cadence®

Virtuoso Schematic Editor Spectre and Ultrasim Interface User Guide

Product Version ICADVM20.1
October 2020

Printed in the United States of America.

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

Contents

1
Virtuoso Schematic Editor Spectre Interface 5
Specifying Input for the Simulation 8
Specifying Input from a Stimulus File 9
Specifying Input from a Netlist File 10
Specifying Input from a Analyses/Options File 11
Specifying Input from a Model Include File 12
Specifying Input from a Model Include Path 13
Specifying Input from a Design Variable 14
Specifying Input from a Configuration File 15
Specifying Input from Other Files 16
Netlisting Your Design and Starting Simulations 17
Showing Outputs 18
Displaying Results 20
Name Mapping 21
Checking the Status of Your Simulations 22
Running a Simulation Remotely 23

2
Virtuoso Schematic Editor UltraSim Interface 25
Starting the Virtuoso Schematic Editor UltraSim Interface 27
Specifying Input from a Stimulus File 29
Specifying Input from a Netlist File 30
Specifying Input from a Analyses/Options File 31
Specifying Input from a Model Include File 32
Specifying Input from a Model Include Path 33
Specifying Input from a Design Variable 34
Specifying Input from a Configuration File 35
Specifying Input from Other Files 36
Netlisting Your Design and Starting Simulations 37
Showing Outputs 38

Displaying Results 39
Checking Simulation Status 39
Running a Simulation Remotely 40

A
Netlisting Overview 43
Global Nets 43
Parameters 45
Component/Subcircuit Parameters 45
Global Parameters (Design Variables) 45
Parameter Inheritance 46
Expressions 47

1

Virtuoso Schematic Editor Spectre Interface

This chapter describes how to use the Virtuoso® Schematic Editor Spectre® interface. The procedures described in this chapter are deliberately broad and generic. Requirements for your specific design might dictate procedures slightly different from those described here.

This product provides a solution for digital circuit designers to netlist a Virtuoso schematic editor transistor-level schematic for the Spectre circuit simulator. It also provides a batch and scripting solution for automating various tasks. Some of the key features are listed below.

- Spectre Netlister

  This solution provides an Open Simulation System (OSS) netlister for the Spectre circuit simulator from the Virtuoso schematic editor. The netlister works on devices with `spectre` views in the Cadence `sample` library. The netlister produces a hierarchical parameterized netlist in native circuit simulator syntax. The netlister supports global parameters and design variables and netlist processor (NLP) inheritance of parameters. All devices are netlisted using native circuit simulator syntax. For information on the circuit simulator, see the *Spectre Circuit Simulator and Accelerated Parallel Simulator User Guide* and *Spectre Circuit Simulator Reference*.

- Updated Sample Library

  The `sample` library provided by Cadence for OSS netlisters has been enhanced for the Spectre circuit simulator. The devices `iprobe`, `phyres`, `phyresgnd`, `nmes`, and `pmes` have been added to the sample library to support additional circuit simulator models. In addition, most of the circuit simulator instance parameters have been added to the symbol cellview to aid in design creation.

- Improved Simulation Environment

  The OSS simulation environment (SE) has been enhanced for the Spectre circuit simulator, which now appears on the menu bar. You can now directly edit netlisting and support files. You can enter model files and design variables through the UI. The combining of the netlist and support files is controlled by the environment.

- Access to Analog Waveform Display for Waveform Display

  This solution provides access to the Analog Waveform Display (AWD) tool from the analog circuit design environment. The circuit simulator default format for writing data out is PSF, which is read in by the AWD tool. The simulation environment provides access to this tool and the Results Browser and Waveform calculator. You can choose which signals you want to plot using the Results Browser. For more information on these tools, see the analog circuit design environment documentation and the *Cadence Online Support* application note “Using AWD with Stand-alone Spectre.”

- OCEAN Batch and Scripting Capabilities

  The Virtuoso schematic editor Spectre interface uses the OCEAN batch and scripting product to control the simulator with scripts or interactive commands. It also gives you access to the AWD tool, which allows you to make measurements of waveform data and print them to the screen or a file. All of the standard measurement capabilities of a signal calculator are available from the OCEAN command prompt, as are any macros that you create. You can output data files with either the `fprintf` or `ocnPrint` commands. For more information on OCEAN, see the *Ocean Reference*.

This chapter discusses the following topics:

- Starting the Virtuoso Schematic Editor Spectre Interface on page 7
- Specifying Input for the Simulation on page 8
- Netlisting Your Design and Starting Simulations on page 17
- Displaying Results on page 20
- Name Mapping on page 21
- Checking the Status of Your Simulations on page 22
- Running a Simulation Remotely on page 23

Starting the Virtuoso Schematic Editor Spectre Interface

To start the Virtuoso® Schematic Editor Spectre® interface,

1. Start the Cadence software.
2. From the CIW, do one of the following:
   - Start a new design by choosing *File – New...*.
   - Start an existing design by choosing *File – Open*....
3. After correct setup, click the *OK* button to display the Virtuoso schematic editing window.
4. Choose *Launch – Simulation – Spectre*.

   The *Spectre* menu is added to the menu bar.

   If you have not accessed another design during this session or you have never created a circuit simulator run directory for this design, all but one of the commands in the *Spectre* menu are grayed out.
5. Choose *Spectre – Initialize*.

   The Initialize Environment form appears.
6. In the *Simulation Run Directory* field, type the path to the directory where you want to place all of the simulation-related files.

   By default, the files are placed in `spectre.run1` in your startup directory.

   If the path defines a new directory, the Initialize Environment form expands:

   You can change the name of the run directory or specify a different design. The *Browse* button brings up the Library Browser, which automatically fills in the *Library Name*, *Cell Name*, and *View Name* fields when you select a new design.

   Note: *spectre* is the only choice for *Simulator Name*. Custom modifications for third-party simulators are possible.
7. Click *OK*.

Specifying Input for the Simulation

The Virtuoso schematic editor Spectre interface offers you several ways to provide input to your simulation and to modify that input. Before simulation, you have the option of combining all of these files with the netlist file to create a run file.

After you set up your simulation options, you can specify all the input files for the simulation. The *Stimulus* menu lists each of these files.

When you select most of these commands, a text editor window opens up, displaying the file. If the editor is empty, you must create the file.

Specifying Input from a Stimulus File

To provide input from a stimulus file to your simulation,

1. From the schematic editor window, choose *Spectre – Stimulus – Edit Stimulus File*.

   A text file opens in a text editor window, for example `emacs`. The file is saved as `spectre.inp` and is placed in the run directory. You do not have to add source statements if you have included sources in your designs.

   Sources are specified as instance statements in Spectre syntax. See “Spectre Netlists” in the *Spectre Circuit Simulator and Accelerated Parallel Simulator User Guide* for more information.

2. If you use the text file, connect the sources to the net names on the schematic.

   Schematic names are mapped to lowercase. If you use circuit simulator syntax, you must use the lowercase form of net names you are connecting to. Connect to global nets such as `ground` from the stimulus file. Global nets use a `!` at the end of the net name and that character must be escaped. In addition, any escaped characters entered in this file must be escaped twice (`gnd\\!`) because one escape is removed during the merging of netlist files. For more information, see Netlisting Overview.

Note: When you add lines to the analysis and options file, the square brackets characters must be escaped: `\[` and `\]`. For example, a sweep function line must like this:

```
sweep1 sweep param=slope values=\[0.5n 1.0n 1.5n\]
```

By default, the file should have Spectre circuit simulator syntax. If you use SPICE syntax, the line

```
simulator lang=spice
```

must precede the SPICE syntax.

Specifying Input from a Netlist File

To provide input from a netlist file to your simulation,

➤ From the schematic editor window, choose *Spectre – Stimulus – Edit Netlist File*.

If you have already netlisted your design using this product, a text editor window opens to display the netlist file, which you can then modify. The netlist file produced in this way is located in the circuit simulator run directory you must create.

If the text editor window opens with an empty file, you have yet to netlist this design.

The file is called `netlist` and is located in the run directory. This is not the final netlist: it does not include any of the data from the other stimulus files. If netlisting has occurred, it will contain connectivity information for the associated schematic. If you modify this file, you might have to escape certain characters. Also, the next time you netlist this file, all modifications to this file will be lost.

Specifying Input from a Analyses/Options File

To specify circuit simulator analyses and output options for your simulation,

➤ From the schematic editor window, choose *Spectre – Stimulus – Edit Analyses/ Options File*.

A text file comes up in a text editor. The file starts with some suggested default values that you can use with most designs. You can keep these, delete them, or modify them. However, you must put at least one analysis statement into this file for a simulation to occur. This file is called `spectre.sim` and is placed in the run directory. This is the default sample file:

```
// Spectre Analyses and Output Options Statements
// Output Options
// simOptions options
// +      reltol = 1.00000000E-03
// +      vabstol = 1.00000000E-06
// +      iabstol = 1.00000000E-12
// +      temp = 27
// +      save = allpub
// +      currents = selected
// Analyses
// dc1 dc oppoint=logfile homotopy=all
// tran1 tran stop=1 errpreset=moderate
```

Note: When you add lines to the analysis and options file, any square bracket characters must be escaped: `\[` and `\]`. For example, a sweep function line must like this:

```
sweep1 sweep param=slope values=\[0.5n 1.0n 1.5n\]
```

Specifying Input from a Model Include File

To provide input from a model include file to your simulation,

1. From the schematic editor window, choose *Spectre – Stimulus – Model Include Files*.

   The Model Include Files form appears.

2. Click *Add File*.

   The *File Name (Full Path)* field is displayed.

3. Type the complete path to the model file.

4. Repeat this process for each model file required.

The files that you specify in this form are placed in `include` statements in a file called `spectre.include` in the run directory.

Specifying Input from a Model Include Path

To provide input from a model include path to your simulation,

1. From the schematic editor window, choose *Spectre – Stimulus – Model Include Paths*.

   The Model Include Paths form appears.

2. Enter the path to the files to be used for input into the simulation.
3. Click the *OK* button.

Specifying Input from a Design Variable

To provide input from design variables to your simulation,

1. From the schematic editor window, choose *Spectre – Stimulus – Design Variables*.

   The Design Variables form appears.

2. Click *Add Variable*.

   A text field is added at the bottom of the form.

   A file called `spectre.var` is created in the run directory.

"Design variables" are global parameters that are used throughout your design. You must define all of the global parameters in this form. If you try to access these parameters in included data in SPICE format (`.model`), you need to limit your parameters to lowercase. Refer to Parameters for more information.

Specifying Input from a Configuration File

To provide input from a configuration file to your simulation,

➤ From the schematic editor window, choose *Spectre – Stimulus – Edit Configuration File*.

The configuration file identifies the files the system uses to create the final run file for the simulation. The file, called `control`, starts with a default list of files from the run directory. You can add another file by copying the syntax shown and placing that file into the run directory for this design. You can also delete a file.

This is the default file content:

```
// Default Spectre Simulation run title card.
[?netlist]
[?spectre.include]
[?spectre.inp]
[?spectre.sim]
// End of Netlist
```

The order listed in this file controls the order in which files are included in the final netlist.

Specifying Input from Other Files

To provide input to your simulation that is not covered by the other files provided,

➤ From the schematic editor window, choose *Spectre – Stimulus – Edit Other File.*

This is a miscellaneous file. It can have any name, but if it is located outside of the run directory, you must provide the full path to its location.

Netlisting Your Design and Starting Simulations

To netlist your design and start simulations,

1. From the schematic editor window, choose *Spectre – Netlist/Simulate*.

   The Netlist and Simulate form appears.

   The *Simulation Run Directory* field is not editable: It displays the run directory you are working with.

   To change to a different directory,

   a. Click *Cancel*.

   b. From the schematic editor window, choose *Spectre – Initialize*.

   c. Do one of the following:

      - Click the *Browse* button, which opens the Library Browser, and find the library you are working on.
      - In the *Library Name*, *Cell Name*, and *View Name* fields, type the names of the library, cell, and view you are working on.

2. Verify that the *Simulator Name* cyclic field is set to *spectre*.

3. Turn on the appropriate *Run Actions* buttons.

   create run file is optional. Refer to Netlisting Overview for details on netlisting and the order of run actions.

4. (Optional) Set the simulation job to run in the background or foreground of your machine or the remote machine you have selected. Select foreground by turning the *Run In Background* button off.

   If you run the simulation job in background mode, you can use the *Job Priority* slider to control the job's access to the processor. A job with a lower priority number gets run sooner. The system tells you when your simulation has finished by displaying a message box.

   If you run the simulation job in foreground mode, you are locked out of menus until the task is complete; the status is displayed in the CIW.

Showing Outputs

Once you have tried to create a netlist or a run file, or run a simulation, the system produces the following output files, stored in the run directory.

File or Directory	Description
`si.env`	Internal environment file, created during the initialization process, describes design and its netlist; do not edit this file
`spectre.inp`	Stimulus file for the netlist, which you create prior to simulation
`netlist`	Netlist file of the design
`spectre.sim`	Analyses/Option file for the netlist, which you create prior to simulation
`spectre.include`	File of include statements to model data
`spectre.var`	File of design variables for the netlist, which you create prior to simulation
`control`	Netlist configuration file
`si.log`	Background log file, produced by the simulation process
`si.foregnd.log`	Foreground log file, produced by the simulation process
`si.out`	Spectre output log file, produced by the simulation process
`si.inp`	Final simulation run file, created with all the input files you specified before simulation
`si.raw`	Spectre simulation results directory, produced by the simulation process
`S.F.`	Link to `si.raw` directory

Each simulation run produces a log. You select background and foreground logs with separate commands. These logs list all the details of the operation of this product. Normally you do not need to read these files, but they are helpful for debugging problems.

➤ To examine the output files, do one of the following:

- To examine the run log file, choose *Spectre – Show Outputs – Show Run Log*.

  These files are useful for debugging.

- To examine the simulation output file, choose *Spectre – Show Outputs – Simulation Output*.

The system displays the `si.out` file. This is the output log from the Spectre simulator. This is the same information that is printed into the `spectre.out` file when running Spectre as a standalone product.

This is a typical example of a short Spectre log:

```
spectre (ver. 4.4.2.60 -- 10 Oct 1997).
Simulating `si.inp' on cds9886 at 5:52:23 PM, Mon Nov 10, 1997.
Convergence achieved in 2 iterations.
Total time required for dc analysis `dc1' was 450 ms.
*******************************************

Transient Analysis `tran1': time = (0 s -> 1 s)
*******************************************

......9......8......7......6......5......4......3......2......1......0
Number of accepted tran steps = 105.
Initial condition solution time = 20 ms.
Intrinsic tran analysis time = 290 ms.
Total time required for tran analysis `tran1' was 320 ms.
Aggregate audit (5:52:30 PM, Mon Nov 10, 1997):
Time used: CPU = 2.3 s, elapsed = 7 s, util. = 32.9%.
Virtual memory used = 1.37 Mbytes.

spectre completes with 0 errors, 0 warnings, and 0 notices.
```

- To examine the run file, choose *Spectre – Show Outputs – Show Run File*.

  A form for viewing your final netlist appears. The default name for your final netlist is `si.inp`.

Displaying Results

The Virtuoso schematic editor Spectre interface includes a waveform display tool suite that includes a waveform window, a calculator, and a results browser. This is the same tool suite available to users in the analog design environment.

The *Waveform Tool* menu varies according to the executable used to start the software.

For more information refer to the refer to the Virtuoso Visualization and Analysis Tool User Guide.

Name Mapping

To map instance terminals (currents) at device level (`nmos`, `pmos`, `npn`, `pnp`, etc.), the Virtuoso schematic editor Spectre interface needs to map information from the pins of those primitives in the `samples` library to the nodes of their models in the Spectre circuit simulator, which are not created by the netlister. If you customize components from the `samples` library, you need to override the `_dcdfTermMapList` variable to provide the pin-name–to–model-node name-mapping information for these customized devices. The `_dcdfTermMapList` variable has the following format:

```
_dcdfTermMapList=list(list("device_cell_name_1"
 list('("pin_name_1"
"model_node_name_in_Spectre_1")
'("pin_name_2"
"model_node_name_in_Spectre_2")
…))
list("device_cell_name_2"
list('("pin_name_"
"model_node_name_in_Spectre_1")
'("pin_name_2"
"model_node_name_in_Spectre_2")
…))
…
)
```

To view the current value of the `_dcdfTermMapList` variable,

➤ In the CIW, type

```
_dcdfTermMapList
```

The complete list for currently supported devices is displayed.

Use the Results Browser to view the currents saved at subcircuit level.

You can avoid using the Results Browser if you use a current probe (iprobe) to measure the currents between instance terminals at subcircuit level. You can select an instance terminal of iprobe, and the current flowing through the current probe is plotted. If you do not use an iprobe, but select an instance terminal at subcircuit level instead, the voltage signal of the net connected to that instance terminal will be plotted in error.

Checking the Status of Your Simulations

You can use the job monitor to track simulations, even if they are being run remotely. The job monitor identifies the location of the run directory; the day and time the simulation was started; on which computer the simulation is running; whether the simulation is running, has succeeded, or has failed, and what the simulation job priority is.

To track your simulation,

➤ Choose *Spectre – Job Monitor* to display the Analysis Job Monitor window.

The commands on the job monitor's *Command* menu affect only jobs whose check box is on.

Show Run Log displays the run log (just like the *Spectre – Output – Show Background Log* command) for the most recent simulation job for the same design. If you apply *Show Run Log* to an earlier simulation job, a dialog box telling you that the log has been overwritten (by the more recent simulation) appears.

Set Priority opens the Set Priority dialog box, which allows you to change the priority of a job, even while it is running.

Kill stops any job whose check box is on.

Suspend suspends any job whose check box is on.

Continue resumes any suspended job whose check box is on.

Remove Entry deletes any line whose check box is on from the job monitor window.

Running a Simulation Remotely

To run simulations on a remote workstation, you must configure the remote host. For remote workstations made by different manufacturers, with different operating systems, additional steps are needed.

The Virtuoso schematic editor Spectre interface can find and use all of the standard files in the local directory and run directory for each design, but special additions, such as include files, require that you provide a path to the location of the file. If you cannot specify a path from the target machine back to the source machine, you must copy the file to the target machine and specify a path local to that machine.

To run a simulation remotely,

1. Establish an account on the remote workstation.
2. From the schematic editing window, choose *Spectre – Options*.

   The Simulation Environment Options form appears.
3. Turn on *Run Simulation Remotely*.
4. In the *Simulation Host Name* field, type the name of the remote workstation.
5. Turn on *Simulation Host is Different Type* if the remote workstation is not the same brand or operating system as the local computer.

   You do not have to do this to simulate on another machine with a different version of the same operating system.
6. If the remote workstation is a different type, create a `.rhosts` file in the remote workstation's home directory and add this line to it:

   source_hostname user_name
7. Test the remote connect by choosing any file on the local workstation and typing

   `rcp` *some_file remote_workstation*`:/tmp`

   If `rcp` does not work, edit or disable your `.login` file and/or your shell file (such as `.cshrc`) on the remote workstation. Once `rcp` works, remote simulation should work.

   - If you get an error message similar to

     `account disabled`

     make sure you have an account on the remote workstation. If you have an active account, your password might have expired.
   - If you get an error message similar to

```
login incorrect
```

make sure that you have the `.rhosts` file in your account on the remote workstation and that it has the correct entry as described above.

2

Virtuoso Schematic Editor UltraSim Interface

This chapter describes how to use the Virtuoso® Schematic Editor UltraSim™ interface. The procedures described in this chapter are deliberately broad and generic. Requirements for your specific design might dictate procedures slightly different from those described here.

This product provides a solution for digital circuit designers to netlist a Virtuoso schematic editor transistor-level schematic for the Virtuoso UltraSim circuit simulator. It also provides a batch and scripting solution for automating various tasks. Some of the key features are listed below.

- UltraSim Netlister

  This solution provides an open simulation system (OSS) netlister for the Virtuoso UltraSim simulator from the Virtuoso schematic editor. The netlister works on devices with `ultrasim` views in the Cadence `sample` library. The netlister produces a hierarchical parameterized netlist in native circuit simulator syntax. The netlister supports global parameters and design variables and netlist processor (NLP) inheritance of parameters. All devices are netlisted using native circuit simulator syntax. For information on the Virtuoso UltraSim simulator, refer to the *Virtuoso UltraSim Simulator User Guide*.

- Updated Sample Library

  The `sample` library provided by Cadence for OSS netlisters has been enhanced for the Virtuoso UltraSim simulator. The devices `iprobe`, `phyres`, `phyresgnd`, `nmes`, and `pmes` have been added to the sample library to support additional circuit simulator models. In addition, most of the circuit simulator instance parameters have been added to the symbol cellview to aid in design creation.

- Improved Simulation Environment

  The OSS simulation environment (SE) has been enhanced for the Virtuoso UltraSim simulator, which now appears on the menu bar. You can now directly edit netlisting and support files. You can enter model files and design variables through the graphical user interface (GUI). The combination of the netlist and support files is controlled by the environment.

- Access to Analog Waveform Display for Waveform Display

  This solution provides access to the analog waveform display (AWD) tool from the analog design environment (ADE). The circuit simulator default format for writing data out is parameter storage format (PSF), which is read in by the AWD tool. The simulation environment provides access to this tool and the Results Browser and Waveform calculator. You can choose which signals you want to plot using the Results Browser. For more information about these tools, refer to the analog circuit design environment documentation.

- OCEAN Batch and Scripting Capabilities

  The Virtuoso schematic editor UltraSim interface uses the OCEAN batch and scripting tool to control the simulator with scripts or interactive commands. It also gives you access to the AWD tool, which allows you to measure waveform data and print the measurements to the screen or a file. All of the standard measurement capabilities of a signal calculator are available from the OCEAN command prompt, as are any macros that you create. You can output data files with either the `fprintf` or `ocnPrint` commands. For more information about OCEAN, refer to the OCEAN Reference.

Starting the Virtuoso Schematic Editor UltraSim Interface

To start the Virtuoso Schematic Editor UltraSim interface,

1. Start the Cadence software.
2. From the command interpreter window (CIW), do one of the following:
   - Start a new design by choosing *File – New*.
   - Start an existing design by choosing *File – Open*.
3. After setup, click *OK*.

   The Virtuoso schematic editing window appears.
4. Choose *Launch – Simulation – UltraSim*.

   The UltraSim menu is added to the main menu bar.

   If you have not accessed another design during this session or you have never created a circuit simulator run directory for this design, all but one of the commands in the *UltraSim* menu are grayed Initout.
5. Choose *UltraSim – Initialize*.

   The Initialize Environment form appears.
6. In the *Simulation Run Directory* field, type the path to the directory where you want to place all of the simulation-related files.

   By default, the files are placed in `UltraSim.run1` in your startup directory.

   If the path defines a new directory, the Initialize Environment form expands. You can change the name of the run directory or specify a different design. The *Browse* button brings up the Library Browser, which automatically fills in the *Library Name*, *Cell Name*, and *View Name* fields when you select a new design.
7. Click *OK*.

Specifying Input for the Simulation

The Virtuoso schematic editor UltraSim interface offers you several ways to provide input to your simulation and to modify the input. Before simulation, you have the option of combining all of these files with the netlist file to create a run file.

After you set up your simulation options, you can specify all the input files for the simulation. The *UltraSim – Stimulus* menu lists each of these files. When you select these commands, a text editor window opens up displaying the file. If an empty text editor window appears, you need to create the file.

For more information about specifying inputs for simulation, see Chapter1, Virtuoso Schematic Editor Spectre Interface.

Specifying Input from a Stimulus File

To provide input from a stimulus file to your simulation,

1. From the schematic editor window, choose *UltraSim – Stimulus – Edit Stimulus File*.

   A text file opens in a text editor window, for example `emacs`. The file is saved as `UltraSim.inp` and is placed in the run directory. You do not have to add source statements if you have included sources in your designs.

2. If you use the text file, connect the sources to the net names on the schematic.

   Schematic names are mapped to lowercase. If you use circuit simulator syntax, you must use the lowercase form of net names you are connecting to. Connect to global nets such as `ground` from the stimulus file. Global nets use a `!` at the end of the net name and that character must be escaped. In addition, any escaped characters entered in this file must be escaped twice (`gnd\\!`) because one escape is removed during the merging of netlist files. For more information, see Netlisting Overview.

Note: When you add lines to the analysis and options file, the square brackets characters must be escaped: `\[` and `\]`. For example, a sweep function line must like this:

```
sweep1 sweep param=slope values=\[0.5n 1.0n 1.5n\]
```

By default, the file should have UltraSim simulator syntax. If you use SPICE syntax, the line

```
simulator lang=spice
```

must precede the SPICE syntax.

Specifying Input from a Netlist File

To provide input from a netlist file to your simulation,

➤ From the schematic editor window, choose *UltraSim – Stimulus – Edit Netlist File*.

If you have already netlisted your design using this product, a text editor window opens to display the netlist file, which you can then modify. The netlist file produced in this way is located in the circuit simulator run directory you must create.

If the text editor window opens with an empty file, you have yet to netlist this design.

The file is called `netlist` and is located in the run directory. This is not the final netlist: it does not include any of the data from the other stimulus files. If netlisting has occurred, it will contain connectivity information for the associated schematic. If you modify this file, you might have to escape certain characters. Also, the next time you netlist this file, all modifications to this file will be lost.

Specifying Input from a Analyses/Options File

To specify circuit simulator analyses and output options for your simulation,

➤ From the schematic editor window, choose *UltraSim – Stimulus – Edit Analyses/ Options File*.

A text file comes up in a text editor. The file starts with some suggested default values that you can use with most designs. You can keep these, delete them, or modify them. However, you must put at least one analysis statement into this file for a simulation to occur. This file is called `UltraSim.sim` and is placed in the run directory. This is the default sample file:

```
// UltraSim Analyses and Output Options Statements
// Output Options
// simOptions options
// +      reltol = 1.00000000E-03
// +      vabstol = 1.00000000E-06
// +      iabstol = 1.00000000E-12
// +      temp = 27
// +      save = allpub
// +      currents = selected
// Analyses
// dc1 dc oppoint=logfile homotopy=all
// tran1 tran stop=1 errpreset=moderate
```

Note: When you add lines to the analysis and options file, any square bracket characters must be escaped: `\[` and `\]`. For example, a sweep function line must like this:

```
sweep1 sweep param=slope values=\[0.5n 1.0n 1.5n\]
```

Specifying Input from a Model Include File

To provide input from a model include file to your simulation,

1. From the schematic editor window, choose *UltraSim – Stimulus – Model Include Files*.

   The Model Include Files form appears.

2. Click *Add File*.

   The *File Name (Full Path)* field is displayed.

3. Type the complete path to the model file.

4. Repeat this process for each model file required.

The files that you specify in this form are placed in `include` statements in a file called `UltraSim.include` in the run directory.

Specifying Input from a Model Include Path

To provide input from a model include path to your simulation,

1. From the schematic editor window, choose *UltraSim – Stimulus – Model Include Paths*.

   The Model Include Paths form appears.

2. Enter the path to the files to be used for input into the simulation.
3. Click the *OK* button.

Specifying Input from a Design Variable

To provide input from design variables to your simulation,

1. From the schematic editor window, choose *UltraSim – Stimulus – Design Variables*.

   The Design Variables form appears.

2. Click *Add Variable*.

   A text field is added at the bottom of the form.

   A file called `UltraSim.var` is created in the run directory.

"Design variables" are global parameters that are used throughout your design. You must define all of the global parameters in this form. If you try to access these parameters in included data in SPICE format (`.model`), you need to limit your parameters to lowercase. Refer to Parameters for more information.

Specifying Input from a Configuration File

To provide input from a configuration file to your simulation,

➤ From the schematic editor window, choose *UltraSim – Stimulus – Edit Configuration File*.

The configuration file identifies the files the system uses to create the final run file for the simulation. The file, called `control`, starts with a default list of files from the run directory. You can add another file by copying the syntax shown and placing that file into the run directory for this design. You can also delete a file.

This is the default file content:

```
// Default UltraSim Simulation run title card.
[?netlist]
[?UltraSim.include]
[?UltraSim.inp]
[?UltraSim.sim]
// End of Netlist
```

The order listed in this file controls the order in which files are included in the final netlist.

Specifying Input from Other Files

To provide input to your simulation that is not covered by the other files provided,

➤ From the schematic editor window, choose *UltraSim – Stimulus – Edit Other File.*

This is a miscellaneous file. It can have any name, but if it is located outside of the run directory, you must provide the full path to its location.

Netlisting Your Design and Starting Simulations

To netlist your design and start simulations,

1. From the schematic editor window, choose *UltraSim – Netlist/Simulate*.

   The Netlist and Simulate form appears.

   The *Simulation Run Directory* field is not editable (displays the run directory you are working with).

   To change to a different directory,

   a. Click *Cancel*.

   b. From the schematic editor window, choose *UltraSim – Initialize*.

   c. Do one of the following:

      - Click the *Browse* button, which opens the Library Browser, and find the library you are working on.
      - In the *Library Name*, *Cell Name*, and *View Name* fields, type the names of the library, cell, and view you are working on.

2. Verify that the *Simulator Name* cyclic field is set to *UltraSim*.
3. Turn on the appropriate *Run Actions* buttons.
4. (Optional) Set the simulation job to run in the background or foreground of your machine or the remote machine you have selected. Select foreground by turning the *Run In Background* button off.

   If you run the simulation job in background mode, you can use the *Job Priority* slider to control the job's access to the processor. A job with a lower priority number gets run sooner. The system tells you when your simulation has finished by displaying a message box.

   Note: If you run the simulation job in foreground mode, you are locked out of menus until the task is complete (job status is displayed in the CIW).

Showing Outputs

Once you have created a netlist or a run file, or run a simulation, the system produces the output files which are stored in the run directory.

For more information about specifying inputs for simulation, see Chapter1, Virtuoso Schematic Editor Spectre Interface.

Displaying Results

The Virtuoso schematic editor UltraSim interface includes a waveform display tool suite that includes a waveform window, a calculator, and a results browser. This is the same tool suite available in ADE.

The *UltraSim – Waveform Tool* menu varies according to the executable used to start the software. For more information on how to use these tools to select results and display them, refer to the Virtuoso Visualization and Analysis Tool User Guide.

Checking Simulation Status

You can use the job monitor to track simulations, even if they are being run remotely. The job monitor identifies the location of the run directory; the day and time the simulation was started; on which computer the simulation is running; whether the simulation is running, has succeeded, or has failed; and what the simulation job priority is.

To track your simulation,

➤ Choose *UltraSim – Job Monitor*.

The Analysis Job Monitor window appears.

For more information about the job monitor commands, see Chapter1, Virtuoso Schematic Editor Spectre Interface.

Running a Simulation Remotely

To run simulations on a remote workstation, you must configure the remote host. For remote workstations made by different manufacturers, with different operating systems, additional steps are needed.

The Virtuoso schematic editor UltraSim interface can find and use all of the standard files in the local directory and run directory for each design, but special additions, such as include files, require that you provide a path to the location of the file. If you cannot specify a path from the target machine back to the source machine, you must copy the file to the target machine and specify a path local to that machine.

To run a simulation remotely,

1. Establish an account on the remote workstation.
2. From the schematic editing window, choose *UltraSim – Options*.

   The Simulation Environment Options form appears.
3. Turn on *Run Simulation Remotely*.
4. In the *Simulation Host Name* field, type the name of the remote workstation.
5. Turn on *Simulation Host is Different Type* if the remote workstation is not the same brand or operating system as the local computer.

   You do not have to do this to simulate on another machine with a different version of the same operating system.
6. If the remote workstation is a different type, create a `.rhosts` file in the remote workstation's home directory and add this line to it:

   source_hostname user_name
7. Test the remote connect by choosing any file on the local workstation and typing

   `rcp` *some_file remote_workstation*`:/tmp`

   If `rcp` does not work, edit or disable your `.login` file and/or your shell file (such as `.cshrc`) on the remote workstation. Once `rcp` works, remote simulation should work.

   - If you get an error message similar to

     `account disabled`

     make sure you have an account on the remote workstation. If you have an active account, your password might have expired.
   - If you get an error message similar to

```
login incorrect
```

make sure that you have the `.rhosts` file in your account on the remote workstation and that it has the correct entry as described above.

A

Netlisting Overview

Global Nets

The Virtuoso Schematic Editor to Spectre netlister is a hierarchical netlister. It supports hierarchical designs and does not flatten out the netlists. The netlister creates parameterized netlists. In other words, you can use parameters throughout the design hierarchy and the netlister does not evaluate the values of these parameters when generating the netlists. This enables you to dynamically alter the values of these parameters during analyses.

Name mapping is performed during the netlisting. Since this is a hierarchical netlister and Spectre allows mixed cases and escaped special characters, the name mapping is minimal. The most noticeable name mapping is that the net names are mapped into lowercase, due to the lowercase requirement of global nets/signals in Spectre syntax.

The language syntax used in the netlisting is Spectre syntax. However, model files can be either in SPICE syntax or Spectre syntax. By default, the model files are in SPICE syntax. If a model file is in Spectre syntax, make sure the first line in the model file is

```
simulator lang=spectre
```

The netlister generates readable, modularized netlists. Each subcircuit is documented in a single block. To create a final simulation input file, two steps are involved: choosing *Spectre – Netlist/Simulate – (netlist* option) and choosing *Spectre – Netlist/Simulate – (create run file)* option. When you choose *netlist*, a structural netlist `netlist` is created in the run directory. When you choose *create run file*, a final simulation input file `si.inp` is created in the run directory. You can also use this file for standalone simulation.

Basically, *netlist* extracts connectivity, parameter values, parameter inheritance, and all the data stored in the schematics, and it reads the global parameters defined by the Design Variables form. The *create run file* option combines the raw netlist file, included model files statements (defined by the Model Include Files form), Spectre source statements (*Edit Stimulus File*), output options, and analyses statements (*Edit Analyses/Options File*) to create a final run file that is ready to be simulated.

In this process, if you modify your schematics, change your parameter values or other data stored in the schematics, or modify your global parameters, you need to choose *netlist* again.

If you only modify your included model files information, or Spectre source statements or output options and analyses statements, you do not need to re-netlist. Choose *create run file* again only to obtain a new `si.inp` file.

During the netlisting, bus signals in schematics are split into single signals in netlists. Iterated instances in schematics are flattened into individual instances. Patch cords in schematics are ignored and their destination signals are used in netlists. Bundles in schematics are split into single signals. Global signals in schematics are retained in netlists.

The netlister does not automatically create model cards for components. Use the Model Include Files form to define the model files to be used. You can also include default model cards for instances for which you did not specify a model name. If you do specify a model name, the model name in the model card is the same as the library cell name.

Additional components can be inserted in the design by directly editing the netlist. You can also use this method to include Verilog-A modules. Use `include` file statements, and in the case of the analog HDLs, use the `ahdl_include` statements. Refer to *Cadence Verilog-A Language Reference* for more information.

Parameters

Component/Subcircuit Parameters

Parameters used for components and subcircuits are stored as DFII database properties. For each component, a set of default properties with their default values is created on the component symbol as cellview properties. When a component is instantiated, you can display these properties in the schematic editor's tool's Edit Object Properties form with their default values shown as master values. You can enter the instance values for certain properties as needed. The instance values overwrite the master values (default values) when the netlist is created. You can also add new properties at the instance level. The netlister writes only the parameters/values that are different from their defaults.

Global Parameters (Design Variables)

Global parameters are global to the entire netlist and can be used anywhere inside the design hierarchy as any device parameter value. For example, on a resistor you can specify the resistance as *r=res* where *res* is a global parameter. The Design Variables form allows you to define the global parameters with their default values. To reduce the netlisting time, the netlister does not embed global parameter searching. It is your responsibility to make sure that all the global parameters used in the entire hierarchy are defined in the Design Variables form.

When entering global parameters in the Design Variables form, you must make sure that parameters that are functions of other parameters are entered after the dependent parameters are entered. For example, *a=5*b* and *b=10* require you to enter parameter *b* before parameter *a* because *a* uses *b*. When you use a global parameter in the design hierarchy, it is entered as a `string` type property (either within an expression or by itself) on the components' instances or the subcircuits' instances.

The netlister creates `parameters` statements based on the variables and their default values defined in the Design Variables form. Variables can be entered as either a constant or as an expression. The netlister does not evaluate the values of the global parameters in the design hierarchy. Later, you can use the `alter` and `sweep` statements to change their values during analyses. For information on the parameter capability in the circuit simulator, see the *Spectre Circuit Simulator and Accelerated Parallel Simulator User Guide* and *Spectre Circuit Simulator Reference*.

Parameter Inheritance

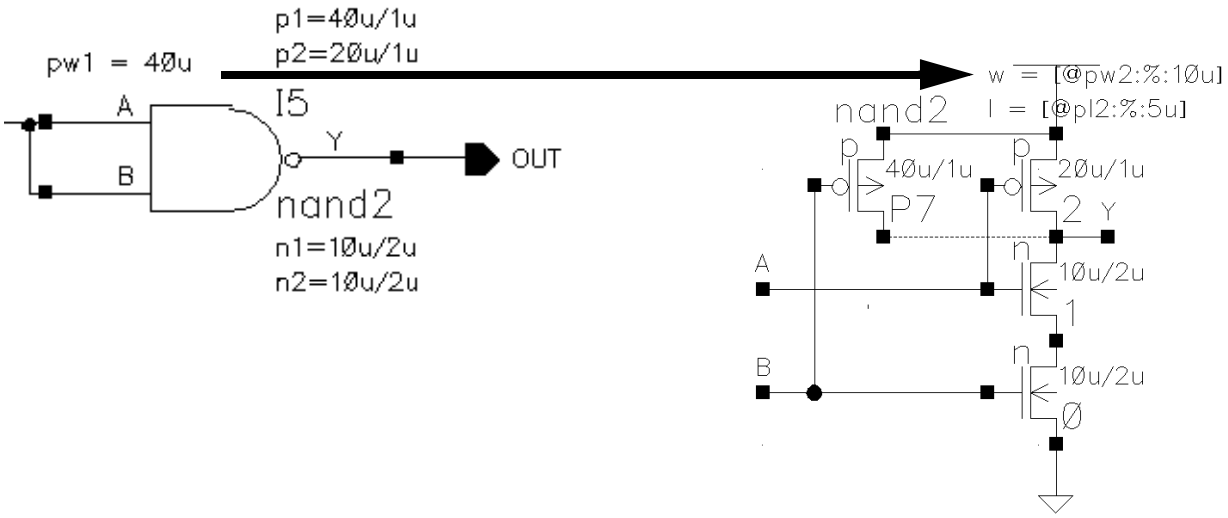
If you are using the `sample` library, parameter inheritance is handled via netlist processor (NLP) expressions in DFII properties. The syntax of the NLP expression used for parameter inheritance is

`[@`*`parameter_name`*`:%:`*`default_value`*`]`

where *`parameter_name`* is the name of the parameter at the upper-level design from which values are inherited and *`default_value`* is the default value of this upper-level parameter.

For example, at the current level of a design, there is a two-input NAND gate instance. A NAND subcircuit consists of two NMOS transistors and two PMOS transistors. A parameter of the NAND instance at the current level, `pw`, is inherited down to the lower-level parameter `w` of the `pmos` instance in the inverter schematic. To make this possible, type the expression `[@pw:%:10u]` as the value of the parameter `w` of the `pmos` instance in the inverter schematic (where `10u` is the default value of `pw`). The parameter `w` is created as an nlpExpr type of a DFII property.

The netlister reads the NLP expressions during netlisting and creates parameter statements and instance statements for the subcircuits accordingly. The netlister does not evaluate the values of the inherited parameters. For information on entering and changing parameters in a design, refer to the Edit Object Properties form in the *Virtuoso Schematic Editor User Guide*.



Expressions

You can use the string-type of properties to define the expressions that Spectre supports. The expressions need to be in Spectre syntax. The netlister reads and passes the expressions without evaluating their values.

Note: You can only use global parameters inside the expressions.

For example, the value of a parameter `w` on a MOS device is an expression `aw+3*bw`, where `aw` and `bw` are global parameters that need to be defined in the Design Variables form.

For information on the expression capability in the circuit simulator, see the *Spectre Circuit Simulator and Accelerated Parallel Simulator User Guide* and *Spectre Circuit Simulator Reference*.