

Simulation Environment Help

**Product Version IC23.1
June 2023**

© 2023 Cadence Design Systems, Inc. All rights reserved.
Printed in the United States of America.

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

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

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

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

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

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

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

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

Contents

1

Introduction to Simulation Environment 5

<u>Licensing Requirements</u>	5
<u>The Simulation Menu</u>	7

2

Netlist Generation and Customization 9

<u>Netlist Customization With the .simrc File</u>	10
<u>Hierarchy of Netlisting Views and Switching Views</u>	10
<u>Selection of Netlisting Views from a Hierarchy Using Stopping Views</u>	10
<u>Default View and Stop List Override</u>	10
<u>Variables for Incremental Netlisting</u>	12
<u>Customization of Scale Factors</u>	13
<u>Substitution Functions</u>	14
<u>Location of the .simrc File</u>	15

3

Simulation Runs 21

<u>Initializing the Simulation Environment</u>	23
<u>Specifying Simulation Environment Options</u>	26
<u>Creating the Input Stimulus in the Control File</u>	28
<u>Running a Simulation from the Schematic</u>	29
<u>Remote Simulation</u>	31
<u>Setting Up Remote Simulation From the Schematic</u>	32
<u>Files Needed for Simulation in the Command-Line Interface</u>	33
<u>si.env file</u>	33
<u>control file</u>	34
<u>Setting Up a Simulation Using the Command-Line Interface</u>	35
<u>Setting Up Remote Simulation From the Command-Line Interface</u>	36
<u>Types of Simulations in Command-Line Mode</u>	37

Simulation Environment Help

<u>Functions Used to Run Simulations in All Modes</u>	39
<u>Running a Simulation from the Command-Line Interface</u>	41
<u>Running si in Replay Mode</u>	42
<u>Running an Interactive Simulation</u>	43
<u>Customization of the Simulation Environment Using the .simrc File</u>	44
<u>Variables to Customize Simulations</u>	45
<u>Viewing Waveform Results in the Schematic</u>	46
<u>Viewing Waveform Results in Register Format</u>	47
<u>Viewing a Specified Text File</u>	48
<u>Viewing the Run Log of a Job</u>	49
<u>Viewing the Text Output in SE</u>	50
<u>Viewing Global Errors in SE</u>	51
<u>Viewing Highlighted Errors in SE</u>	52
<u>Managing Jobs Using the Job Monitor</u>	53

A

<u>SE Form Reference</u>	57
<u>Analysis Job Monitor Form</u>	58
<u>Edit Run Directory File Form</u>	59
<u>Initialize Environment Form</u>	60
<u>Netlist and Simulate Form</u>	61
<u>Set Priority Form</u>	63
<u>Show Registers Form</u>	64
<u>Show Run File Form</u>	65
<u>Show Waveforms Form</u>	66
<u>Simulation Environment Options Form</u>	67

B

<u>Files Used by the Cadence SILOS II Simulator</u>	69
<u>Sample control File for Cadence SILOS II</u>	70
<u>Sample si.inp File Generated for Cadence SILOS II</u>	72

Introduction to Simulation Environment

This document describes the Virtuoso® Simulation Environment (SE) and is aimed at designers who want to netlist and simulate designs maintained in the Virtuoso Studio design environment and assumes that you are familiar with:

- The Virtuoso Studio design environment and application infrastructure mechanisms designed to support consistent operations between all Cadence tools.
- The applications used to design and develop integrated circuits in the Virtuoso Studio design environment, notably Virtuoso Schematic Editor.

The Cadence Simulation Environment (SE) allows you to run simulations from the graphical user interface (GUI) or a command line. The GUI lets you use SE menus and forms. The non-graphical environment can be accessed using SE commands in a UNIX® xterm window running si or SKILL commands in the Virtuoso CIW.

SE supports user-defined simulators and the following standard simulators:

- System HILO
- HSPICE
- Verilog-XL Simulator

Licensing Requirements

For information on licensing in the Virtuoso Simulation Environment, see [Virtuoso Software Licensing and Configuration Guide](#).

Related Topics

[The Simulation Menu](#)

[Initializing the Simulation Environment](#)

[Setting Up a Simulation Using the Command-Line Interface](#)

Simulation Environment Help

Introduction to Simulation Environment

Customization of the Simulation Environment Using the .simrc File

Digital Design Netlisting and Simulation SKILL Reference

The Simulation Menu

To access the Simulation menu from the schematic:

- ➡ In the schematic window, choose *Launch — Plugins — Simulation — Other*.

The *Simulation* menu gets added to the menu bar. It has the following commands to simulate your design.

Command	Description
<i>Initialize</i>	Initializes the environment for simulation.
<i>Options</i>	Sets additional simulation run options.
<i>Application Options</i>	Displays forms that are specific to your simulator. You must write the necessary SKILL code to activate this menu choice.
<i>Stimulus</i>	Lets you edit any file in the current simulation run directory.
<i>Netlist/Simulate</i>	Generates a netlist, runs a simulation using an existing netlist, or runs a complete simulation, including generating a new netlist.
<i>Interactive</i>	Starts an interactive simulation session.
<i>Show Outputs</i>	Displays the run log (<code>si.log</code>) of a background simulation, the run log (<code>si.foregnd.log</code>) of a foreground simulation, the text output (<code>si.out</code>) of the simulator, or global errors (<code>global.err</code>) found during netlisting or simulation with supported interfaces.
<i>Show Waveforms</i>	Displays the waveforms produced during a simulation.
<i>Show Registers</i>	Displays the waveforms produced during a simulation in register form.
<i>Job Monitor</i>	Lets you check the status of a background analysis job, monitor its output, suspend it, change its execution priority, or stop it.

Related Topics

[SE Form Reference](#)

[Viewing Waveform Results in the Schematic](#)

Simulation Environment Help

Introduction to Simulation Environment

[Viewing Waveform Results in Register Format](#)

[Viewing a Specified Text File](#)

Netlist Generation and Customization

When the Simulation Environment (SE) netlists your design, it takes your design hierarchy (extracted schematic or layout) and the simulator primitive component library and generates a network description containing all instances, nets, and models in an appropriate format for your simulator.

When you extract and save your schematic, the system takes the connectivity information from the drawing and saves it to the disk. The netlister uses this data, the simulation data in the Cadence `basic` and `sample` libraries, and the modeling data to create the netlist for simulation.

The traversal method used in the design hierarchy to produce the netlist and the netlist syntax depends on your simulator choice. Consider that you want the netlist for a Verilog simulation to be at the logic gate level because Verilog can simulate primitives such as AND gates and AOIs. Additionally, you want the netlist for a SPICE simulation of the same design to be at the transistor level because SPICE cannot simulate logic gates.

Some Cadence netlisters flatten the hierarchy and produce an expanded description of the design. For HSPICE, the netlister can create either a flattened or a hierarchical netlist. The Verilog-XL netlister produces only a hierarchical netlist.

Related Topics

[Netlist Customization With the .simrc File](#)

[Variables for Incremental Netlisting](#)

[Customization of Scale Factors](#)

[Substitution Functions](#)

[Location of the .simrc File](#)

Netlist Customization With the .simrc File

Based on your requirement, you can customize netlisting in SE. The following methods let you add or modify SKILL variables in the `.simrc` file for the required customization:

- Hierarchy of Netlisting Views and Switching Views
- Selection of Netlisting Views from a Hierarchy Using Stopping Views
- Default View and Stop List Override

Hierarchy of Netlisting Views and Switching Views

The view switch list is defined using the `simViewList` SKILL variable. SE searches for each view in this list in the given order in the cell hierarchy. The first view found is switched with the symbol.

When a device referenced in the schematic is located, the instance of the symbol must be associated with its corresponding schematic or simulator primitive. This process is called switching views. Using the `simViewList` variable, you can define the list of valid views that must be used as switching views for the target simulator. If no valid view is found, an error message is displayed to indicate netlist generation failure.

Selection of Netlisting Views from a Hierarchy Using Stopping Views

A stopping view is the most detailed description of devices for a simulation. The stopping point view list for the target simulator is defined using the `simStopList` SKILL variable. SE checks whether each view in the `simViewList` also exists in the `simStopList`.

- If the view exists in both `simViewList` and `simStopList`, the expansion stops and the SE netlist function prints the connectivity information of the instance in the netlist file.
- If the view only exists in the `simStopList`, SE continues to locate the next view that exists only in the `simViewList` and expands it.

Default View and Stop List Override

To override the default view list or stop list, specify the new list for the current simulator in the `.simrc` file. This sets the internal list for the current simulator to the `.simrc` value. You must set the view and stop lists separately for each simulator.

Simulation Environment Help

Netlist Generation and Customization

The following table summarizes the `simViewList` and `simStopList` variables that correspond to the internal variables listed for SPICE and HSPICE simulators.

simSimulator	simViewList	simStopList
SPICE	<code>spiceSimViewList</code>	<code>spiceSimStopList</code>
HSPICE	<code>hspiceSimViewList</code>	<code>hspiceSimStopList</code>

The following example shows how the netlister uses the view and stop list variables to control its traversal of the design hierarchy:

```
spiceSimViewList = '("spice" "cmos.sch" "schematic")  
spiceSimStopList = '("spice")
```

Here, the netlister stops expansion when it finds a cell with a view named `spice` and writes the device into the netlist file. If a `spice` view does not exist, the netlister tries to find the `cmos.sch` view to expand it. If neither a `spice` nor a `cmos.sch` view exists, the netlister looks for the `schematic` view. If none of these views exist, the netlister generates an error message.

Related Topics

[simViewList](#)

[simStopList](#)

[Netlist Generation and Customization](#)

[Variables for Incremental Netlisting](#)

[Customization of Scale Factors](#)

[Substitution Functions](#)

[Location of the .simrc File](#)

Variables for Incremental Netlisting

The following incremental netlisting variables let you reduce netlisting time by eliminating unnecessary netlisting.

- `simNotIncremental`: When set to its default value `nil`, lets you netlist only the parts of your design modified since the last netlisting of the design.
- `simReNetlistAll`: When set to `t`, generates a new netlist on all the cellviews in your entire design. The default is `nil`.
- `simNetlistHier`: When set to `t`, runs the hierarchical netlister.

Related Topics

[`simNotIncremental`](#)

[`simReNetlistAll`](#)

[`simNetlistHier`](#)

[Netlist Generation and Customization](#)

[Netlist Customization With the `.simrc` File](#)

[Customization of Scale Factors](#)

[Substitution Functions](#)

[Location of the `.simrc` File](#)

Customization of Scale Factors

The netlister can scale time and capacitance values. The SE variables `simTimeUnit` and `simCapUnit` let you define the scale factors for time and capacitance. For both variables, the value to be scaled is divided by the scale factor. The default value of `simTimeUnit` is `1e-9` (nanoseconds), and the default value of `simCapUnit` is `1e-15` (femtofarads). You can customize the scale factors by specifying new `simTimeUnit` and `simCapUnit` values in your `.simrc` file.

Related Topics

[`simTimeUnit`](#)

[`simCapUnit`](#)

[Netlist Generation and Customization](#)

[Netlist Customization With the `.simrc` File](#)

[Variables for Incremental Netlisting](#)

[Substitution Functions](#)

[Location of the `.simrc` File](#)

Substitution Functions

If needed, Simulation Environment (SE) maps the names assigned on your schematic to names that are valid in the simulator syntax. When you refer to the name of a net, instance, or file in your control file or an included file, you must use the appropriate substitution to ensure correct translation.

The following table lists the substitution functions provided by SE:

Function	Description
[#netname]	Replaces [#netname] with the netlister-assigned name of the net corresponding to <i>netname</i> .
[\$instname]	Replaces [\$instname] with the netlister-assigned name of the instance corresponding to <i>instname</i> .
[!filename]	Replaces [!filename] with the contents of the file named <i>filename</i> and continues to do substitutions in the included file.
[?filename]	Same as [!filename] except no error message is reported if the file does not exist.
[n!filename]	Replaces [n!filename] with the contents of the file named <i>filename</i> and does not do substitutions in the included file.
[n?filename]	Same as [n!filename], except no error message is generated if the file does not exist and no substitutions are done in the included file.

Related Topics

[Netlist Generation and Customization](#)

[Netlist Customization With the .simrc File](#)

[Variables for Incremental Netlisting](#)

[Customization of Scale Factors](#)

[Location of the .simrc File](#)

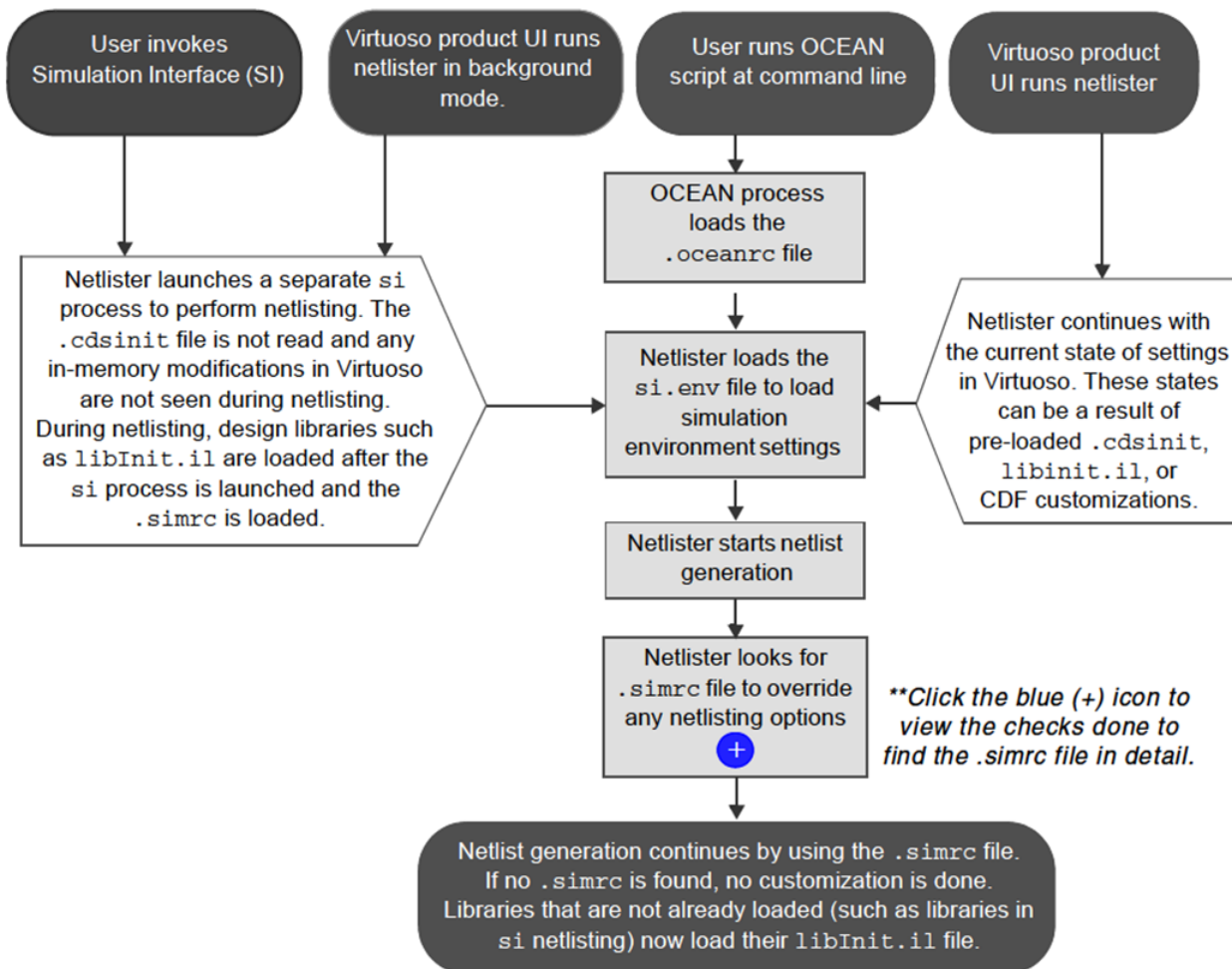
Location of the .simrc File

Various Virtuoso tools load the `.simrc` file during the netlisting process. Depending on the project requirements or setup, a designer may keep the `.simrc` file at different locations. If your simulation setup contains a `.simrc` file at multiple locations, refer to the lookup order flowchart in the following figure to understand which of these files is selected for netlist customization.

This flowchart describes the locations and the sequence in which Virtuoso looks for a `.simrc` file. It also provides guidance to designers on the best location in which to place the `.simrc` file if they want Virtuoso to use it for netlist customization.

Lookup order for the .simrc file

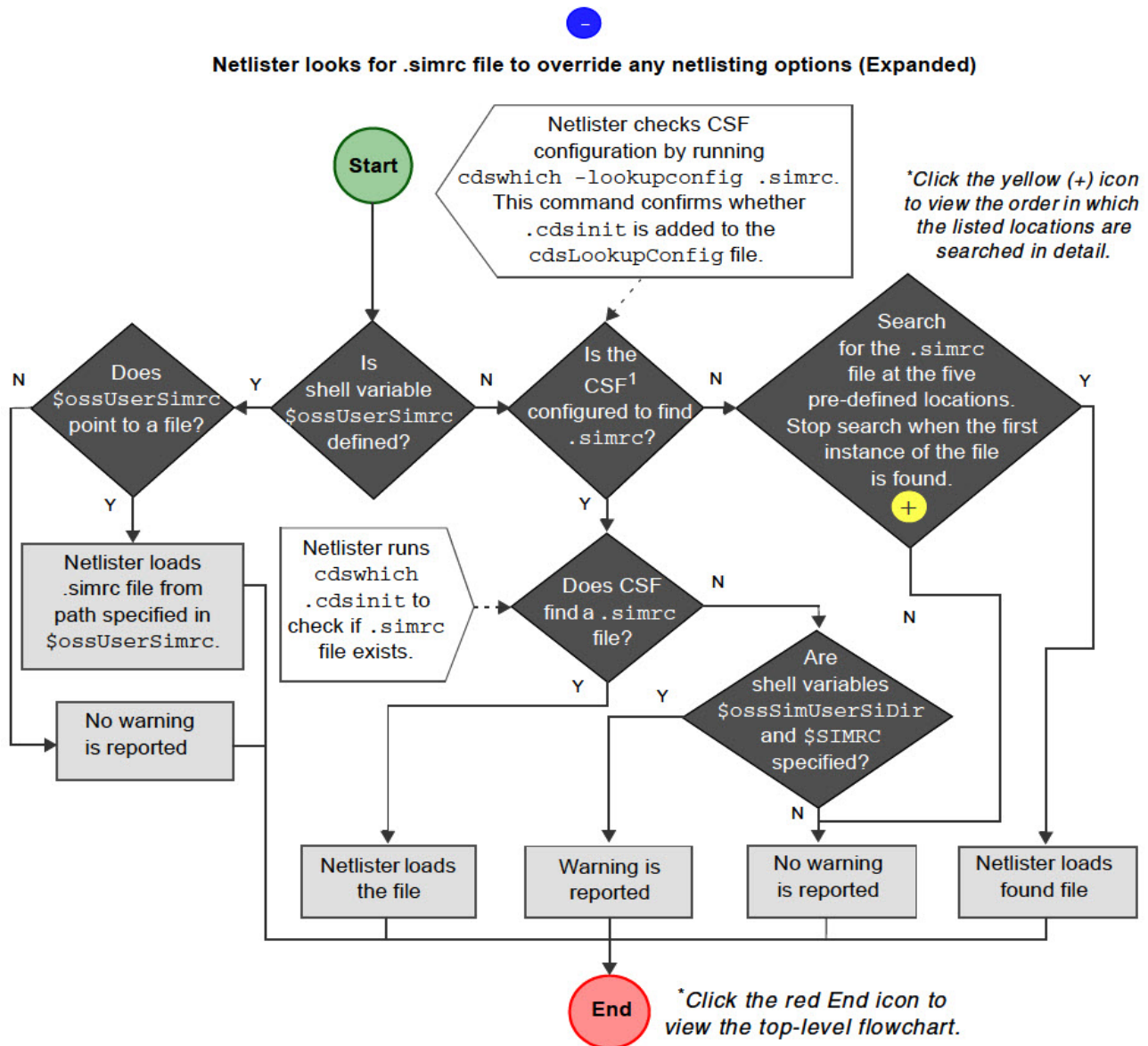
***Click the topmost gray boxes to know related details.*



Simulation Environment Help

Netlist Generation and Customization

Checks to detect the .simrc file



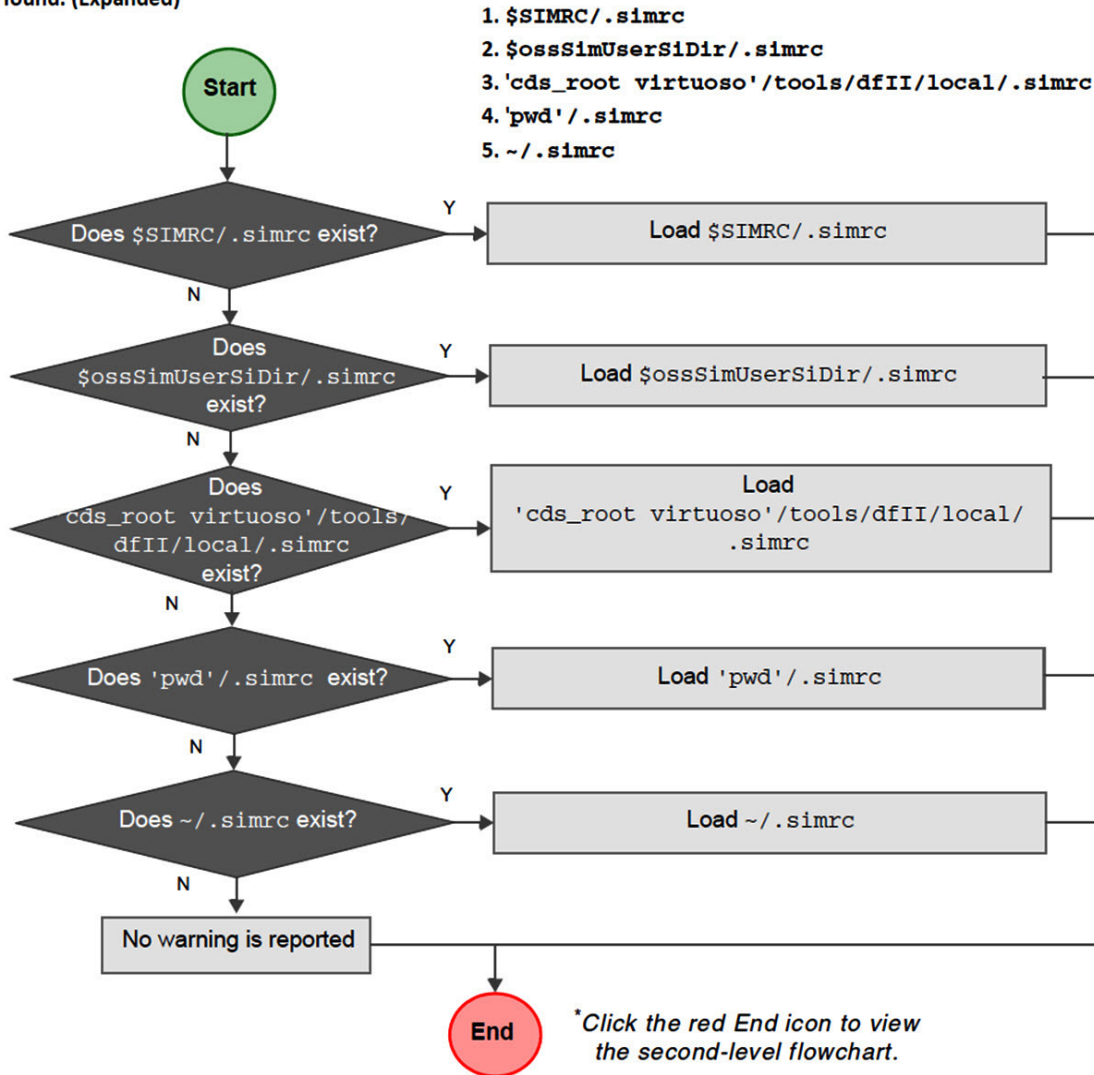
*Cadence Search Framework. For more details, see [Cadence Setup Search File](#).

Simulation Environment Help

Netlist Generation and Customization

Search to locate the .simrc file

Search for a .simrc file in the following pre-defined locations in the given order. Stop search when first instance of file is found. (Expanded)



Important

Set the `$ossUserSimrc` variable to a dummy value to avoid loading the .simrc file.

The `$SIMRC` and `$ossSimUserSiDir` shell environment variables are set using the `setenv` command before you start a Virtuoso session.

For example: `setenv SIMRC <path>`

Simulation Environment Help

Netlist Generation and Customization

All netlisters first load the `si.env` file, which prepares the simulation environment. The `.simrc` file, if it exists, is loaded whenever netlisting starts in the simulation flow. This file overrides the settings defined in the `si.env` file and sets default variables for simulations run by a specific user or system without impacting other simulations.

The netlisters that use the `.simrc` file can be grouped into the following categories:

- Netlisters that are invoked by running the `si -batch` command at the command line or through interactive simulation.

In this case, the `.cdsinit` file is not loaded and a separate `si` process is launched to perform the netlisting. The `si.env` is loaded, followed by the `.simrc` file, and changes in the current session, such as effective CDF modifications or other in-memory modifications, are visible in the netlist.

- Netlisters that run in the background when the *Run in Background* check box is selected in the Virtuoso GUI.

For example, the CDL Out form used by Virtuoso lets you select or deselect the *Run in Background* check box. The `.cdsinit` file or the current environment is not used and in addition to the background `virtuoso` process, a separate `si` process is launched to perform the netlisting. The `si.env` is loaded, followed by the `.simrc` file. For example, CDL, Verilog, or VHDL. Changes in the current session are not visible in the netlist.

- Netlisters that are invoked when an OCEAN script is run at the command-line.

The `.oceanrc` file is loaded, followed by `si.env`, and then the `.simrc` file during netlisting. In-memory modifications are visible in the netlist.

- Netlisters that are invoked by the Virtuoso GUI.

For example, Spectre, HspiceD, CDL, Verilog, or VHDL. The `si.env` file is loaded, followed by the `.simrc` file. Changes in the current session, such as effective CDF modifications, `.cdsinit` modifications, or other in-memory modifications are visible in the netlist.

Note: Modifications through the `.cdsinit` file are only possible with the netlisters used by the Virtuoso GUI. For all other netlisters, simulations must be customized using the `.simrc` file.

The `.simrc` file is loaded during each netlist generation and can be loaded multiple times in a Virtuoso session. You can specify simulator-specific customizations by setting the `simSimulator` variable.

For example,

```
when( and( boundp('simSimulator) (simSimulator == "spectre")))
  printf("Specify Spectre options\n")
```

Simulation Environment Help

Netlist Generation and Customization

```
)  
  
when( and( boundp('simSimulator) (simSimulator == "auCdl"))  
      printf("Specify CDL options\n")  
      )  
)
```

Related Topics

[Netlist Generation and Customization](#)

[Netlist Customization With the .simrc File](#)

[Variables for Incremental Netlisting](#)

[Customization of Scale Factors](#)

[Substitution Functions](#)

Simulation Environment Help

Netlist Generation and Customization

Simulation Runs

After you complete your design, you extract it, correct errors, and save the design for simulation input. You must correct all the errors reported during extraction before you simulate your design.

SE performs the following steps during the simulation process.

1. **Run Directory Initialization:** When a simulation is run on the Cadence system, all inputs and outputs of the simulation process are contained in a single directory. This directory is called the Simulation Run Directory (or run directory). The first step in the simulation process is to ensure required files exist in this directory.
2. **Netlisting:** Netlisting is the process of converting the connectivity of a design into a textual description suitable as input to a design analysis tool. Netlisting is the most complex step in integrating your simulator into the Cadence system. Netlisters frequently perform name mapping. Part of the SE functionality automatically translates between these names as needed by the application.
3. **Simulation Input Translation:** To enable the designer to specify the same names in the design and the input to the simulator (stimulus and commands), the `control` file is translated before it is provided as input to the simulator. Any names that were mapped to a different name during the netlisting process are then substituted with the name used for the netlist.
4. **Running the Simulator:** Once the input for the simulator has been prepared, the simulator is run. After the simulation completes, the simulator output needs to be prepared for user analysis. The simulation output is in textual format.
5. **Simulator Output Translation:** The simulator text output also requires translation. The names in the output file referencing the design are the same ones that appeared in the netlist. These may not be the same names as those entered in the design; therefore, the names need to be converted back to the names the designer entered.

Related Topics

[Initializing the Simulation Environment](#)

Simulation Environment Help

Simulation Runs

[Specifying Simulation Environment Options](#)

[Running a Simulation from the Schematic](#)

[Remote Simulation](#)

[Running an Interactive Simulation](#)

[Types of Simulations in Command-Line Mode](#)

Initializing the Simulation Environment

The first step in simulation is setting up the simulation environment. When you initialize the simulation environment, you specify the following:

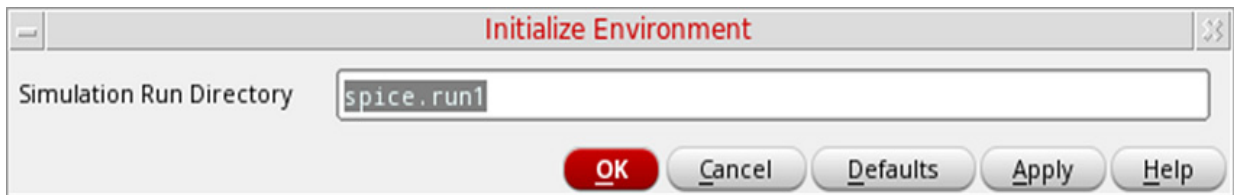
- Design
- Simulator
- Simulation run directory

Initially, all commands on the *Simulation* menu, except *Initialize*, are disabled. The remaining menu commands are enabled after you have used the *Initialize* command to initialize the simulation environment.

To initialize a new simulation run directory:

1. In the schematic window, choose *Simulation – Initialize*.

The Initialize Environment form appears.



2. Specify the name of the simulation run directory using a full or a relative path.
3. Click *OK*.

Simulation Environment Help

Simulation Runs

The Initialize form is overlaid by an expanded version of the same form that shows the following additional fields.



The values on this form are the current window and default SE values. You can edit these values by using the *Browse* button or specifying new values in the form.

4. From the *Simulator Name* list, select a simulator.

The possible values are `hspice`, `spice`, `sage`, and `other`. If your preferred simulator is not listed, select `other` and specify the name of the simulator in the adjoining text field.

5. Specify the name of the library containing the top level of your design.
6. Specify the cell name of your design.
7. Specify the view name of your design (for example, `schematic`).
8. Click *OK*.

The system initializes the simulation environment with the specified directory.

To use the Initialize Environment form for an existing run directory:

1. In the schematic window, choose *Simulation – Initialize*.

The Initialize Environment form appears.

2. Specify a new name for the *Simulation Run Directory*.
3. Click *OK*.

The system reinitializes the simulation environment with the specified directory.

Simulation Environment Help

Simulation Runs

Related Topics

[simInitEnv](#) (SKILL function)

[Initialize Environment Form](#)

[The Simulation Menu](#)

Specifying Simulation Environment Options

You can use the *Simulation – Options* command before you run a simulation to specify the following:

- If the system creates a hierarchical or flat netlist
- If the simulation runs on a remote machine

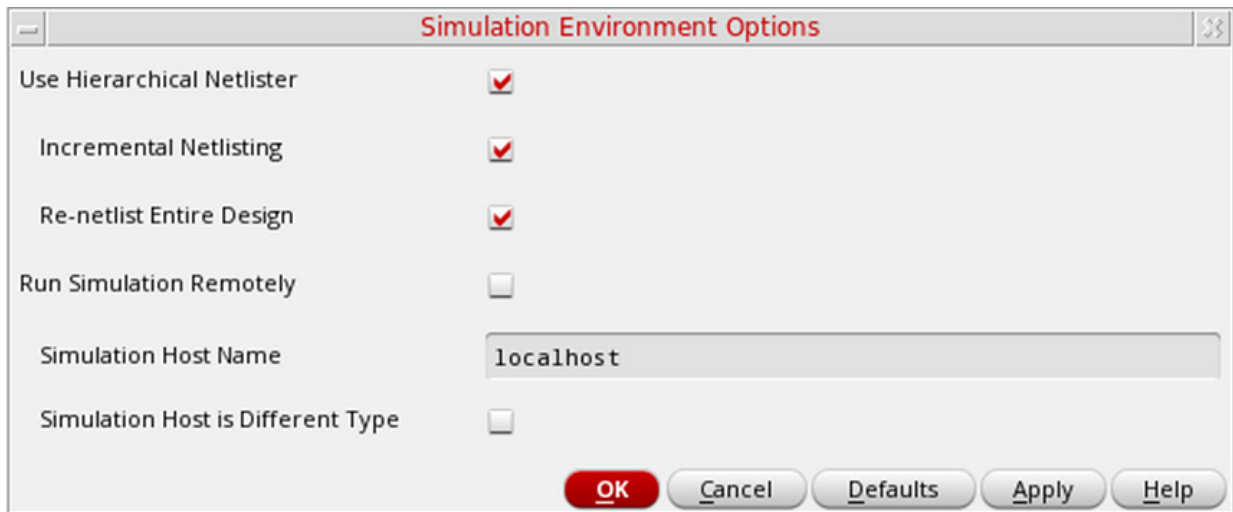
These options determine whether the system creates a flat or hierarchical netlist, runs remote simulation, or runs the *simdiff* command during the simulation process. The fields on the *Options* form have the default values or the values you specified when you last ran a simulation in the current simulation run directory.

The *Simulation – Options* command is enabled only after you have used the Initialize command to initialize the simulation environment.

To specify the simulation environment options:

1. In the schematic window, select *Simulation – Options*.

The Simulation Environment Options form appears.



The screenshot shows a dialog box titled "Simulation Environment Options". It contains several settings:

Option	Value
Use Hierarchical Netlist	<input checked="" type="checkbox"/>
Incremental Netlisting	<input checked="" type="checkbox"/>
Re-netlist Entire Design	<input checked="" type="checkbox"/>
Run Simulation Remotely	<input type="checkbox"/>
Simulation Host Name	localhost
Simulation Host is Different Type	<input type="checkbox"/>

At the bottom of the dialog box are five buttons: **OK** (highlighted in red), **Cancel**, **Defaults**, **Apply**, and **Help**.

2. Select *Use Hierarchical Netlist* to netlist using the hierarchical netlist.
 - ☐ Select *Incremental Netlisting* to enable incremental netlisting using the hierarchical netlist.
 - ☐ Select *Re-netlist Entire Design* to renetlist the entire design using the hierarchical netlist.

Simulation Environment Help

Simulation Runs

3. Select *Run Simulation Remotely* to run the simulation remotely.

- ☐ Specify the hostname for the remote simulation in the *Simulation Host Name* field. This field is enabled when you select the *Run Simulation Remotely* check box.
- ☐ Select *Simulation Host is Different Type* to indicate that the host computer has a different binary storage format than the local computer.

4. Click *OK*.

The system updates the simulation environment with the specified simulation options.

Related Topics

[Netlist and Simulate Form](#)

[Variables for Incremental Netlisting](#)

[Remote Simulation](#)

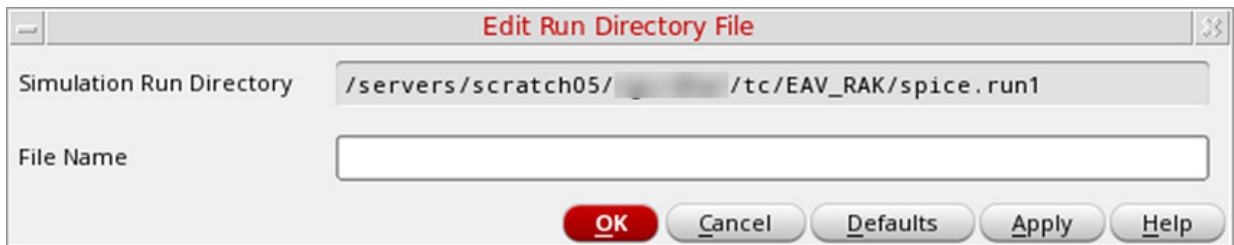
Creating the Input Stimulus in the Control File

You can edit a control file saved within a simulation run directory by creating an input stimulus.

To create an input stimulus in the current simulation run directory:

1. In the schematic window, select *Simulation – Stimulus – Edit File*.

The Edit Run Directory File form appears.



The *Simulation – Stimulus – Edit File* command is enabled only after you have used the *Initialize* command to initialize the simulation environment.

2. In the *File Name* field, specify the name of the simulation run directory file that you want to edit.

The *Simulation Run Directory* field in this form is read-only. To change this directory, click *Cancel* and choose the *Initialize* command again.

3. Click *OK*.

The system brings up a new window displaying the file you specified.

4. Edit the file in the syntax of your analysis tool using a text editor.

You can specify all your customizations directly into the control file or you can type SE substitution functions to translate names and merge the customizations in other files.

5. Exit the text editor when you have finished editing the file.

The system automatically closes the window.

Related Topics

[Netlist and Simulate Form](#)

[Substitution Functions](#)

[Sample si.inp File Generated for Cadence SILOS II](#)

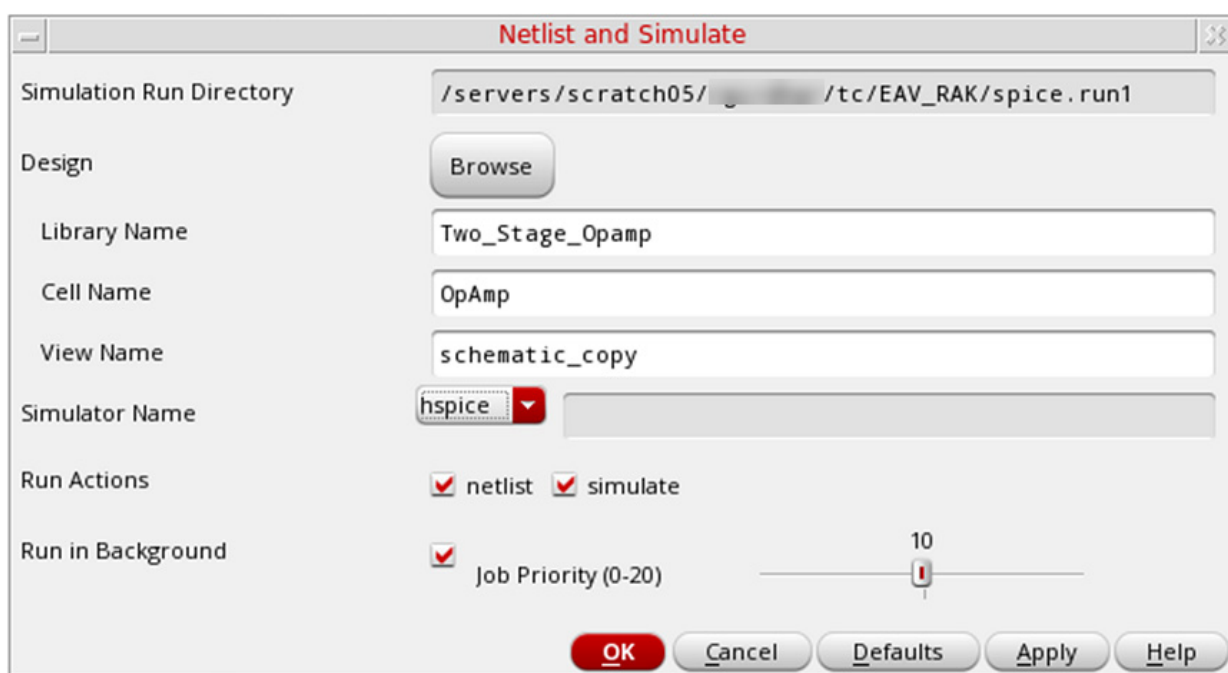
Running a Simulation from the Schematic

You can either run a simulation using an existing netlist or run a complete simulation after generating a new netlist.

To generate a netlist:

1. In the schematic window, select *Simulation – Netlist/Simulate*.

The Netlist and Simulate form appears.



The *Simulation – Netlist/Simulate* command is enabled only after you have used the *Initialize* command to initialize the simulation environment.

Most fields in this form display the values that you specified using the *Initialize* command. You can edit the values manually or by using the *Browse* button.

The *Simulation Run Directory* field in this form is read-only. To change this directory, click *Cancel* and choose the *Initialize* command again.

2. Specify the values and select the options that you want.
3. Click *OK*.

If you click *Cancel*, the simulation is not run. If you click *Defaults*, the values in the form are reset to the default values.

Simulation Environment Help

Simulation Runs

When a background simulation completes, a dialog box appears with the completion status of the simulation.

Note: You cannot run more than one simulation concurrently in the same run directory. Any attempt to run such concurrent simulations displays a message that a simulation is already running in the directory. You can terminate the currently running job and start a new simulation.

Related Topics

[Netlist and Simulate Form](#)

[Initializing the Simulation Environment](#)

Remote Simulation

You can set up the system to run remote simulations using the Verilog-XL and HSPICE simulators.

- When using the SE GUI in the schematic, you can set the required options in the Simulation Environment Options form.
- When using the command-line interface, you can set the required SKILL variables in the CIW or the .simrc file. The local machine and the remote host must both run the X Window System™.

Related Topics

[Netlist and Simulate Form](#)

[Setting Up Remote Simulation From the Schematic](#)

[Setting Up Remote Simulation From the Command-Line Interface](#)

Setting Up Remote Simulation From the Schematic

To set up remote simulation using the SE GUI:

1. In the schematic window, choose *Simulation – Options* to open the Simulation Environment Options form.
2. Select the *Run Simulation Remotely* check box.
The *Simulation Host Name* field becomes editable.
3. Specify the name of the remote workstation in the *Simulation Host Name* field.
For example, `cds17`.
4. Select *Simulation Host is Different Type* to indicate that the host computer has a different binary storage format than the local computer.
5. Click *OK*.

After setting these options, any simulation that you run will be running remotely, and you can view the results, similar to a locally run simulation.

Related Topics

[Netlist and Simulate Form](#)

[Remote Simulation](#)

[Setting Up Remote Simulation From the Command-Line Interface](#)

Files Needed for Simulation in the Command-Line Interface

Before you start a simulation in the UNIX environment, you must create the following files in the simulation run directory:

- si.env file
- control file

The system creates these files automatically when you run a simulation using the Netlist and Simulate form in the SE GUI, but you must manually create the files if you want to use the command-line interface to run a simulation.

si.env file

The `si.env` file directs SE on which design to simulate and what simulator to use.

The following table lists the variables you must define in the `si.env` file. Each interface might store additional simulator-specific variables in the `si.env` file.

Variables to be defined in the si.env file

Variable	Description
<code>simSimulator</code>	Simulator to be run
<code>simLibName</code>	Name of the library containing the top-level cellview
<code>simCellName</code>	Name of the top-level cell to be simulated
<code>simViewName</code>	View name of the top-level cell to be simulated
<code>simHost</code>	Host name of remote simulator. This is optional.

A sample si.env file:

```
simLibName = "testLib"  
simCellName = "74169"  
simViewName = "schematic"  
simSimulator = "silos"  
simHost = "cds642"
```

control file

You can directly specify all your customizations in the control file or you can use SE substitution functions to translate names and merge the changes in other files. For example, to merge an input stimulus file named `simsub.inp` into the control file, prefix the file name with an exclamation mark and enclose the text in square brackets. For example:

```
[!simsub.inp]
```

If `simsub.inp` is not in the current simulation run directory, specify the full path as follows:

```
[!/<level1InstName/.../levelnInstName>/simsub.inp]
```

Related Topics

[simSimulator](#) (SKILL Function)

[simLibName](#) (SKILL Function)

[simCellName](#) (SKILL Function)

[simViewName](#) (SKILL Function)

[simHost](#) (SKILL Function)

[Sample si.inp File Generated for Cadence SILOS II](#)

Setting Up a Simulation Using the Command-Line Interface

It is recommended that you run a simulation using the menus and forms available in the GUI. However, you can also run an interactive or batch mode simulation using SE commands in the CIW or in the shell environment using the `si` binary.

To set up SE to run a simulation from the command-line interface:

1. In a terminal window, enter the following command to change to the directory that contains the simulation run directory.

```
cd <parent_directory_name>
```

2. Enter the following command to create the run directory.

```
mkdir directoryname
```

Here `directoryname` is the name of the simulation run directory. For example, if your simulation run directory is `spice.run1`, enter the following command:

```
mkdir spice.run1
```

3. Enter the following command to change to the newly created directory.

```
cd spice.run1
```

4. Create the simulation environment file `si.env` using a text editor.

5. Save the `si.env` file.

SE is set to run simulations from the command-line interface.

Related Topics

[Running an Interactive Simulation](#)

[Types of Simulations in Command-Line Mode](#)

[Files Needed for Simulation in the Command-Line Interface](#)

Setting Up Remote Simulation From the Command-Line Interface

To set up a remote simulation from the command-line interface:

1. Set the SE variable `simHost` to the name of the remote workstation.

For example:

```
simHost = "cds17"
```

2. Set the SE variable `simHostDiffers` to `t` if the host computer has a different binary storage format than the local computer.

For example:

```
simHostDiffers = t
```

After setting these variables, all simulations are run remotely, and you can view the results, similar to a locally run simulation.

Related Topics

[simHost](#) (SKILL Function)

[simHostDiffers](#) (SKILL Function)

[Remote Simulation](#)

[Setting Up Remote Simulation From the Schematic](#)

Types of Simulations in Command-Line Mode

The command-line interface lets you run the following types of simulations: :

Type	Running the Simulation
Full Simulation	<p>After you start SE, you can run the <code>sim</code> SKILL function on a terminal window to run a full simulation automatically.</p> <pre>> sim</pre> <p>The <code>sim</code> function initializes simulation variables appropriate for your simulator, specified by the <code>simSimulator</code> variable, and runs the SE functions required for simulation.</p>

Simulation Environment Help

Simulation Runs

Type	Running the Simulation
Simulation in Batch Mode	<p>You can run a simulation in batch mode by starting the <code>si</code> binary with the <code>-batch</code> option.</p> <pre>si -batch [run_directory_name]</pre> <p>SE initializes the environment with the specified run directory and runs the <code>sim</code> command. Consequently, the <code>sim</code> command runs the required functions in a pre-defined order.</p> <p>If your simulation run directory is <code>/mnt/dave/aluSimulations/silos.run1</code>, enter the following command to run a batch simulation with a log of events:</p> <pre>si -batch /mnt/dave/aluSimulations/silos.run1 >& /mnt/dave/aluSimulations/silos.run1/si.log &</pre> <p>The simulation run directory must already exist and contain the <code>si.env</code> file. Redirecting the messages from SE into the <code>si.log</code> file in the simulation run directory automatically creates a log of run events. This log is created automatically when you run background simulations from the SE UI.</p> <p>You can run a single SE command in batch mode with the <code>-command</code> option:</p> <pre>si -batch -command command_name run_directory_name</pre> <p>If your run directory name is <code>/mnt2/deborah/spice.run1</code>, enter the following command to generate a netlist:</p> <pre>si -batch -command netlist /mnt2/deborah/spice.run1</pre> <p>For using the Spectre simulator, specify <code>nl</code> as the parameter for <code>-command</code>.</p>
Simulation in Manual Mode	<p>You can manually control a simulation by running the functions used by the <code>sim</code> SKILL function in the pre-defined order.</p>

Related Topics

[Functions Used to Run Simulations in All Modes](#)

[Running a Simulation from the Command-Line Interface](#)

Functions Used to Run Simulations in All Modes

The `sim` function runs the following functions in the given order to run simulations in full and batch modes. You can control a manual simulation by running the following functions in the order listed in the table.

Function	Description
<code>simCheckVariables</code>	<p>Checks whether the following variables required to run simulations have been set:</p> <p><code>simSimulator</code>, <code>simCellName</code>, <code>simLibName</code>, <code>simViewName</code>, <code>simRunDir</code>, <code>simViewList</code>, <code>simStopList</code>, <code>simSedFile</code>, <code>simCommand</code>, <code>simNlpGlobalLibName</code>, and <code>simNlpGlobalCellName</code>.</p>
<code>simInitRunDir</code>	<p>Sets up your simulation run directory. It performs the following actions:</p> <ul style="list-style-type: none">■ Copies a default control file into the simulation run directory.■ Generates a default input stimulus file.■ Creates a waveform (<code>raw</code>) directory in the simulation run directory. <p>Each interface might have additional initialization procedures. Some interfaces create input files in addition to the basic <code>control</code> file.</p>
<code>netlist</code>	<p>Produces a text description of the design specified by the <code>simLibName</code>, <code>simCellName</code>, and <code>simViewName</code> variables in a file named <code>netlist</code>. The <code>netlist</code> file contains the elements, signals, models, and their interconnections in the format required by the target simulator.</p>
<code>simin</code>	<p>Translates names in the control file from user-assigned names in the schematic to netlist-generated names acceptable to the target simulator. This function also merges all input stimulus and command files specified in the <code>control</code> file into the simulator input file <code>si.inp</code>. You can only run this command if a netlist is already generated.</p>

Simulation Environment Help

Simulation Runs

Function	Description
<code>runsim</code>	<p>Runs the simulator. It performs the following actions:</p> <ul style="list-style-type: none">■ Runs the specified simulator using the <code>si.inp</code> file as its input.■ Translates waveform output from the simulator to the Cadence Waveform Storage Format (if necessary).■ Translates names in the text simulator output back to the user-assigned names in your design. <p>You can only use this command after running the <code>netlist</code> and <code>simin</code> commands to generate the simulator input files.</p>
<code>exit</code>	<p>Exits the simulation environment.</p>

Related Topics

[simCheckVariables](#) (SKILL Function)

[simInitRunDir](#) (SKILL Function)

[netlist](#) (SKILL Function)

[simin](#) (SKILL Function)

[runsim](#) (SKILL Function)

[Types of Simulations in Command-Line Mode](#)

Running a Simulation from the Command-Line Interface

To run a simulation from the command-line interface:

1. Change to the directory that contains the `cds.lib` file. You can also specify the `cds.lib` file at the command line with the `-cdslib` option.

The `cds.lib` file contains the library path to the design. If you invoke `si` in a directory that does not contain the `cds.lib` file, specify a fully-qualified path to the `cds.lib` file following the `-cdslib` option. For example:

```
si /mnt/temp/spice.run1 -cdslib /mnt/dave/cds.lib
```

2. Enter the `si` command followed by the full system path to the simulation run directory.

For example, if the path to your run directory is `/mnt/temp/spice.run1`, enter the following command:

```
si /mnt/temp/spice.run1
```

When the system has been initialized, SE displays a command prompt (`>`). You can enter the `si` command with one or both options, `-noenv`, and `-diffptest`.

If you do not want `si` to specify the environment file but let OSS pick the default environment file, use the `-noenv` option. For example:

```
si -noenv
```

To remove any date stamps that appear in the `si.log` file or the standard output during netlisting, use the `-diffptest` option. For example:

```
si -diffptest -cdslib ./cds.lib -batch -command netlist
```

You can now enter the SE commands to run any simulation supported in the command-line interface.

Related Topics

[Types of Simulations in Command-Line Mode](#)

[Functions Used to Run Simulations in All Modes](#)

Running si in Replay Mode

To specify the time taken to suspend the running process:

- In a terminal window, run the `simIlSleep()` function.

It is recommended to use the `simIlSleep()` function, instead of the `ipcSleep()` and `sleep()` functions, while running `si` in replay mode. Replay mode does not support `ipcSleep()` and `sleep()` functions.

Related Topics

[simIlSleep](#) (SKILL Function)

Running an Interactive Simulation

You can run an interactive simulation if your simulation interface supports this functionality. The *Interactive* command is enabled only after you have used the *Initialize* command to initialize the simulation environment.

To run an interactive simulation:

1. In the Schematic window, select *Simulation – Interactive*.
2. Choose to netlist your design, or not, and click *OK*.

By default, the system opens the following three windows:

- ☐ A window for displaying waveforms produced during the simulation (at the top half of the screen)
- ☐ A window for editing the design you are simulating (at the bottom right)
- ☐ A window for textual interaction with the simulator (at the bottom left)

Commands in the simulator window vary depending on the simulator you use.

To customize the window placement, in the CIW, choose *Options – User Preferences* and click *Place Manually*.

3. When you finish simulation, choose *Finish Interactive* from the SE menu of the simulator window.

Related Topics

`iseStartInteractive`

Customization of the Simulation Environment Using the .simrc File

When you initialize SE, it first loads the `si.env` file. This file notifies SE on what design to simulate and what simulator to use.

SE then loads the simulation run control file `.simrc` if it exists. If you set a variable in `.simrc` that also sets options in the graphical environment using the Options command, SE uses the `.simrc` file settings and ignores the Options settings.

The `.simrc` file must be in SKILL syntax. The following is a sample `.simrc` file. The first line in this file overrides the default view list used for view switching with SILOS. The second line overrides the default stopping list that stops hierarchy expansion for SILOS.

```
hspiceSimViewList = ("hspice" "cmos_sch" "schematic")
hspiceSimStopList = ("hspice" "cmos_sch")
```

Related Topics

[si.env file](#)

[control file](#)

[Location of the .simrc File](#)

[Netlist and Simulate Form](#)

Variables to Customize Simulations

The following table describes some of the SE variables that you can set in your `.simrc` file to customize simulation.

Variable	Description
<code>simSimulator</code>	Specifies the simulator to run
<code>simControlFile</code>	Specifies the path of default control file
<code>simDefaultControl</code>	Specifies the name of the default <code>control</code> file if stored in <code>install_dir/etc/src</code> .
<code>simTimeUnit</code>	Specifies the scaling factor for delay times. This value should match the first argument of the <code>deftiming</code> command.
<code>simCapUnit</code>	Specifies the scaling factor for capacitance
<code>simNlpGlobalLibName</code>	Specifies the name of the library containing global formatting instructions for flat netlister
<code>simNlpGlobalCellName</code>	Specifies the name of the cell containing global formatting instructions for flat netlister
<code>simNlpGlobalViewName</code>	Specifies the name of the view of the cell containing global formatting instructions for flat netlister
<code>simNotIncremental</code>	Specifies incremental netlisting when set to <code>nil</code> .
<code>simReNetlistAll</code>	Specifies non-incremental netlisting
<code>simNetlistHier</code>	Specifies hierarchical netlisting

Reated Topics

[SE Variables](#)

[control file](#)

[Location of the .simrc File](#)

[Creating the Input Stimulus in the Control File](#)

[Customization of the Simulation Environment Using the .simrc File](#)

Viewing Waveform Results in the Schematic

To display the waveforms produced during a simulation:

1. In the schematic window, choose *Simulation – Show Waveforms*.

The Show Waveforms form appears.



The *Simulation - Show Waveforms* command is enabled only after you have used the *Initialize* command to initialize the simulation environment.

2. In the *Waveform File Name* field, specify the name of the waveform file to be displayed.
3. Click *OK*.

The system opens the waveform window and displays information for the simulation run and the waveform file you specified. You can now execute any of the waveform commands to manipulate and display specific waveforms.

Related Topics

[Show Registers Form](#)

[Viewing Waveform Results in Register Format](#)

[Initializing the Simulation Environment](#)

[The Simulation Menu](#)

Viewing Waveform Results in Register Format

To view the waveforms produced during a simulation in register format:

1. In the schematic window, select *Simulation – Show Registers*.

The Show Registers form appears.



The *Simulation - Show Registers* command is enabled only after you have used the *Initialize* command to initialize the simulation environment.

2. In the *Waveform File Name* field, specify the name of the waveform file to be displayed.
3. Click *OK*.

The system opens the register display window.

Related Topics

[Show Registers Form](#)

[Viewing Waveform Results in the Schematic](#)

[Initializing the Simulation Environment](#)

[The Simulation Menu](#)

Viewing a Specified Text File

To display any text file from the current simulation run directory:

1. In the schematic window, select *Simulation – Show Outputs – Show Run File*.

The Show Run File form appears.



The *Simulation – Outputs – Show Run File* command is enabled only after you have used the *Initialize* command to initialize the simulation environment.

2. In the *File Name* field, specify the name of the file you want to view.
3. Click *OK*.

A window appears showing the text file.

4. To close the window, choose *File – Close Window*.

Related Topics

[Show Run File Form](#)

[Initializing the Simulation Environment](#)

[The Simulation Menu](#)

Viewing the Run Log of a Job

The *Show Run log* command displays the run log (`si.log`) of a background simulation or the run log (`si.foregnd.log`) of a foreground simulation. Both run logs contain the output from the simulation environment. It lists the simulation steps and their completion status. It also lists any error messages from SE, including the netlisters. The *Show Outputs – Show Run Log* command is enabled only after you have used the *Initialize* command to initialize the simulation environment.

To display the run log of a simulation:

1. Choose one of the following:

- ☐ *Simulation – Show Run Log – Show Foreground Run Log*
- ☐ *Simulation – Show Run Log – Show Background Run Log*

A window appears showing the run log file of the foreground or background job.

2. Close the window by choosing *File – Close Window*.

Related Topics

[Initializing the Simulation Environment](#)

[The Simulation Menu](#)

Viewing the Text Output in SE

The *Show Output* command in the *Simulation* menu displays the `si.out` file from the simulation run directory. This file contains the text output of the simulator, for example, error messages. The *Show Outputs – Show Output* command is enabled only after you have used the *Initialize* command to initialize the simulation environment.

To displays the `si.out` file from the simulation run directory:

1. Choose *Simulation – Show Outputs – Show Output*.

A window appears showing the `si.out` file.

2. Close the window by choosing *File – Close Window*.

Related Topics

[Initializing the Simulation Environment](#)

[The Simulation Menu](#)

Viewing Global Errors in SE

Global errors are not associated with a particular net or instance. The *View Global Error* command in the *Simulation* menu displays global errors (`global.err`) found during netlisting or simulation with interfaces that support this feature. The *Show Outputs – Show Global Error* command is enabled only after you have used the *Initialize* command to initialize the simulation environment.

To display global errors (`global.err`) found during netlisting or simulation:

1. Choose *Simulation – Show Outputs – View Global Error*.

A window appears showing the error messages generated by the flat netlister or by your analysis tool. If no error file exists, a dialog box informs you that there were no global errors.

You can see global errors generated by the simulator only if you use Cadence SILOS II.

2. Close the window by choosing *File – Close Window*.

Related Topics

[Initializing the Simulation Environment](#)

[The Simulation Menu](#)

Viewing Highlighted Errors in SE

The *Highlight Errors* command in the *Simulation* menu places a probe on any object in your schematic that has an error.

To highlight errors associated with a specific net or instance:

- Choose *Simulation – Show Outputs – Highlight Errors*.

The system places probes on objects in your design with errors.

For example, the system places a probe on a gate with an undriven input if your simulator does not support these. The system can also place probes lower in the design hierarchy and reflect them up the hierarchy to the instance that contains them.

After probes have been placed, you can use the standard probing functions to manipulate and remove them. If an error cannot be isolated to a particular net or instance, the system does not place a probe on it. Use the *Show Global Error* command to view such errors.

The *Show Outputs – Highlight Errors* command is enabled only after you have used the *Initialize* command to initialize the simulation environment.

Related Topics

[Initializing the Simulation Environment](#)

[The Simulation Menu](#)

Managing Jobs Using the Job Monitor

The *Job Monitor* command in the *Simulation* menu lets you perform the following operations for a background analysis job:

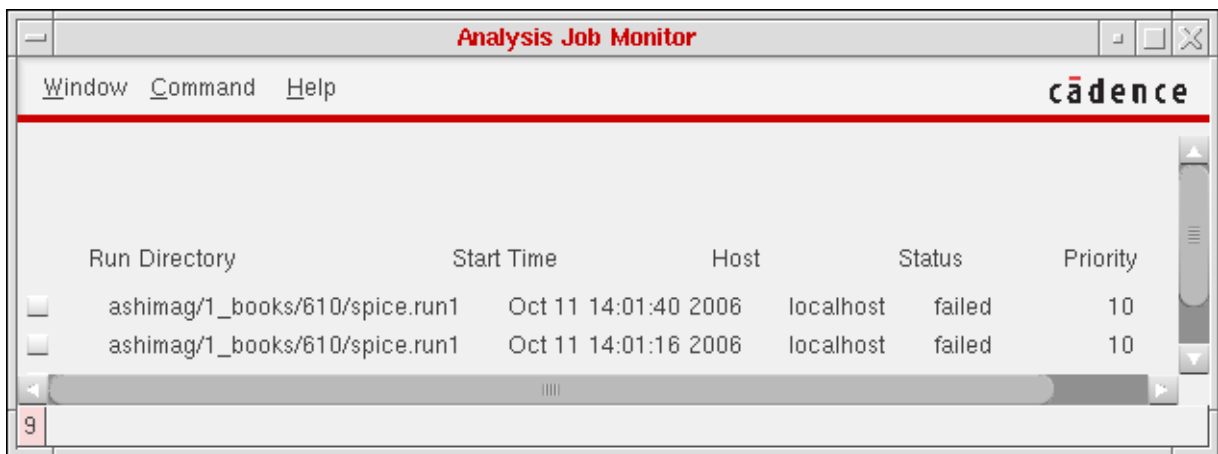
- Check the job status
- Monitor the output
- Suspend the job
- Change the execution priority
- Stop the job
- View the run log

The background analysis jobs from SE appear in the *Job Monitor* form irrespective of whether you started them using the *Simulation* menu. The *Simulation - Job Monitor* command is enabled only after you have used the *Initialize* command to initialize the simulation environment and then run a simulation.

To access the Job Monitor:

1. In the schematic window, choose *Simulation – Job Monitor*.

The Analysis Job Monitor form appears.



The form displays all SE background analysis jobs, whether they were started using the *Simulation* menu or otherwise.

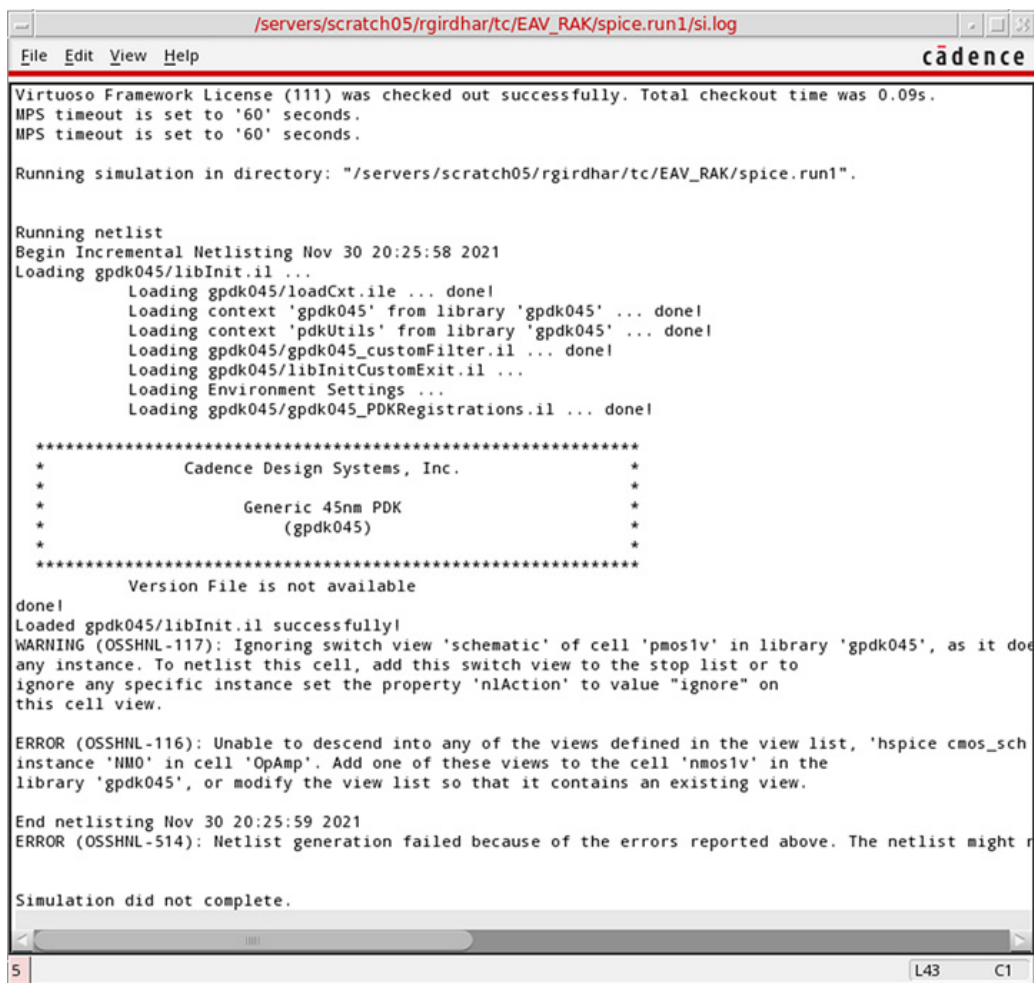
2. Select the jobs for which you want to run the Job Monitor.

Simulation Environment Help

Simulation Runs

3. Choose an appropriate command from the *Command* menu.

- ❑ *Show Run Log* to display the run log (*si.log*) for the selected jobs in a new window.



```
/servers/scratch05/rgirdhar/tc/EAV_RAK/spice.run1/si.log
File Edit View Help
cadence

Virtuoso Framework License (111) was checked out successfully. Total checkout time was 0.09s.
MPS timeout is set to '60' seconds.
MPS timeout is set to '60' seconds.

Running simulation in directory: "/servers/scratch05/rgirdhar/tc/EAV_RAK/spice.run1".

Running netlist
Begin Incremental Netlisting Nov 30 20:25:58 2021
Loading gpdK045/libInit.il ...
  Loading gpdK045/loadCxt.ile ... done!
  Loading context 'gpdK045' from library 'gpdK045' ... done!
  Loading context 'pdkUtils' from library 'gpdK045' ... done!
  Loading gpdK045/gpdK045_customFilter.il ... done!
  Loading gpdK045/libInitCustomExit.il ...
  Loading Environment Settings ...
  Loading gpdK045/gpdK045_PDKRegistrations.il ... done!

*****
*          Cadence Design Systems, Inc.          *
*                                                                 *
*          Generic 45nm PDK                        *
*          (gpdK045)                                *
*                                                                 *
*****
Version File is not available
done!
Loaded gpdK045/libInit.il successfully!
WARNING (OSSHNL-117): Ignoring switch view 'schematic' of cell 'pmos1v' in library 'gpdK045', as it does not exist. To netlist this cell, add this switch view to the stop list or to ignore any specific instance set the property 'nlAction' to value "ignore" on this cell view.
ERROR (OSSHNL-116): Unable to descend into any of the views defined in the view list, 'hspice_cmos_sch' instance 'NM0' in cell 'OpAmp'. Add one of these views to the cell 'nmos1v' in the library 'gpdK045', or modify the view list so that it contains an existing view.
End netlisting Nov 30 20:25:59 2021
ERROR (OSSHNL-514): Netlist generation failed because of the errors reported above. The netlist might not be correct.
Simulation did not complete.
```

This run log contains the output from the simulation environment. It lists the simulation steps and their completion status. It also lists any error messages from SE, including the netlisters.

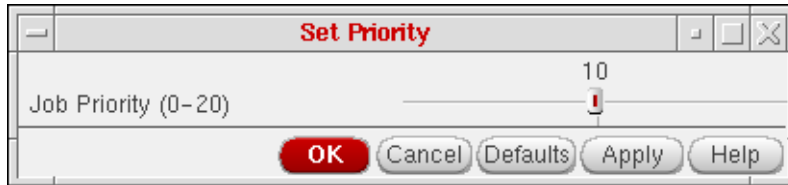
Alternatively, to access the run log of a job from SE:

- Choose *Simulation – Show Outputs – Show Run Log – Show Foreground Run Log* to view the foreground job log.
- Choose *Simulation – Show Outputs – Show Run Log – Show Background Run Log* to view the background job log.

Simulation Environment Help

Simulation Runs

- ❑ *Set Priority* to display the Set Priority form which lets you change the priority of the selected jobs.



You can click and drag the *Job Priority* slider bar to the required value in the range 0–20.

- ❑ *Kill* to terminate the selected jobs.
- ❑ *Suspend* to suspend the selected jobs.
- ❑ *Continue* to resume the jobs you suspended with the *Suspend* command.
- ❑ *Remove Entry* to delete the selected jobs from the Job Monitor window.

4. Choose *Window – Close* to close the Analysis Job Monitor form.

Related Topics

[Analysis Job Monitor Form](#)

[Set Priority Form](#)

Simulation Environment Help

Simulation Runs

SE Form Reference

The following forms are available in the Simulation Environment.

[Analysis Job Monitor Form](#)

[Edit Run Directory File Form](#)

[Initialize Environment Form](#)

[Netlist and Simulate Form](#)

[Set Priority Form](#)

[Show Registers Form](#)

[Show Run File Form](#)

[Show Waveforms Form](#)

[Simulation Environment Options Form](#)

Analysis Job Monitor Form

Use the Analysis Job Monitor form to check the status of a background analysis job, monitor its output, suspend it, change its execution priority, or stop it.

The following table describes the fields available in the Analysis Job Monitor form.

Field	Description
<i>Show Run Log</i>	Displays the run log (<code>si.log</code>) for the selected jobs in a new window.
<i>Set Priority</i>	<p>Displays the Set Priority form that lets you change the priority of the selected jobs.</p> <p>Unless you have root access privilege, you cannot lower the priority of a job once it is started. You can only decrease the priority of a job. For example, to decrease the priority of a job set at 10, change it to 12. This slows down the background job and increases the speed of the Cadence graphics shell.</p>
<i>Kill</i>	<p>Terminates the selected jobs.</p> <p>If you kill a job, you cannot continue it with the <i>Continue</i> command.</p>
<i>Suspend</i>	Suspends the selected jobs.
<i>Continue</i>	Resumes the jobs that you suspended using the <i>Suspend</i> command.
<i>Remove Entry</i>	<p>Removes the selected jobs from the Analysis Job Monitor form but does not terminate them.</p> <p>As a result, you can no longer change their priorities or kill them. You can still see output from these simulations with the <i>Show Run Log</i> command if you first initialize the environment and specify their simulation run directories.</p>

Related Topics

[simJobMonitor \(SKILL Function\)](#)

[Managing Jobs Using the Job Monitor](#)

Edit Run Directory File Form

Use the Edit Run Directory File form to edit any file in the current simulation run directory.

The following table describes the fields available in the Edit Run Directory File form.

Field	Description
<i>Simulation Run Directory</i>	<p>Specifies the name of the current simulation run directory that you specified using the <i>Initialize</i> command.</p> <p>This field is not editable. To change this directory, click <i>Cancel</i> and select the <i>Initialize</i> command again.</p>
<i>File Name</i>	<p>Specifies the name of the file you want to edit in the current simulation run directory. Specifying the full path is not required.</p> <p>You can edit or create any file in the simulation run directory by editing this field. For example, type <code>control</code> to edit the control file created in the simulation run directory by the <i>Initialize</i> command. The system displays the file in a new window. You can change the text editor by modifying the <i>EDITOR</i> shell variable or the <i>editor</i> SKILL variable.</p>

Related Topics

[Creating the Input Stimulus in the Control File](#)

[Initializing the Simulation Environment](#)

[Initialize Environment Form](#)

Initialize Environment Form

Use the Initialize Environment form to initialize the environment for simulation.

The following table describes the fields available in the Initialize Environment form.

Field	Description
<i>Simulation Run Directory</i>	<p>Specifies the directory in which the system stores simulation input and output files.</p> <p>If you specify a relative path, the system creates the run directory in the directory from which you launched the software. By default, the run directory is <code>spice.run1</code>.</p> <p>The system saves all simulation input and output files in the simulation run directory. As the system initializes the environment, it lists the files it has loaded and any overridden variables in the <i>si.foregnd.log</i> file, which is located in the run directory.</p>
<i>Simulator Name</i>	<p>Specifies the simulator to be used for analysis.</p> <p>This field shows the available simulators. If your preferred simulator is not listed, select <code>other</code> and specify the name of the simulator in the adjoining text field. The text field becomes editable when you select <code>other</code>.</p>
<i>Library Name</i>	Specifies the library containing the top level of your design.
<i>Cell Name</i>	Specifies the top-level cell to simulate, for example, <i>inverter</i> .
<i>View Name</i>	Specifies the name of the view to simulate, for example, <i>schematic</i> .

Related Topics

[Initializing the Simulation Environment](#)

[Specifying Simulation Environment Options](#)

[The Simulation Menu](#)

[simSimulator](#) (SKILL Function)

Netlist and Simulate Form

Use the Netlist and Simulate form to generate a netlist, run a simulation using an existing netlist, or run a complete simulation, which includes generation of a new netlist.

The following table describes the fields available in the Netlist and Simulate form.

Field	Description
<i>Simulation Run Directory</i>	<p>Displays the path to the simulation run directory.</p> <p>The simulation run directory contains simulation input and output files. This is the same directory you specified when you ran the <i>Initialize</i> command. This field is read-only. To change the simulation run directory, you must run the <i>Initialize</i> command.</p>
<i>Library Name</i>	<p>Specifies the library that contains the top-level cellview you want to simulate.</p>
<i>Cell Name</i>	<p>Specifies the cell name of the top-level cellview to simulate.</p>
<i>View Name</i>	<p>Specifies the view name of the top-level cellview to simulate.</p>
<i>Simulator Name</i>	<p>Specifies the system on which the simulation runs.</p> <p>The default is <i>localhost</i>, which is your system. When you select <i>other</i> from the list, the corresponding text field becomes editable. You can specify the name of the system on which you want to run remote simulation. This field requires that <i>Run Simulation Remotely</i> is selected.</p>
<i>Run Actions</i>	<p>Specifies the analysis tool.</p> <p>This field shows the analysis tools available. If you want to use a tool that is not listed, select <i>other</i> and type the name of that tool in the adjoining text entry field. You can only type in the text entry field if you have selected <i>other</i>.</p>

Simulation Environment Help

SE Form Reference

Field	Description
<i>Run in Background</i>	<p>Determines whether to run the simulation in the background.</p> <p>When selected, runs the simulation in the background. While the simulation is running, you can continue to use the graphics shell to monitor the simulation or perform other editing or analysis tasks. If you do not run the simulation in the background, you cannot use the graphical environment until the simulation is done.</p> <p>When deselected, runs the simulation in the foreground, which is most useful for small designs.</p>
<i>Job Priority</i>	<p>Specifies the UNIX priority of the background simulation.</p> <p>The lower the value, the faster the background simulation and the slower your system performance on other tasks. You cannot modify the priority of a foreground simulation. Unless you have root access privilege, you cannot lower the priority of a job once it is started.</p>

Related Topics

[simRunNetAndSim](#) (SKILL function)

[Running a Simulation from the Schematic](#)

[Initializing the Simulation Environment](#)

[Initialize Environment Form](#)

Set Priority Form

Use the Set Priority form to set an appropriate job priority before you start the job.

The following table describes the fields available in the Set Priority form.

Field	Description
<i>Job Priority</i>	<p>Specifies the priority of the job.</p> <p>A higher number in this field indicates lower priority, faster job simulation, and slower system performance on other tasks. Unless you have root access privilege, you cannot lower the priority of a job once it has started.</p> <p>For example, to lower the priority of a job set at 10, change it to 12. This slows down the job simulation and increases the speed of your local Cadence graphics shell.</p>

Related Topics

[Managing Jobs Using the Job Monitor](#)

[Analysis Job Monitor Form](#)

Show Registers Form

Use the Show Registers form to display the waveforms produced during a simulation in register format.

Note: *Show Registers* is supported only with icfb and icds workbenches.

The following table describes the fields available in the Show Registers form.

Field	Description
<i>Simulation Run Directory</i>	<p>Specifies the name and path of the current simulation run directory as a read-only field.</p> <p>To change the directory, click <i>Cancel</i> and select the <i>Initialize</i> command again.</p>
<i>Waveform File Name</i>	<p>Specifies the name of the waveform file to display.</p> <p>The system expects this file to be within <code>/run_dir/raw</code>. The default waveform file is <code>waves</code>. Most of the standard interfaces do not require that you change this field. If you are using a tool that produces multiple waveform files, such as the Cadence HSPICE interface, you need to change this field to look at the waveforms for each analysis run.</p>

Related Topics

[Viewing Waveform Results in Register Format](#)

[Initialize Environment Form](#)

[Initializing the Simulation Environment](#)

[The Simulation Menu](#)

Show Run File Form

Use the Show Run File form to display a text file from the current simulation run directory.

The following table describes the fields available in the Show Run File form.

Field	Description
<i>Simulation Run Directory</i>	Specifies the name of the current simulation run directory that you specified using the <i>Initialize</i> command. This field is not editable. To change this directory, click <i>Cancel</i> and select the Initialize command again.
<i>File Name</i>	Specifies the name of the file you want to display from the current simulation run directory. The system expects the file to be in the simulation run directory and displays it in a new window.

Related Topics

[Viewing a Specified Text File](#)

[Initialize Environment Form](#)

[Initializing the Simulation Environment](#)

[The Simulation Menu](#)

Show Waveforms Form

Lets you display the waveforms produced during a simulation.

The following table describes the fields available in the Show Waveforms form.

Field	Description
<i>Simulation Run Directory</i>	<p>Specifies the name and path of the current simulation run directory as a read-only field.</p> <p>To change the directory, click <i>Cancel</i> and select the <i>Initialize</i> command again.</p>
<i>Waveform File Name</i>	<p>Specifies the name of the waveform file to display.</p> <p>The system expects this file to be in the <code>raw</code> directory in the simulation run directory. The default waveform file is <code>waves</code>. Most of the standard interfaces do not require that you change this field. If you are using a tool that produces multiple waveform files, such as the Cadence HSPICE interface, you need to change this field to look at the waveforms for each analysis run.</p>

Related Topics

[simWaveOpen \(SKILL Function\)](#)

[Viewing Waveform Results in Register Format](#)

[Initialize Environment Form](#)

[Initializing the Simulation Environment](#)

[The Simulation Menu](#)

Simulation Environment Options Form

Use the Simulation Environment Options form to set simulation run options that are additional to the ones that appear on the Netlist and Simulate form.

The following table describes the fields available in the Simulation Environment Options form.

Field	Description
<i>Use Hierarchical Netlist</i>	Specifies whether the system creates a hierarchical netlist. For tools and interfaces that do not support both types of netlists, this field is ignored.
<i>Incremental Netlisting</i>	Specifies whether the system netlists only the parts of your design you modified since you last netlisted.
<i>Re-netlist Entire Design</i>	Specifies whether the system runs a new netlist for all cellviews in your design.
<i>Run Simulation Remotely</i>	Specifies whether the simulation runs on a remote file server or on your local computer. <ul style="list-style-type: none">■ When selected, runs simulation on a different computer. <i>Simulation Host Name</i> and <i>Simulation Host is Different Type</i> specify the computer that will be used for remote simulation.■ When deselected, runs simulation on your computer. <i>Simulation Host Name</i> and <i>Simulation Host is Different Type</i> have no effect.
<i>Simulation Host Name</i>	Specifies the computer on which the simulation runs. The default is <i>localhost</i> , which is your machine. Type the name of the computer on which you want to run remote simulation. This field requires that <i>Run Simulation Remotely</i> is turned on.
<i>Simulation Host is Different Type</i>	Specifies whether the computer on which you run the simulation has a different binary storage format. <ul style="list-style-type: none">■ When selected, indicates that the storage format is different. If you do not set this field correctly, you cannot view your waveform results after the simulation finishes. This field requires that <i>Run Simulation Remotely</i> is turned on.■ When deselected, indicates that the storage format is the same.

Simulation Environment Help

SE Form Reference

Related Topics

[Specifying Simulation Environment Options](#)

[Initializing the Simulation Environment](#)

[Initialize Environment Form](#)

Files Used by the Cadence SILOS II Simulator

The following files are used by the Cadence SILOS II simulator:

- control file
- si.inp file

When you run a simulation, the system automatically translates the `control` file and creates the `si.inp` file, which it uses as input for the simulator. This file contains the same text as the `control` file, but with commands in square brackets replaced by their interpreted value.

Related Topics

[Sample control File for Cadence SILOS II](#)

[Sample si.inp File Generated for Cadence SILOS II](#)

[Substitution Functions](#)

Sample control File for Cadence SILOS II

The following is a sample `control` file for Cadence SILOS II.

Line	#
1	input netlist
2	input .term
3	\$This is test #1 of the 74LS169A counter.
4	.TABfsLE [#QD] [#QC] [#QB] [#QA] [#RCO]
5	[#CLOCK] .CLK 0 S0 20 S1 30 S0 .REP 0
6	.PATTERN [#UP] [#LOAD*] [#P*] [#T*] [#D] [#C] [#B] [#A]
7	0 0 0 0 0 0000
8	60 0 0 0 0 0000
9	120 0 0 0 0 0001
10	180 0 0 0 0 0010
11	240 0 0 0 0 0011
12	.EOP
13	!type errors
14	!simul 0 to 6960
15	!save
16	!type errors
17	!type outputs on change
18	!type network
19	.end
20	!exit

Simulation Environment Help

Files Used by the Cadence SILOS II Simulator

The following table provides a line-wise description of the `control` file.

Line Number	Line Function
1	Directs Cadence SILOS II to read the netlist file using the Cadence SILOS II file inclusion command.
2	Directs Cadence SILOS II to read network data from the terminal input that is the <code>si.inp</code> file created from this <code>control</code> file.
3	Indicates a comment line.
4, 5, 6	Specifies name translations.
7, 8, 9, 10, 11	Specifies stimulus patterns used to drive the simulator.
13, 14, 15, 16, 17, 18	Indicates Cadence SILOS II commands. The exclamation point (!) is an escape character required for commands that follow the <code>input.term</code> line.

Related Topics

[Files Used by the Cadence SILOS II Simulator](#)

[Sample si.inp File Generated for Cadence SILOS II](#)

[Substitution Functions](#)

Sample si.inp File Generated for Cadence SILOS II

The following sample `si.inp` file has been created from a `control` file. In this file, the net names are replaced by N, followed by the net number given on lines 4, 5, and 6 of the `control` file. These net names have been generated by the flat netlister.

```
input netlist
input .term
$This is test #1 of the 74LS169A counter.
.TABLE N4 N3 N2 N1 N23
N14 .CLK 0 S0 20 S1 30 S0 .REP 0
.PATTERN N15 N22 N16 N17 N21 N20 N19 N18
0 0 0 0 0 0000
60 0 0 0 0 0 0000
120 0 0 0 0 0 0001
180 0 0 0 0 0 0010
240 0 0 0 0 0 0011
.EOP
!type errors
!simul 0 to 6960
!save
!type errors
!type outputs on change
!type network
.end
!exit
```

Related Topics

[Files Used by the Cadence SILOS II Simulator](#)

[Sample control File for Cadence SILOS II](#)

[Substitution Functions](#)