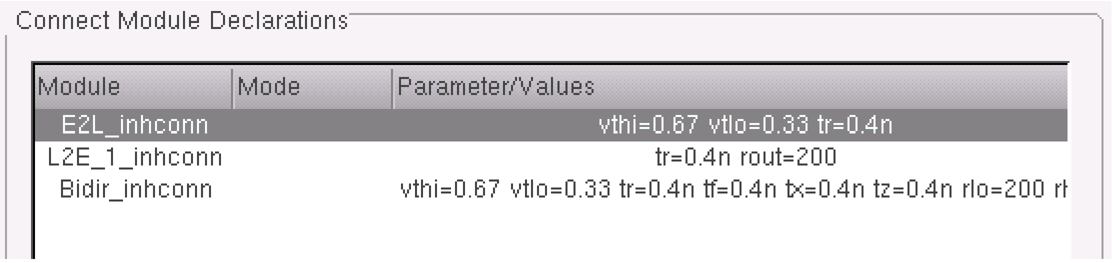
The connect module parameters appear in the *Parameters* group box on the form.



Parameters

Parameter	Value
vthi	0.67
vtlo	0.33
tr	0.4n

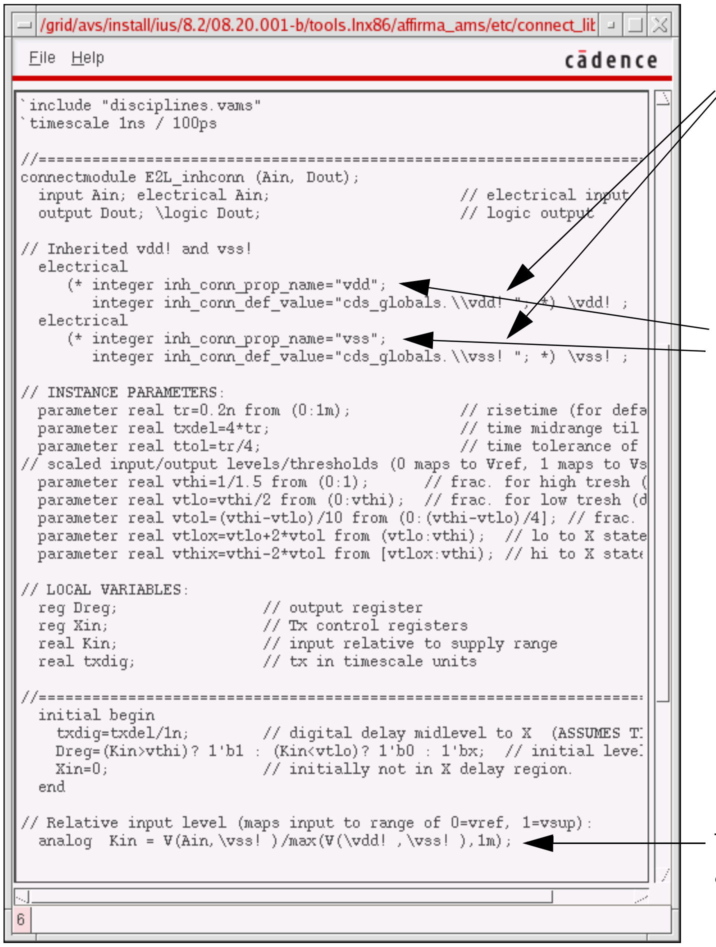
15. In the *Connect Module Declarations* group box, click *View connect module*.



Connect Module Declarations

Module	Mode	Parameter/Values
E2L_inhconn		vthi=0.67 vtlo=0.33 tr=0.4n
L2E_1_inhconn		tr=0.4n rout=200
Bidir_inhconn		vthi=0.67 vtlo=0.33 tr=0.4n tf=0.4n tx=0.4n tz=0.4n rlo=200 rh

16. Scroll down past the initial comment lines in the Verilog-AMS file to view the following:


/grid/avs/install/ius/8.2/08.20.001-b/tools.lnx86/affirma_ams/etc/connect_lit
File Help
cādence
6

```
`include "disciplines.vams"
`timescale 1ns / 100ps

//======================================================================
connectmodule E2L_inhconn (Ain, Dout);
  input Ain; electrical Ain;                         // electrical input
  output Dout; \logic Dout;                          // logic output

// Inherited vdd! and vss!
  electrical
     (* integer inh_conn_prop_name="vdd";
        integer inh_conn_def_value="cds_globals.\\vdd! "; *) \vdd! ;
  electrical
     (* integer inh_conn_prop_name="vss";
        integer inh_conn_def_value="cds_globals.\\vss! "; *) \vss! ;

// INSTANCE PARAMETERS:
  parameter real tr=0.2n from (0:1m);                // risetime (for defa
  parameter real txdel=4*tr;                         // time midrange til
  parameter real ttol=tr/4;                          // time tolerance of
// scaled input/output levels/thresholds (0 maps to Vref, 1 maps to Vs
  parameter real vthi=1/1.5 from (0:1);      // frac. for high tresh (
  parameter real vtlo=vthi/2 from (0:vthi);  // frac. for low tresh (d
  parameter real vtol=(vthi-vtlo)/10 from (0:(vthi-vtlo)/4]; // frac.
  parameter real vtlox=vtlo+2*vtol from (vtlo:vthi);  // lo to X state
  parameter real vthix=vthi-2*vtol from [vtlox:vthi); // hi to X state

// LOCAL VARIABLES:
  reg Dreg;                    // output register
  reg Xin;                     // Tx control registers
  real Kin;                    // input relative to supply range
  real txdig;                  // tx in timescale units

//======================================================================
  initial begin
    txdig=txdel/1n;            // digital delay midlevel to X  (ASSUMES T
    Dreg=(Kin>vthi)? 1'b1 : (Kin<vtlo)? 1'b0 : 1'bx;  // initial level
    Xin=0;                     // initially not in X delay region.
  end

// Relative input level (maps input to range of 0=vref, 1=vsup):
  analog  Kin = V(Ain,\vss! )/max(V(\vdd! ,\vss! ),1m);
```

Inherited connection attributes containing net expressions

Property names (vdd and vss) match netSet property names

vdd and vss net expression names appear throughout the built-in code

The built-in connect module uses net expressions to specify the inherited connection attributes. The property names are `vdd` and `vss`, which we use in the `netSet` properties in our design. The `vdd` and `vss` net expression names appear throughout the built-in code.

17. Close the viewing window.

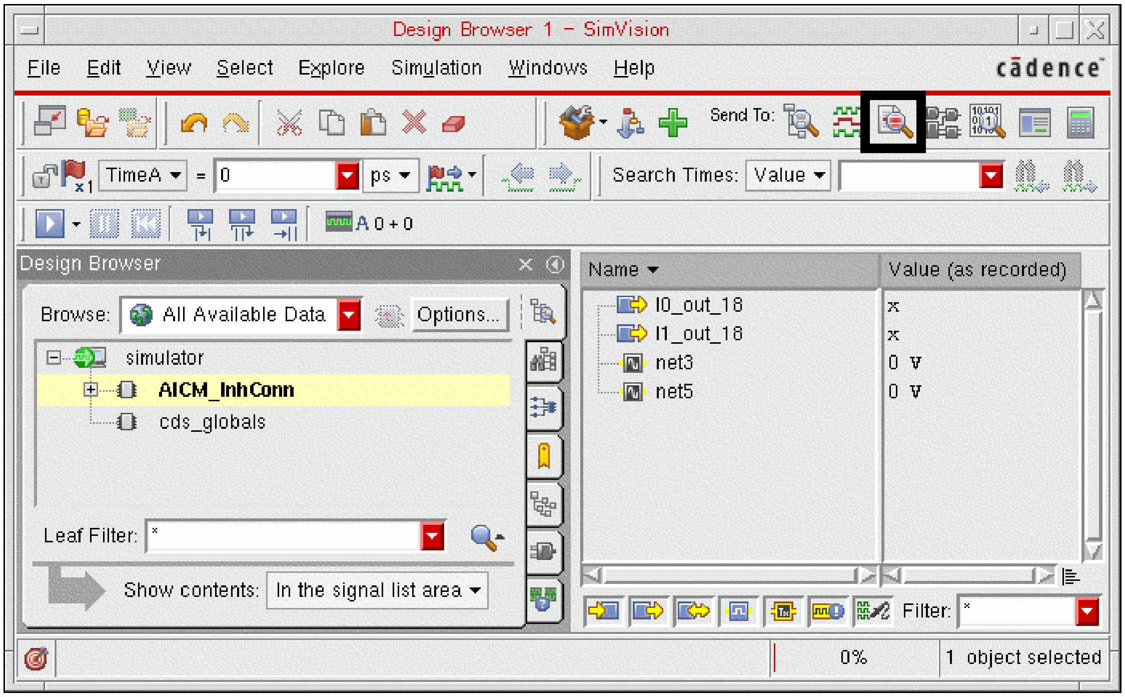
18. On the Select Connect Rules form, click *OK* or *Cancel*.

Using SimVision to Browse the Design Source

To use SimVision to browse the design, do the following:

1. In ADE, click the green Netlist and Run button.

   Several windows appear including the Console window and the Design Browser window.



TimeA = 0 ps

Search Times: Value

A 0 + 0

Browse: All Available Data

- simulator
  - AICM_InhConn
  - cds_globals

Leaf Filter: *

Show contents: In the signal list area

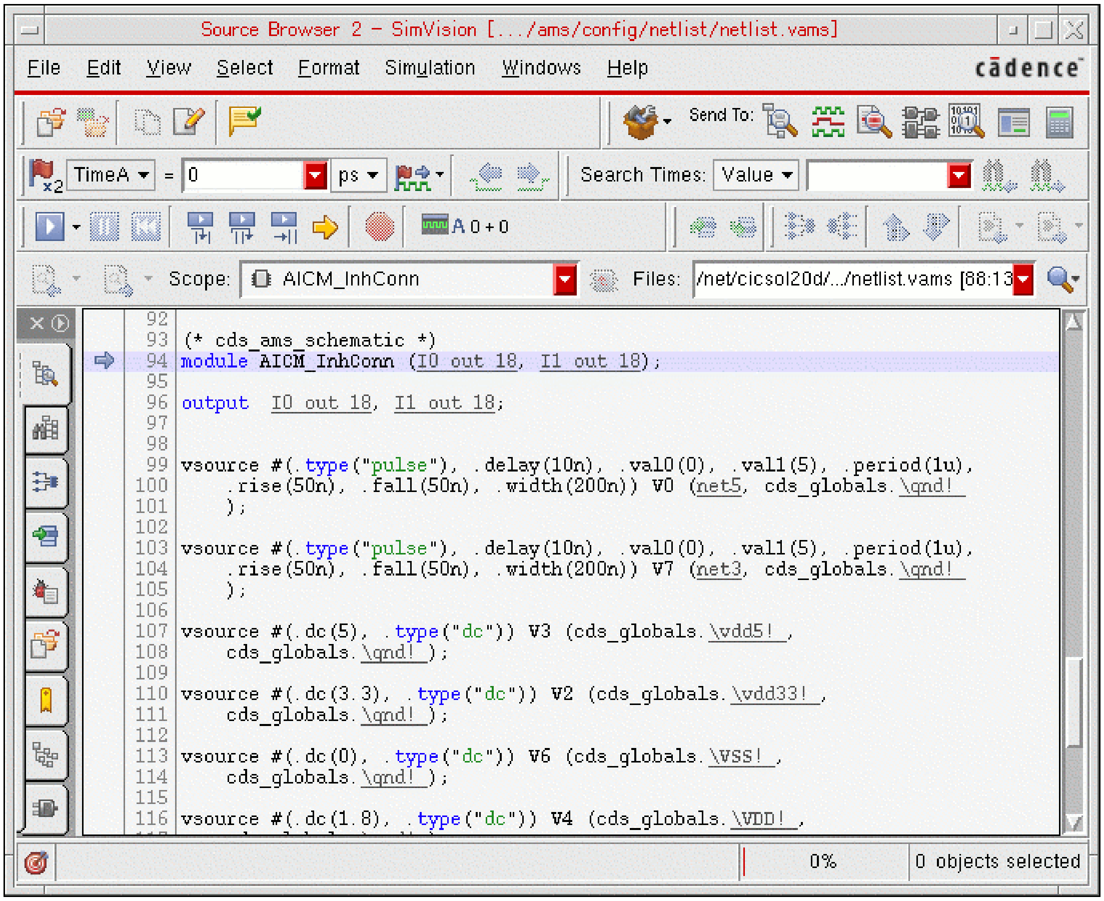
Name	Value (as recorded)
I0_out_18	x
I1_out_18	x
net3	0 V
net5	0 V

0%

1 object selected

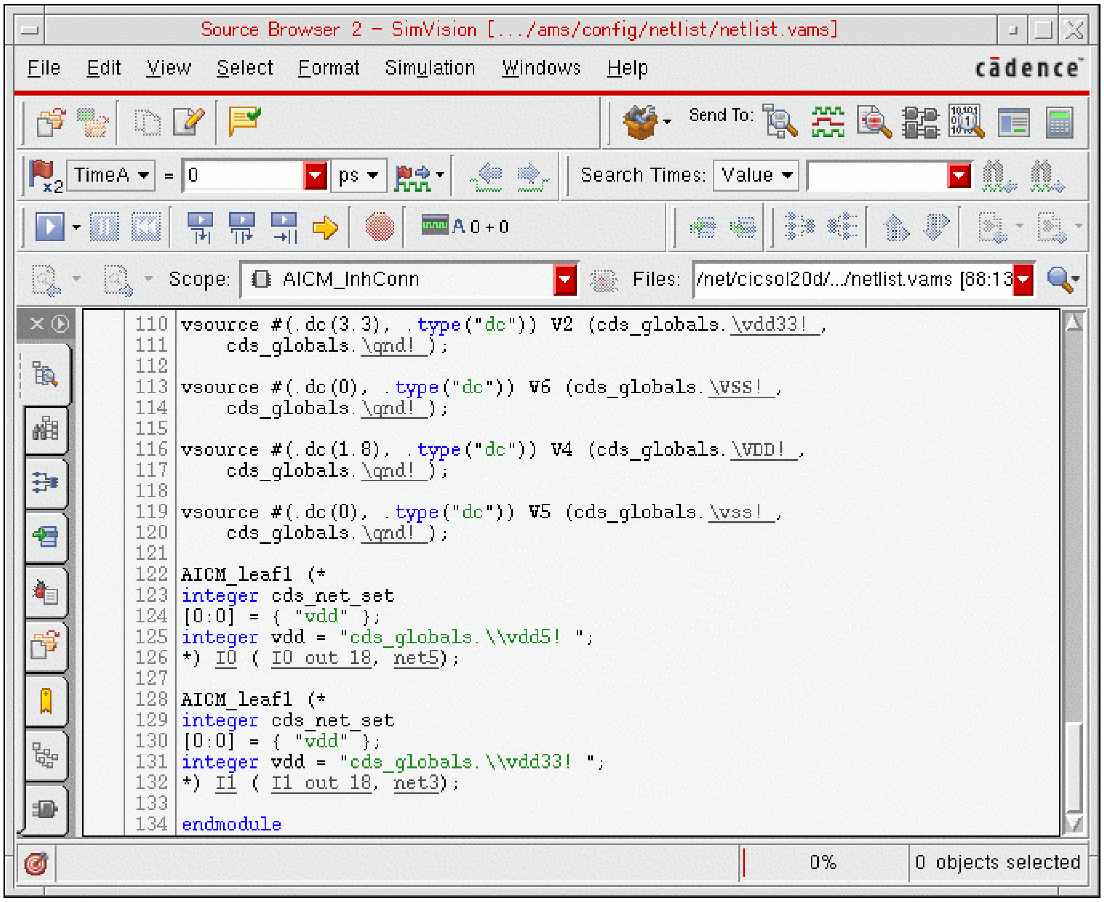
2. In the Design Browser window, click *AICM_InhConn*.

   The Source Browser window appears with an arrow to the left of the `module` statement for `AICM_InhConn`.



```
(* cds_ams_schematic *)
module AICM_InhConn (I0_out_18, I1_out_18);

output  I0_out_18, I1_out_18;


vsource #(.type("pulse"), .delay(10n), .val0(0), .val1(5), .period(1u),
    .rise(50n), .fall(50n), .width(200n)) V0 (net5, cds_globals.\gnd! 
    );

vsource #(.type("pulse"), .delay(10n), .val0(0), .val1(5), .period(1u),
    .rise(50n), .fall(50n), .width(200n)) V7 (net3, cds_globals.\gnd! 
    );

vsource #(.dc(5), .type("dc")) V3 (cds_globals.\vdd5! ,
    cds_globals.\gnd! );

vsource #(.dc(3.3), .type("dc")) V2 (cds_globals.\vdd33! ,
    cds_globals.\gnd! );

vsource #(.dc(0), .type("dc")) V6 (cds_globals.\VSS! ,
    cds_globals.\gnd! );

vsource #(.dc(1.8), .type("dc")) V4 (cds_globals.\VDD! ,
```

3. Scroll down to see the `AICM_leaf1` instances, `I0` and `I1`.


Source Browser 2 – SimVision [.../ams/config/netlist/netlist.vams]
File Edit View Select Format Simulation Windows Help
cādence
Send To:
TimeA
ps
Search Times: Value
A 0 + 0
0%
0 objects selected

Scope: AICM_InhConn

Files: /net/cicsol20d/.../netlist.vams [88:13

```
vsource #(.dc(3.3), .type("dc")) V2 (cds_globals.\vdd33! ,
     cds_globals.\gnd! );

vsource #(.dc(0), .type("dc")) V6 (cds_globals.\VSS! ,
     cds_globals.\gnd! );

vsource #(.dc(1.8), .type("dc")) V4 (cds_globals.\VDD! ,
     cds_globals.\gnd! );

vsource #(.dc(0), .type("dc")) V5 (cds_globals.\vss! ,
     cds_globals.\gnd! );

AICM_leaf1 (*
integer cds_net_set
[0:0] = { "vdd" };
integer vdd = "cds_globals.\\vdd5! ";
*) I0 ( I0_out_18, net5);

AICM_leaf1 (*
integer cds_net_set
[0:0] = { "vdd" };
integer vdd = "cds_globals.\\vdd33! ";
*) I1 ( I1_out_18, net3);

endmodule
```

The AMS netlister translates `netSet` properties into `cds_net_set` attributes in the Verilog-AMS netlist. Notice that `vdd` has the value `vdd5!` in instance `I0` and value `vdd33!` in instance `I1`.

Note: You can expand *AICM_InhConn* in the Design Browser and similarly view the `cds_net_set` attribute for `I0.I28` by clicking *I0 (AICM_leaf1)*.

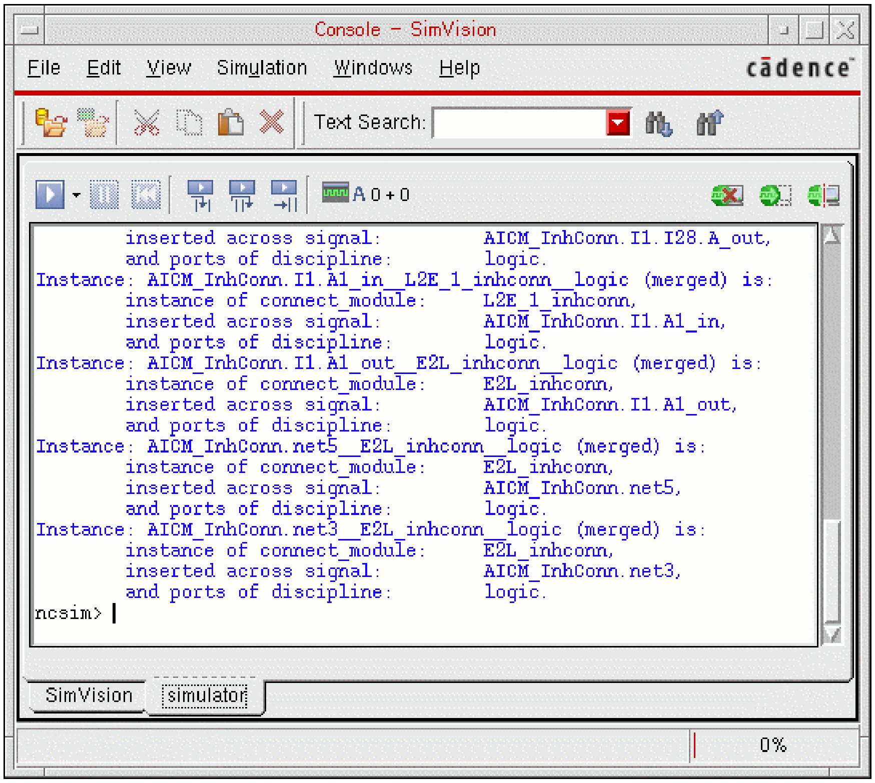
Using SimVision to Investigate AICMs

AICM stands for *automatically-inserted connect module*. The elaborator automatically inserts inherited connection connect modules that have appropriate power supply values wherever analog and digital design units connect. To use SimVision to investigate AICMs in this tutorial example, do the following:

1. In the Console window, type the following scope command:

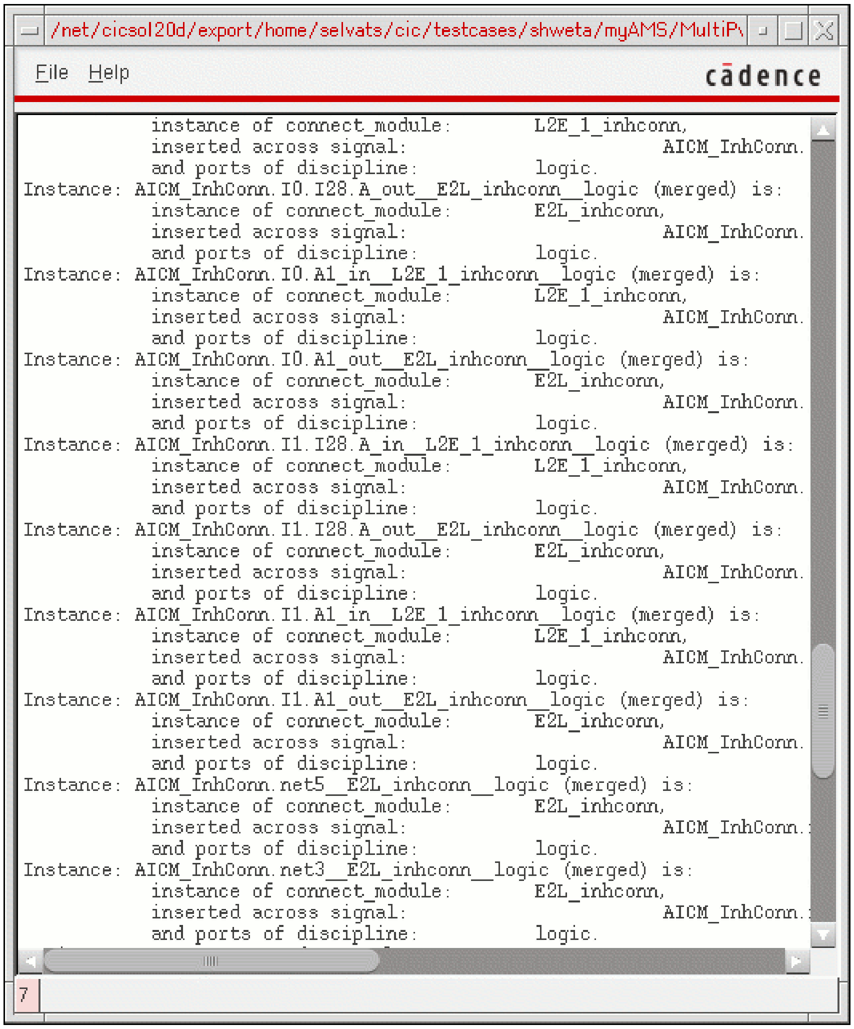
```
scope -aicm -recur -all
```

The program reports having inserted ten connect module instances, both to the Console window:



```
        inserted across signal:         AICM_InhConn.I1.I28.A_out,
        and ports of discipline:        logic.
Instance: AICM_InhConn.I1.A1_in__L2E_1_inhconn__logic (merged) is:
        instance of connect_module:     L2E_1_inhconn,
        inserted across signal:         AICM_InhConn.I1.A1_in,
        and ports of discipline:        logic.
Instance: AICM_InhConn.I1.A1_out__E2L_inhconn__logic (merged) is:
        instance of connect_module:     E2L_inhconn,
        inserted across signal:         AICM_InhConn.I1.A1_out,
        and ports of discipline:        logic.
Instance: AICM_InhConn.net5__E2L_inhconn__logic (merged) is:
        instance of connect_module:     E2L_inhconn,
        inserted across signal:         AICM_InhConn.net5,
        and ports of discipline:        logic.
Instance: AICM_InhConn.net3__E2L_inhconn__logic (merged) is:
        instance of connect_module:     E2L_inhconn,
        inserted across signal:         AICM_InhConn.net3,
        and ports of discipline:        logic.
ncsim>
```

and to the `irun.log` file:


/net/cicsol20d/export/home/selvats/cic/testcases/shweta/myAMS/MultiP\
File
Help
cādence
7

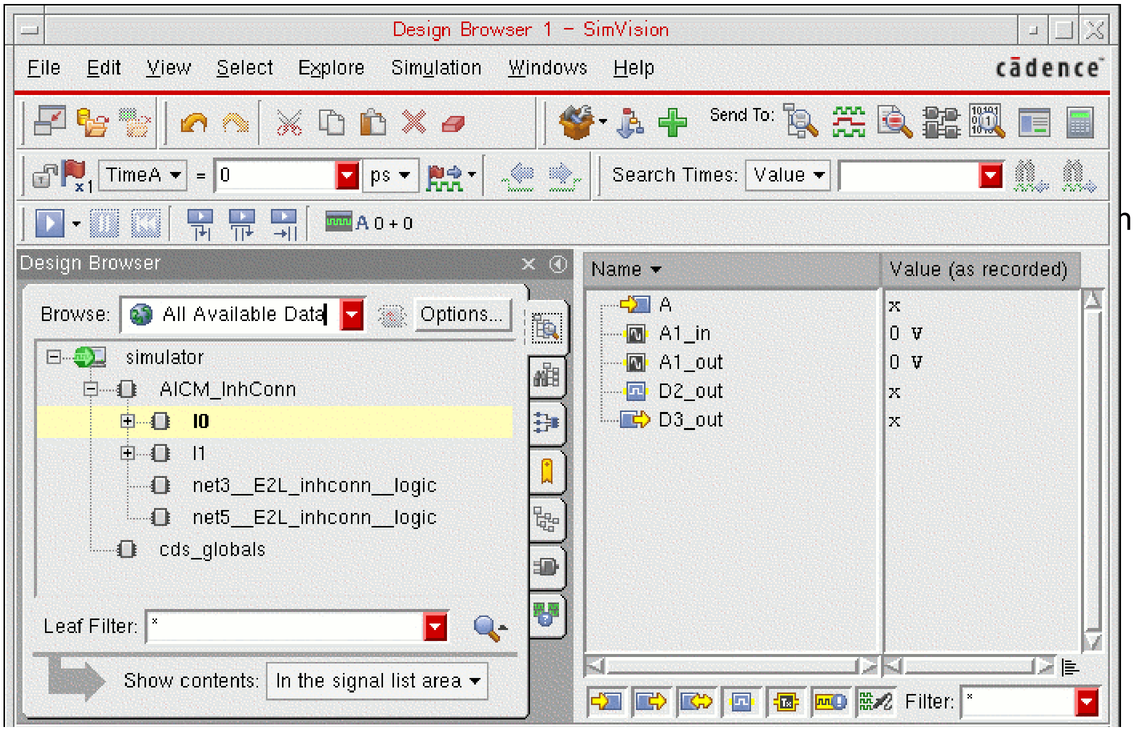
```
           instance of connect_module:       L2E_1_inhconn,
           inserted across signal:                      AICM_InhConn.
           and ports of discipline:          logic.
Instance: AICM_InhConn.I0.I28.A_out__E2L_inhconn__logic (merged) is:
           instance of connect_module:       E2L_inhconn,
           inserted across signal:                      AICM_InhConn.
           and ports of discipline:          logic.
Instance: AICM_InhConn.I0.A1_in__L2E_1_inhconn__logic (merged) is:
           instance of connect_module:       L2E_1_inhconn,
           inserted across signal:                      AICM_InhConn.
           and ports of discipline:          logic.
Instance: AICM_InhConn.I0.A1_out__E2L_inhconn__logic (merged) is:
           instance of connect_module:       E2L_inhconn,
           inserted across signal:                      AICM_InhConn.
           and ports of discipline:          logic.
Instance: AICM_InhConn.I1.I28.A_in__L2E_1_inhconn__logic (merged) is:
           instance of connect_module:       L2E_1_inhconn,
           inserted across signal:                      AICM_InhConn.
           and ports of discipline:          logic.
Instance: AICM_InhConn.I1.I28.A_out__E2L_inhconn__logic (merged) is:
           instance of connect_module:       E2L_inhconn,
           inserted across signal:                      AICM_InhConn.
           and ports of discipline:          logic.
Instance: AICM_InhConn.I1.A1_in__L2E_1_inhconn__logic (merged) is:
           instance of connect_module:       L2E_1_inhconn,
           inserted across signal:                      AICM_InhConn.
           and ports of discipline:          logic.
Instance: AICM_InhConn.I1.A1_out__E2L_inhconn__logic (merged) is:
           instance of connect_module:       E2L_inhconn,
           inserted across signal:                      AICM_InhConn.
           and ports of discipline:          logic.
Instance: AICM_InhConn.net5__E2L_inhconn__logic (merged) is:
           instance of connect_module:       E2L_inhconn,
           inserted across signal:                      AICM_InhConn.:
           and ports of discipline:          logic.
Instance: AICM_InhConn.net3__E2L_inhconn__logic (merged) is:
           instance of connect_module:       E2L_inhconn,
           inserted across signal:                      AICM_InhConn.:
           and ports of discipline:          logic.
```

Using SimVision to Verify Simulation Results

To use SimVision to verify simulation results, do the following:

1. In the Design Browser, click *I0 (AICM_leaf1)*.

   Signal names appear on the right side of the window.

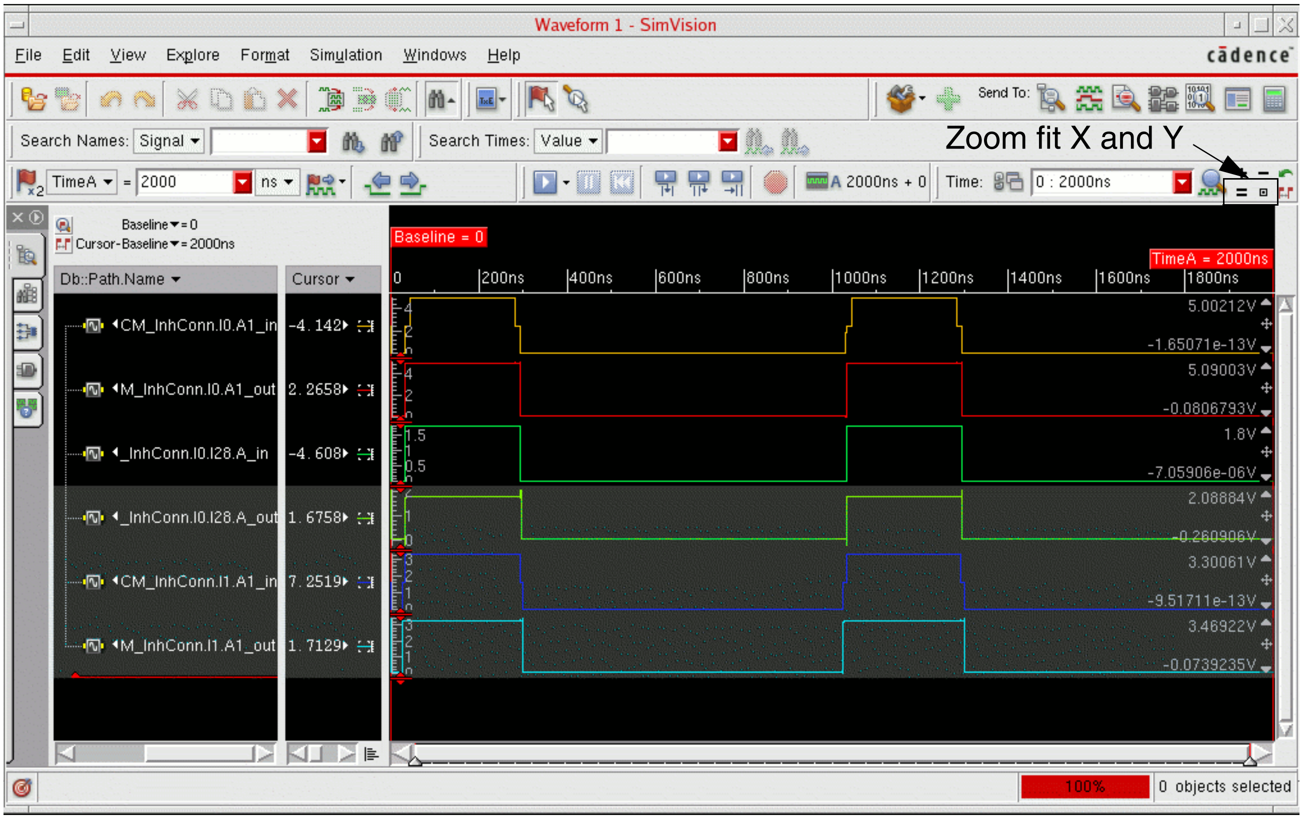


Name	Value (as recorded)
A	x
A1_in	0 V
A1_out	0 V
D2_out	x
D3_out	x

2. Shift-click to select both *A1_in* and *A1_out* at the same time.
3. Click the Send To Waveform icon.
4. In the Design Browser, click *I28 (AICM_leaf3)*.
5. Shift-click to select both *A_in* and *A_out* at the same time.
6. Click the Send To Waveform icon.
7. In the Design Browser, click *I28 (AICM_leaf3)*.
8. Shift-click to select both *A1_in* and *A1_out* at the same time.
9. Click the Send To Waveform icon.
10. Click the Play button.
11. When the simulation finishes, you can zoom to see the waveforms.

Tip

Be sure to zoom out fully to fit data along both the X and Y axes.



Zoom fit X and Y

The final simulation results show the “high” and “low” levels of the pulses, which are correct and allow proper translation from digital to analog and back again.

Tip

You can turn on cross-probing between the schematic and SimVision by choosing *Options – Editor* in the schematic editor and selecting the *on* radio button for *Cross Selection*. When you select a net on the schematic, SimVision highlights the corresponding signal waveform.

7

Using Digital Disciplines for Multiple Power Supply Design

A mixed-signal design with multiple power supplies has nets of different domains (continuous and discrete) crossing analog/digital boundaries. The program needs to know how to convert the signal values. For example, does a logic `1` digital net become a 2.5-volt analog net or a 1-volt analog net? Does an electrical value of 1 volt become a logic `0` or a logic `1`?

AMS Designer lets you use discipline-based connect rules and connect modules, and specify the discipline definition in the Virtuoso® Schematic Editor before simulating from the Virtuoso Analog Design Environment (ADE). This method has the following advantages:

- You do not need a global signal, neither in the connect modules nor in the design.
- You can have different ports with different supplies in the same cell with no leaf schematic under the cell.
- You can have more than one set of parameter values (for `vth`, `tr`, `tf`, and so on) in your connect rules file.

See the following topics for tutorial details:

- Getting Started on page 158
- Preparing Connect Modules, Connect Rules, and Discipline Definitions Using a Text Editor on page 159
- Preparing Connect Modules, Connect Rules, and Discipline Definitions Using the Select Connect Rules Form in the Environment on page 162
- Specifying the Default Digital Discipline in the Schematic Editor on page 182
- Selecting Connect Rules to Use on page 188

See also

- "Using Block-Based Discipline Resolution for Multiple Power Supply Design" in the *Virtuoso® AMS Designer Simulator User Guide*

- "Using Inherited Connections for Multiple Power Supply Design" on page 137

Getting Started

Important

Before starting this tutorial, see "Before You Begin" on page 13.

To begin, do the following in the `MultiPowerDis` directory:

1. Source the setup file:

   ```
   source sourceme
   ```

   The `sourceme` file sets the `CDIR` environment variable to your current directory:

   ```
   setenv CDIR `pwd`
   ```

2. Start Cadence software:

   ```
   virtuoso &
   ```

Preparing Connect Modules, Connect Rules, and Discipline Definitions Using a Text Editor

Cadence provides a set of connect modules (CMs) and connect rules in the IUS installation hierarchy:

your_IUS_install_dir/tools/affirma_ams/etc/connect_lib

You can create a custom file that contains connect rules, connect module parameters, and discipline definitions to suit your multiple power supply design requirements.

In AMS Designer, the default digital discipline is `logic` and the default analog discipline is `electrical`. You can create custom disciplines that contain information about the power supplies in your design. The elaborator uses block-based discipline resolution (BDR) to determine which part of the design has which power supply.

To create custom connect rules, connect modules, and discipline definitions, do the following:

1. Copy the template connect rules file (`ConnRules*.vams`) from the `connect_lib` directory in the installation hierarchy to your local area. For example:

   ```
   cp $AMSHOME/tools/affirma_ams/etc/connect_lib/ConnRules5.vams .
   ```

2. Save the file under another name, such as `ConnRules_discipline.vams`.

   Note: This is the name of the file we have created for this tutorial example.
   See `./myconnectLib/ConnRules_discipline.vams`.

3. Customize the connect module parameters (such as `Vsup`, `Vthi`, and so on) according to the what is required for your design.

   For this example, we type the parameter values directly in the connect module calls, such as `connect L2E #(` **`.vsup(1.2)`**, **`.vthi(0.6), .vtlo(0.3),`** and so on.

   Optionally, you can do this by specifying `` `define `` parameters for the different power supply values. For example, you might define three different sets of `Vsup`, `Vthi`, and `Vtlo` for 1.0V, 1.8V, and 3.3V power supplies:

   ```
   ----------------------------------
   // Parameter Customization
   `define Vsup1  1.0
   `define Vsup2  1.8
   `define Vsup3  3.3
   `define Vthi1  0.66
   `define Vthi2  1.2
   `define Vthi3  2.2
   `define Vtlo1  0.33
   `define Vtlo2  0.6
   `define Vtlo3  1.1
   -----------------------------------
   ```

4. Create customized connect rules using custom disciplines.

For our example, the `ConnRules_discipline.vams` connect rules file contains all the custom parameters, disciplines, and connect rules that the elaborator needs during discipline resolution to detect the discipline pairs (such as `logic_12` and `electrical`) and insert the proper connect module with the proper power supply.

For digital ports or nets that have the custom discipline `logic_12`, the program uses a 1.2-volt supply value such that the logic value of `1` on the digital side converts to 1.2 volts on the analog side. For analog-to-digital conversions, the program uses the `vthi` and `vtlo` threshold voltage values to determine whether to convert an analog signal to logic `1` or logic `0`. Because the `vthi` parameter for our example has a value of 0.6, the program converts any voltage greater than 0.6 volts to logic 1. Because the `vtlo` parameter for our example has a value of 0.3, the program converts any voltage less than 0.3 volts to logic `0`.

The default discipline is `logic`, which translates to 2.5 volts on the analog side.

Here are the contents of `my_connectLib/ConnRules_discipline.vams`, formatted for readability:

```
`include "disciplines.vams"

connectrules ConnRules_discipline;

// logic_12 discipline section, logic 1 -> 1.2V
// These rules are for connections between discipline logic_12 and electrical

connect L2E #(
              .vsup(1.2), .vthi(0.6), .vtlo(0.3), .tr(0.4n), .tf(0.4n),
              .tx(0.4n), .tz(0.4n), .rlo(200), .rhi(200), .rx(40), .rz(10M))
              input logic_12, output electrical;

connect E2L #(
              .vsup(1.2), .vthi(0.6), .vtlo(0.3), .tr(0.4n))
              input electrical, output logic_12;

connect Bidir #(
              .vsup(1.2), .vthi(0.6), .vtlo(0.3), .tr(0.4n), .tf(0.4n),
              .tx(0.4n), .tz(0.4n), .rlo(200), .rhi(200), .rx(40), .rz(10M))
              inout electrical, inout logic_12;

// default section, logic 1 -> 2.5V
// These rules are for connections between the default discipline (logic)
// and electrical

connect L2E #(
              .vsup(2.5), .vthi(1.75), .vtlo(0.75), .tr(1n), .tf(1n),
              .tx(1n), .tz(1n), .rlo(200), .rhi(200), .rx(40), .rz(10M)) ;

connect E2L #(
              .vsup(2.5), .vthi(1.75), .vtlo(0.75), .tr(1n)) ;

connect Bidir #(
              .vsup(2.5), .vthi(1.75), .vtlo(0.75), .tr(1n), .tf(1n),
              .tx(1n), .tz(1n), .rlo(200), .rhi(200), .rx(40), .rz(10M)) ;

endconnectrules
```

Note: The default discipline (`logic`) and our custom discipline (`logic_12`) are of `discrete` domain. The program uses the following discipline definitions:

```
discipline logic_12
    domain discrete;
enddiscipline

discipline logic
    domain discrete;
enddiscipline
```

Preparing Connect Modules, Connect Rules, and Discipline Definitions Using the Select Connect Rules Form in the Environment

To prepare connect modules, connect rules, and discipline definitions using the Select Connect Rules form in the environment, do the following:

1. In the CIW, choose *Tools – ADE L – Simulation*.

   The Virtuoso® Analog Design Environment window appears.

2. Choose *Setup – Design*.

   The Choosing Design form appears.

3. Select the following:

Form Field	**Value**
Library Name	*worklib*
Cell Name	*foo_top*
View Name	*config*

4. Click *OK*.
5. Choose *Setup – Simulator*.
6. Verify that *ams* appears in the *Simulator* field.

   The *Project Directory* is `./simulation`.

7. Click *OK*.
8. Choose *Setup – Connect Rules*.

   The Select Connect Rules form appears. *Built-in* and *ConnRules_18V_full_fast* appears in the *List of Connect Rules Used in Simulation*.

We will demonstrate the following tasks:

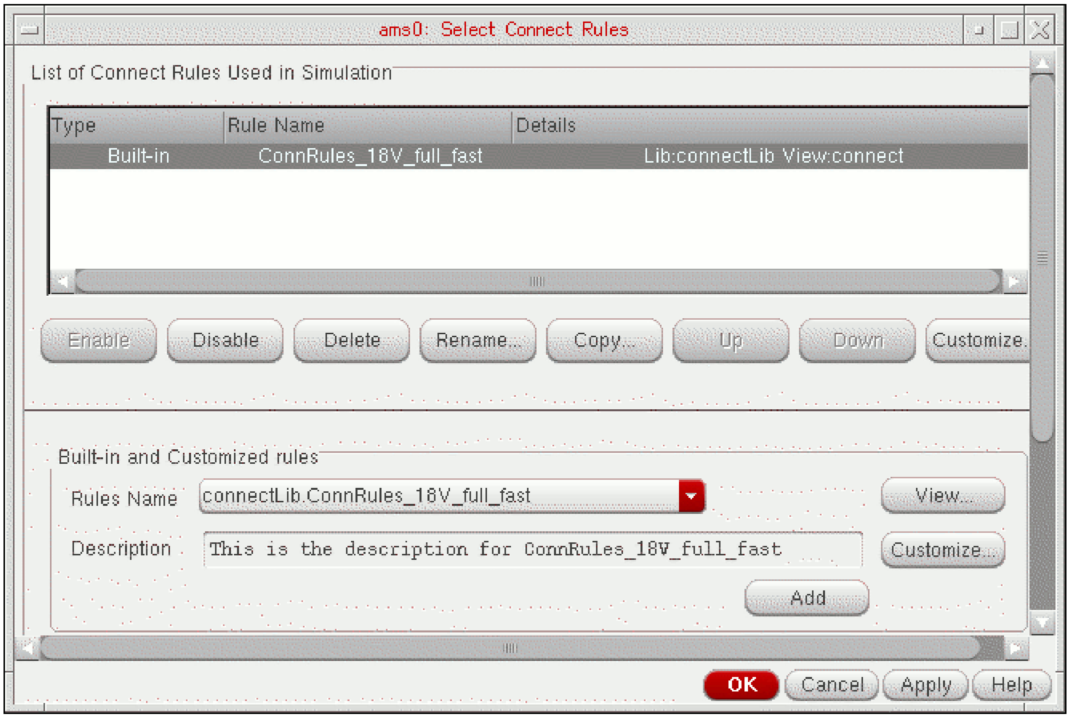
1. Selecting the ConnRules_18V_full_fast Built-In Rules to Customize on page 164
2. Customizing the L2E_2 Connect Module for the 1.2-Volt Supply on page 166
3. Customizing the E2L_2 Connect Module for the 1.2-Volt Supply on page 168

4. Customizing the Bidir_2 Connect Module for the 1.2-Volt Supply on page 169
5. Copying Custom Connect Rules for the 1.2-Volt Supply to my_connectLib on page 170
6. Selecting the ConnRules_5V_full_fast Rules to Customize on page 171
7. Customizing the L2E_2 Connect Module for the 2.5-Volt Supply on page 174
8. Customizing the E2L_2 Connect Module for the 2.5-Volt Supply on page 176
9. Customizing the Bidir_2 Connect Module for the 2.5-Volt Supply on page 178
10. Copying Custom Connect Rules for the 2.5-Volt Supply to my_connectLib on page 180

Selecting the ConnRules_18V_full_fast Built-In Rules to Customize

To select and customize the `ConnRules_18V_full_fast` built-in rules for our 1.2-volt supply, do the following on the Select Connect Rules form:

1. Select the row containing *Built-in ConnRules_18V_full_fast*.



ams0: Select Connect Rules

List of Connect Rules Used in Simulation

Type	Rule Name	Details
Built-in	ConnRules_18V_full_fast	Lib:connectLib View:connect

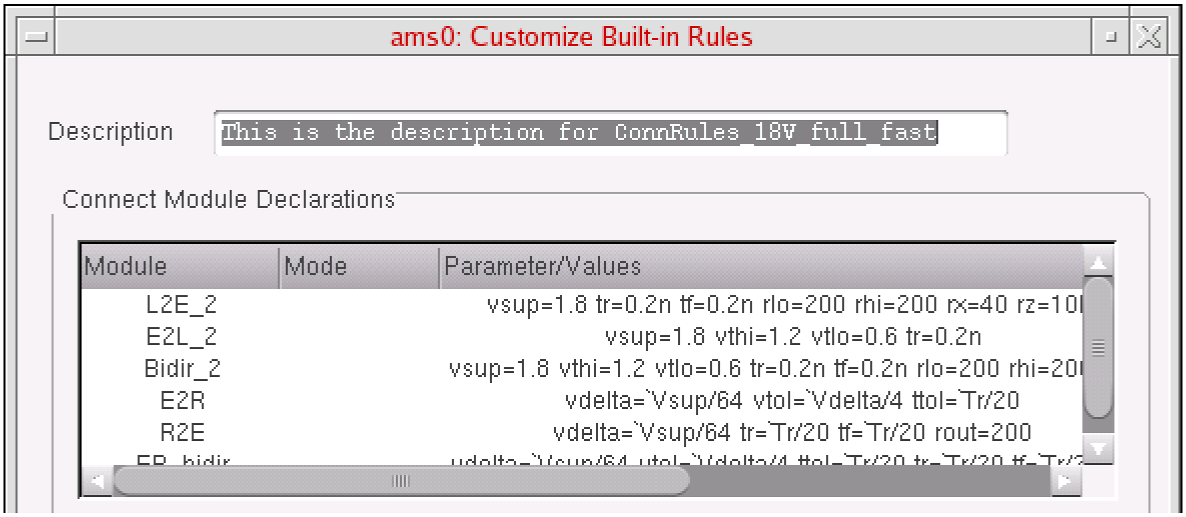
Built-in and Customized rules

Rules Name: connectLib.ConnRules_18V_full_fast

Description: This is the description for ConnRules_18V_full_fast

2. Click *Customize*.

The Customize Built-in Rules form appears.



Description: This is the description for ConnRules_18V_full_fast

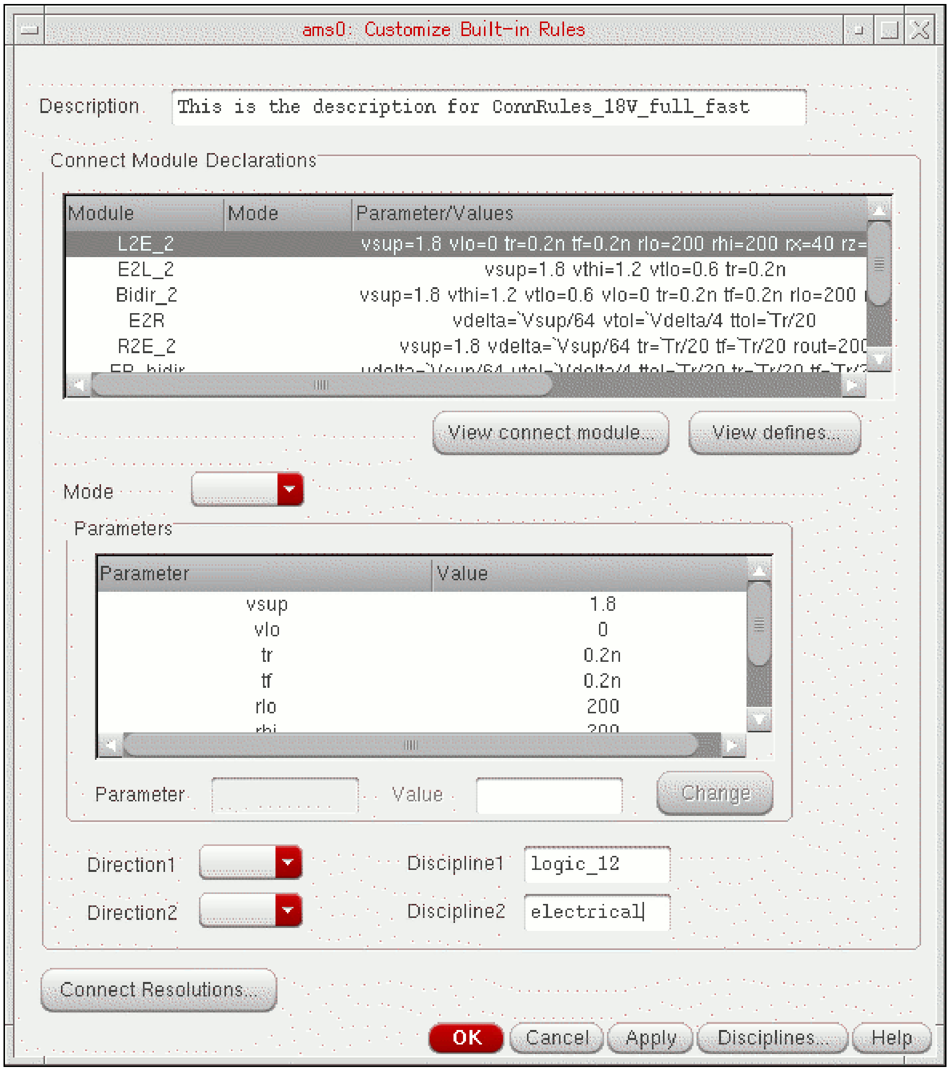
Connect Module Declarations

Module	Mode	Parameter/Values
L2E_2		vsup=1.8 tr=0.2n tf=0.2n rlo=200 rhi=200 rx=40 rz=10
E2L_2		vsup=1.8 vthi=1.2 vtlo=0.6 tr=0.2n
Bidir_2		vsup=1.8 vthi=1.2 vtlo=0.6 tr=0.2n tf=0.2n rlo=200 rhi=20
E2R		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20
R2E		vdelta=`Vsup/64 tr=`Tr/20 tf=`Tr/20 rout=200
ER_bidir		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20 tr=`Tr/20 tf=`Tr/2

Customizing the L2E_2 Connect Module for the 1.2-Volt Supply

The `L2E_2` connect module has a `logic` input and `electrical` output interface. To specify a `logic_12` input instead, and to modify the connect module parameters, do the following:

1. In the *Connect Module Declarations* table, select the row containing *L2E_2*.



Description: This is the description for ConnRules_18V_full_fast

Connect Module Declarations

Module	Mode	Parameter/Values
L2E_2		vsup=1.8 vlo=0 tr=0.2n tf=0.2n rlo=200 rhi=200 rx=40 rz=
E2L_2		vsup=1.8 vthi=1.2 vtlo=0.6 tr=0.2n
Bidir_2		vsup=1.8 vthi=1.2 vtlo=0.6 vlo=0 tr=0.2n tf=0.2n rlo=200
E2R		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20
R2E_2		vsup=1.8 vdelta=`Vsup/64 tr=`Tr/20 tf=`Tr/20 rout=200

Mode:

Parameters

Parameter	Value
vsup	1.8
vlo	0
tr	0.2n
tf	0.2n
rlo	200
rhi	200

Direction1:

Discipline1: logic_12

Direction2:

Discipline2: electrical

2. In the *Direction* and *Discipline* fields near the bottom of the form, select or type the following:

Form Field	**Value**	**Form Field**	**Value**
Direction1	*input*	*Discipline1*	`logic_12`
Direction2	*output*	*Discipline2*	`electrical`

3. In the *Parameters* table, do the following to customize the connect module parameter values for this example:

a. Select the parameter you want to change.

Its name appears in the *Parameter* field. Its value appears in the *Value* field.

b. In the *Value* field, click and drag the mouse to highlight the value for editing.

c. Type the new value.

d. Click *Change*.

Change the following values:

Parameter	*Value*	Change it to
vsup	*1.8*	`1.2`
tr	*0.2n*	`0.4n`
tf	*0.2n*	`0.4n`
vthi	*1.2*	`0.6`
vtlo	*0.6*	`0.3`

Customizing the E2L_2 Connect Module for the 1.2-Volt Supply

The `E2L_2` connect module has an `electrical` input and a `logic` output interface. To specify a `logic_12` output instead, and to modify the connect module parameters, do the following:

1. In the *Connect Module Declarations* table, select the row containing *E2L_2*.
2. In the *Direction* and *Discipline* fields near the bottom of the form, select or type the following:

Form Field	**Value**	**Form Field**	**Value**
Direction1	*input*	*Discipline1*	`electrical`
Direction2	*output*	*Discipline2*	`logic_12`

3. In the *Parameters* table, do the following to customize the connect module parameter values for this example:

   a. Select the parameter you want to change.

   b. In the *Value* field, click and drag the mouse to highlight the value for editing.

   c. Type the new value.

   d. Click *Change*.

   Change the following values:

Parameter	*Value*	Change it to
vsup	*1.8*	`1.2`
vthi	*1.2*	`0.6`
vtlo	*0.6*	`0.3`
tr	*0.2n*	`0.4n`

Customizing the Bidir_2 Connect Module for the 1.2-Volt Supply

To customize the `Bidir_2` connect module to use the custom `logic_12` discipline, and to modify the connect module parameters, do the following:

1. In the *Connect Module Declarations* table, select the row containing *Bidir_2*.
2. In the *Direction* and *Discipline* fields near the bottom of the form, select or type the following:

Form Field	**Value**	**Form Field**	**Value**
Direction1	*inout*	*Discipline1*	`logic_12`
Direction2	*inout*	*Discipline2*	`electrical`

3. In the *Parameters* table, do the following to customize the connect module parameter values for this example:
   - **a.** Select the parameter you want to change.
   - **b.** In the *Value* field, click and drag the mouse to highlight the value for editing.
   - **c.** Type the new value.
   - **d.** Click *Change*.

   Change the following values:

Parameter	*Value*	Change it to
vsup	*1.8*	`1.2`
vthi	*1.2*	`0.6`
vtlo	*0.6*	`0.3`
tr	*0.2n*	`0.4n`
tf	*0.2n*	`0.4n`

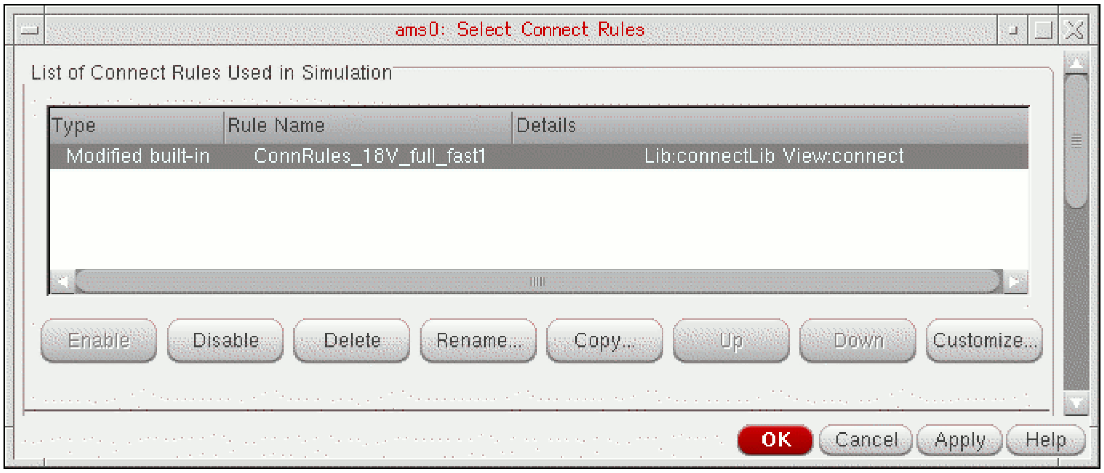
Copying Custom Connect Rules for the 1.2-Volt Supply to my_connectLib

To save the connect rules we customized in the previous steps and copy these connect rules to `my_connectLib`, do the following:

1. On the Customize Built-in Rules form, click *OK*.

   The Select Connect Rules form appears.

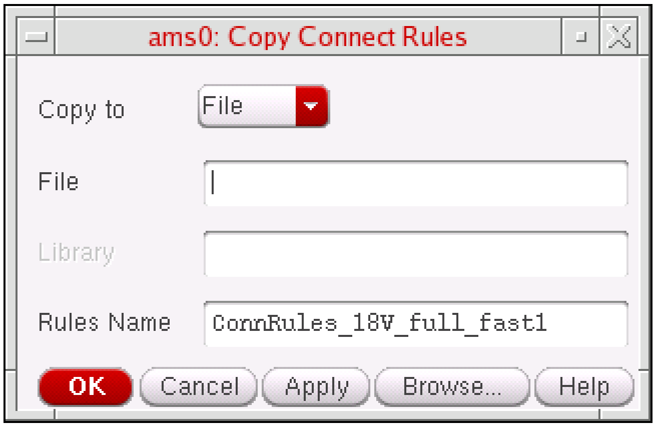
   The custom connect rules for the 1.2-volt supply we created by modifying the `ConnRules_18V_full_fast` built-in connect rules appear in the *List of Connect Rules Used in Simulation* as *Modified built-in, ConnRules_18V_full_fast1*.



Type	Rule Name	Details
Modified built-in	ConnRules_18V_full_fast1	Lib:connectLib View:connect

2. Click *Copy*.

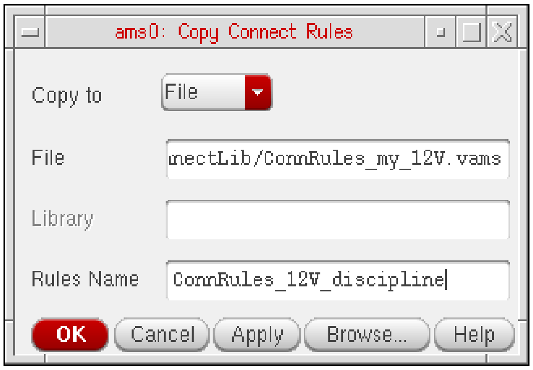
   The Copy Connect Rules form appears.



Copy to: File
File:
Library:
Rules Name: ConnRules_18V_full_fast1

3. In the *File* field, type `./my_connectLib/ConnRules_my_12V.vams`.

4. In the *Rules Name* field, type `ConnRules_12V_discipline`.



Copy to: File

File: nectLib/ConnRules_my_12V.vams

Library:

Rules Name: ConnRules_12V_discipline

5. Click *OK*.

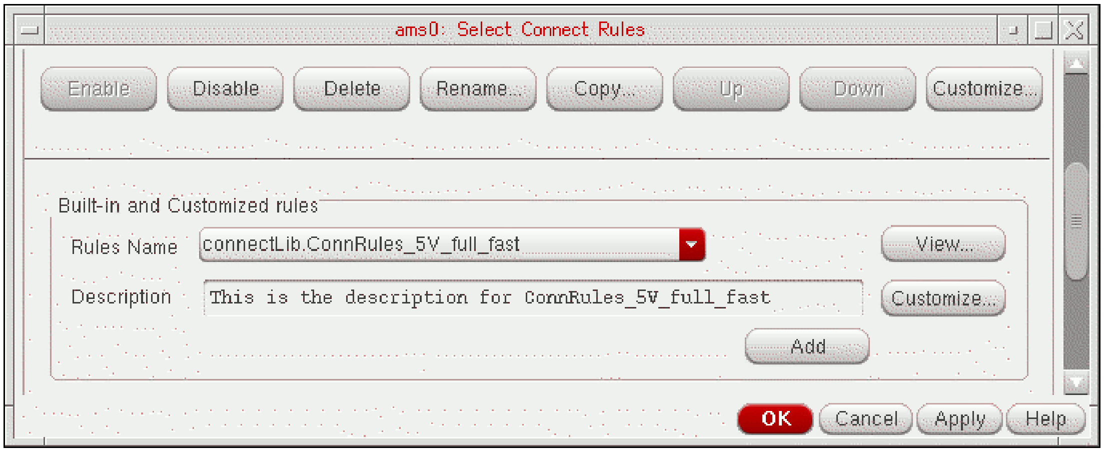
Note: Cadence provides this file in `my_connectLib`, so a prompt appears asking whether you want to overwrite the existing file. You can choose to overwrite the file or click *No* and *Cancel* at this point.

Selecting the ConnRules_5V_full_fast Rules to Customize

We will modify the 5-volt built-in connect rules to create a custom set of rules for our 2.5-volt supply. Because this example has only two supply values (1.2-voltand 2.5-volt), we do not need another custom discipline for the 2.5-volt connect rules: We can use the default discipline (`logic`).

To select and customize the `ConnRules_5V_full_fast` built-in rules for our 2.5-volt supply, do the following on the Select Connect Rules form:

1. In the *Built-in rules* group box, use the *Rules Name* drop-down combo box to select *connectLib.ConnRules_5V_full_fast*.



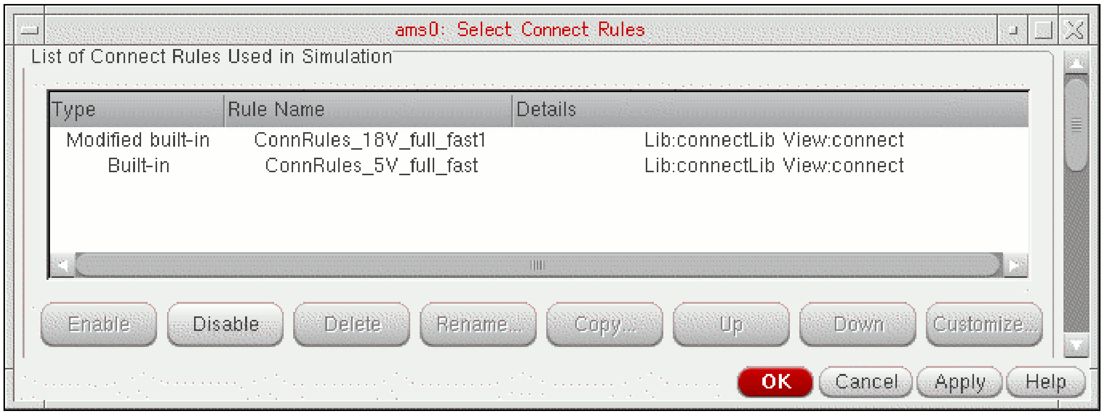
Built-in and Customized rules

Rules Name connectLib.ConnRules_5V_full_fast

Description This is the description for ConnRules_5V_full_fast

2. Click *Add*.

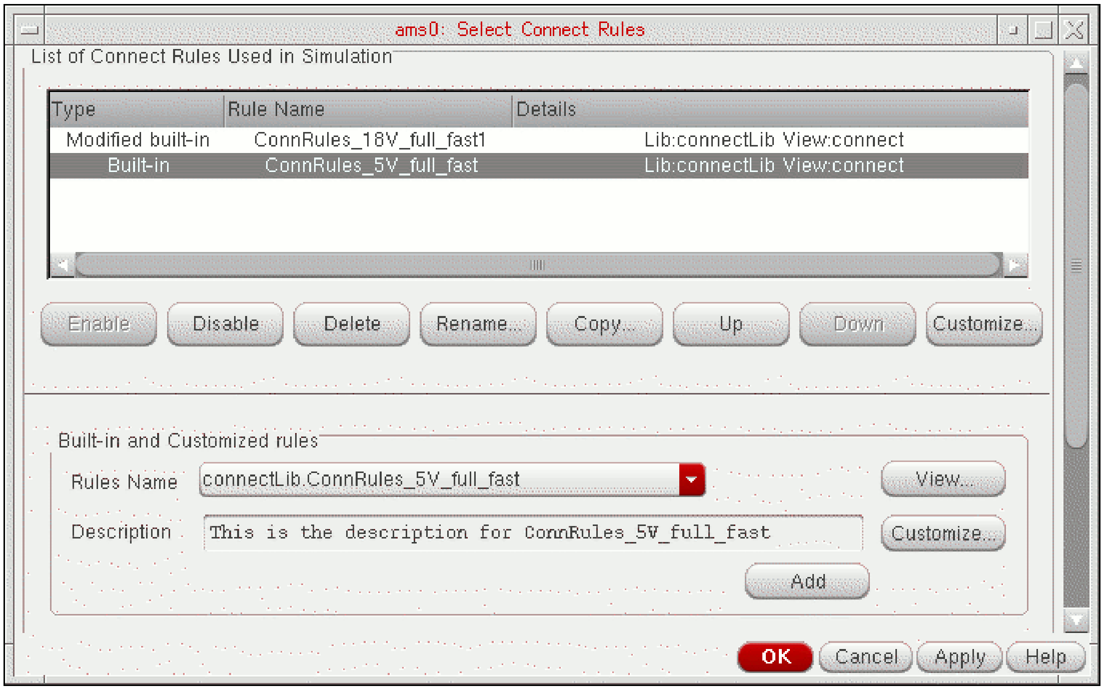
   Built-in and *ConnRules_5V_full_fast* appears in the *List of Connect Rules Used in Simulation*.



List of Connect Rules Used in Simulation

Type	Rule Name	Details
Modified built-in	ConnRules_18V_full_fast1	Lib:connectLib View:connect
Built-in	ConnRules_5V_full_fast	Lib:connectLib View:connect

3. Select the row containing *Built-in ConnRules_5V_full_fast*.



ams0: Select Connect Rules

List of Connect Rules Used in Simulation

Type	Rule Name	Details
Modified built-in	ConnRules_18V_full_fast1	Lib:connectLib View:connect
Built-in	ConnRules_5V_full_fast	Lib:connectLib View:connect

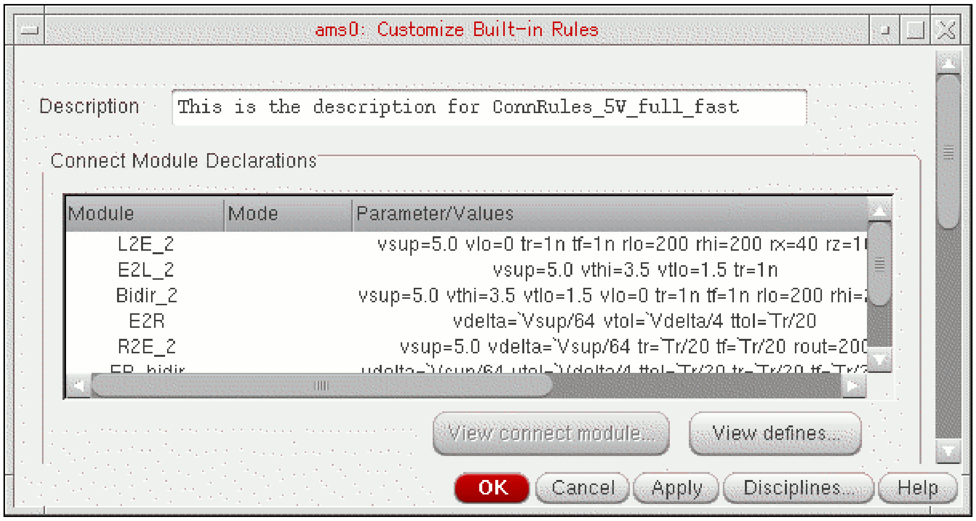
Built-in and Customized rules

Rules Name connectLib.ConnRules_5V_full_fast

Description This is the description for ConnRules_5V_full_fast

4. Click *Customize*.

The Customize Built-in Rules form appears.



ams0: Customize Built-in Rules

Description This is the description for ConnRules_5V_full_fast

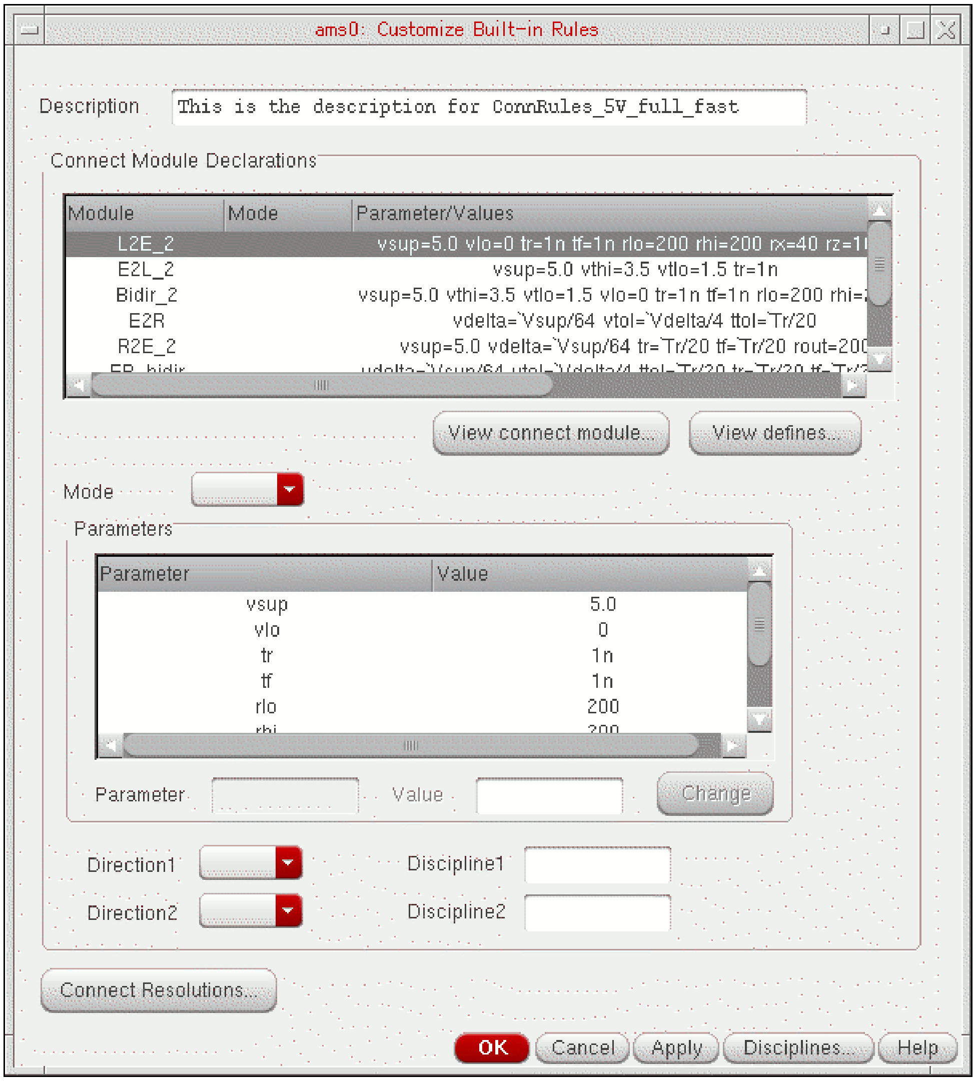
Connect Module Declarations

Module	Mode	Parameter/Values
L2E_2		vsup=5.0 vlo=0 tr=1n tf=1n rlo=200 rhi=200 rx=40 rz=1
E2L_2		vsup=5.0 vthi=3.5 vtlo=1.5 tr=1n
Bidir_2		vsup=5.0 vthi=3.5 vtlo=1.5 vlo=0 tr=1n tf=1n rlo=200 rhi=
E2R		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20
R2E_2		vsup=5.0 vdelta=`Vsup/64 tr=`Tr/20 tf=`Tr/20 rout=200
ER_bidir		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20 tr=`Tr/20 tf=`Tr/2

Customizing the L2E_2 Connect Module for the 2.5-Volt Supply

To customize the `L2E_2` connect module parameters, do the following:

1. In the *Connect Module Declarations* table, select the row containing *L2E_2*.



Description: This is the description for ConnRules_5V_full_fast

Connect Module Declarations

Module	Mode	Parameter/Values
L2E_2		vsup=5.0 vlo=0 tr=1n tf=1n rlo=200 rhi=200 rx=40 rz=1
E2L_2		vsup=5.0 vthi=3.5 vtlo=1.5 tr=1n
Bidir_2		vsup=5.0 vthi=3.5 vtlo=1.5 vlo=0 tr=1n tf=1n rlo=200 rhi=
E2R		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20
R2E_2		vsup=5.0 vdelta=`Vsup/64 tr=`Tr/20 tf=`Tr/20 rout=20
ER_bidir		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20 tr=`Tr/20 tf=`Tr/2

Parameters

Parameter	Value
vsup	5.0
vlo	0
tr	1n
tf	1n
rlo	200
rhi	200

2. In the *Parameters* table, do the following to customize the *vsup* connect module parameter value for this example:

   a. Select *vsup*.

   Its name appears in the *Parameter* field. Its value appears in the *Value* field.

b. In the *Value* field, click and drag the mouse to highlight the value for editing.

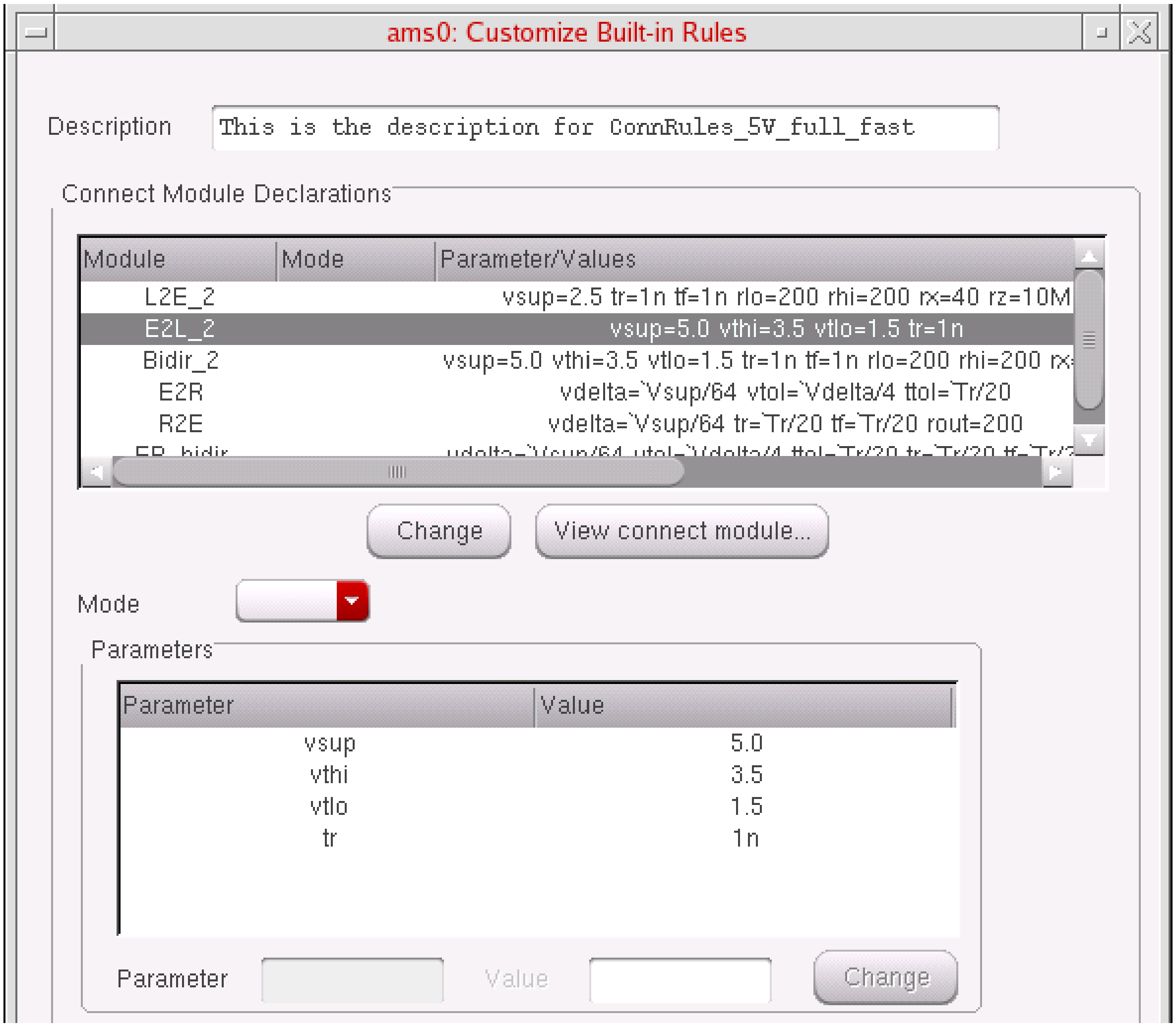
c. Type `2.5`.

d. Click *Change*.

Customizing the E2L_2 Connect Module for the 2.5-Volt Supply

To customize the `E2L_2` connect module parameters, do the following:

1. In the *Connect Module Declarations* table, select the row containing *E2L_2*.



ams0: Customize Built-in Rules

Description: This is the description for ConnRules_5V_full_fast

Connect Module Declarations

Module	Mode	Parameter/Values
L2E_2		vsup=2.5 tr=1n tf=1n rlo=200 rhi=200 rx=40 rz=10M
E2L_2		vsup=5.0 vthi=3.5 vtlo=1.5 tr=1n
Bidir_2		vsup=5.0 vthi=3.5 vtlo=1.5 tr=1n tf=1n rlo=200 rhi=200 rx
E2R		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20
R2E		vdelta=`Vsup/64 tr=`Tr/20 tf=`Tr/20 rout=200
ER_bidir		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20 tr=`Tr/20 tf=`Tr/2

Mode:

Parameters

Parameter	Value
vsup	5.0
vthi	3.5
vtlo	1.5
tr	1n

2. In the *Parameters* table, do the following to customize the connect module parameter values for this example:

   e. Select the parameter you want to change.

   f. In the *Value* field, click and drag the mouse to highlight the value for editing.

   g. Type the new value.

   h. Click *Change*.

Change the following values:

Parameter	*Value*	Change it to
vsup	*5.0*	`2.5`
vthi	*3.5*	`1.75`
vtlo	*1.5*	`0.75`

Customizing the Bidir_2 Connect Module for the 2.5-Volt Supply

To customize the `Bidir_2` connect module parameters, do the following:

1. In the *Connect Module Declarations* table, select the row containing *Bidir_2*.
2. In the *Parameters* table, do the following to customize the connect module parameter values for this example:

   a. Select the parameter you want to change.

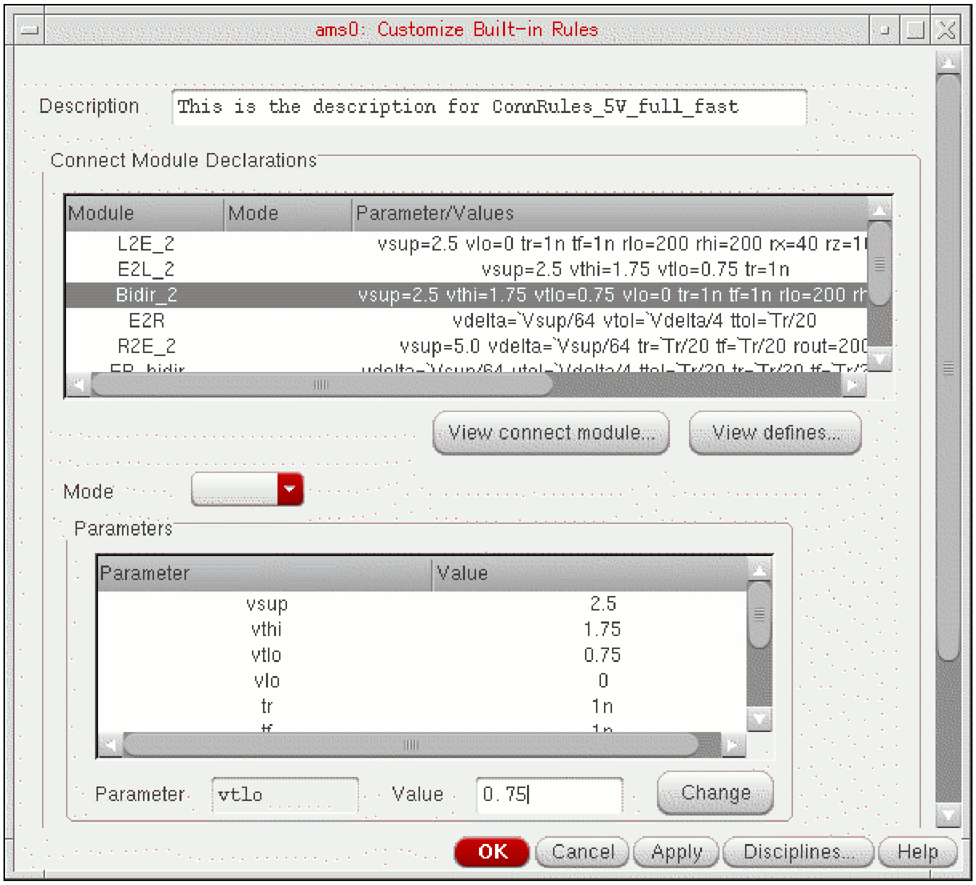
   b. In the *Value* field, click and drag the mouse to highlight the value for editing.

   c. Type the new value.

   d. Click *Change*.

Change the following values:

Parameter	*Value*	Change it to
vsup	*5.0*	`2.5`
vthi	*3.5*	`1.75`
vtlo	*1.5*	`0.75`



ams0: Customize Built-in Rules

Description: `This is the description for ConnRules_5V_full_fast`

Connect Module Declarations

Module	Mode	Parameter/Values
L2E_2		vsup=2.5 vlo=0 tr=1n tf=1n rlo=200 rhi=200 rx=40 rz=1
E2L_2		vsup=2.5 vthi=1.75 vtlo=0.75 tr=1n
Bidir_2		vsup=2.5 vthi=1.75 vtlo=0.75 vlo=0 tr=1n tf=1n rlo=200 rh
E2R		vdelta=`Vsup/64 vtol=`Vdelta/4 ttol=`Tr/20
R2E_2		vsup=5.0 vdelta=`Vsup/64 tr=`Tr/20 tf=`Tr/20 rout=200

Mode

Parameters

Parameter	Value
vsup	2.5
vthi	1.75
vtlo	0.75
vlo	0
tr	1n

Parameter: vtlo   Value: 0. 75

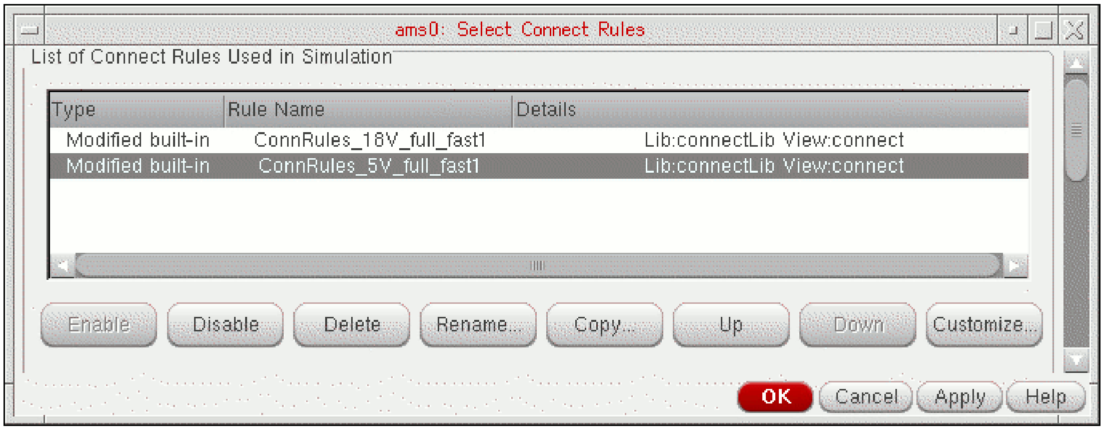
Copying Custom Connect Rules for the 2.5-Volt Supply to my_connectLib

To save the connect rules we customized in the previous steps and copy these connect rules to `my_connectLib`, do the following:

1. On the Customize Built-in Rules form, click *OK*.

   The Select Connect Rules form appears.

   The custom connect rules we created for the 2.5-volt supply by modifying the `ConnRules_5V_full_fast` built-in connect rules appear in the *List of Connect Rules Used in Simulation* as *Modified built-in, ConnRules_5V_full_fast1*.



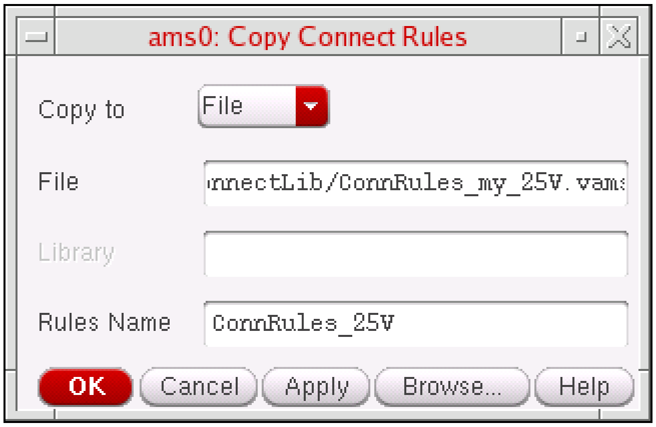
Type	Rule Name	Details
Modified built-in	ConnRules_18V_full_fast1	Lib:connectLib View:connect
Modified built-in	ConnRules_5V_full_fast1	Lib:connectLib View:connect

2. Click *Copy*.

   The Copy Connect Rules form appears.

3. In the *File* field, type `./my_connectLib/ConnRules_my_25V.vams`.

4. In the *Rules Name* field, type `ConnRules_25V`.



Copy to	File
File	nnectLib/ConnRules_my_25V.vams
Rules Name	ConnRules_25V

5. Click *OK*.

   Note: Cadence provides this file in `my_connectLib`, so a prompt appears asking whether you want to overwrite the existing file. You can choose to overwrite the file or click *No* and *Cancel* at this point.

Specifying the Default Digital Discipline in the Schematic Editor

To specify the default digital discipline in the schematic editor, do the following:

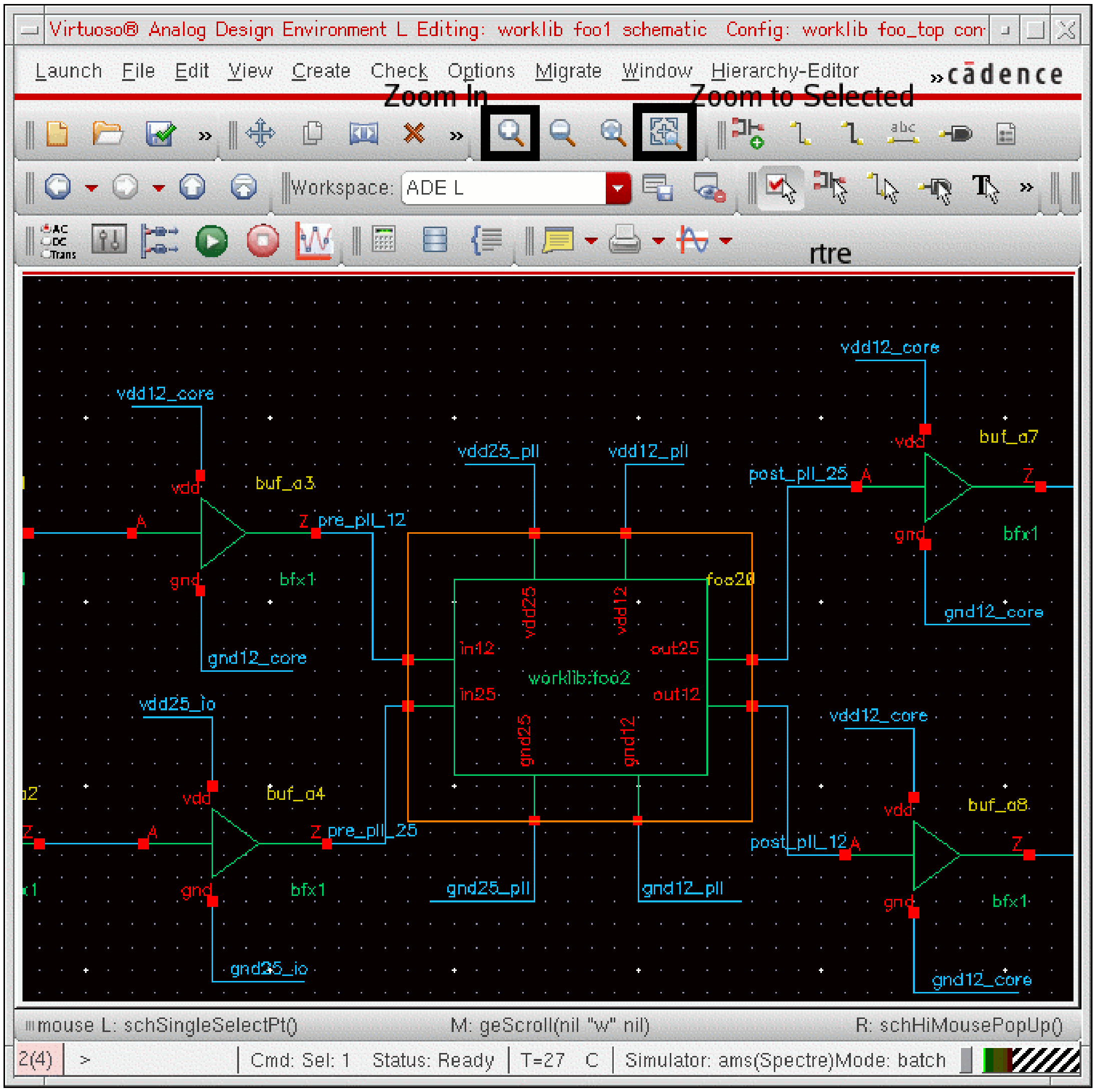
1. In ADE, choose *Session – Schematic Window*.

   The schematic window appears.

   Because *ams* appears in the *Simulator* field of the Choosing Simulator form in ADE, the *AMS* menu appears on the menu bar in the schematic window.

2. In the schematic window, select the *foo10* instance and type `e` to descend into it.
3. On the Descend form, click *OK*.
4. Select instance *foo20* and click the *Zoom to Selected* icon.

5. Use *Zoom In* to get a closer view.



Virtuoso® Analog Design Environment L Editing: worklib foo1 schematic Config: worklib foo_top con

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

2(4) > Cmd: Sel: 1 Status: Ready T=27 C Simulator: ams(Spectre)Mode: batch

Note the following:

- ❑ `pre_pll_12` is an analog net that connects to a digital port, `in12`. The connect module for this case is `E2L_2` from connect rules `ConnRules_12V_discipline`.

  If the voltage value on `pre_pll_12` is greater than

  `vthi = 0.6 V`

  the program converts the value on digital port `in12` to `logic1`.

The same is true for `pre_vdd_12`, which connects to digital port `A` on instance `buf_d5`.

❑ `pre_pll_25` is an analog net that connects to a digital port, `in25`. The connect module for this case is E2L_2 from connect rules `ConnRules_25V`.

If the voltage value on `pre_pll_25` is greater than

```
vthi = 1.75 V
```

the program converts the value on digital port `in25` to `logic1`.

❑ `out25` is a digital port that connects to analog net `post_pll_25`. The connect module for this case is L2E_2 from connect rules `ConnRules_25V`.

The program converts a digital value of `logic1` on `out25` to voltage value 2.5 volts on `post_pll_25`.

❑ `out12` is a digital port that connects to analog net `post_pll_12`. The connect module for this case is L2E_2 from connect rules `ConnRules_12V_discipline`.

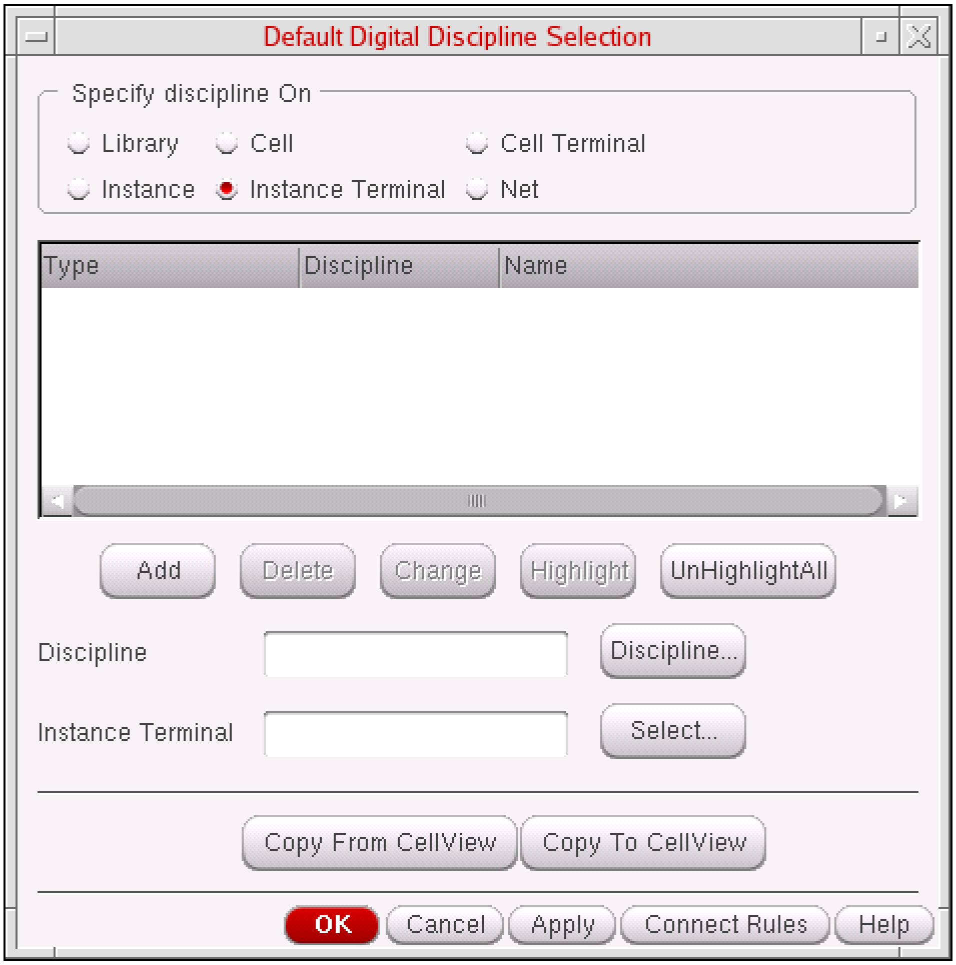
The program converts a digital value of `logic1` on `out12` to voltage value 1.2 volts on `post_pll_12`.

The same is true for digital port `Z` on instance `buf_d5`, which connects to analog net `post_vdd_12`.

To specify `logic_12` as the default discipline for analog/digital boundaries with a 1.2-volt supply, do the following:

1. In the schematic window, choose *AMS – Default Discipline Selection – Instance Terminal*.

   The Default Digital Discipline Selection form appears.



Default Digital Discipline Selection

Specify discipline On

○ Library ○ Cell ○ Cell Terminal
○ Instance ● Instance Terminal ○ Net

Type	Discipline	Name

Discipline:

Instance Terminal:

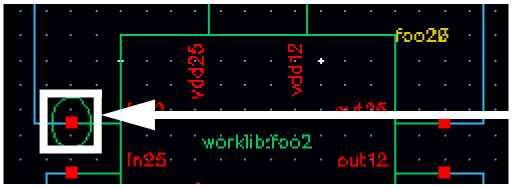
2. In the *Discipline* field, type `logic_12`.

3. To the right of the *Instance Terminal* field, click *Select*.

   The schematic window appears in the foreground.

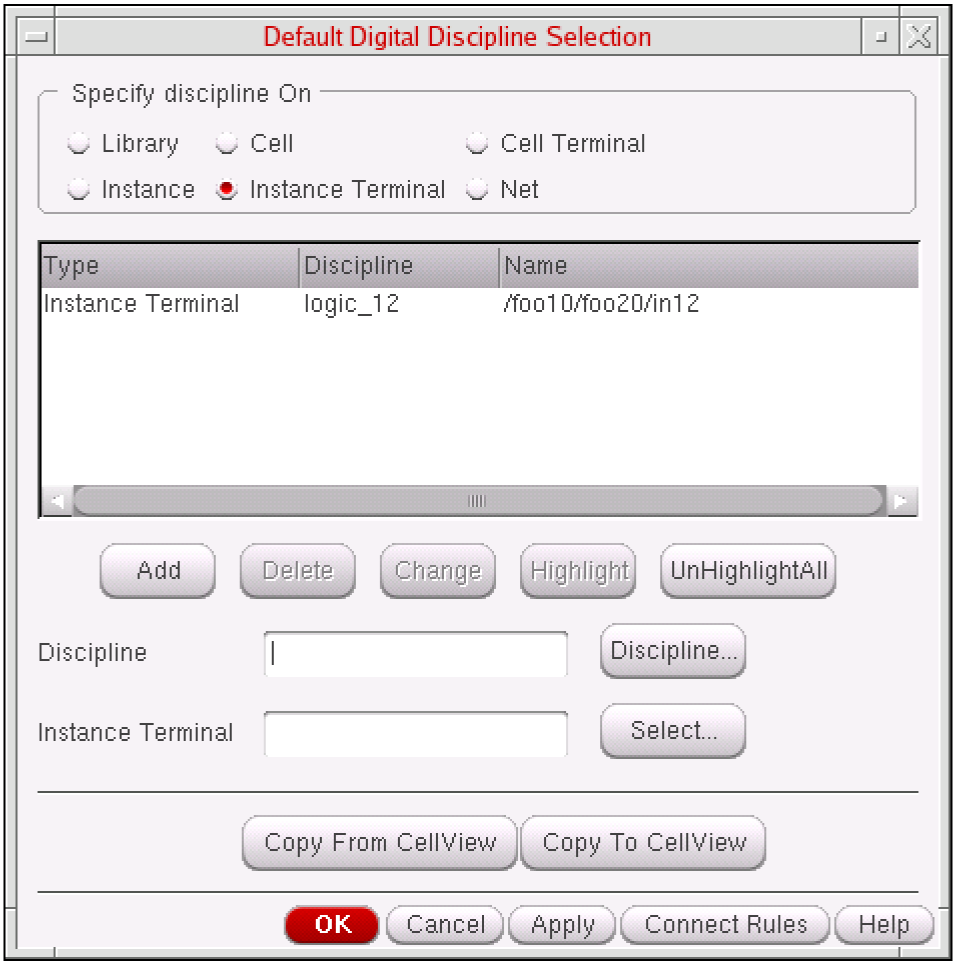
4. Select instance terminal `in12`.

A selection circle appears around the instance terminal.



Selected instance terminal

5. Return to the Default Digital Discipline Selection form and click *Add*.



Default Digital Discipline Selection

Specify discipline On

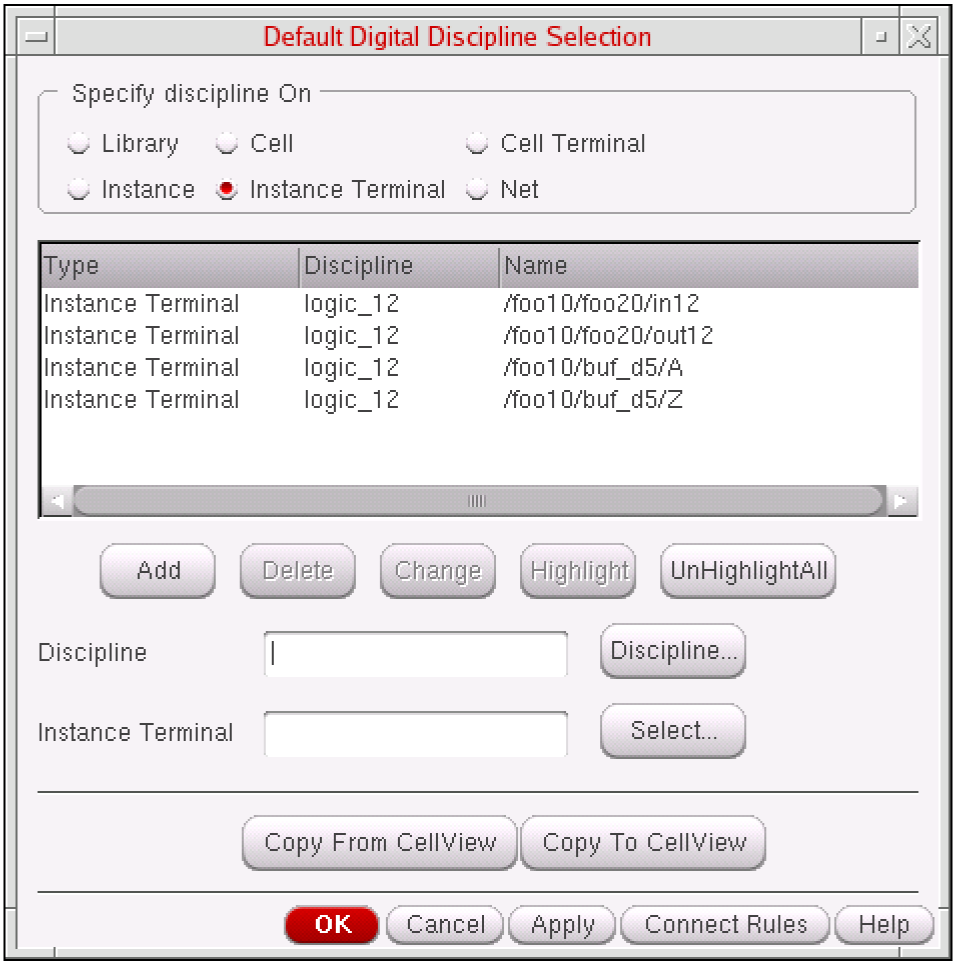
○ Library ○ Cell ○ Cell Terminal
○ Instance ● Instance Terminal ○ Net

Type	Discipline	Name
Instance Terminal	logic_12	/foo10/foo20/in12

Discipline: |

Instance Terminal:

6. Repeat Steps 1 through 5 for instance terminals `foo20/out12`, `buf_d5/A`, and `buf_d5/Z`.



Default Digital Discipline Selection

Specify discipline On

○ Library ○ Cell ○ Cell Terminal
○ Instance ● Instance Terminal ○ Net

Type	Discipline	Name
Instance Terminal	logic_12	/foo10/foo20/in12
Instance Terminal	logic_12	/foo10/foo20/out12
Instance Terminal	logic_12	/foo10/buf_d5/A
Instance Terminal	logic_12	/foo10/buf_d5/Z

Discipline: |

Instance Terminal:

7. Click *Copy To CellView*.

   Note: This step enables us to use *Copy From CellView* at some future point.

8. Click *OK*.

Selecting Connect Rules to Use

In “Preparing Connect Modules, Connect Rules, and Discipline Definitions Using a Text Editor” on page 159, we created `ConnRules_discipline`. In “Preparing Connect Modules, Connect Rules, and Discipline Definitions Using the Select Connect Rules Form in the Environment” on page 162, we created `ConnRules_12V_discipline` and `ConnRules_25V`. You could choose either connect rules setup.

To load the state we will use for this example, do the following:

1. In ADE, choose *Session – Load State*.

   The Loading State form appears.

2. In the *State Name* area, choose *state_ams_no_setup*.

   This state contains basic setup information, such as the `hdl.var` file and transient stop time.

3. Click *OK*.

   This setup uses the OSS-based netlister with `irun`.

See the following topics for information about how to select each set of connect rules:

- Selecting ConnRules_Discipline on page 189
- Selecting ConnRules_12V_Discipline and ConnRules_25V on page 191

Selecting ConnRules_Discipline

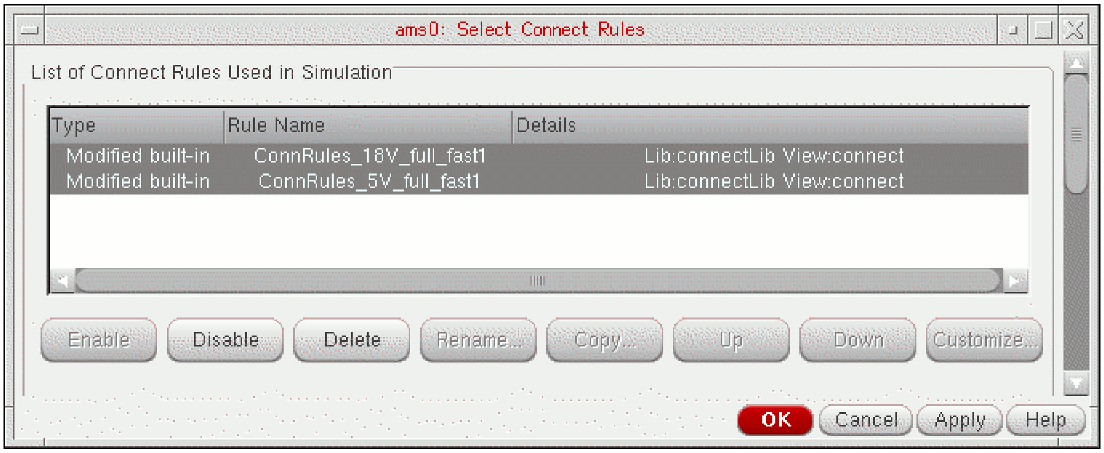
For this example, we will select the `ConnRules_discipline` we created in "Preparing Connect Modules, Connect Rules, and Discipline Definitions Using a Text Editor" on page 159.

To select the `my_connectLib/ConnRules_discipline` custom connect rules to use, do the following:

1. In ADE, choose *Setup – Connect Rules*.

   The Select Connect Rules form appears.

2. If there are any built-in or modified built-in rules names from the *List of Connect Rules Used in Simulation*, do the following:



ams0: Select Connect Rules

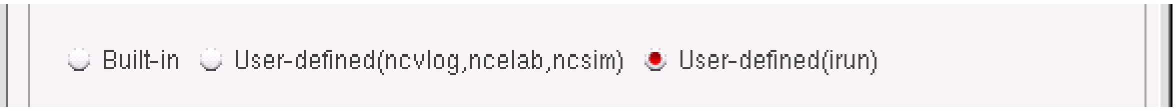
List of Connect Rules Used in Simulation

Type	Rule Name	Details
Modified built-in	ConnRules_18V_full_fast1	Lib:connectLib View:connect
Modified built-in	ConnRules_5V_full_fast1	Lib:connectLib View:connect

   a. Select (highlight) the rules in the table.

   b. Click *Delete*.

3. Choose *User-defined(irun)*.



○ Built-in ○ User-defined(ncvlog,ncelab,ncsim) ● User-defined(irun)

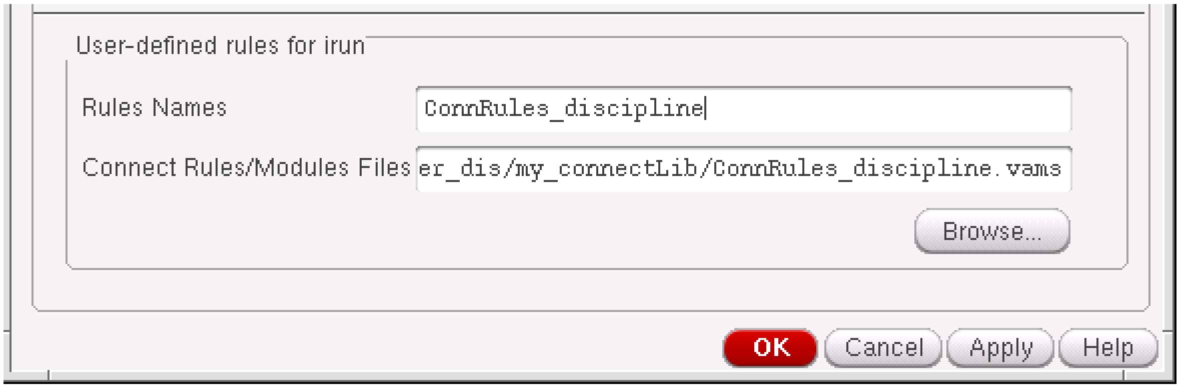
The *User-defined rules for irun* group box becomes active.

Remember, we are using the OSS-based netlister with `irun`. User-defined rules for this setup do not require a Virtuoso$^{®}$ database. The software reads and compiles the connect modules when it reads and compiles your other source files.

Note: User-defined rules for the cellview-based netlister with `ncvlog`, `ncelab`, and `ncsim` must have a Virtuoso database, so you would have to compile the modules into the Cadence lib/cell/view structure.

4. Click *Browse*.
5. On the Browser form that appears, navigate to and choose *my_connectLib/ Conn_Rules_discipline.vams*.
6. Click *Open*.

   ...my_connectLib/ Conn_Rules_discipline.vams appears in the *Connect Rules/ Modules Files* field.



User-defined rules for irun

Rules Names `ConnRules_discipline`

Connect Rules/Modules Files `er_dis/my_connectLib/ConnRules_discipline.vams`

7. In the *Rules Names* field, type `ConnRules_discipline`.
8. Click *OK*.
9. Proceed to "Simulating and Analyzing Results" on page 193.

Selecting ConnRules_12V_Discipline and ConnRules_25V

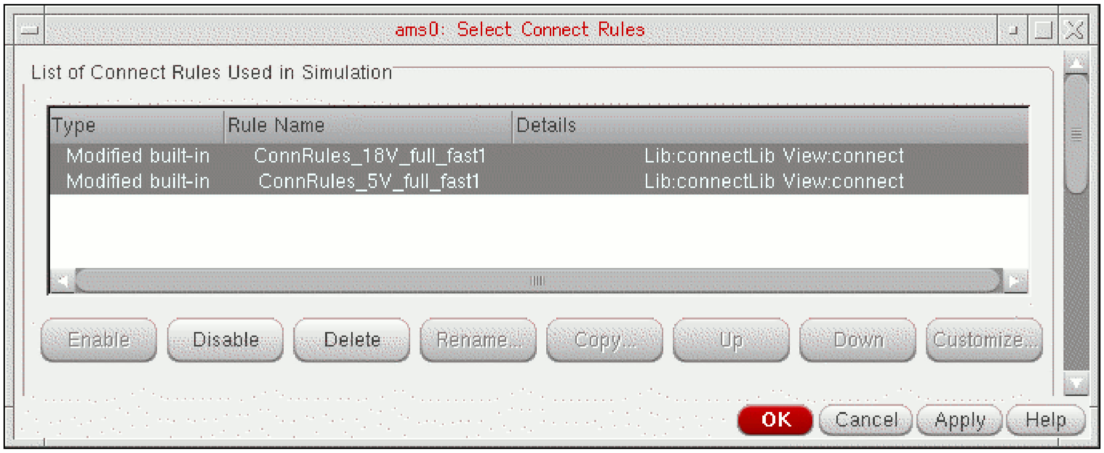
For this example, we will select the `ConnRules_12V_discipline` and `ConnRules_25V` we created in "Preparing Connect Modules, Connect Rules, and Discipline Definitions Using the Select Connect Rules Form in the Environment" on page 162.

To select the `my_connectLib/ConnRules_12V_discipline` and `my_connectLib/ConnRules_25V` custom connect rules to use, do the following:

1. In ADE, choose *Setup – Connect Rules*.

   The Select Connect Rules form appears.

2. If there are any built-in or modified built-in rules names from the *List of Connect Rules Used in Simulation*, do the following:



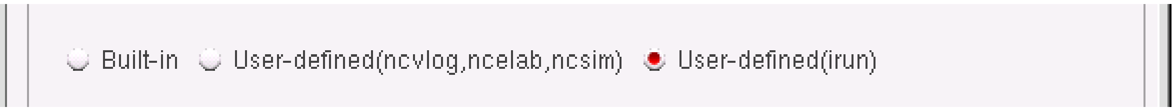
List of Connect Rules Used in Simulation

Type	Rule Name	Details
Modified built-in	ConnRules_18V_full_fast1	Lib:connectLib View:connect
Modified built-in	ConnRules_5V_full_fast1	Lib:connectLib View:connect

   a. Select (highlight) the rules in the table.

   b. Click *Delete*.

3. Choose *User-defined(irun)*.



○ Built-in ○ User-defined(ncvlog,ncelab,ncsim) ● User-defined(irun)

The *User-defined rules for irun* group box becomes active.

Remember, we are using the OSS-based netlister with `irun`. User-defined rules for this setup do not require a Virtuoso® database. The software reads and compiles the connect modules when it reads and compiles your other source files.

Note: User-defined rules for the cellview-based netlister with `ncvlog`, `ncelab`, and `ncsim` must have a Virtuoso database, so you would have to compile the modules into the Cadence lib/cell/view structure.

4. Click *Browse*.
5. On the Browser form that appears, navigate to and choose *my_connectLib/ ConnRules_my_12V.vams*.
6. Click *Apply*.
7. Navigate to and choose *my_connectLib/ ConnRules_my_25V.vams*.
8. Click *Open*.

   ...my_connectLib/ ConnRules_my_12V.vams and *...my_connectLib/ ConnRules_my_25V.vams* appear in the *Connect Rules/Modules Files* field.

9. In the *Rules Names* field, type `ConnRules_12V_discipline ConnRules_25V`.

Important

There is a space between the rules names.

10. Click *OK*.
11. Proceed to "Simulating and Analyzing Results" on page 193.

Simulating and Analyzing Results

We have selected the AMS simulator (*ams* appears in the *Simulator* field of the Choosing Simulator form in ADE), the OSS-based netlister with `irun` (the state we loaded is set up to use *OSS-based netlister with irun* on the Netlist and Run Options form), and SimVision for viewing and debugging our results (*Interactive (debugger)* appears on the Netlist and Run Options form). We have specified our custom connect modules, connect rules, and disciplines (see “Preparing Connect Modules, Connect Rules, and Discipline Definitions Using a Text Editor” on page 159 and “Specifying the Default Digital Discipline in the Schematic Editor” on page 182 and “Selecting Connect Rules to Use” on page 188).

To simulate and analyze results using SimVision, do the following:

Note: Cadence provides the SimVision signal probing setup in the file, `restore.tcl.sv`.

1. In ADE, click the Netlist and Run icon.

   SimVision windows appear.

2. In SimVision, choose *Files – Source Command Script*.

   The Select SimVision Command Script browser appears.

3. Navigate to `restore.tcl.sv` in the starting directory, choose it, and click *Open*.

   SimVision runs the setup script.

4. Click the Play button to run the simulation.

5. When the simulation finishes, be sure to zoom out fully to fit data along both the X and Y axes.

6. Verify the results.