cadence®

Virtuoso Parasitic Aware Design User Guide

Product Version IC23.1
November 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

© 2023 All Rights Reserved.

© 2023 All Rights Reserved.

© 2023 All Rights Reserved.

© 2023 All Rights Reserved.

© 2023 All Rights Reserved.

© 2023 All Rights Reserved.

© 2023 All Rights Reserved.

© 2023 All Rights Reserved.

1

Parasitic Aware Design in Virtuoso ADE Explorer, Virtuoso ADE Assembler, and Virtuoso Schematics L/XL

Important

Virtuoso Parasitic Aware Design (also previously known as MSPS and *Virtuoso Parasitic Simulation*) has been renamed *Virtuoso Parasitic Aware Design* from the IC6.1.2 release.

This topic describes how to use Cadence® tools to investigate the effects of parasitics on circuits using Virtuoso® Parasitic Aware Design in the analog design simulation environments (ADE Explorer and ADE Assembler) and Virtuoso® Schematic Editor (Schematics L/XL) applications.

From the IC6.1.1 release, the availability of parasitic aware design functionality differs depending upon the software you access it from.

Licensing Requirements

You do not need any additional license to run Virtuoso Parasitic Aware Design from Schematic L, or Schematic XL, ADE Explorer, or ADE Assembler. The license already checked out by these products is used.

For information about licensing in the Virtuoso Studio design environment, see *Virtuoso Software Licensing and Configuration Guide*.

© 2023 All Rights Reserved.

Introducing Parasitic Aware Design in ADE Explorer, ADE Assembler and Schematics L/XL

You can access parasitic aware design functionality through the following applications:

- Virtuoso® ADE Explorer
- Virtuoso® ADE Assembler
- Virtuoso Layout Suite XL
- Virtuoso Schematic Editor (Schematics L/XL)
- Virtuoso Specification-driven Environment (SdE)
- Virtuoso AMS Simulator (AMS)

This chapter discusses parasitic aware design functionality available in ADE Explorer, ADE Assembler and Schematics L/XL. It describes how to setup parasitics, display parasitics, and generate parasitic reports for your design.

This chapter contains sections on the following:

© 2023 All Rights Reserved.

© 2023 All Rights Reserved.

Analog Design Flow Supported by Parasitic Aware Design

The following information outlines a typical analog design flow supported by parasitic aware design:

Summary Action	Description
Create a Schematic View	The analog designer creates a schematic design using the Virtuoso Schematic Editor. The result of this is the generation of a schematic view. The designer would then simulate the schematic view using the Virtuoso Analog Design Environment (as a front-end to the Spectre or UltraSim simulators) **Note:** This schematic contains discrete components and values that the designer will require to achieve desired performance levels.
Create a Layout View	The schematic design is then laid out on silicon. This is done either manually using the Virtuoso Layout Suite L, and referencing the schematic view, or it can be auto-assisted using Virtuoso Layout Suite XL. The result of this is the generation of a layout view.
Use PVS	PVS (Physical Verification Schematic) performs LVS (Layout Versus Schematics) and confirms that the layout view accurately corresponds to the schematic view. Generally, the LVS result must be “clean” before you can proceed to the next stage.

© 2023 All Rights Reserved.

Use Quantus

Quantus (Resistance / Capacitance and Inductance Extraction) establishes all the extra, non-designed, parasitic components that result from the physical layout. This includes, for example:

- The resistances (R) that arise from the lengths, widths and resistivity of metal or polysilicon, as well as vias, used to implement connecting wires on silicon.
- The capacitances (C) between these connecting wires and power or ground supply planes, or between the connecting wires and adjacent wires that are crossed on neighboring layers.
- The inductance (L) and mutual inductance that arises because of the coupling between wires.

Quantus creates a *parasitic netlist* that can be output in one of the following formats:

1. SPICE netlist variant (DSPF), which cannot be processed by parasitic aware design
2. *Extracted* view
3. *Smart* view.

For more information, see Views Supported by Parasitic Aware Design.

Use of Quantus also provides the designer with feedback on the modified parasitic values and their impact on the performance.

See also the *Quantus QRC Extraction Users Manual.*

Use Parasitic Aware Design - Parasitic Probing

A key parasitic aware design function is *parasitic probing*.

Here, a designer can open a schematic view, provide parasitic aware design details of the associated extracted view, and then probe (report on) nets on the schematic. This will generate summary (report) information on the parasitics (for example, parasitics for a net, parasitics between two terminals of a net, or the parasitics between two nets).

For more information see Parasitics – Report Parasitics on page 56.

© 2023 All Rights Reserved.

Use Parasitic Aware Design - Parasitic Backannotation

A further key parasitic aware design function is to provide *parasitic backannotation*.

Backannotation involves annotating each net on the schematic with summary information about the parasitics associated with the net.

For more information see Parasitics – Show/Hide Parasitics on page 55.

Use Parasitic Aware Design - Parasitic Refinement

Another key function is called *parasitic refinement*.

An extracted view can contain thousands of parasitics that can have a dramatic impact on analog simulation of the laid out design. Refinement allows a designer to specify which net parasitics are considered significant and to create a new version of the extracted view only containing these.

Important

This generates the *refined extracted* view which is given the default name `av_analog_extracted`.

For more information see Parasitics – Refine Extracted View on page 65.

Using the Analog Design Environment for Post Layout Simulation

Post layout simulation allows a designer to explore the parasitic impact on design performance.

A *testbench* schematic is created containing instances of the design. A *configuration* is created (in the Cadence Hierarchy Editor) specifying which view should be used when simulating the design instances, for example *schematic*, *av_extracted*, or *av_analog_extracted*. Once again, Spectre or UltraSim can be used via the Analog Design Environment.

For more information see Simulating the Design in ADE Explorer on page 107.

© 2023 All Rights Reserved.

Use Parasitic Aware Design - out-of-context Probing

Once simulation has completed you may want to monitor the signals at certain points. Another key parasitic aware design function is therefore to allow you to descend from the testbench schematic into the schematic view of the design instance (even though the extracted/refined view was simulated).

When in this schematic view, the designer is said to be “out-of-context”. The Analog Design Environment allows a designer to perform out-of-context probing on the schematic, and parasitic aware design handles the mapping between the points probed on the schematic and the equivalent points in the much more complex netlist with parasitics.

For more information see Out-Of-Context Probing on page 69.

© 2023 All Rights Reserved.

Views Supported by Parasitic Aware Design

Parasitic aware design features are used to analyze the effect of parasitic devices on a circuit. Once the physical design is complete, you can create an Extracted View or a Smart View using Quantus to support parasitic simulation. In ADE, parasitic aware design supports the following views for parasitic extraction:

- Extracted View
- Smart View

Extracted View

The Extracted View is a graphical representation of the parasitic netlist that is used to debug circuits, and to verify and simulate post-layout designs.

Quantus QRC creates a parasitic netlist that can be output as a SPICE netlist variant (DSPF), which cannot be processed by parasitic aware design, or as an *extracted view*.

The extracted view typically includes parasitic components that are standard parasitic components from a given library (typically called "`pres`", "`pcap`", "`pind`", and "`pmind`").

A typical schematic design might contain, for example, 50 design elements, but the extracted view could contain many thousands of parasitic elements.

Important

The Quantus QRC extracted view is given the default view name *av_extracted*.

Smart View

The Smart View is an enhanced version of the extracted view. It provides the same functionality as the Extracted View, but with a few limitations. It uses a highly efficient and scalable storage mechanism and stores parasitic elements in a compact data structure. Because of this improved storage scheme, the Smart View can manage larger, more complex designs and advanced nodes with a reduced overall extraction run time and netlist size. Consequently, netlist generation is faster in Virtuoso ADE when you are using the Smart View.

Note: The Smart View is given the default view name *smart_view*.

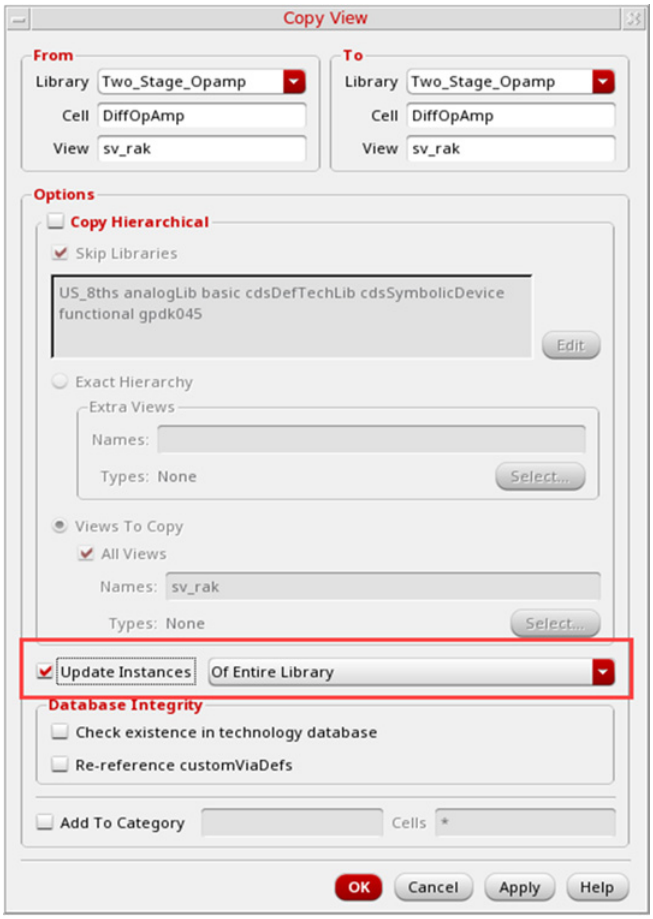
© 2023 All Rights Reserved.

Important

The Smart View is supported only in Quantus version 18.2 or higher.

Tip

If you need to create a copy of a Smart View, choose *Library Manager – Edit–Copy*. In the *Copy View* form, select the *Update Instances* check box to ensure that data integrity is maintained.



For more information, see:

- Benefits of Smart View
- Known Limitations of Using Smart View
- Features Supported in the Smart View Flow
- Prefixes in Instance Names in the DSPF File

© 2023 All Rights Reserved.

Benefits of Smart View

The Smart View offers the following benefits during parasitic extraction:

- Delivers the fastest path to post-layout verification and simulation with the following performance improvements:
  - Up to two times faster than extracted view.
  - Up to five times smaller netlist size than extracted view.
  - Up to three times faster Virtuoso ADE netlisting turnaround time.
- Provides same usability and integration in Virtuoso and Virtuoso ADE platforms.
- Provides key post-layout simulation verification functionality, such as in-context cross-probing and back annotation.
- Uses DSPF file format output to allow direct support for Fast SPICE tools like Spectre XPS.
- Enables faster verification and simulation run times with Spectre APS and Spectre XPS.
- Enables the hierarchical extraction flow in Virtuoso ADE platform.
- Enables reporting of inductance and mutual inductance values in the netlist.
- Requires no additional foundry enablement to support files such as `.trp` or `icellmap`.

 Video

For a video overview of Smart View, see Using Smart View in the ADE Flow on Cadence Online Support. Also read a blog related to this at Virtuosity: A Smart Extracted View.

Known Limitations of Using Smart View

Smart view output format does not support the following flows or extraction features:

- Parasitic Estimates
- Parasitic Filters
- SNA substrate extraction
- Wide band inductance model
- Third-party simulator support

© 2023 All Rights Reserved.

- Multi-corner extraction for AMS

For more information on extracted view and Smart View, see the *Quantus QRC Extraction User Manual*.

 Video

For a video overview of Smart View, see Using Smart View in the ADE Flow on Cadence Online Support. Also read a blog related to this at Virtuosity: A Smart Extracted View.

Features Supported in the Smart View Flow

To know about the additional features or flows supported in the Smart View flow, see the following sections:

- DSPF Support
- Multi-process Corner Support
- Resistor Bounding-box Support

© 2023 All Rights Reserved.

DSPF Support

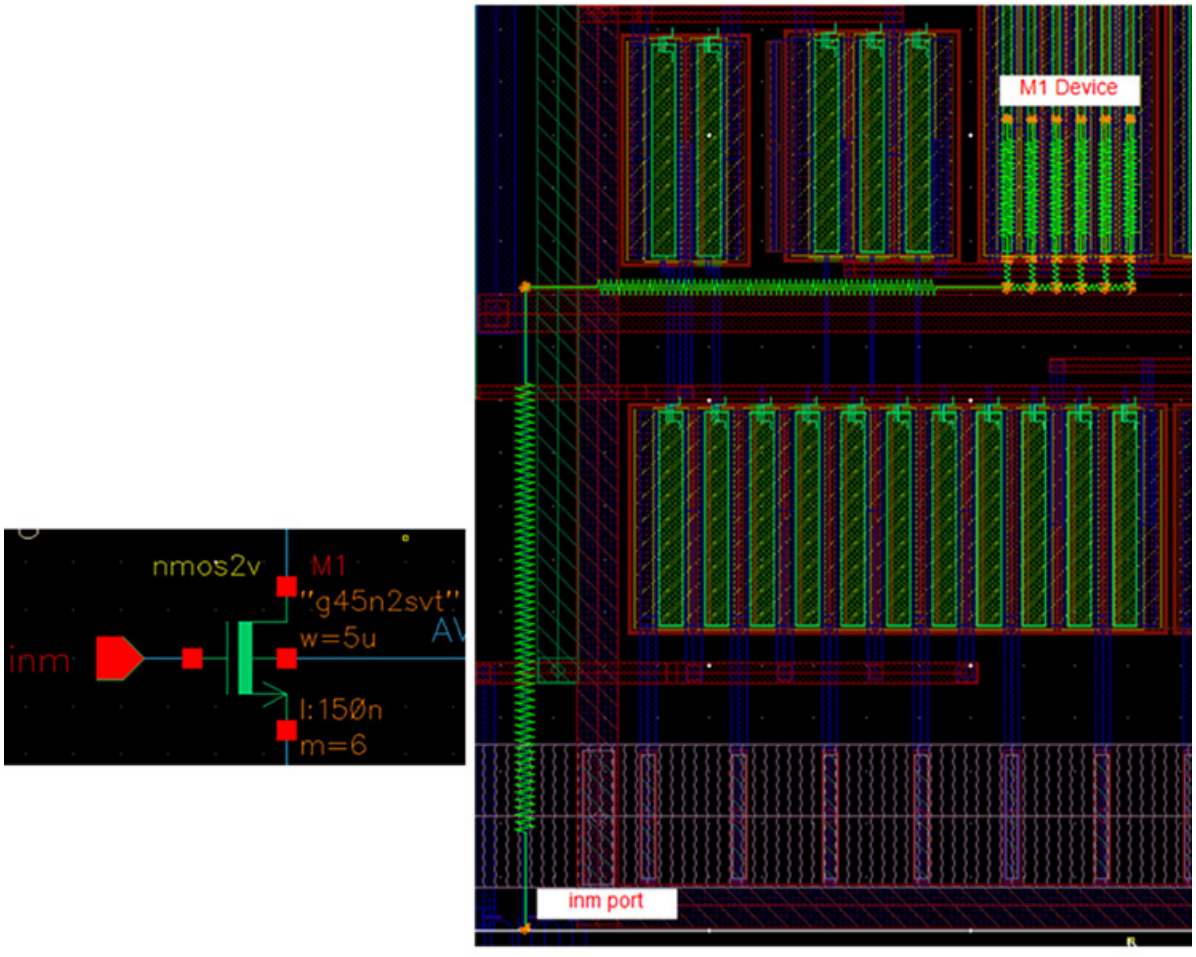
The netlist generated for the Smart View is saved in the DSPF format. By default, this DSPF file is saved in `./simulation/<Lib>/<Cell>/<View>/results/maestro/SmartViewDSPF/.tmpADEDir_<login>/<lib_cell_view>/<subckt>.dspf`.

The syntax of the netlist matches the SPF standards for connections. This means that the name of the last node in the netlist must be printed as `<instance><delimiter><instance_pin>`.

Generating a netlist in compliance with the SPF standards also provides significant performance improvement by reducing the netlisting time and peak memory usage.

Consider an example, where the Smart View includes the following:

- An `M1` device
- A net `inm` containing a series of sub-nodes `inm#`*`x`* where *`x`* is an integer



© 2023
All Rights Reserved.

Subsequently, a finger of M1 will have it its gate connected to one of these sub-nodes.

- Earlier, the netlist would print the connections as follows:

```
*|I (inm#3 XM1_4__rcx g X 0 21.165 15.295) // $llx=21.165 $lly=15.295
$urx=21.165 $ury=15.295 ¶
r_1_12291 inm#12 inm#3 7.5287 $poly_conn $l=0.0995 $w=0.144¶
```

- Now, the netlist prints the connections as follows:

```
*|I (XM1_4__rcx#g XM1_4__rcx g X 0 21.165 15.295) // $llx=21.165 $lly=15.295
$urx=21.165 $ury=15.295¶
r_1_3 inm#12 XM1_4__rcx#g 7.5287 $l=0.0995 $w=0.144 $poly_conn¶
```

Here, the sub-node `XM1_4__rcx#g` is related to the net `inm` by the same connection to the sub-node `inm#12`.

The DSPF output must have the following `.ccl` settings:

- `output_setup -net_name_space schematic`
- `output_db`
  - `-include_parasitic_res_width "true"`
  - `-include_parasitic_res_length "true"`
  - `-include_parasitic_res_model_by_sub_conductor "true"`
  - `-output_xy canonical_cap canonical_res diode mos bipolar generic parasitic_cap parasitic_res`
  - `-add_bulk_terminal true`

If the Smart View has been extracted using the following command, the OA database contains bounding box information for that instance:

```
output_db  -include_instance_bounding_box "true"
```

Therefore, when Virtuoso ADE netlists the DSPF, the instance section also includes the bounding box information as follows:

```
M1 S#1 G#1 D#1 B#1 g45n1vt L=0.15u W=5u $X=15.725 $Y=42.985 $LLX=15.65 $LLY=40.485 $URX=15.8 $URY=45.485
```


© 2023 All Rights Reserved.

The DSPF file supports the QRC options. By using the QRC options, you can prevent cells from a file, or cells of a specific type from being included in the DSPF file. For example, you can specify the following QRC options:

```
-parasitic_blocking_device_cells_file
-parasitic_blocking_device_cells_type
```

For more details on the DSPF format, see the *Quantus QRC Extraction User Manual*

You can specify a new location to save the DSPF file by using the environment variable `smartViewDSPFDirectory`. Saving the DSPF file to the specified location allows it to be shared by every run instead of copying it for each run.

The number of parasitic resistors may be higher when you use Smart View, as compared to when you use the extracted view. This is because Smart View uses DSPF, and unlike SPICE, DSPF does not support abutment connections between `M1.s` and `M2.d` without a resistor. Additional resistors in the Smart View connect two device pins. These resistors are located on layers such as `mwires`, `dwires`, `rwires`, `bwires`, and more, and their values are `0.001`.

Note: By setting the environment variable `"spectre.envOpts" "case sensitive"` to `true` in the `.cdsinit` file and re-netlisting, the Smart View automatically enables case sensitivity in the regenerated netlist and adds `case_sensitive = true` in the `dspf_include` statement.

The DSPF generated by Virtuoso ADE relies on parameters being defined in the simulation information section of the Edit CDF form. If parameters are missing from this section, the results from EM/IR simulations might be inconsistent. However, timing simulations are not affected. For example, the netlist generated by ADE Assembler might contain the following statement:

```
XR0[1] net1#1 net2#2 GND#2 RES1 semiResL=1.4 w=99 + multFactor=1
semiResHeadNum=1 _par0="RES1" $x==2 + $y=3
```

Observe that since the simulation information section in the Edit CDF form does not contain the parameter `l`, it is not netlisted in the DSPF generated for the Smart View. This is because the resistor RES1 is a pcell and the parameter `l` is not referenced in the lower hierarchy, and this is why it is not used for the instance XR0[1].

To ensure that the `l` parameter is added to the netlist, add the following SKILL flag in the simrc file before netlisting:

```
nlCustomHierEmirParamList  = 'list(list("RES1" list("l")))
```

This flag requires arguments in a format of `list(list(model list(params))`.

© 2023 All Rights Reserved.

When this SKILL flag is set, the netlist will print the following statement for resistor RES1:

```
XR0[1] net1#1 net2#2 GND#2 RES1 semiResL=1.4 w=99 + multFactor=1
semiResHeadNum=1 _par0="RES1" l=1.4  $x==2 + $y=3
```

`dspf_include` provides special features for reading the DSPF format data, for example, port order adjustments, or handling of duplicated subcircuits. Ensure that you do not use the `include` or `.include` commands to read the DSPF format data because these commands do not have the special functions that the `dspf_include` or `.dspf_include` command provides.

Note: Setting the `.simrc` variable `hnlMaxLineLength` does not affect the `dspf_include` links in the input.scs file.

Related Topics

hnlMaxLineLength

Port Order Handling in Running a Simulation

Multi-process Corner Support

If you use a design block that is bound to a Smart View that contains multi-process corners (MPCs), you must set the environment variable `autoSyncMPC` to `t`. These multi-process corners are assigned a variable name in the format <*`ViewName_extractionCorner`*>. The `maestro` cellview displays these multi-process corners as a design variable.

There may be a degradation in performance when this environment variable is set, and you open a cellview that contains multi-process corners. By default, the environment variable is set to `nil`.

Important

Creating a `maestro` view from a `config` file sets the AMS simulator as the default simulator. Changing the simulator to Spectre might not update the MPC variables as expected. To allow `autoSyncMPC` to automatically sync the MPC variables, it is recommended that you create a `maestro` view using the schematic and bind it to the config view.

© 2023 All Rights Reserved.

Resistor Bounding-box Support

To view multiple resistor bounding box components in the DSPF file generated for a Smart View, ensure the following:

- The design must have bends or U-shape routing.
- The Smart View must have the following `.ccl` setting:

  ```
  -include_parasitic_res_conductor_bounding_box "Multiple"
  ```

- If the `.ccl` setting is included but the bounding box components are not visible in the generated DSPF file, it is possible that the design does not contain the components or the design is not appropriate. In such a case, choose a different design and regenerate the Smart View.

  The following example shows the resistor bounding box components (such as `llx1`, `lly1`, `urx1`, `ury1`, and others) in the generated DSPF file:

  ```
  r_2_11276 11#a1 12#a1 0.0183713 $metal1_conn $lvl=3 $l=0.03 $w=0.1 $llx1=6.9
  $lly1=19.2045 $urx1=6.98 $ury1=19.205 $llx2=0.05
  $lly2=19.205 $urx2=573994 $ury2=16.3835 $llx3=0.008 $lly3=0 $urx3=6.97
  $ury3=19.235 $x=6.94 $y=19.2195
  ```

When the design contains trapezoid resistors, the generated DSPF file shows the details in the following format:

```
r_11_141451 138#net021 139#net021 1.78345 $metal2_conn $lvl=10 $l=9.4395 $w=0.25
$llx1=40.145000 $lly1=39.390000 $urx1=40.225000
$ury1=39.640000 $llx2=31.520000 $lly2=39.390000 $urx2=40.145000 $ury2=39.640000
$llx3=31.270000 $lly3=39.390000 $urx3=31.520000
$ury3=39.640000 $llx4=31.270000 $lly4=39.640000 $urx4=31.520000 $ury4=39.825000
$llx5=31.270000 $lly5=39.825000 $urx5=31.520000
$ury5=39.920000 $llx6=31.190000 $lly6=39.920000 $urx6=31.270000 $ury6=40.155000
$llx7=31.270000 $lly7=39.920000 $urx7=31.300000
$ury7=40.155000 $llx8=31.170000 $lly8=39.905000 $urx8=31.190000 $ury8=40.155000
$t1=31.190000,39.905000,31.190000,39.920000,31.270000,39.920000,31.270000,39.8250
00
$t2=31.315000,39.920000,31.315000,40.140000,31.520000,39.935000,31.520000,39.9200
00
$t3=31.300000,39.920000,31.300000,40.155000,31.315000,40.140000,31.315000,39.9200
00 $x=35.687 $y=39.515
```

For more details on the DSPF format, see the *Quantus QRC Extraction User Manual*.

© 2023 All Rights Reserved.

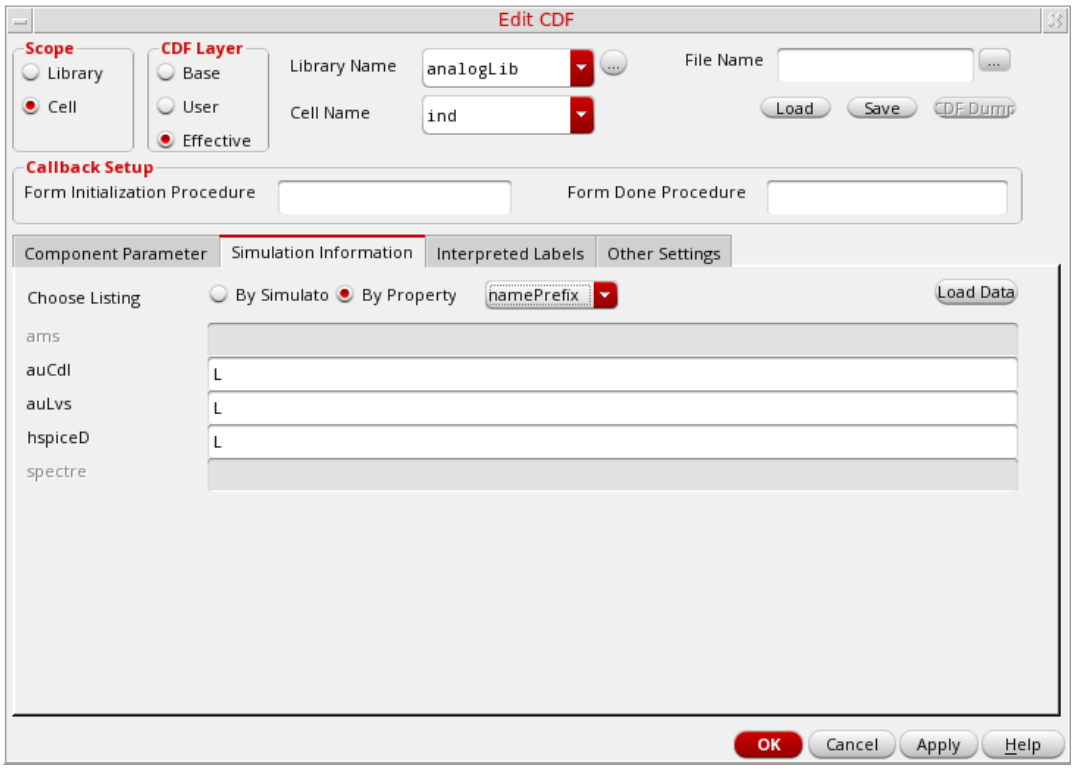
Prefixes in Instance Names in the DSPF File

The device names in a DSPF file generated during Smart View netlisting may not have the prefix required by the simulator to understand their types. Therefore, the tool needs to add a prefix to such devices. For example, a user-defined capacitor, `R1`, will require a prefix `C` so that the simulator can consider it a capacitor.

To set naming prefixes for devices, choose one of the following methods:

- Use the default flow. In this case, no environment variables are set.The prefixes are applied as follows:
  - If the `namePrefix` is the same as the first character of the device name, the netlister does not add a prefix to the instance name in the netlist.
  - If the `namePrefix` is not the same as the first character of the device name, the netlister adds the `namePrefix` value to the instance name in the netlist.

  This prefix is determined by the `namePrefix` property defined in the *Simulation Information* section of the Edit CDF form.



  For example, the `ind` device is netlisted in the DSPF with the `L` prefix, when the simulator used is *auCdl*, *auLvs*, or *hspiceD*.

© 2023
All Rights Reserved.

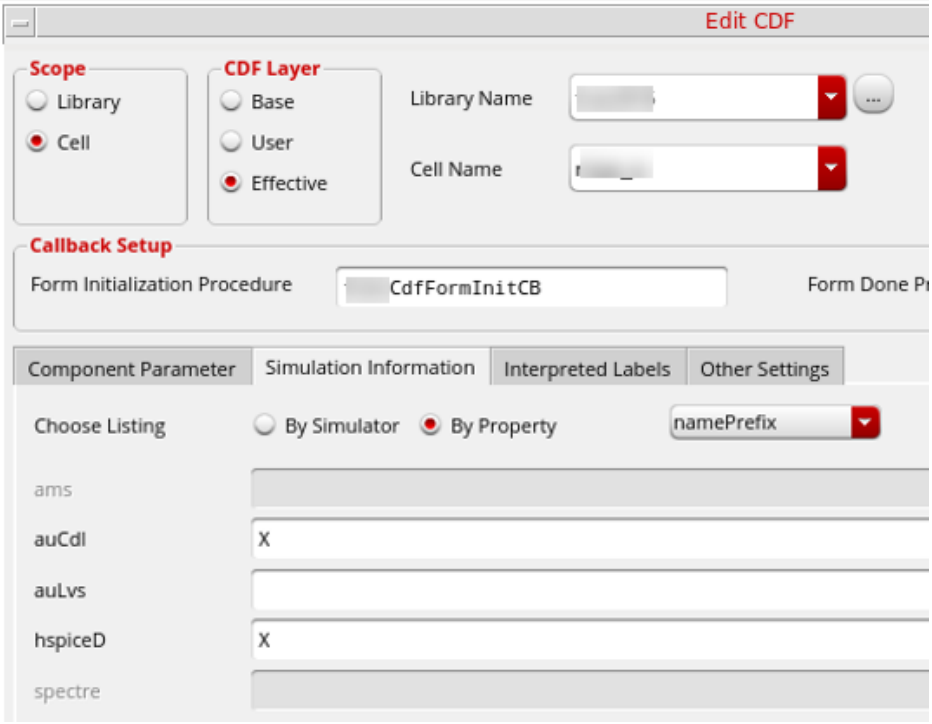
Typically, the Edit CDF form does not have any `namePrefix` settings for *ams* and spectre simulator. In such cases, the simulation fails due to the lack of correct prefixes.

- Set the dspfNamePrefixSimInfo environment variable to the name of the simulator whose setting you want to use. For example, against the *auCdl* simulator, the `namePrefix` property for a device is set to `R`. By default, this property will ensure that all resistor devices have the `R` prefix. However, instead you want to use `RX`, which is the `namePrefix` for *hspiceD*. In this case, set the `dspfNamePrefixSimInfo` variable to `hspiceD`. With this setting, the netlister ignores the `namePrefix` value set for *auCdl* and uses the prefix specified for *hspiceD*. In this case, useSelfCDFInPostLayoutNetlist is ignored.

  Consider that you have resistors in the netlist as shown below:

  ```
  R1@2 (net1#1 net02#2 gnd#1 ) resistor l=3.5e-06 wr=8e-07 + multi=(1) $x=62.422
  $y=32.544
  ```

  To netlist this correctly with an `X` prefix, set `dspfNamePrefixSimInfo` to *auCdl* because the `namePrefix` for *auCdl* is `X`.




  ```
  XR1@2 (net1#1 net02#2 gnd#1 ) resistor l=3.5e-06 wr=8e-07 + multi=(1) $x=62.422
  $y=32.544
  ```

  This is applicable in cases when the *auCdl* prefix is not the same as the first character of the device name.

© 2023
All Rights Reserved.

- Use the useSelfCDFInPostLayoutNetlist environment variable to specify the list of simulators from which the simulation-specific CDF properties are to be used to apply name prefixes to devices. In this case, dspfNamePrefixSimInfo must not be set.

  If the `useSelfCDFInPostLayoutNetlist` environment variable is set to include your current simulator, which is usually *spectre*, the `namePrefix` value from the *Simulation Information* section of the Edit CDF form for the *spectre* simulator is used after considering the following rules:

  - If the `namePrefix` is the same as the first character of the device name, the netlister does not add a prefix to the instance name in the netlist.
  - If the `namePrefix` is not the same as the first character of the device name, the netlister adds the `namePrefix` value to the instance name in the netlist.
  - If the `namePrefix` is not specified, the netlister adds the prefix `X` to the instance name in the netlist.

  If none of the above rules are applicable, `useSelfCDFInPostLayoutNetlist` uses the `namePrefix` value set for the *auCdl* simulator in the *Simulation Information* section of the Edit CDF form.

  For example, when you specify the following:

  ```
  envSetVal("maestro.simulation" "useSelfCDFInPostLayoutNetlist" 'string "ams hspiceD")
  ```

  - If the current simulator is `ams`, the netlister will get the `namePrefix` value for `ams` from the *Simulation Information* section of the Edit CDF form.
  - If the current simulator is `hspiceD`, the netlister will get the `namePrefix` value for `hspiceD` from the *Simulation Information* section of the Edit CDF form.
  - If the current simulator is `auLvs`, the netlister will get the `namePrefix` value for `auCdl` from the *Simulation Information* section of the Edit CDF form.

  The default is `""`, which means that the `namePrefix` value from the `auCdl` simulator is used.

Related Topic

Rules for Adding Prefixes to Instance Names

dspfNamePrefixSimInfo

useSelfCDFInPostLayoutNetlist

© 2023 All Rights Reserved.

Accessing Parasitic Aware Design Functionality in Virtuoso Layout Suite XL/GXL

You can use the Smart View to view and analyze parasitic elements independent of the ADE flow. You can access parasitic aware design functionality if the current view is one of the following types:

- a layout view
- an extracted view
- a Smart View
- a configuration view (the schematic can refer back to a configuration)

To use an extracted or a Smart View, you need to setup your design environment to use Quantus QRC. For more details, see Step 2: Building an Extracted View using Cadence Quantus QRC Extraction.

The parasitic display using Smart View provides a visual display of the parasitic elements in the view within a user defined threshold. You can probe nodes to view the net IDs and the net names for the selected nodes. Smart view properties can be used to analyze layer and connectivity details.

To access Smart View in Virtuoso Layout Suite XL/GXL, choose *Smart-Parasitics* from the Layout XL/GXL menu bar.

Figure 1-1 The Smart-Parasitics menu in Virtuoso Layout Suite XL/GXL

Important points to note:

- The *Smart-Parasitics* menu is displayed only when you are working with a Smart View in Virtuoso Layout XL or GXL.
- In ICADVM20.1, you can use Smart View functionality in Virtuoso Layout Suite XL/EXL.

© 2023 All Rights Reserved.

Smart-Parasitics Menu

Selecting *Smart-Parasitics* displays the following menu options:

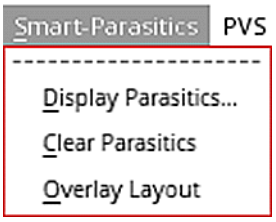


Figure 1-2 The Smart-Parasitics menu commands

A summary of the *Smart-Parasitic*s menu commands is listed below:

- Smart-Parasitics – Display Parasitics displays the parasitics on the selected nets or nodes on a highlighted set in the Smart View.
- Smart-Parasitics – Clear Parasitics removes the highlighted set of displayed parasitics in the Smart View.
- Smart-Parasitics – Overlay Layout opens a layout view in the background while keeping the Smart View in the foreground.

Note: The *Smart-Parasitic*s menu is not displayed if you open a Smart View that contains multi-process corners.

© 2023 All Rights Reserved.

Smart-Parasitics – Display Parasitics

Select *Smart-Parasitics – Display Parasitics* to display the *Display Parasitics* form with settings related to *Display* and *Threshold* for capacitances and resistances.

The values entered in this form control the threshold values of parasitics to display. Set the minimum value to 0 to display all parasitics.

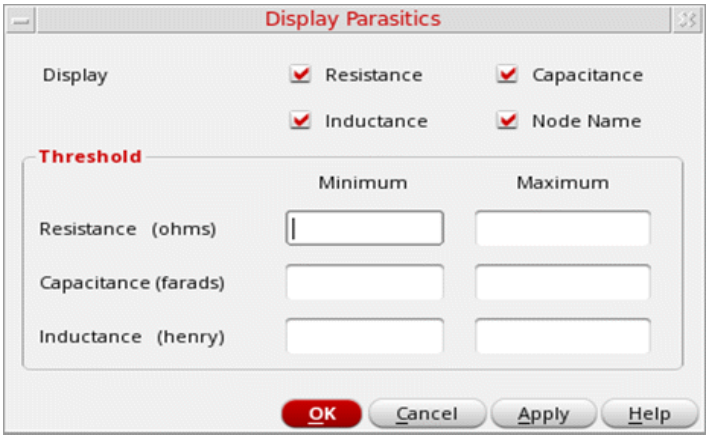


Figure 1-3 The Display Parasitics Form

The following table describes the fields on the *Display Parasitics* form:

GUI Item	Description
Display	
- *Resistance*	Enables or disables the display of the resistance parasitics in the *smart_view*.
- *Capacitance*	Enables or disables the display of the capacitance parasitics in the *smart_view*.
- *Inductance*	Enables or disables the display of the inductance parasitics in the *smart_view*.
- *Node Name*	Enables or disables the display of the net fragment names for parasitic nodes in the *smart_view*.
Threshold	
- *Resistance (Ohms)*	Specifies the minimum and maximum threshold for resistances to be used when probing parasitics on nets.
- *Capacitance (farads)*	Specifies the minimum and maximum threshold for capacitances to be used when probing parasitics on nets.

© 2023 All Rights Reserved.

GUI Item	Description
- Inductance (henry)	Specifies the minimum and maximum threshold for inductances to be used when probing parasitics on nets.

To display parasitics in the Smart View:

1. Open a Smart View in Virtuoso Layout Suite XL/GXL.
2. Select *Window – Assistants – Navigator* to display the Navigator assistant.
3. Choose *Nets* and select a net or an instance in the schematic.
4. Select *Smart-Parasitics – Display Parasitics.*

   The Display parasitics form is displayed.
5. Specify the display settings and the threshold values for resistance, capacitance, and inductance.
6. Click *OK*.

   The resistors and capacitors on the selected net are highlighted on the Smart View parasitic display.

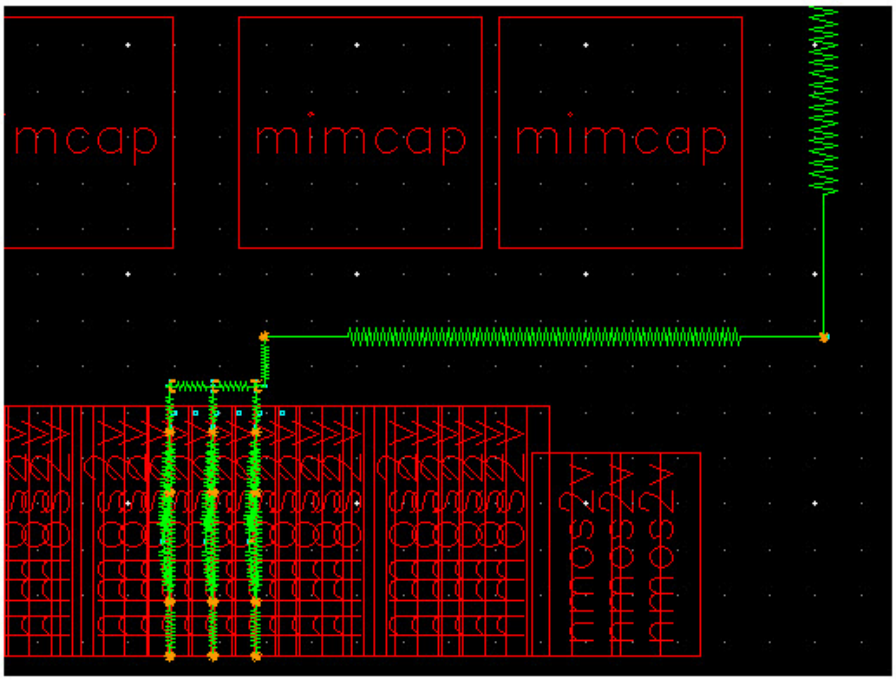


Figure 1-4 The Smart View after resistance and capacitance parasitics are displayed

© 2023
All Rights Reserved.

Similarly, the inductances on the selected net are highlighted on the Smart View parasitic display.

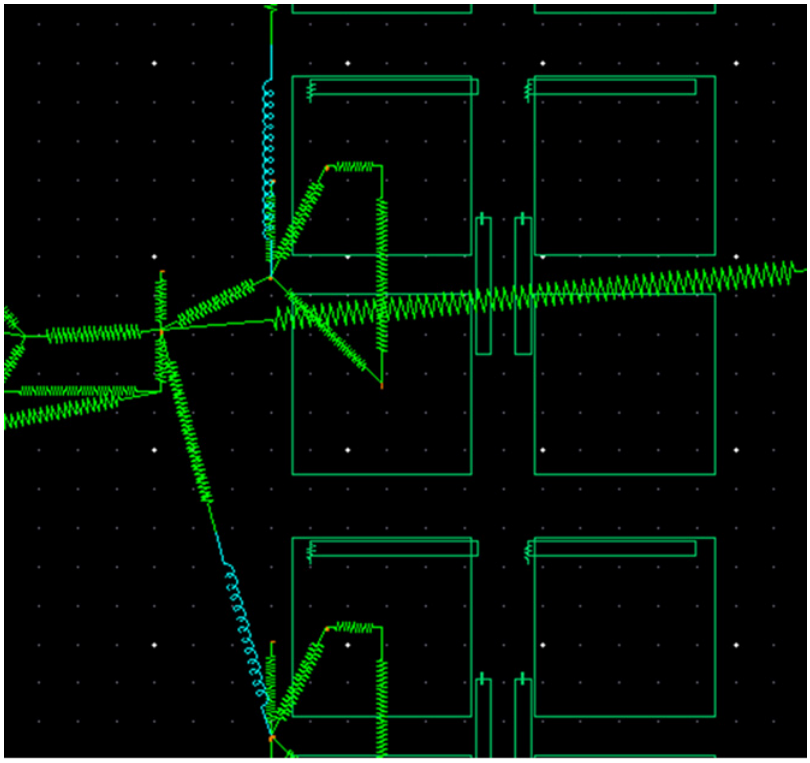
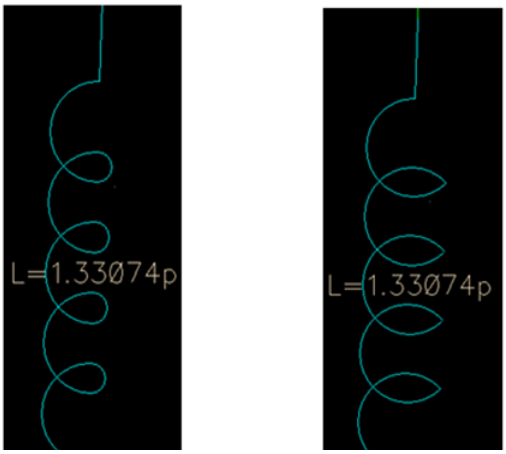


Figure 1-5 The Smart View after resistance, capacitance, and inductance parasitics are displayed

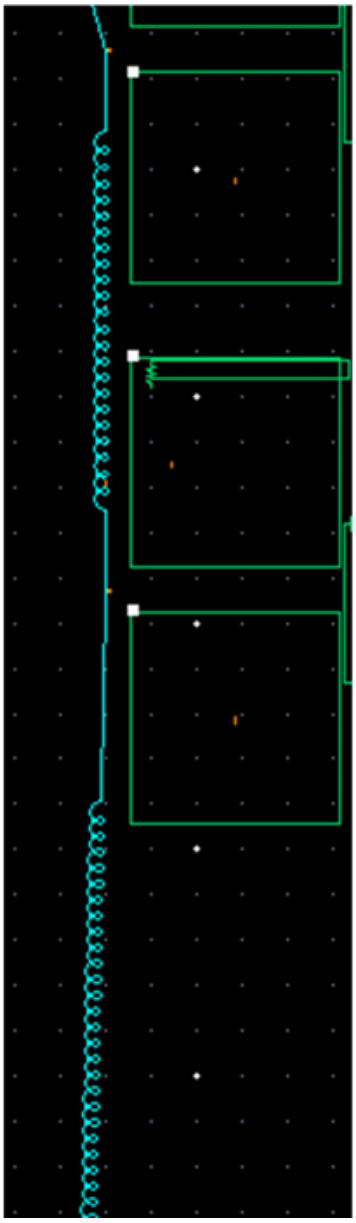
To change the style of inductance annotation, set the `svInductorStyle` environment variable to `ROUND` or `SHARP`.



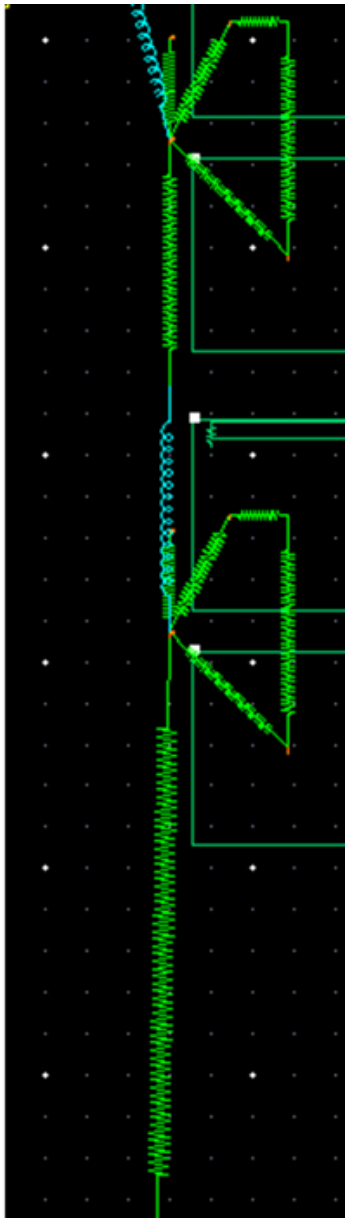
ROUND SHARP

© 2023 All Rights Reserved.

You can annotate the inductance with or without the resistors connected in series by toggling the *Resistance* option in the Display Parasitics form.



Resistance is not selected



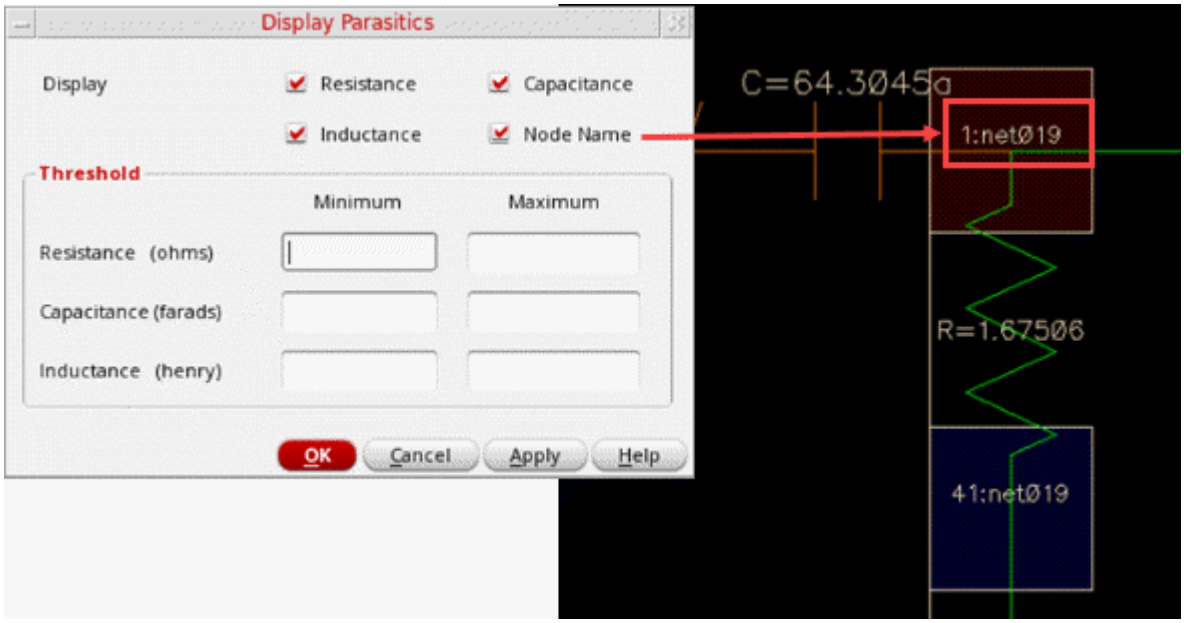
Resistance is selected

Note: The Smart View only displays self-inductance values.

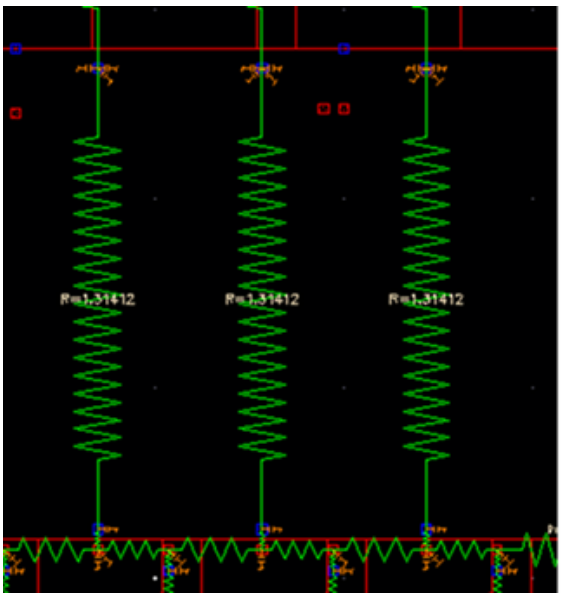
By default, the Smart View does not display the names of net fragments for parasitic nodes. To display these names, select the *Node Name* option in the Display Parasitics

© 2023 All Rights Reserved.

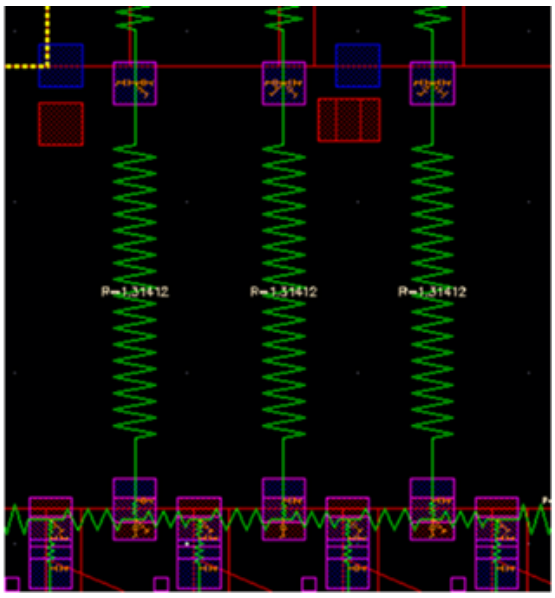
form or set the svDisplayNodeName environment variable to t. Alternatively, select a square on the parasitic display and choose *Parasitics – Probe Node*



The size of the parastic nodes might be small, especially in case of complex technologies. In such cases, you can use the svDotWidth environment variable to control the display size of the parasitic nodes.



svDotWidth = 100



svDotWidth = 500

© 2023
All Rights Reserved.

If the Smart View has been generated without any layer information, a `Y3:drawing` layer is created for the parasitic nodes. This ensures that the nets can be selected directly on the parasitic display or using the *Direct Plot* command, and then added to the ADE outputs.

Smart-Parasitics – Clear Parasitics

Select *Smart-Parasitics – Clear Parasitics* to remove the parasitics that are displayed by the *Display Parasitics* form.

Smart-Parasitics – Overlay Layout

Select *Smart-Parasitics – Overlay Layout* to display the corresponding layout view in the background while keeping the Smart View in the foreground. The background layout view is displayed in a new adjacent tab.

© 2023 All Rights Reserved.

The following illustration shows a Smart View design with the parasitics displayed by selecting *Smart-Parasitics – Display Parasitics.*

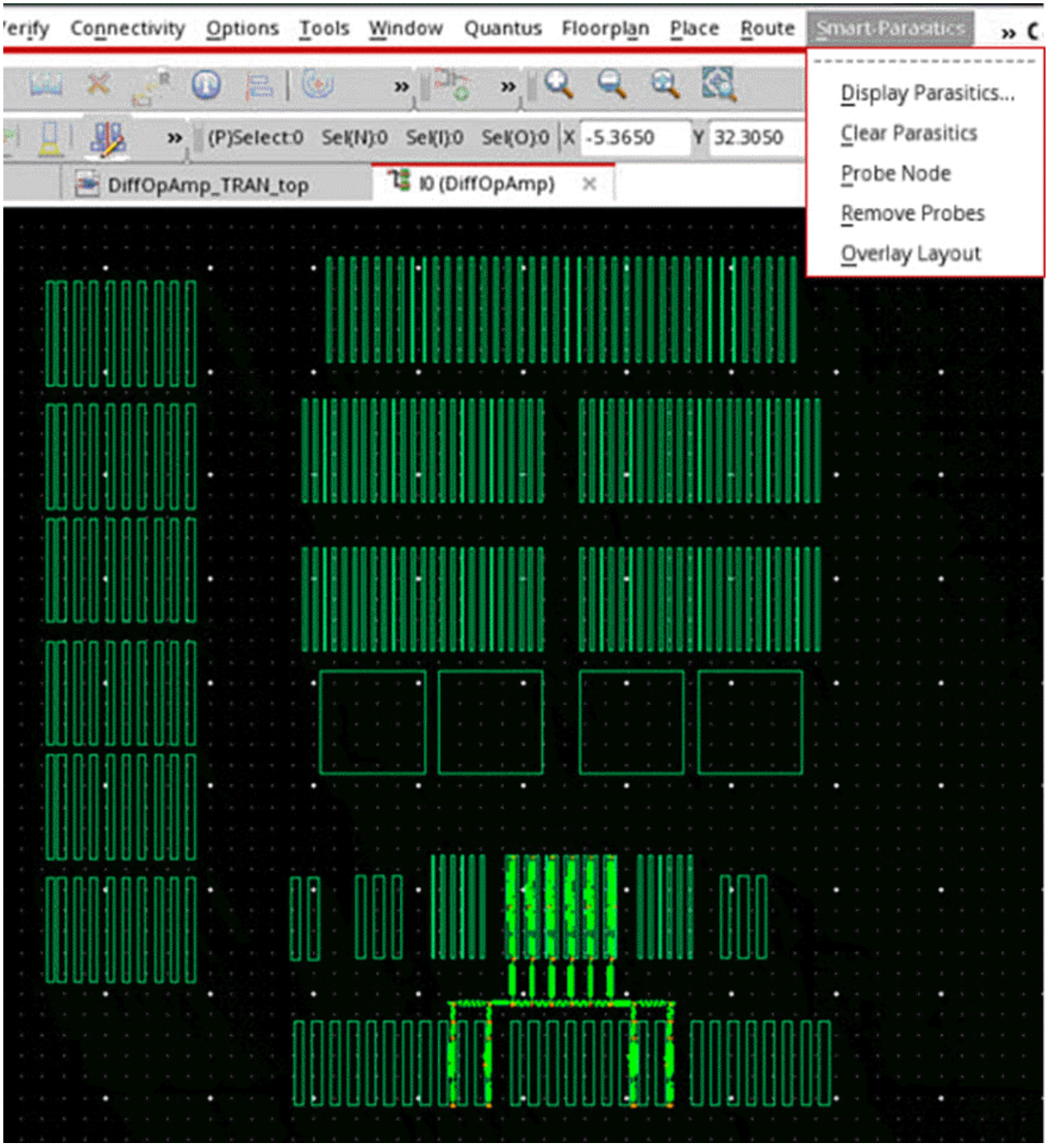


Figure 1-6 The Smart View before the layout is overlaid

© 2023
All Rights Reserved.

Selecting *Smart-Parasitics – Overlay Layout* displays the following view:

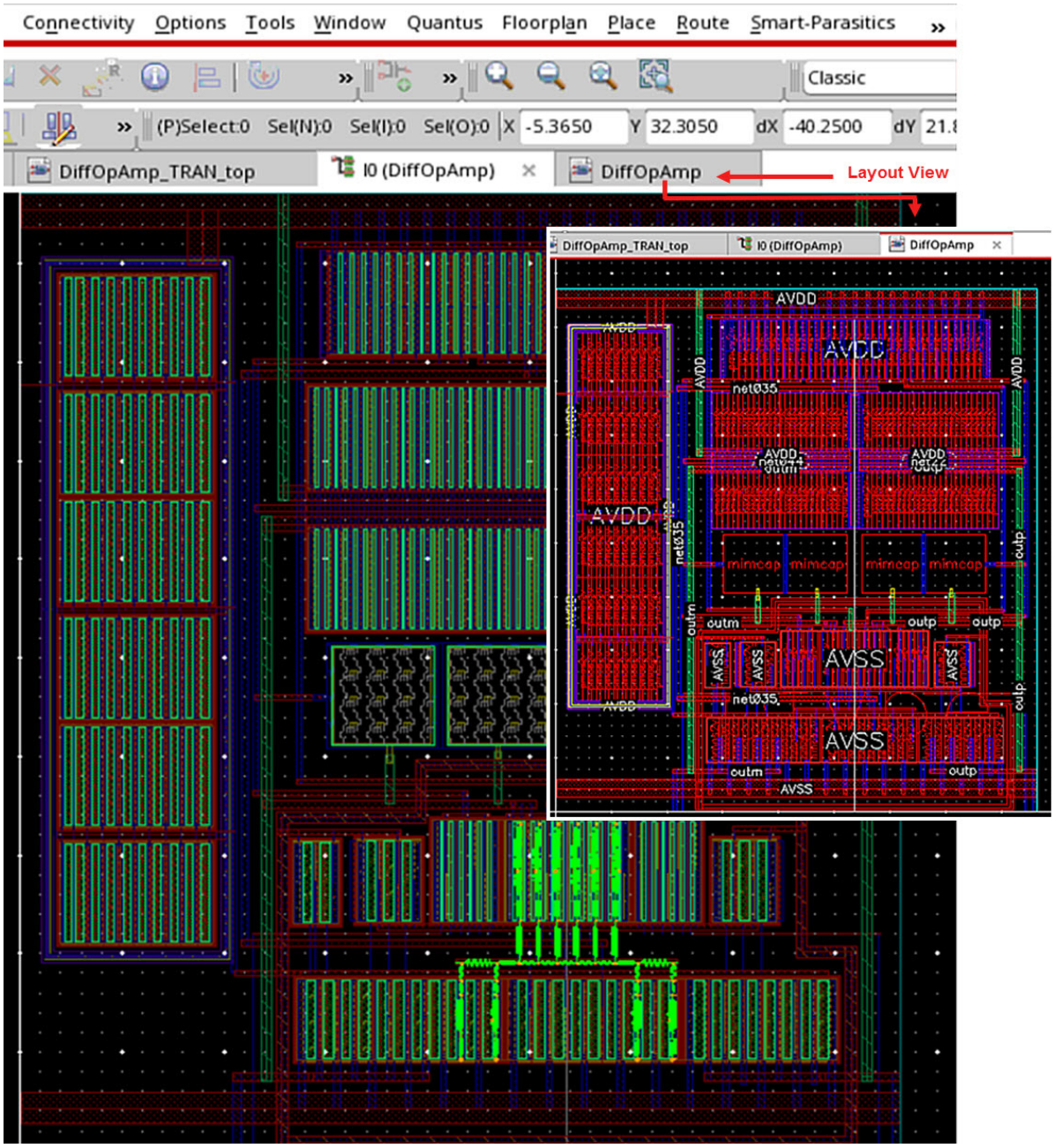


Figure 1-7 The Smart View after layout is overlaid

Note: If the layout view has been modified since the Smart View creation, overlaying the layout could show inconsistencies. A dialog box appears where you can choose to continue to overlay the layout view.

© 2023 All Rights Reserved.

Accessing Smart View in ADE

Parasitic aware design allows you to access the Smart View feature in the ADE flow. You can display and probe parasitics, generate reports and make comparisons between the parasitic elements in the Smart View.

The typical flow to use the Smart View in ADE is as follows:

1. Create the Smart View using Quantus.

Note: Ensure that the *Quantus* setup is complete before you attempt to create the Smart View. See Step 2: Building an Extracted View using Cadence Quantus QRC Extraction.

2. Open a `maestro` cellview in ADE Assembler.

   Note: The design block in the schematic for this cellview should be bound to the Smart View. This can be a view or a configuration that is bound to the Smart View.

3. Run the simulation in your ADE Assembler setup. When the simulation run completes you can select outputs and probe the design on the schematic.

Note: Smart view functionality is available in ADE Explorer and ADE Assembler.

4. In *ADE Assembler – Parasitics/LDE – Setup* form, select the Smart View in the *Extracted* tab.
5. Select the output on the Results tab.
6. Right-click and choose *Direct Plot* to descend into the Smart View to probe nodes directly in the parasitic display. See In-Context Probing.

Using the Smart View with ADE, you can analyze the extracted parasitics and their extraction results and determine the cause of unexpected outputs.

© 2023 All Rights Reserved.

Accessing Parasitic Aware Design Functionality in ADE Explorer, ADE Assembler and Schematics L/XL

You can access parasitic aware design functionality if the current view is:

- a schematic view
- an extracted view

  Note: Virtuoso Parasitic Aware Design supports simulation cross-probing of schematic nets even when the extracted view is used in the simulation.
- a Smart View
- a configuration view (the schematic can refer back to a configuration).

To access *Parasitics* from Schematics L/XL:

➤ Select *Launch – Plugins – Parasitics* from the Schematics L/XL menu bar.

This will add a Parasitics Menu to the Schematics L/XL menu bar.

Note: The *Parasitics* menu is displayed by default when you are working with ADE Explorer or ADE Assembler.

© 2023 All Rights Reserved.

Parasitics Menu

Selecting *Launch – Plugins – Parasitics* adds a *Parasitics* menu to Schematics L/XL with the following menu options:

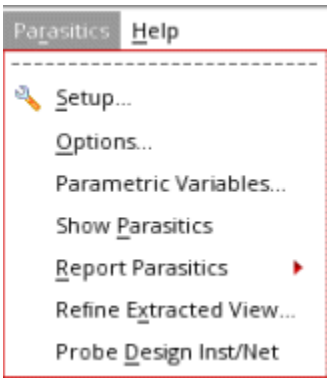


Figure 1-8 The Parasitic menu items

Note: The *Parasitics* menu is displayed by default when you are working with ADE Explorer or ADE Assembler.

A summary of the *Parasitic*s menu options in Schematics L/XL is detailed below:

- *Parasitics – Set Up* requires that you complete the necessary schematic, extracted, or configuration details before proceeding with parasitic aware design.

  Note: Along with the *Parasitics – Options* menu-item, *Parasitics – Set Up* is the only menu-item initially enabled.

- *Parasitics – Options* displays the Options form whose value settings determine how parasitic aware design handles certain backannotation and parasitic probing tasks.

- Parasitics – Parametric Variables displays the Specify Parameters Values form which can be used to set and edit parametric values for use in parasitic simulation.

- *Parasitics – Show/Hide Parasitics* lets you choose to turn on or off the display of parasitic values in the current schematic view. This in turn controls parasitic backannotation.

- *Parasitics – Report Parasitics* lets you undertake parasitic reporting on a selection of nets and terminals.

- *Parasitics – Refine Extracted View* can be used to generate a refined extracted view that can be used for circuit simulation.

© 2023 All Rights Reserved.

- *Parasitics – Probe Design Inst/Net* can be selected to probe a design instance or net, from the current schematic view, to extracted instances in the extracted view, or vice versa.

© 2023 All Rights Reserved.

Parasitics – Set Up

When parasitic aware design is enabled only the *Set Up* and *Options* menu options are available.

- If you access *Parasitics – Setup* from a schematic or extracted cellview you will have to complete the required *Schematic Cellview* or *Extracted Cellview* sections in the Setup form before you can continue to use parasitic aware design functionality (see Parasitics Set Up for Schematic and Extracted Views).
- If you access *Parasitics – Setup* from a top level configuration, or after you descend into a sub-block of the configuration, you will have to complete the required *Config Cellview* or *Simulation Data* sections in the Setup form before you can continue to use parasitic aware design functionality (see Parasitics Set Up for Configuration Views).

Selecting Cellviews to use with Parasitic Aware Design

You will not be able to proceed with parasitic functionality until either the schematic and extracted cellviews have been entered correctly, or a configuration view and simulation results have been chosen.

You can specify a cellview to use as follows:

- By clicking on the *Browse* button on the Parasitics Setup form and locating the view in the Library Browser form.
- By choosing the *Select by Cursor* option and then clicking on the relevant view.
- By selecting the view from the appropriate form fields.

Initial Parasitic Setup

Parasitic setup can take one of three forms dependent upon the following scenarios:

1. If you open a **schematic view** you will be displayed with a version of the Setup Parasitics Form (Figure 1-9 on page 46) that asks for extracted view details.

   If you open a **configuration** you are asked for a simulation database (Figure 1-10 on page 49).

   Note: The decision as to which Parasitics Setup form is displayed is dependent upon what cellview type is current.

© 2023 All Rights Reserved.

2. If a **schematic in the design below the top-level view has been opened directly**, just knowing the top-level view, or configuration, is not enough to determine which extracted view or Smart View (if any) corresponds to that schematic.

   This is because the schematic cannot be associated with a hierarchical instance path. In this scenario, you **must** specifically select the **extracted view**.

   However, as there is a risk that you may inadvertently select an extracted view that has not been simulated, the annotation of resistance (R) (Effective R Calculations) values will be disabled to avoid annotating incorrect effective values for the selected extracted view.

3. You have an **extracted view** open, in which case you must specify a matching schematic.

If required, you can change the Setup Parasitics form options at any time, with any changes made to the form being recorded for future use.

Note: Here, **extracted view** implies the extracted view or the Smart View.

© 2023 All Rights Reserved.

Parasitics Set Up for Schematic and Extracted Views

The *Setup Parasitics* form below will appear if you access parasitic aware design functionality from a schematic or extracted cellview.

Note: If the Setup Parasitics form has been invoked from a schematic window you will only have to specify an extracted view name and vice versa.

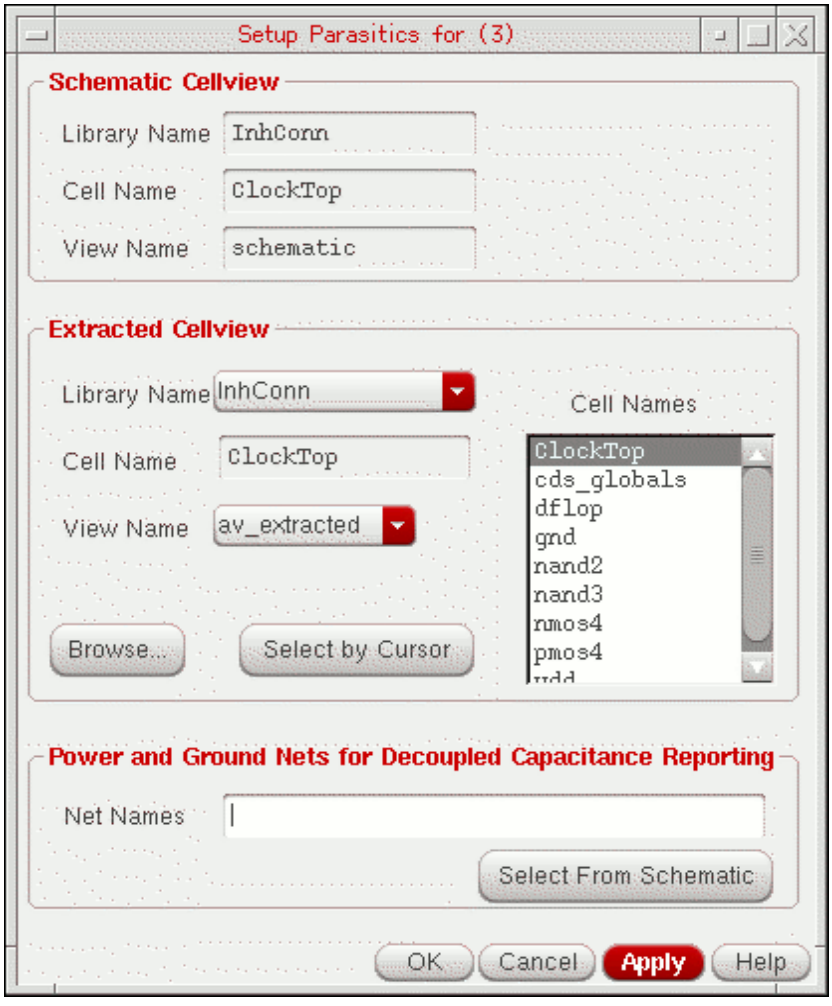


Figure 1-9 The Setup Parasitics form for schematic and extracted cellviews

© 2023
All Rights Reserved.

GUI Item	Description
Schematic Cellview	
- Library Name	Enter or select the library that contains the schematic cellview information.
- Cell Names	A cyclic menu that shows the cells in the selected library.
- View Name	Enter or select the schematic cell view.
- Browse	Use to search through hierarchies locating extracted cellview information.
- Select by Schematic	Select this option, then click directly on the schematic window you want to use.
Extracted Cellview	
- Library Name	Enter or select the library that contains the extracted cellview information.
- Cell Name/s	A cyclic menu that shows the cells in the selected library.
- View Name	Enter or select the extracted cell view. The default (but not fixed) Extracted Cellview *View Name*, from Quantus QRC, displayed in the setup form is the first instance of a view containing the string "`av_extracted`". If this string is not found the default view will be the first instance of a view containing the string "`extracted`". The list is sorted alphabetically.
- Browse	Use to move through hierarchies locating extracted cellview information.
- Select by Cursor	Select this option, then click directly on the extracted window you want to use.
Power and Ground Nets for Decoupled Capacitance Reporting	Use this field to set power and ground nets. Parasitic aware design uses these to determine which parasitic capacitances are decoupled (one side of the capacitor is connected to a power or ground net) or coupled (the capacitance is between two other nets). Parasitic reports allow you to select which types of capacitance to report. When you are selecting power and ground nets for the capacitance report, from a configuration, these selections must be made out-of-context. That is, you need to make your selections while in the schematic view of a block that is bound to an extracted view of a configuration. **Note:** Power and ground nets **must** be specified for coupled/decoupled parasitics reporting to work correctly. If no power and ground nets are specified, all capacitors are considered to be coupled capacitors.

© 2023 All Rights Reserved.

GUI Item	Description
- *Net Names*	Identify the power and ground nets that you want to use when differentiating between coupled and decoupled nets (see also Parasitics – Report Parasitics on page 56). **Note:** When entering a net name you can optionally add a "/" at the beginning of each net name. If omitted, the "/" is automatically added when comparing nets (this will not however be reflected in the *Net Names* field if a "/" was not originally entered). This is required for coupled and decoupled parasitic reporting to work correctly.
- *Select From Schematic*	This option allows you to select the power and ground nets directly from the schematic rather than physically typing them in.

© 2023 All Rights Reserved.

Parasitics Set Up for Configuration Views

If you select *Parasitics – Setup* from a configuration view you are requested to enter database information.

Note: The version of the Setup form shown below is also displayed if you launch parasitic aware design functionality after descending into a sub-block of the configuration.

If you have opened a schematic via a simulation configuration you will not need to specify either view as this will information will be derived when you perform Out-Of-Context Probing.

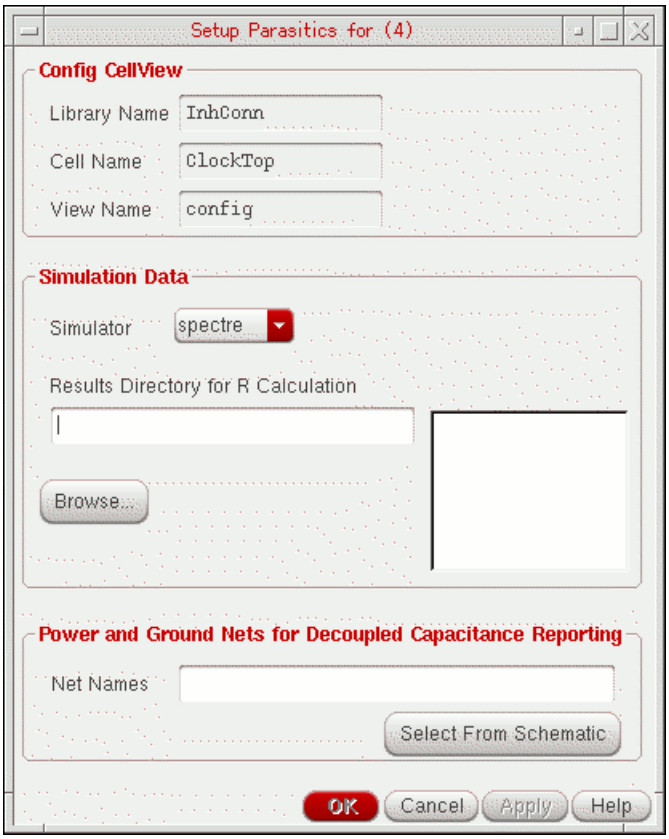


Figure 1-10 The Setup Parasitics Form (for configuration cellview in Schematics L/XL)

© 2023
All Rights Reserved.

GUI Item	Description
Config Cellview	
- Library Name	Enter or select the library that contains the configuration cellview information.
- Cell Name	A cyclic menu that shows the cells in the selected library.
- View Name	Enter or select the configuration cell view.
Simulation Data	
- Simulator	Choose the simulator that you want to use simulation data with. Currently, Ultrasim and Spectre are supported. **Note:** Where a schematic is associated with an ADE Explorer or ADE Assembler session, this option will mirror the settings in the ADE Explorer or ADE Assembler session and will consequently be disabled. For more information on ADE Explorer or ADE Assembler, see the *Virtuoso ADE Explorer User Guide* and *Virtuoso ADE Assembler User Guide*
- Results Directory for R Calculation	Enter or choose the directory containing the dc op point simulation results that are required to calculate effective resistance (see Effective R Calculations). **Note:** Where a schematic is associated with an ADE Explorer or ADE Assembler session, this option will mirror the settings in the ADE Explorer or ADE Assembler session and will consequently be disabled. For more information on ADE Explorer or ADE Assembler see the *Virtuoso ADE Explorer User Guide* and *Virtuoso ADE Assembler User Guide*.
- Browse	Use this option to move through hierarchies locating extracted cellview information.
Power and Ground Nets for Decoupled Capacitance Reporting	Use this field to set power and ground nets. Parasitic aware design uses these to determine which parasitic capacitances are decoupled (one side of the capacitor is connected to a power or ground net) or coupled (the capacitance is between two other nets). Parasitic reports allow you to select which types of capacitance to report. When you are selecting power and ground nets for the capacitance report from a configuration, these selections must be made out-of-context. That is, you need to make your selections while in the schematic view of a block that is bound to an extracted view of a configuration. **Note:** Power and ground nets **must** be specified for coupled/decoupled parasitics reporting to work correctly. If no power and ground nets are specified, all capacitors are considered to be coupled capacitors.

© 2023 All Rights Reserved.

- *Net Names*	Identify those power and ground nets that you want to use when differentiating between coupled and decoupled nets (see also Parasitics – Report Parasitics on page 56). **Note:** When entering a net name, you can optionally add a "/" at the beginning of each net name. If omitted, the "/" is automatically added when comparing nets (this will not however be reflected in the *Net Names* field if a "/" was not originally entered). This is required for coupled and decoupled parasitic reporting to work correctly.
- *Select From Schematic*	This option allows you to select the power and ground nets directly from the schematic rather than typing them in.

© 2023 All Rights Reserved.

Parasitics – Options

The parasitic aware design Options form contains a range of options related to *Backannotation* and *Parasitic Probing Reports*.

Note: The values entered here will determine the content of other parasitic aware design forms.

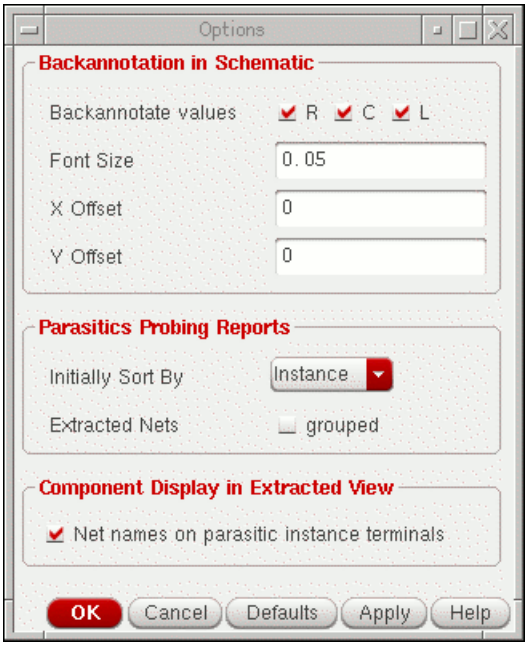


Figure 1-11 The Options Form

GUI Item	Description
Backannotation	(See also Parasitics – Show/Hide Parasitics on page 55)
- *Font Size*	Specify the label font size for displaying parasitic backannotation.
- *X Offset*	Sets the horizontal offset from the centre of the net when displaying parasitics.
- *Y Offset*	Sets the vertical offset from the centre of the net when displaying parasitics.
- *Backannotate values*	Specify the backannotation values that you want to view (*R*, *C* and/or *L*).
Parasitics Probing Reports	(See also Parasitics – Report Parasitics on page 56).
- *Initially Sort By*	Specify the initial sorting method (*Instance*, *Type*, *Value*, *From*, or *To*) to be used when probing parasitics on nets.

© 2023
All Rights Reserved.

GUI Item	Description
- Extracted Nets	Check the *grouped* option to report parasitics for the whole design net when probing an extracted net (this will work as if probing in the schematic). For detailed debugging however, you should uncheck the *grouped* option. In this case the report will be specific to the selected extracted net fragment. **Note:** If Parasitics is in a config set up, with simulation results available, probing in-context in the extracted view will also display effective R results in the corresponding report.
Component Display in Extracted View	(See also the Extracted Parasitics).
- Net names on parasitic instance terminals	Enables the display of the names of nets connected to terminals of parasitic instances in the extracted view.

© 2023 All Rights Reserved.

Parasitics – Parametric Variables

The Specify Parametric Values form can be used to set and edit parametric values specified on parasitic instances for use in parasitic simulation.

Quantus QRC extraction controls the specification of parametric variables, which are the result of using Multiple Rule Sets to run extraction on different process corners. For more information on this see *Use Multiple Rule Sets in the Quantus QRC Graphical User Interface* section of the *Quantus QRC Extraction Users Manual*.

Note: Variables, such as `Cmin` and `Cmax`, and their current values, can also be loaded directly from the Virtuoso Analog Design Environment. Report results, for example when reporting capacitances on a net, will differ dependent upon changes made to variable values in the Specify Parametric Values form (for example any value changes made to `Cmin` and/or `Cmax`).

An example of parametric C:

```
C = Cmin * iPar("c-min") + Cmax * iPar("c-max")
```

To set parametric variables:

1. Select *Parasitics – Parametric Variables*.

   The Specify Parametric Values form is displayed.

2. Select the *Library Name* and *Cell Name* whose variables you want to set (you can use the *Browse* button to invoke the Library Browser to assist this, or use the *Select By Cursor* button to select directly from the design canvas).

   Note: *Cell Name* must be the Analog Design Environment (ADE) cellview where the variables were originally set.

   For information on how parametric variables are set and edited in ADE, see Design Variables and Simulation Files for Direct Simulation in the Virtuoso ADE Explorer User Guide.

   You can also click the *Load Values* button to load parametric variable values that were set in the Edit Design Variables form in ADE. Here, you should select *Variables – Copy To Cellview* in ADE before choosing to load these values in.

3. Optionally, edit the values of variable to suit your requirements then choose *Apply* or *OK*.

   To save changes to parametric values for use in ADE select *Save Values*. Once these edits have been applied you can load the new values back into ADE by selecting *Variables – Copy From Cellview* in ADE.

4. Click *OK* to apply the new parametric variable values and close the Specify Parametric Variables form.

© 2023 All Rights Reserved.

Parasitics – Show/Hide Parasitics

You can choose to turn on or off the display of generated parasitic values in the current **schematic** view.

Important

The *Show/Hide Parasitics* menu option is only available in a relevant schematic view. If you are in a schematic view that has not been specified in the Parasitics Setup form, for example if you open up a schematic from the CIW that is completely unrelated to the Setup schematic, the *Show/Hide Parasitics* menu-item will be disabled.

If you show/hide parasitics on a particular schematic (which, as mentioned, must be defined in the schematic hierarchy in the *Schematic Cellview* section of the Parasitics – Set Up form), and descend/ascend the hierarchy, the state will be remembered so that in the new level parasitics will continue to be displayed as in the previous level.

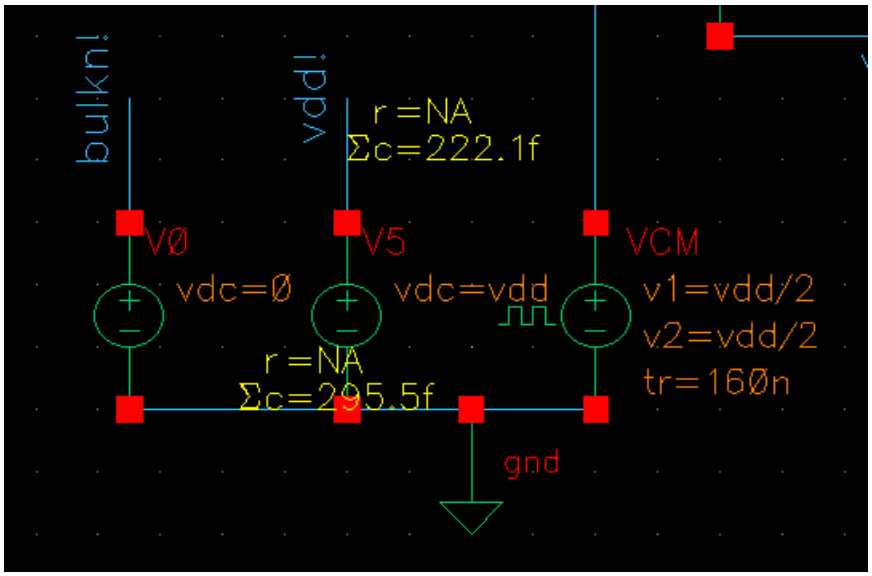


Figure 1-12 The display of parasitic values turned on

The use and operation of *Show/Hide Parasitics* is discussed further in Performing Backannotation on page 96.

© 2023
All Rights Reserved.

Parasitics – Report Parasitics

The following parasitic probing report options are available under the *Report Parasitics* menu option:

- Reporting on Nets (*Net* and *Net to Net*)
- Reporting on Terminal to Terminal
- Reporting on Net Capacitors
- Reporting on All Nets

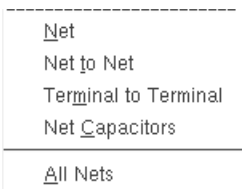


Figure 1-13 Available Parasitic menu reports

Note: The Report Parasitics Assistant is available for single-point simulations run from ADE Explorer and ADE Assembler.

Sum Totals

Any sum *Totals* at the bottom of a report displays the effective and summed values (C, L, and K will always be summed, while R depends on the type of probing) for each parasitic type that has been found for the particular type of probing.

For example, even if you filter out R or C by deselecting their respective options at the top of the Parasitics Report form, the summary totals will still remain the same. If available, effective totals will also be displayed (these will not be available if a DC op point simulation has not been run).

For information on running a DC simulation, see Preparing for Resistance Backannotation (Running a DC Analysis) on page 90.

© 2023 All Rights Reserved.

Reported Values

Parasitic aware design uses the `aelSuffixNotation` function to print the values of capacitance and resistance, with 4 as the default number of digits set to be displayed.

It is possible to view rounded values in a report form rather than the value that is displayed on the extracted view.

For example, you may see `r = 143.1453m` on the extracted view but in the parasitics report the 4 digit setting would display `r = 143.1m`.

You can however change the number of digits displayed. For example, to get full precision you should set:

```
aelPushSignifDigits (15)
```

Sorting the Report Table

Parasitic report tables can be sorted by clicking on the desired column option header.

The default sorting method is specified using the *Initially Sort By* option in the Parasitics – Options form.

Saving Report Data

You can save the contents of a report table, for example, a parasitic report or an instance probing report, to a specified report file by clicking on the *Save* button at the bottom of the applicable form.

The save action will record the information as per any sort and filter specifications you have already made.

You also have the option to save the information as either a *Text* file (`.txt`), or a *Comma separated value* file (`.csv`). If you want to change the name of the report you can edit the *Filename* as required.

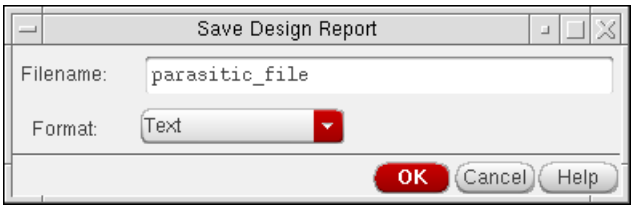


Figure 1-14 Saving report information

© 2023
All Rights Reserved.

Reporting on Nets

If you choose to report on a *Net* or on a *Net to Net* basis, this will invoke the following Parasitic report form (Figure 1-15 on page 58 or Figure 1-16 on page 59).

The title bar of the Parasitic report displays the net location in a hierarchical format, for example: *Parasitics for net /gnd!* or *Parasitics from net /gnd! to net /compout*.

Before the Parasitics report form is displayed you will be prompted to select the appropriate net/nets that you want to report on.

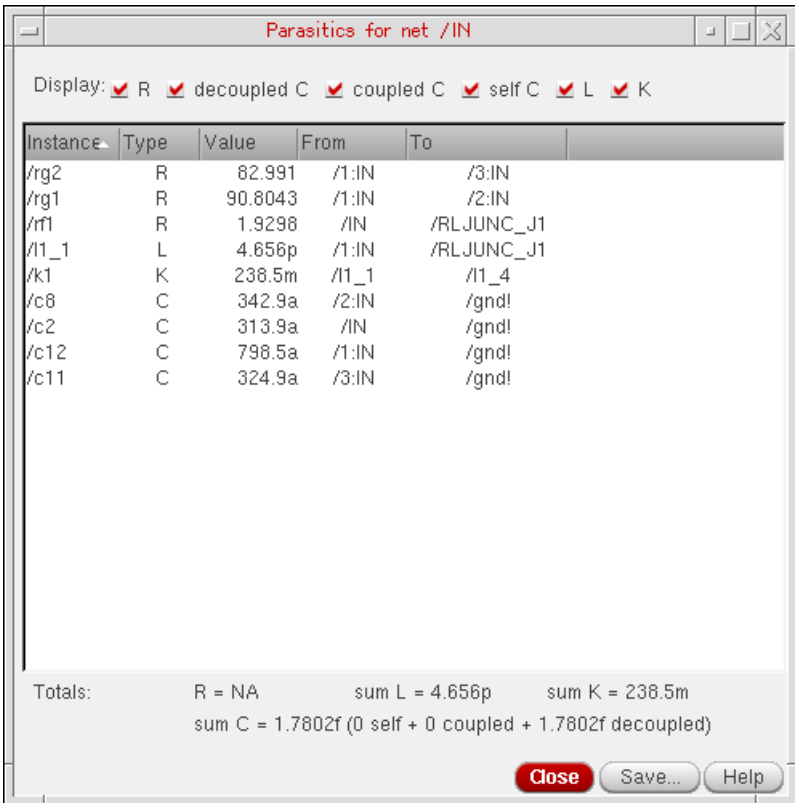


Figure 1-15 Parasitic report for net /IN

© 2023 All Rights Reserved.

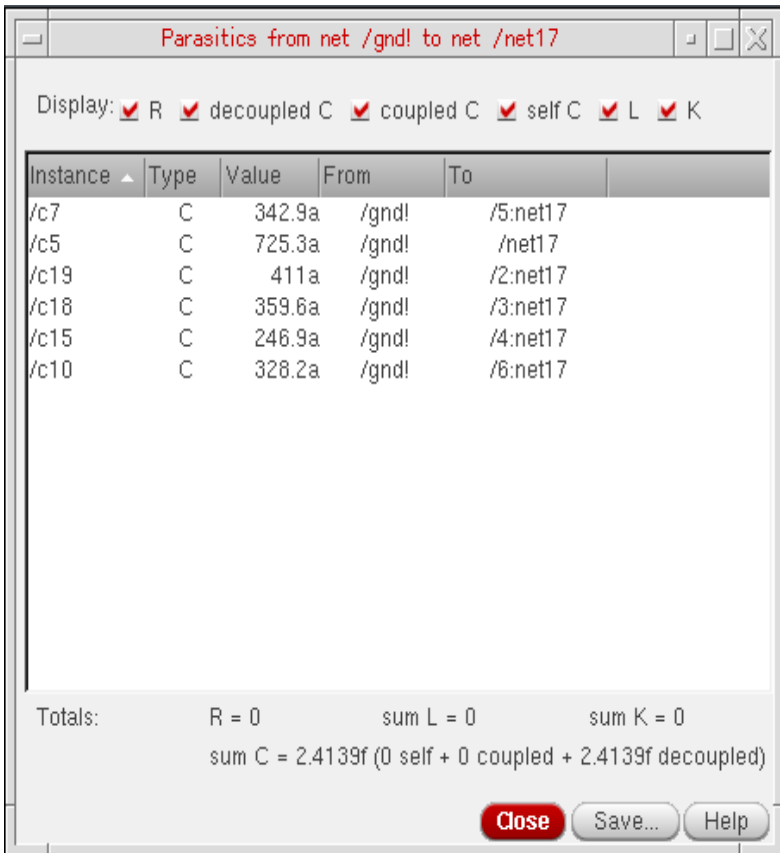


Figure 1-16 Parasitic report for parasitics from net /gnd! to net /net17

Important

You can click an individual *Instance* and be taken directly to that parasitic instance in the extracted view (only if this view is already open).

Note: You can also specify how to identify power and ground nets using the *Power and Ground Nets for Decoupled Capacitance Reporting* section in the Parasitics – Set Up form.

© 2023 All Rights Reserved.

The content of the Parasitics nets report forms can be specified in the *Display* section at the top of the report.

Display options	Description
R	Displays parasitic resistance elements.
decoupled C	Displays capacitors that are tied to power or ground nets.
coupled C	Displays capacitors between ordinary nets.
self C	Displays coupled capacitors between segments of the same net.
L	Displays parasitic inductances.
K	Displays the mutual inductance (*K*) ratio. For example, given two circuits, A and B, the mutual inductance is the ratio of the flux through circuit A caused by the current in circuit B.

The columns displayed in the Parasitics nets report forms are related to:

Column option	Description
Instance	Displays the name of the parasitic element.
Type	Displays the parasitic type (R, C, L or K).
Value	Depending on the parasitic *Type*, displays either a resistance, capacitance, inductance, or mutual-inductance value.
From	Displays the name of a net (in the case of R, L, and C), or an inductor (in the case of K).
To	Displays the name of a net (in the case of R, L, and C), or an inductor (in the case of K).

© 2023 All Rights Reserved.

Reporting on Terminal to Terminal

Using the terminal to terminal probing, you can report parasitics between two instance terminals or between a terminal and a pin on the same net.

The *Parasitic Report* assistant will report parasitics between the two most recent selections only. If multiple selections are done, all terminals or pins other than the last two selected will be ignored. Other object type selections are also ignored.

Note: The terminal to terminal probing does not report parasitics between two pins.

The format displayed in the title bar of the *Terminal to Terminal* Parasitics report is:

```
from (terminal|net) (<schTerm>(<extNets>+))+ to (terminal|net)
(<schTerm>(<extNets>+))+
```

Here, the term `net` is used instead of `terminal` when a selection is taken from an extracted view, as only extracted nets would be available.

Also, `(<schTerm>(<extNets>+))` will be replaced by `<bus_name>` if you select *All* when you have a list of buses.

Note: `schTerm` = schematic terminal and `extNets` = extraction nets.

Terminal to terminal report values can differ from whole net report values

When performing a terminal to terminal probe or generating report (also known as a *point-to-point* probe), where two selected terminals map completely to the same extracted nets, a warning describing the situation will be issued. The resultant report will also reveal that no parasitics have been found between these terminals. This happens because the terminals are considered to be shorted, and parasitics will therefore have no influence on the interaction between the two terminals in question.

However, if you subsequently perform a whole net probe on the same net, parasitics could still be reported, even if the net is a two terminal only net.

For example, if you consider two mFactored transistors (mFactor = 2) `T1` and `T2`, with the source of `T1` connected to the drain of `T2`. Selecting `T1-source` and `T2-drain`, in the schematic, will identify that `T1_m1-source` is shorted with `T2_m1-drain` and `T1_m2-source` shorted with `T2_m2-drain`. As mentioned, terminal to terminal probing would complete, reporting no parasitics, however whole net probing will identify and report any parasitics that might exist between `T1_m1-source` and `T2_m2-drain`.

© 2023 All Rights Reserved.

Selecting Leaf Terminals

The Select leaf terminals form is displayed when you perform terminal to terminal probing and select hierarchical terminals that have more than one leaf terminal.

From the Select leaf terminals form, you must select the terminals (for example, gate and drain) that you want to measure resistance values for.

Selecting individual terminals will prevent the results of net point probing (that has parallel resistors) being reported as a sum of the resistance values.

© 2023 All Rights Reserved.

Reporting on Net Capacitors

To create a net capacitors report select *Parasitics – Report Parasitics – Net Capacitors*, and then select the net you want to report on.

Reporting on net capacitors generates the Capacitors for net report which displays summed capacitances (*Sum C*) between the selected net and all nets that it is connected to via parasitic capacitors.

Note: No report will be generated for “0” capacitances, and no distinction is made between power/ground, and other nets.

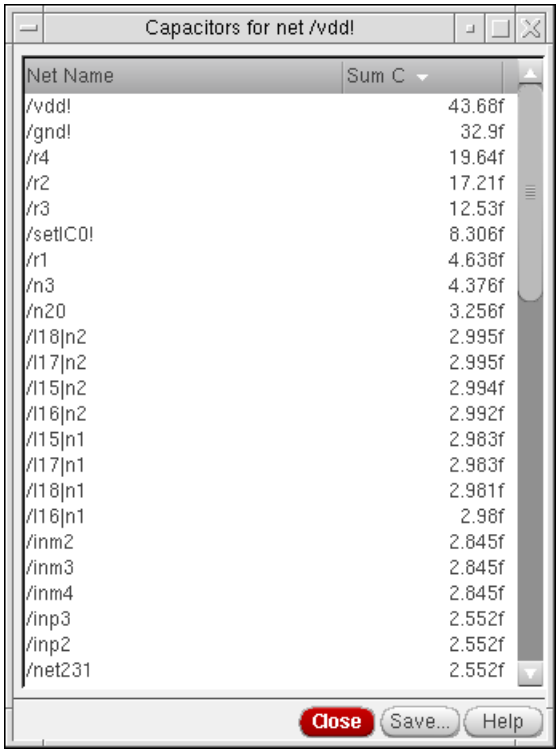


Figure 1-17 Capacitors report for net /vdd!

© 2023 All Rights Reserved.

Reporting on All Nets

Reporting on all nets generates the All Parasitics design report form.

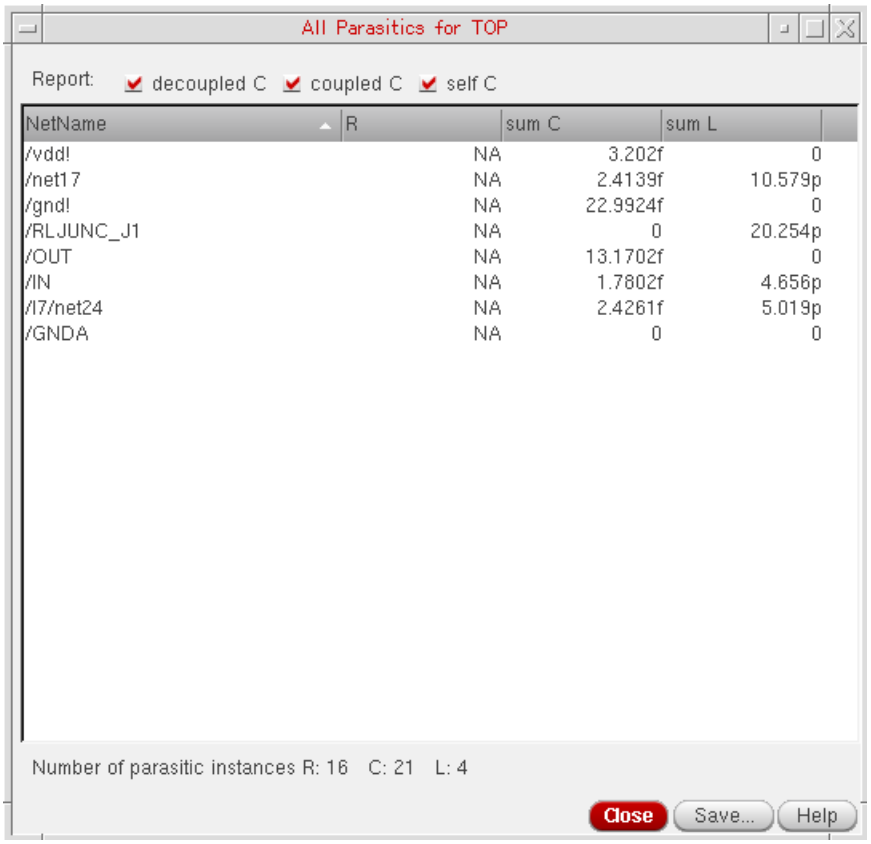


Figure 1-18 All parasitics report for TOP design

The All Parasitics report lists each net in the design along with the total value for the net based on either the effective value or the summed value.

The *decoupled C*, *coupled C*, and *coupled self C* check boxes control what capacitors are summed for the *sum C* column.

You can also save the All Parasitics report information to a file using the *Save* button with the option to save the information in either *text* (`.txt`) or *comma separated* (`.csv`) format.

© 2023 All Rights Reserved.

Parasitics – Refine Extracted View

Selecting the *Parasitics – Refine Extracted View* menu option will display the Refine Extracted View form with the name of the current extracted view in the title bar.

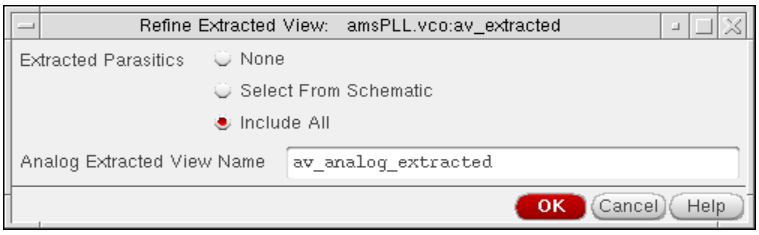


Figure 1-19 Refine Extracted View form with current view in title bar

The purpose of this option is to generate a refined extracted view that can be used for circuit simulation. The refined extracted view is based on the full extracted view created in Quantus QRC.

After the refine extracted view has been created, a window is displayed summarizing the parasitics in the new view. This information is also output to the CIW.

Note: The default Quantus QRC full extracted view is `av_extracted`, while the default parasitic aware design refined extracted view name is `av_analog_extracted`.

For more information on this process, see Step 3: Building a Refined (Analog) Extracted View Using Parasitic Aware Design on page 87.

Within this form, you can specify the extracted parasitics required for the extraction.

GUI Item	Description
None	Generate an extracted view with none of the parasitics.
Select From Schematic	Select the parasitics to be included in the simulation by placing special symbols (`spresistor`, `spcapacitor`, `spinductor`, and `spcapacitor2`) on nets in the schematic view. **Note:** The above symbols are taken from the standard `sbaLib` library.
Include All	Generate a (copy of the) extracted view with all the parasitics that have been extracted.
Analog Extracted View Name	Enter a name for the new extracted view, the default being `av_analog_extracted`.

© 2023 All Rights Reserved.

The refine extracted view will not remove any resistance that is associated with inductors as long as the inductors are not filtered out. This will prevent the creation of zero-resistance loops in the refined parasitic net.

© 2023 All Rights Reserved.

Parasitics – Probe Design Inst/Net

Note: You should ensure that the schematic and extracted views are both currently open to facilitate viewing and probing.

Select *Parasitics – Probe Design Inst/Net* to:

- probe from a design instance or net in the current schematic view to extracted instances in the extracted view. The probe performed will raise the Extracted insts associated with... form (see Figure 1-20 on page 68) from where you can pan and zoom to selected instances.

  or

- probe from a design instance or net in the extracted view to the schematic view. The probe performed will highlight, but not zoom in on, the schematic instance. If the schematic instance is within a hierarchical block in an open schematic then the block will be highlighted.

  Note: When probing from the extracted view highlighting in the schematic view will only work when the schematic is open in its own right. It will not work when the schematic is open through a config.

If attempting to probe from hierarchical blocks/instances a warning message will be displayed in the CIW. To resolve this, you must descend the hierarchy (*Edit – Hierarchy – Descend...*) and probe at the leaf level.

If you are probing a schematic net that exits the current hierarchy level the probe will be restricted to the instances attached to that net in the current level and not the full design. Probing therefore remains localized.

After you have selected a regular device (instance or net) to probe, the relevant *Extracted/Schematic insts associated...* form is displayed.

This tabular form lists all of the *Extracted/Schematic inst* names, and their associated *Type*, that are related to the device that you selected. The results are viewed in sortable columns.

© 2023 All Rights Reserved.

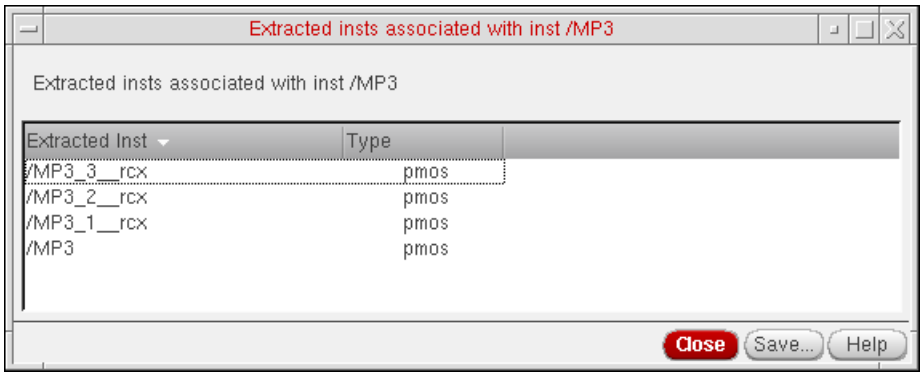


Figure 1-20 Extracted instances report form

For a design net, the displayed list contains all of the extracted or schematic instances that are attached to that net in the other view type.

If you select one or more extracted instances from the list (using the `Control/Shift` keys), this causes the extracted view to automatically zoom into those instances, highlighting the selected items in both the schematic and extracted views.

Note: There is no corresponding zoom facility on the schematic view.

If performing a multiple factor (m-factor) selection, it is generally the case that the instances will be found, and highlighted in the same area of the chip. An m-factor device is where a single cell instance represents multiple instances of a parallel connected cell.

Note: Highlighting of the instances only applies to an extracted view and not a layout view.

Clicking on the *Save* button in the *Extracted insts...* form displays the *Save Instance Report form*. For information on saving reports in parasitic aware design see Saving Report Data on page 57.

© 2023 All Rights Reserved.

Probing Out-Of-Context and In-Context

In-context and out-of-context probing has particular relevance when you are performing backannotation and reporting on parasitics. A parasitic aware design permits out-of-context probing of post-layout simulation results, containing additional parasitic components. This type of probing is done on a schematic where the simulation used an extracted view.

With an out-of-context probe, parasitic aware design is referring to the view that is currently being displayed against the view that is specified in the configuration used for simulation.

You can run out of-context or in-context probing on simulation results only when the current designs have been simulated. See Simulating the Design in ADE Explorer.

Out-Of-Context Probing

Out-of-context probing occurs when you descend into any cellview other than the simulation cellview. For extracted views, parasitic aware design provides the functionality to map information from the schematic view to the post-layout view. You can reuse expressions created with schematic net and terminal names when simulating the extracted view and those net and terminal names are automatically mapped to their equivalent post-layout names.

Note: Out-of context probing is supported in extracted view and Smart View.

Performing Out-Of-Context Probing in Extracted Views

To perform out-of-context probing in ADE Explorer or ADE Assembler, do one of the following:

1. Select the *Results* tab.
2. Right-click on an output value and select *Plot*.

   Or

1. In the *Results* tab, specify the plotting mode.
2. Choose the *Extracted View* or the *Smart View* next to *Plotting Template*.
3. Click the *Plot All* icon.

   The simulation results are plotted in the *Virtuoso Visualization and Analysis XL* window.

Alternatively, you can cross-probe using the Direct Plot functionality to plot the extracted net segments connected to instance terminals on different parts of the same net.

© 2023 All Rights Reserved.

To use Direct Plot, do the following:

1. In the *Results* tab, right-click the extracted view output, and choose *Direct Plot – Main Form.* The *Direct Plot Form* appears.
2. Specify the *Plotting Mode* and click *OK*.
3. In the schematic view, right-click and choose *Descend Read.* Alternatively, use the bindkey E to descend into the extracted view on the schematic. The *Descend* form appears.
4. Specify the schematic view to descend into and click *OK*.
5. Choose a net or terminal in the schematic view.

Selecting a point on a schematic net probes the voltage on the nearest instance terminal. The selected points are plotted as waveforms in the *Virtuoso Visualization and Analysis XL* window and the schematic view is displayed.

In-Context Probing

In the parasitic aware design environment, *in-context* probing implies probing the same cellview that is specified in the simulation configuration.

Note:

- A root cellview is always *in-context*.

Performing In-Context Probing in Extracted Views

To perform in-context probing, do the following:

1. In the ADE Assembler *Results* tab, right-click the simulation results for the `Smart View` and choose *Direct Plot – Main Form.* The *Direct Plot Form* appears.
2. In the schematic view, right-click and choose *Return.* Alternatively, use the bindkey Ctrl+E to return to the top-level testbench.
3. Select the instance that contains the Smart View. Ensure that the `Smart View` is selected to descend into and the mode is set to read-only. The Smart View is displayed.
4. In the Smart View, set the workspace to *Basic*.
5. Choose *Window – Assistants – Navigator*.
6. Choose a net in the *Navigator* Assistant. This plots the voltage on the net that you choose.

© 2023 All Rights Reserved.

7. Do the following:

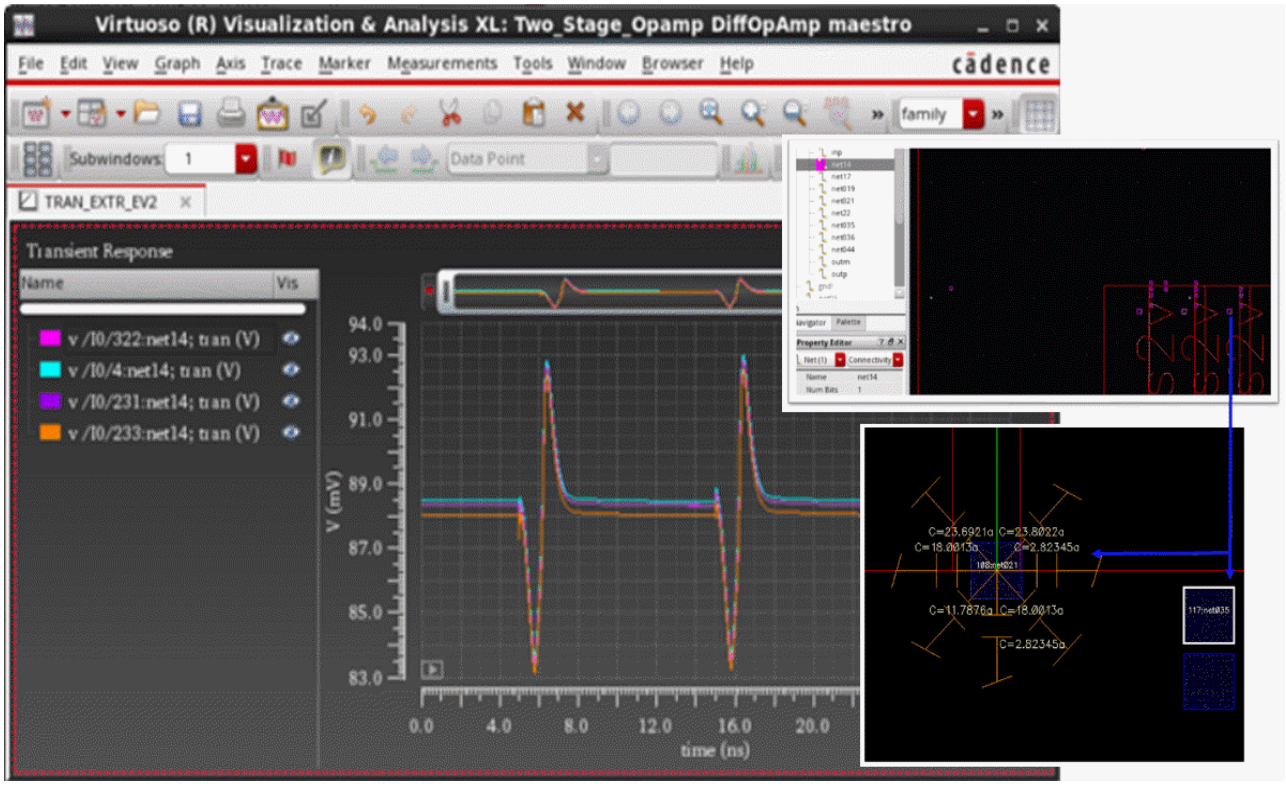
- If you are using the extracted view:
  - Press escape to exit the *Direct Plot* mode.
- If you are using the Smart View:
  - Choose *Smart-Parasitics – Display Parasitics.* The *Display Parasitics* form is displayed.

    The Smart-Parasitics Menu is displayed only in Virtuoso Layout Suite XL/GXL when you are working with a Smart View.
  - Specify the display settings and the threshold values for resistance and capacitance and click *Apply*. The parasitic display is updated with highlighted resistances and capacitances on the selected net.
  - In the Smart View window, click the *Zoom In* toolbar button to view the values of the parasitic resistances and capacitances.
  - Right-click a net in the *Navigator* and select *Probe – Add*.
    All nodes on the specified net are highlighted.
  - In the Smart View window, click the *Zoom In* toolbar button and select a square dot shape in the parasitic display. A small dot square represents a node on the net.

  Note: Nodes can only appear in the parasitic display if the layer and location information for the specific node is available. The layer and location attributes are essential to create a physical handle to the node to allow viewing and selecting the node. This information is available in the `display_map_file` file. Ensure that this file is used if you want to display parasitics in the Smart View.
  - Choose *Smart-Parasitics – Probe Node.* A concatenated string consisting of the node ID and the net name of the node is displayed on the selected node. The same node information is also displayed in the *Virtuoso Visualization*

© 2023 All Rights Reserved.

and Analysis XL window with the segments plotted, as illustrated in the following figure.



© 2023
All Rights Reserved.

An Analog Simulation Flow using PVS

This section discusses how parasitic aware design interacts with the Cadence Physical Verification and Extraction tools, PVS (Physical Verification System) and Quantus QRC (Resistance / Capacitance and Inductance Extraction), to create an extracted view.

Note: Extracted, configuration and schematic views can be used as input into parasitic aware design. The parasitic aware design flow also supports simulation cross-probing of schematic nets even when the extracted view is used in the simulation.

Parasitic aware design functionality, such as parasitic probing and backannotation, can then be used to take account of the effect of parasitics. This can consequently improve the accuracy of your circuit simulation.

Parasitic aware design provides you with the functionality to annotate parasitic values including resistance (R), capacitance (C) and inductance (L) values on the schematic.

The following sections (not all of which are compulsory) will take you through a typical parasitic aware design flow in PVS:

© 2023 All Rights Reserved.

- Reporting Parasitic Instances on page 101
- Probing Parasitic Instances on page 102
- Probing Buses for Parasitics on page 103

- Creating a Configuration on page 105
- Simulating the Design in ADE Explorer on page 107

Note: See also an Analog Design Flow Supported by Parasitic Aware Design on page 14.

© 2023 All Rights Reserved.

Preparing Cell Libraries

Before proceeding to create an extracted view using Quantus QRC, you need to provide the following views and component description format (CDF) information for analog primitives and parasitic cells.

Analog primitives must have:

- a symbol cellview

  Note: For information on how to create a symbol cellview, see the "*Creating The Symbol View*" section in the *SpectreHDL Reference* manual.

- a layout cellview or extraction rules that the extractor will recognize
- a model that matches the simulator you use
- an `auLvs` cellview that provides parameter values used by layout versus schematic (LVS)
- the `permuteRule` parameter added to the CDF simulation information for the auLvs view

  Often, device pins in the SmartView may be swapped by LVS or Quantus QRC. Parasitics require `permuteRule` in the CDF simulation information to properly bind such swapped pins. If your PDKs do not have the `permuteRule` specified in CDF and the CDF properties cannot be changed, it can result in the SmartView missing the pins or nets for parasitic devices.

  To apply binding rules in the absence of CDF properties, the tool detects whether the `layoutXL.bindPermuteRules` environment variable is specified, and if found, applies its settings.

Parasitic cells (e.g. presistors and pcapacitors) must have:

- a symbol cellview
- an `auLvs` cellview
- CDF simulation information for the `auLvs` view
- CDF component parameters for resistance (R), inductance (L), and capacitance (C)

If required, the `analogLib` library contains examples of analog primitives and parasitic cells that you can copy to create your cell library.

You can find the `analogLib` in the following hierarchy:

```
$CDS_INST_DIR/tools/dfII/etc/cdslib/artist/analogLib
```


© 2023 All Rights Reserved.

Preparing Library for Extraction

The following actions are required when preparing a library for parasitic extraction:

- Describe the technology layers.

  Note: For details about technology layers refer to the *Incremental Technology Databases and Display Resources User Guide*.

- Add or modify the verification rules used by the PVS processes.

  PVS processes include:

  – Designed device extraction

  – Layout versus schematic (LVS)

  – Parasitic resistor and capacitor extraction (Quantus QRC)

  Refer to the *Quantus QRC Extraction Users Manual* for information on creating verification and extraction rules.

  Note: You can use capacitance generation (`capgen`) to create the parasitic coefficients and files required for parasitic aware design.

- Prepare the PVS technology directory.

  This directory must contain all the designed device extraction rules, verification rules, and those files required for parasitic extraction.

  Again, refer to the *Quantus QRC Extraction Users Manual* for details about preparing the PVS technology directory.

© 2023 All Rights Reserved.

Planning Your Design Guidelines

If you intend to extract parasitic components from the layout view and run a simulation with parasitics, you should use the following guidelines to avoid problems as you plan your design:

- Devices, with the `componentName` parameter (in `auLvs` simInfo), should be set to either:
  - `pcapacitor`
  - `presistor`
  - `pinductor`
  - `pdiode`
  - `pmind`

  Note: This information is not directly required to run simulations with parasitics, but is necessary for parasitic probing between schematic and extracted views.

  If you do use these names to define your parasitics, be sure to set the appropriate values to ensure that they are properly reported.
- Do not use the LVS `permuteDevice` parameter to match groups of components in a series as that makes it impossible to determine which device to use for waveform probing.
- You can use the Quantus QRC `?LvsMatchRequired` parameter to ensure that the extracted view can only be created if the layout and schematic match up.
- Set the `permuteRule` parameter for devices that have permutable terminals. Parasitic Aware Design uses this rule to detect terminals that have been permuted in the extracted view.

  The syntax to set the `permuteRule` is `"(p term1 term2)"`. For example, for a MOS device with a permutable drain and source, you can set the `permuteRule` parameter as `"(p D S)"`. Similarly, for a resistor, you can set this parameter as `"(p PLUS MINUS)"`.

  Note: If the `permuteRule` parameter is not set correctly, out-of-context probing and parasitic reporting may fail.

© 2023 All Rights Reserved.

Creating an Analog Extracted View

Important

The extracted view provides the crucial link between physical verification and mixed-signal simulation. When you create an extracted view, the layout and schematic views need to match in order to correctly associate the parasitics with the entire schematic network.

Before attempting to generate an extracted view you should ensure that the following requirements are met:

- You have successfully prepared your cell library (see Preparing Cell Libraries on page 75).
- The design adheres to the suggested design guidelines (Planning Your Design Guidelines on page 77).

With these tasks successfully completed, you will now utilize PVS to compare the layout view to the schematic view.

PVS creates an internally formatted database from the layout view (this database contains only the designed devices). Quantus QRC then reads the database to extract the parasitics and to create the extracted view. This extracted view contains designed devices and parasitic devices.

Note: For more information see *Quantus QRC Extraction Users Manual.*

The steps required for the successful creation of a refined analog extracted view are as follows:

- Step 1: Comparing the Schematic and Layout Views Using PVS on page 79
- Step 2: Building an Extracted View using Cadence Quantus QRC Extraction on page 83
- Step 3: Building a Refined (Analog) Extracted View Using Parasitic Aware Design on page 87

© 2023 All Rights Reserved.

Step 1: Comparing the Schematic and Layout Views Using PVS

1. Start the Cadence software by entering an appropriate command at the prompt, for example:

   ```
   virtuoso &
   ```

2. Choose *File – Open* in the CIW and open up a cell layout view to invoke the Virtuoso® Layout Suite Editing window (for example, Layout XL).

3. In the Layout Suite Editing window, choose *Launch – Plugins – PVS*.

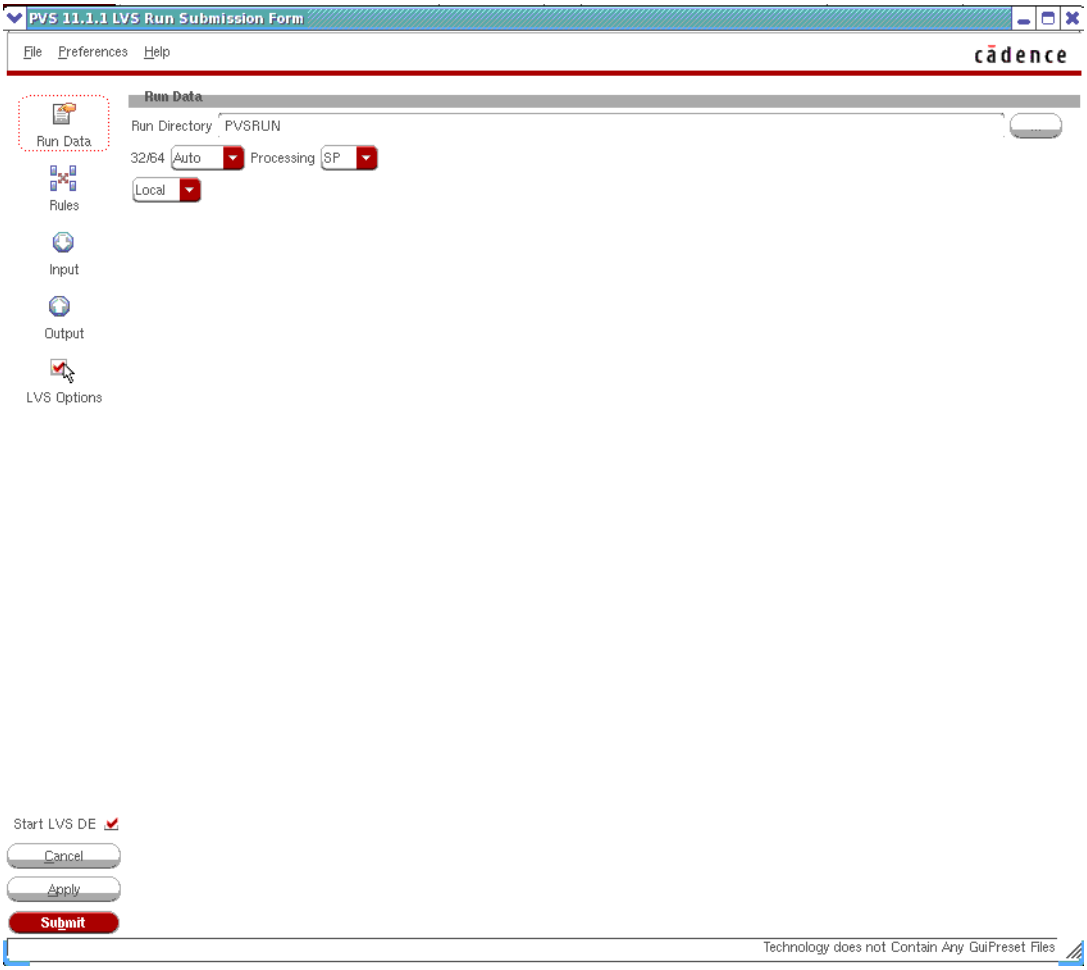
   The *PVS* menu is added to the menu bar of this window.

   Note: PVS will need to be installed to be available as a plugin in the Layout window.

4. Choose *PVS – Run LVS*.

   The **PVS LVS Run Submission** form appears.

© 2023 All Rights Reserved.



You can use `avCompareRule formGate(none)` to ensure that the connectivity between the layout and schematic views are fixed. For more information on this, refer to the *Physical Verification System User Guide*.

5. If you want to load previously saved LVS run settings into the form, choose *File – Load Presets* and select a preset file name from the *Load Presets File* form.
6. To create a new LVS run, perform the following steps:
   a. Click *Run Data* to open the Run Data tab.
   b. On the Run Data tab, specify the path to the run directory in the *Run Directory* field.
   c. Click *Rules* to open the Rules tab.
   d. Specify the path of the technology mapping file in the *Technology Mapping File* field.

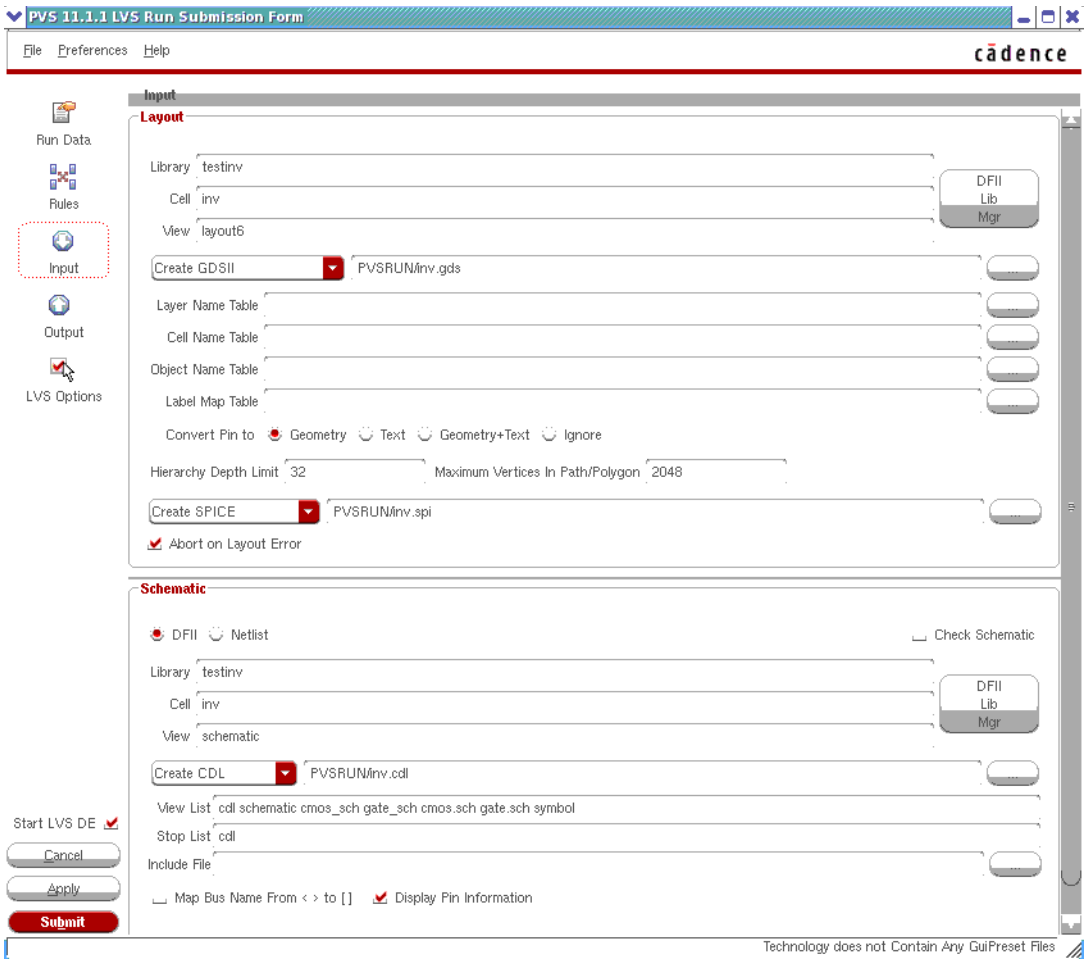
© 2023
All Rights Reserved.

e. From the *Technology* drop-down list, select a technology definition file for the process to be used.

f. From the *Rule Set* drop-down list, select the name of the rule set to be used.

g. Click *Input* to open the *Input* tab.

h. Specify the library name, cell name, and view name of the layout and schematic views in the *Layout* and *Schematic* sections, as shown below.



Note: To search for the available cellviews, click *DFII Lib Mgr* to open the PVS: Select Cell View form.

i. Click *Output* to open the *Output* tab.

j. Ensure that the *Create QRC Input Data* option in the *Additional Output* section is selected.

© 2023
All Rights Reserved.

This option generates the data required by Quantus QRC to perform parasitic extraction.

k. Specify a name of directory in the *QRCDataDir* field.

A subdirectory is created by this name and the data required by Quantus QRC is saved in it.

l. Click *LVS Options* to open the LVS Options tab and select appropriate options.

7. Click *Submit* to start a new LVS run.

The **PVS Reports** form is displayed. This form provides run status information. After the run is complete, the PVS LVS Run Status message box is displayed indicating the success status of the extraction results and the comparison results.

You can click *Errors List* on the **PVS Reports** form to display errors as the run progresses. For more details, see *Checking LVS Run Report* in the *Cadence Physical Verification User Guide*.

If there are errors, the tool prompts you to open LVS DE.

8. Click Yes to open the **LVS Debug Environment** and view the results and files of the LVS run.

For more details on how to debug an LVS run, refer to *Debugging LVS Comparison Runs* in *Cadence Physical Verification User Guide*.

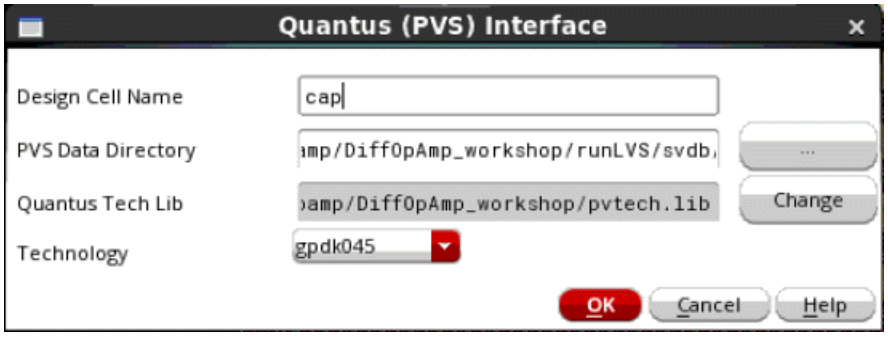
© 2023 All Rights Reserved.

Step 2: Building an Extracted View using Cadence Quantus QRC Extraction

Once the comparison between the schematic and layout views has completed, you now need to run Quantus QRC (Resistance/Capacitance and Inductance Extraction) to extract the parasitic devices and build the `av_extracted view`.

1. Select *Quantus QRC – Run PVS-QRC* from a layout session window.

   The **QRC (PVS) Interface** form appears.

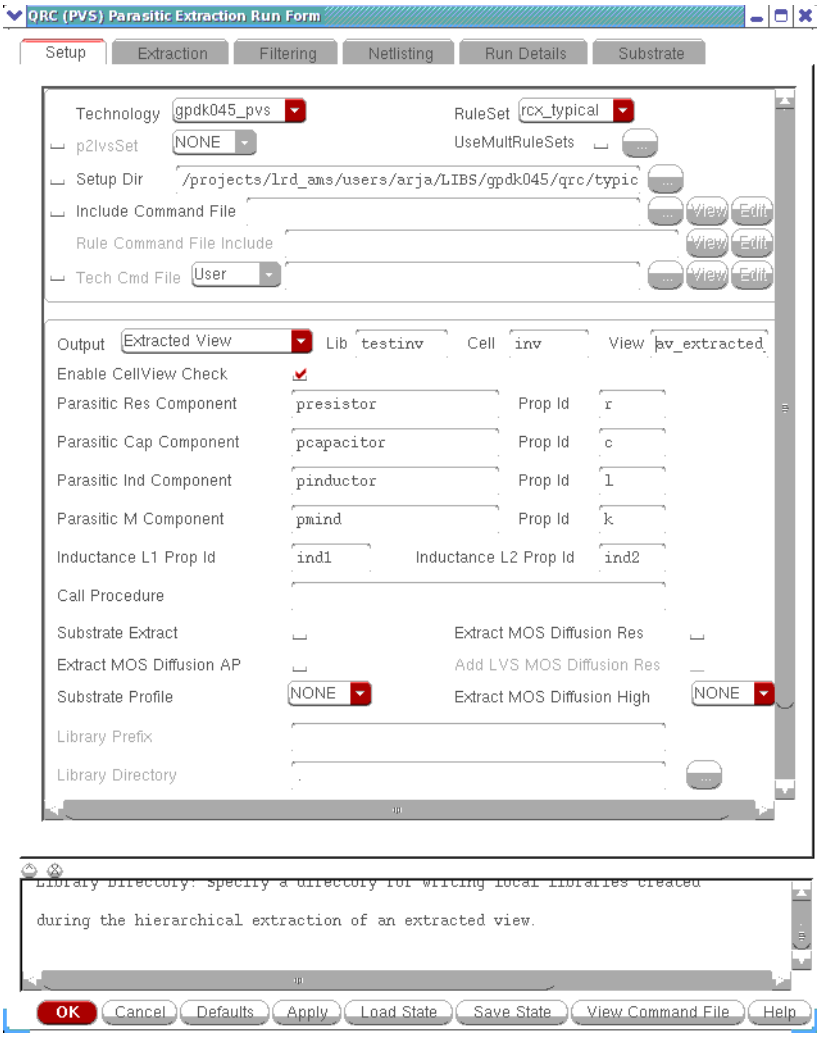


2. In the *Design Cell Name* field, specify a name for the design cell.
3. In the *PVS/QRC Data Directory* field, specify the path to the directory where the results of PVS were saved for Quantus QRC.

   This is the same subdirectory name that you specified in the *Output* tab of the LVS Run Submission form. Refer to **step k** in the previous section.
4. Ensure that the required QRC technology library is selected in the *QRC Tech Lib* field and the technology name is selected in the *Technology* field.
5. Click *OK* to select the run.

   The **QRC (PVS) Parasitic Extraction Run** form is displayed, as shown below.

© 2023
All Rights Reserved.

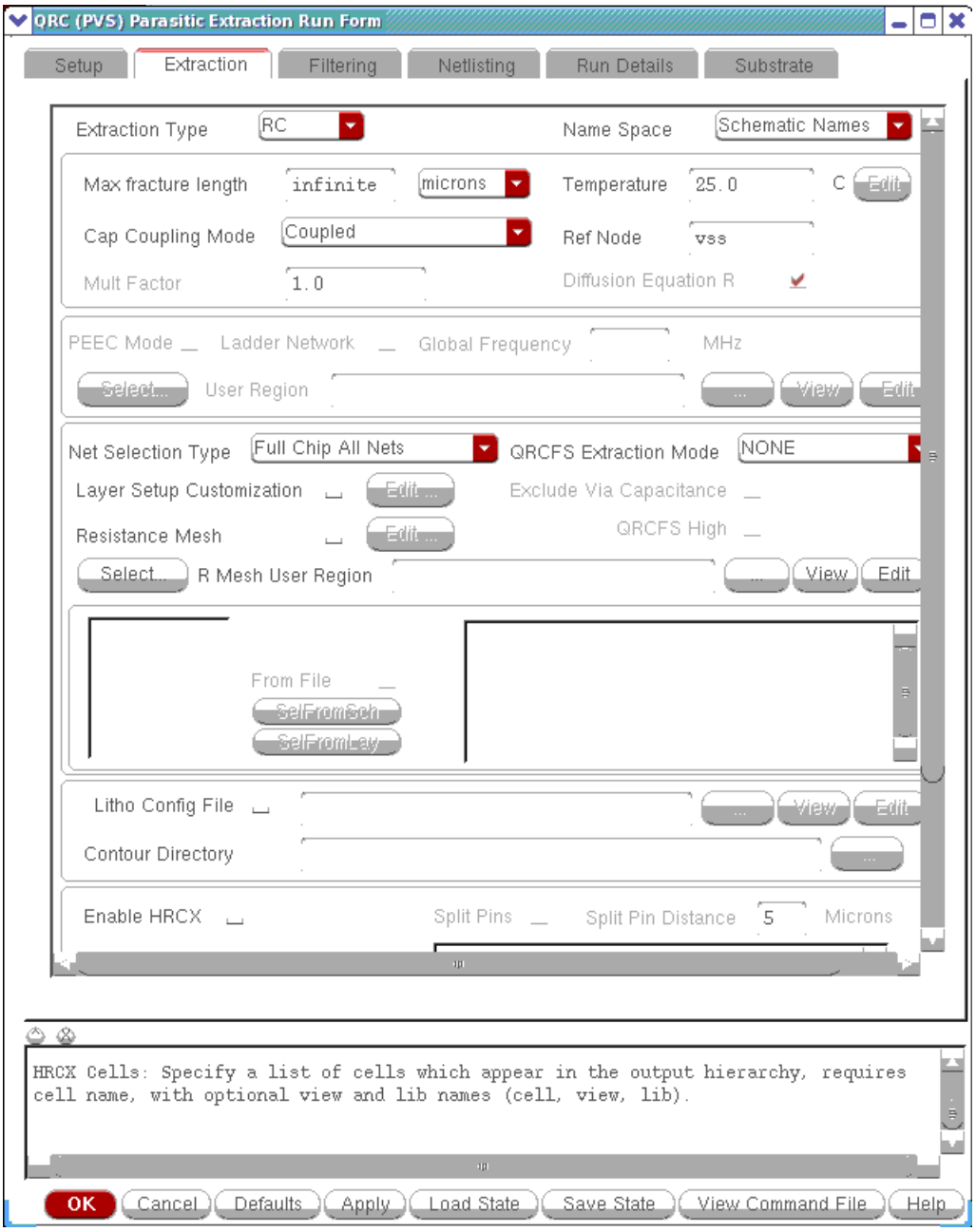


6. On the *Setup* tab, ensure that either *Extracted View* or *Smart View* is selected from the *Output* drop-down box.
7. In the *View* field of the *Setup* tab, type the name of the view.

   The default name for *Extracted View* is `av_extracted`, and in case of *Smart View* it is `smart_view`, but you can create the extracted view using another name.

   On the *Extraction* tab, select an extraction type by using the *Extraction Type* drop-down list and specify options related to that extraction type.

© 2023
All Rights Reserved.



8. Enter an appropriate entry (for example `gnd!`) in the *Ref Node* field of the *Extraction* tab.

   Ref Node is a capacitance extraction option for grounding caps. Capacitance is decoupled to the specified *Ref Node*.

9. Click *OK* to start the Quantus QRC run.

© 2023
All Rights Reserved.

A progress form, which shows the progress of the Quantus QRC run, is displayed. When the Quantus QRC run is complete, a message box is also displayed showing the success status.

With the extracted view created, you can now use parasitic aware design to build an analog extracted view which can be used to filter parasitic devices.

For more information on Quantus QRC, see the *Quantus QRC Extraction Users Manual*.

Note: It is not recommended to edit the extracted view. To modify parasitic values and check results without running simulations, use the Refine Extracted View to filter multiple extracted parasitics by values or use the parasitic filters to filter parasitics based on values. You can proceed to run simulations with this refined extracted view.

© 2023 All Rights Reserved.

Step 3: Building a Refined (Analog) Extracted View Using Parasitic Aware Design

The final step in this process is to build an analog extracted view using parasitic aware design functionality that is derived from the extracted view created using Quantus QRC.

Note: You cannot set up parasitic aware design on a non-Quantus QRC view, for example a layout. If you attempt to do this, a warning message is displayed advising you to open either a schematic or a Quantus QRC extracted view, before continuing to select *Launch – Parasitics* (in Schematics L/XL) to add the *Parasitics* menu.

You do not need to perform this step to *simulate* the Quantus QRC extracted view, or to perform parasitic probing or back annotation of parasitic values (other than resistance). Building an `av_analog_extracted` view is only necessary if you want to filter parasitic devices out of the extracted view for a quicker simulation.

1. Select *Launch – Parasitics* (in Schematics L/XL) from an extracted or schematic view.

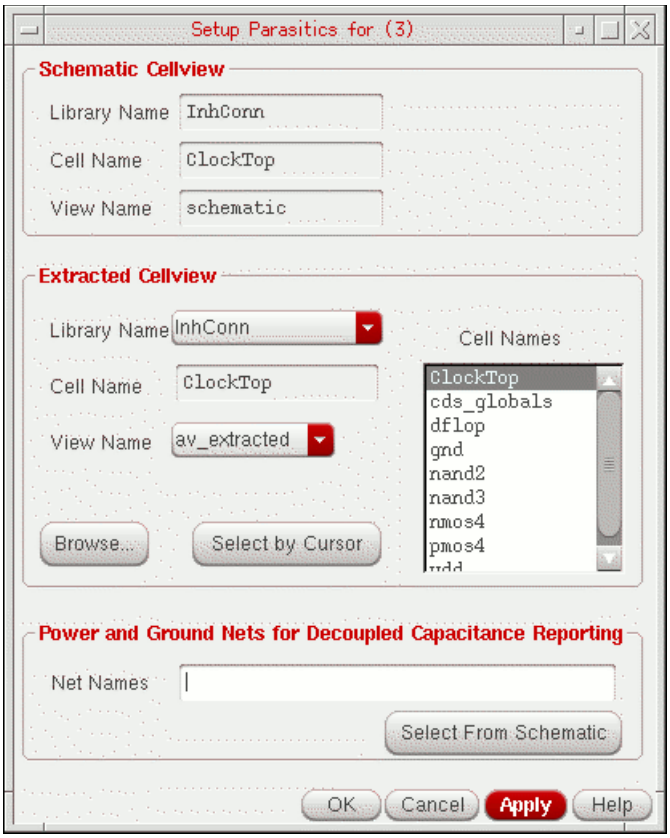
   This will add a *Parasitics* menu to the schematic editor menu bar.

   Note: The *Parasitics* menu is displayed by default when you are working with ADE Explorer or ADE Assembler.

2. Select *Parasitics – Setup* from the *Parasitics* menu.

© 2023 All Rights Reserved.

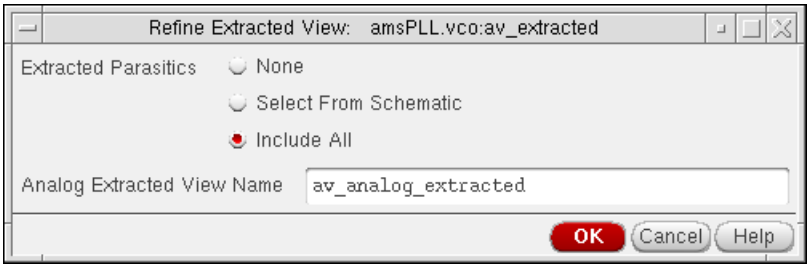
This will display the Setup Parasitics form where you can specify which views to use. If opened from a schematic you will only need to specify which extracted view to use (and vice versa).



3. In the *Extracted Cellview* section, ensure that you have selected the `av_extracted` view that was created in Step 2: Building an Extracted View using Cadence Quantus QRC Extraction on page 83.
4. Click the *OK* button to setup parasitics.
5. Select *Parasitics – Refine Extracted View* from the Parasitics menu.

© 2023
All Rights Reserved.

This will display the Refine Extracted View form.



6. Enter the new extracted view name in the *View Name* text field. The default refined analog extracted view name is `av_analog_extracted`.

   Note: If this field is not completed, and you attempt to run *Refine Extracted View*, the following message will be displayed in the CIW log window:

   ```
   WARNING: No analog view name specified. Specify a name in the 'Refine Extracted
   View' form under the Analog Extracted View Name field.
   ```

7. Click *OK*.

   The Refine Extracted View form closes and `av_analog_extracted` (or your renamed view) is created and available for parasitic aware design.

   After the refine extracted view has been created, a window is displayed summarizing the parasitics in the new view. This information is also outputted to the CIW.

© 2023 All Rights Reserved.

Backannotating Parasitic Values

Note: If you are using Virtuoso® UltraSim, you can use your `mmsim` license to run Spectre for backannotation purposes.

Preparing for Resistance Backannotation (Running a DC Analysis)

Backannotation of resistance values is dependent upon the results of a DC analysis of the extracted view. Whole net probing and the design report (see Parasitics – Report Parasitics on page 56) also make use of the DC analysis results. In all cases, the requirements on the simulation data are the same.

This section therefore describes the actions and requirements you must perform and meet to provide suitable simulation data.

Note: The simulators currently supported are Ultrasim and Spectre.

➤ **Action 1**: *Provide and configure a suitable testbench that uses Spectre to simulate the chosen extracted view*

Requirements:

- The testbench must ensure that all nets in the design are stimulated.
- It does not matter where in the hierarchy the extracted view appears, as parasitic aware design uses the hierarchical instance path to locate the simulation results for the extracted view that corresponds to the out-of-context schematic.
- When backannotation is invoked, the configuration (see Creating a Configuration) bindings must match those that were in place during simulation. This is required, as problems could arise if you change and save the config after a simulation is run.

© 2023 All Rights Reserved.

➤ **Action 2:** *Run a simulation of the testbench that includes a suitably configured DC analysis*

Requirements:

- The analysis must be configured to save DC operating Point. This is part of the DC Analysis in Spectre and the Transient Analysis in Ultrasim.

  Note: For information on running a DC operating point analysis, see Setting Up for an Analysis in the *Virtuoso ADE Explorer User Guide*.

- The analysis must not be swept.

- The DC analysis must use the `gmin` simulator option to ensure that the current flows through nonlinear devices and nets with no DC path to ground.

  The `gmin` value should be set low enough so as not to impact the DC solution. Setting the `gmin` value as low as possible will also provide for more accurate effective R results. Spectre or Ultrasim - the simulators that support Effective R Calculations - will issue a warning if the `gmin` value is too high. However, the lower the `gmin` value, the increase in the likelihood that the simulation will not converge. Additionally, you may also want to consider attempting to create a R-only extracted view which could reduce the influence of capacitors on the net.

  Using the `gmin` convergence option is the default.

  Note: If the `gmin` influence is significant, in terms of the net current, then the resistors placed around the capacitors, for a `dcOp` calculation, will create a significant contribution to results, and will be reflected as distorted resistance values. For example, if you receive the `gmin` warning, R values may be reported in the 100 megohm range, when in fact they are in the ~10 ohm range.

Important

If you are caught between the choice of continuing with no convergence or proceeding with a value that generates a `gmin` warning, you should proceed with the warning-generating value, but only consider returned backannotation values as “approximations”, suitable only for quick feedback.

Note: You can set the *gmin* option in the Simulator Options form, which is accessible via an ADE Explorer window by selecting *Simulation – Options – Analog* in the *Resistance Options* section (in ADE Explorer or ADE Assembler if you right-click on a Spectre or UltraSim test, an *Options – Analog* selection will be displayed). For more information on adjusting Spectre’s tolerances, see the *Spectre Circuit Simulator User Guide - Identifying Problems and Troubleshooting* chapter.

© 2023 All Rights Reserved.

Running the testbench simulation will produce two datasets of interest:

1. `dcOp`

   A dataset that contains all saved node voltages.

2. `dcOpInfo`

   A dataset which contains DC operating-point data for each instance in the design. This will include terminal currents, voltage drop between terminals, and device power consumption.

Effective R Calculations

An effective R value is a single number distillation of complex physical parasitic networks, including power dissipation and current. The use of a single number makes annotation to each logical net of the schematic possible. You can use this single effective R value, per net, to monitor changes to the layout that are made at each iteration request. This will check and confirm that you are correctly reducing the parasitics on key nets.

This value also normalizes the parasitic network on each net, even though the parasitic network itself may be considerably different between iterations. This could be due to changes in the layout of the net, or the layout around it, that influences its parasitics, for example the number of parasitic elements and connectivity. Alternatively, parasitic networks could also change as a result of Assura parameters, such as fracture length.

The DC simulation results are used to calculate the backannotated resistance. These results can have different values for the same design, based on the value of `gmin` compared to the design component values (see also Preparing for Resistance Backannotation (Running a DC Analysis).

Effective resistance (effective R) backannotated values are only intended for use as guideline estimates. They are not intended as accurate reference values for use in simulations. The values generated are not “real” resistance values but can be useful for comparisons with effective resistance of other nets, or between extractions of different versions of the layout (for example one version of a layout could have significantly reduced the parasitics on a particular net). It is impossible to accurately reduce a complex net to a single resistance value. Effective resistances are calculated using a “delta power” method.

Calculation of effective R is based on the current at the source (iS) and drain (iD). If iS == 0 and iD == 0 then effective R is not calculated as there is no current flow and no power. If either of iS or iD is a non-zero value, effective R is calculated as:

© 2023 All Rights Reserved.

$$R = \frac{\Sigma P}{I^2}$$

Where,

I is current at source if $S \neq 0$, or

I is current at drain if $S = 0$

There may be a few thresholds involved while working out the current. For example, there can be an initial threshold from the DC-OP attribute “abstol(I)”. If this does not exist, then the environment variable, mspsAv currentThreshold, is used. If the environment variable is missing, $1 \cdot e^{-12}$ is used.

Note: The use of effective R calculations with MOS devices can be difficult. This is due to MOS devices normally being set to either *on* or *off*; in these states there is a reliance on leakage currents to perform effective R computation. This can result in less meaningful results if the current is minuscule.

Effective R value is a measure of the power lost through the parasitic resistor network (or equivalently, the delta between the power entering the network and the power leaving the network), normalized by the total current entering (or leaving) the network.

Combining `V=IR` with `P=VI`, it is calculated as:

$$EffectiveR = \frac{P}{I^2}$$

where:

`P` is the sum of the power through the parasitic resistors in the network.

`I` is the current entering (or leaving) the network.

There could be situations where the capacitor was annotated correctly, but the effective resistance is not annotated on each node. In such cases, the Spectre log reports that a large number of parasitic resistors, `pRes`, have been reduced when the *Enable Post Layout*

© 2023 All Rights Reserved.

Optimization option in the High Performance Simulation Options form is used. This is why currents for parasitic resistors, `pRes`, are not saved by Spectre and parasitic aware design reports effective R as `0` for most of the nets.

To see the effective R when effective R annotations are not available, deselect the *Enable Post Layout Optimization* option in the High Performance Simulation options form.

Backannotation Labelling

Assuming that you have suitable simulation results at hand (see Preparing for Resistance Backannotation (Running a DC Analysis) on page 90), backannotation can be performed.

Parasitic aware design provides you with the functionality to annotate parasitic resistance (R), capacitance (C) and inductance (L) values on the schematic when Parasitics – Show/Hide Parasitics is used.

- Analog Designers use many tools and techniques, including dynamic-analysis tools such as simulators, and static-analysis like parasitic aware design, to understand the impact that the post-layout parasitics have had on their previously “ideal” design. In order to identify spots where values of parasitic resistance are a concern, designers want to display and analyze the resistance values of these networks of resistors using parasitic aware design and ADE Explorer or ADE Assembler/Schematics L/XL. Physical hot spots are identified as unacceptably large values annotated onto logical schematic nets by parasitic aware design. For resistances, these values represent the ‘effective resistance’ of each net (see Effective R Calculations on page 92), regardless of the complexity of the parasitic networks that each represents, or the number of net terminals.

  - **~ label**

    If a net has **two or more terminals**, the R value will use an approximation symbol (~) to indicate that the value uses the delta power method to approximate a resistance for the net.

  - **- - label**

    The “**- -**” label indicates that you have backannotated without any simulation results being set up.

  - **NA label**

    If the necessary **simulation results are not available to allow delta power calculations** for specific nets, for example, where a subset of voltages or currents have not been saved, the annotated R value will display the string “**NA**” for “Not Available”.

© 2023 All Rights Reserved.

If a string with NA is displayed, a (one off) message will be displayed in the CIW as follows:

```
"Parasitic R values are not available due to missing/incomplete simulation
results. Please run a DC analysis and supply the path to the results
directory via the Parasitics Setup form."
```

- **NS label**

  If a **net is not stimulated during simulation**, the current through it will be 0, which makes it impossible to calculate an effective R value. in such cases, the annotated R value will be the string "**NS**" for "Not Stimulated".

  If a string with NS is displayed, a (one off) message will be displayed in the CIW as follows:

```
"Effective R values are not available on nets with no stimulus (nets with
'r=NS'). Review the simulation test bench to ensure that there is current
through the net and resimulate."
```

  Note: "Not stimulated" basically means having no current flowing through the net. This can occur in DC simulation if all the parasitic resistances are removed from the net. For example, if the extraction did not include R extraction, or if the extracted view, that is being simulated, has been refined so that parasitic Rs have been removed from the net.

- The value displayed for C is an arithmetic sum of parasitic capacitance.

  - **Σ label**

    The C value will use a sigma (Σ) sign to indicate that it is a sum of the parasitic capacitances associated with the net.

- The value displayed for L is an arithmetic sum of parasitic inductance.

  - **Σ label**

    The L value will also use a sigma (Σ) sign to indicate that it is a sum of the parasitic inductance associated with the net.

Persistence of Backannotation Labels

If you open a schematic cellview in edit mode, and perform backannotation, the backannotated labels will be persistent, and be written to the schematic database. However, if you open the schematic in read-only mode, the backannotated labels will only be temporary.

© 2023 All Rights Reserved.

Performing Backannotation

Backannotation functionality is available through:

- The *Parasitic* menu in Schematics L/XL
- The *Results – Annotate* menu option in ADE Explorer

Backannotating From the Parasitics Menu

For backannotation purposes, you can access parasitic aware design functionality, from a schematic window, by selecting *Launch – Parasitics* (in Schematics L/XL) to display a *Parasitics* menu (for more information see Parasitics Menu on page 42). There is a *Show/Hide Parasitics* toggle menu-item to inform you if annotation is currently active or not.

Note: If you want to specify backannotation settings (*Font Size*, *X Offset*, *Y Offset*, and *Backannotate Value*), see Parasitics – Options on page 52.

If you are attempting to backannotate via the *Parasitics* menu, you should first of all consider the following scenarios:

Scenario 1: The Current Schematic Window is Associated With a Current ADE Session

Here, the *Show/Hide Parasitics* toggle menu-item is identical to that in ADE Explorer (see Backannotating From ADE Explorer on page 98) and operates in the same manner. That is, no parasitic aware design setup is required to enable the *Show Parasitics* menu because the required information is available from the ADE session.

Note: The ADE Explorer *Show/Hide Parasitics* toggle menu-item is only available when out-of-context, so you have to descend from the top schematic.

Scenario 2: The Schematic Window is NOT Associated With a Current ADE Session

In this scenario, you are required to perform the parasitics setup step (see Parasitics – Set Up on page 44) so that parasitic aware design can locate the simulation results directory and extracted views.

Only after setup is completed will you be able to annotate parasitic values.

© 2023 All Rights Reserved.

Verifying a Configuration Setup

When a configuration setup (Parasitics – Set Up on page 44) is used, you can specify the design that was simulated, the simulator, and the simulation results directory. When you click the *OK* or the *Apply* buttons in the Setup form, parasitic aware design will check that the results directory is valid and will also verify that it contains the `dcOp` and `dcOpInfo` datasets. If the directory specified does not meet these requirements, an error will be displayed and the *Show Parasitics* menu-item will remain disabled.

Where a schematic is associated with an ADE session (*Scenario 1* above), the Setup parasitics form will still be available. The simulation cellview, *Simulator*, and *Results Directory for R Calculation* fields will mirror the settings in the Analog L session and will consequently be disabled. This therefore means that you cannot change any parasitic aware design settings so that they are out of sync with ADE. It also indicates that other parasitic aware design functionality must use the simulated design from ADE, when available, to determine what extracted view to use, and that other parasitic aware design functionality must be able to operate under a configuration.

© 2023
All Rights Reserved.

Backannotating From ADE Explorer

Setting up parasitics (see Parasitics – Set Up on page 44) is normally required before parasitic aware design functions are enabled. However, with all the information, that parasitic aware design requires to determine the extracted view being available in the ADE session, along with the simulation results directory, the setup step is not required.

Important

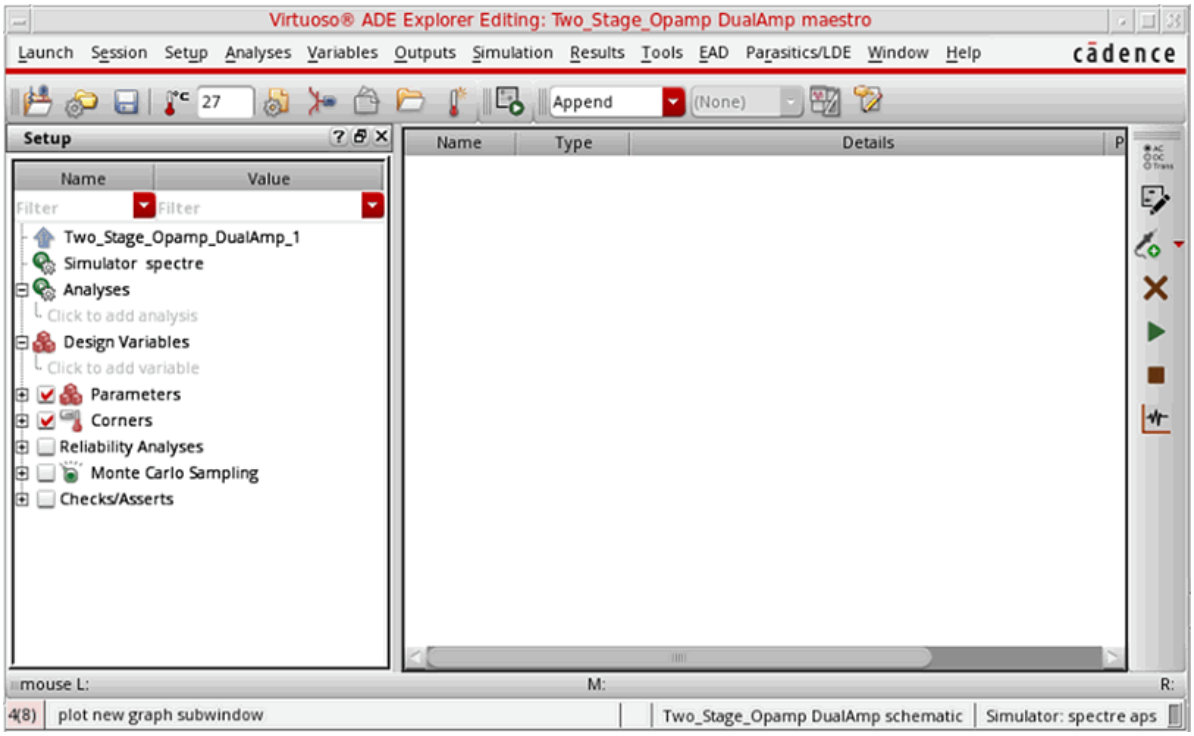
The only requirement criteria to enable the *Show Parasitics* menu option (see Step 4 below), is the existence of the `dcOp` and `dcOpInfo` datasets in the current simulation results read directory, and that you are currently out-of-context.

Backannotating from ADE Explorer therefore requires the following steps:

1. ADE Explorer must be started from the top-level schematic of a configuration view. You must therefore open a configuration view, then open the top-level schematic view.

   Note: If ADE Explorer is already open, you can choose the configuration by selecting *Setup – Design* and choosing the appropriate design.

2. Select *Launch – ADE* Explorer to display the Virtuoso ADE Explorer window.



3. From here, select *Results – Annotate*.

© 2023
All Rights Reserved.

This displays a sub-menu which includes *Show Parasitics* and *Hide Parasitics*. The *Show/Hide Parasitics* menu entries are enabled dependent upon whether parasitic aware design is active or not.

Note: *Results – Annotate* is only enabled when `dcOp point` simulation results are available. That is, parasitics will only be shown when effective resistances (see Effective R Calculations) can be calculated using the results. It will not be enabled to show basic results with `R=N/A` with any other type of simulation results.

4. Click the *Show Parasitics* menu option.

   Selecting *Show Parasitics* causes the schematic editor window, associated with the current ADE Explorer session, to show parasitic annotations when displaying in an out-of-context schematic. When the schematic is in-context, no parasitics will be displayed and a message is displayed in the CIW to inform you why there are no annotations. As you navigate through the schematic hierarchy, the annotations will change as appropriate.

   If the DC simulation results do not include the necessary data for a specific net, then an annotation of “NA” (Not Available) will be set.

   Parasitic annotation will remain active until you choose to *Hide Parasitics*.

   Note: If there are no DC simulation results available (see Backannotation Labelling on page 94), and C annotation is disabled, *Show Parasitics* will be grayed out. Likewise, if there has been no extracted view specified in the config.

Reading Simulation Results

You can change where the directory simulation results are read from by using the *Results – Select* option in ADE Explorer.

ADE Explorer uses this directory when plotting data, annotating schematics, and annotating parasitics. Parasitic aware design looks for the `dcOp` and `dcOpInfo` datasets in the current simulation results directory. If you run a new simulation in ADE Explorer, ADE Explorer will send a message to parasitic aware design to hide backannotation labels, if they are visible, and discard any backannotation that is cached.

Note: You can change the directory that simulation results are written to by selecting *Setup – Simulator/Directory/Host*.

© 2023 All Rights Reserved.

Reporting and Probing Parasitic Values

By probing the schematic `av_extracted` or Smart View, you can examine the instances of parasitic components. The name, value, and net connectivity can be displayed for each parasitic instance.

To view the parasitic instances in the `av_extracted` view for probing, the interconnect layers need to be marked as selectable in the layer selection window (LSW). Use the *AS* (All Selectable) button in the LSW to make all the layers selectable (*AS* is the default).

Important

Schematic View Terminals Correspond to Extracted View Net Segments

In the extracted view, the terminals are not readily accessible. For example, they could be minuscule terminals on the pcapacitors, or they may not exist at all, as in the case of devices which are now just rectilinear layers.

Therefore, in the extracted view, you are required to select the net segments. The net names, such as "`1:netX`", "`2:netX`", and so on, come from PVS, and parasitic aware design does not have any control over the name ordering or continuous nature of the numbers. For example, for the schematic net "`NetX`", the extracted view segment could be "`1:netX`".

In summary, net-to-net probing, in the extracted view, returns parasitics between the selected net *segments*, rather than maintaining the same definition, as in the schematic view, where parasitics between the two "whole" schematic nets are returned.

© 2023 All Rights Reserved.

Reporting Parasitic Instances

To probe parasitic values, you should perform the following:

1. Select *Parasitics – Report Parasitics*.

   If the *Parasitics* menu is not on view in the current session window, you can display it by selecting *Launch – Parasitics* (in Schematics L/XL).

2. Choose, from the sub-menu items, what kind of probing you want to perform: *Net*, *Net to Net*, *Terminal to Terminal*, *Net Capacitors* or *All Nets*.

   For more information on these options see Parasitics – Report Parasitics on page 56.

   Selecting any of the first three report options listed (*Net*, *Net to Net*, *Terminal to Terminal*) will display the Parasitics report form. Go to Step 3.

   Selecting *Net Capacitors* will display the Capacitors for net form. Go to step 4.

   Selecting *All Nets* will display the All Parasitics (Design Report) form. Go to step 5.

   Note: The `NA` label indicates that Parasitic R values are not available due to missing or incomplete simulation results.

3. The Parasitics report form allows you to choose which parasitic values and information to *Display*: *R*, *decoupled C*, *coupled C*, *self C*, *L*, and/or *K* (for descriptions of these values see Display options on page 60).

   You can sort the table by clicking on the required column header and save the report contents to a separate `.txt` or `.csv` file.

   See also Probing Parasitic Instances on page 102.

4. The Capacitors for net report form displays the summed capacitances (*Sum C*) between the net and all nets that it is connected to via parasitic capacitors. No report will be generated for “0” capacitances, and no distinction is made between power/ground, and other nets.

5. The All Parasitics form (Design Report) lists each net in the design, and the total value for the net, based on either the estimated or summed value. Again, the report information can be saved in either `.txt` or `.csv` format.

   For more information on the All Parasitics form, see Reporting on All Nets on page 64.

If you select a parasitic instance from the Parasitics report form, and the extracted view is open, then the extracted view is zoomed to the component symbol associated with the parasitic instance, and the cursor is moved to rest on the symbol (the *From* and *To* nets are also highlighted).

© 2023 All Rights Reserved.

This zoom feature works for the *Net*, *Net to Net*, *Terminal to Terminal* and *Net Capacitors* options.

Probing Parasitic Instances

- For *Terminal to Terminal* options click two pins or instance pins in the schematic or nets in the extracted view to collect all the parasitics between two points.
- For *Net to Net* options click two nets in the schematic or extracted view to collect parasitic capacitances between two different nets.

  To perform *Net to Net* probing, you must run Quantus QRC extraction using coupled capacitors (choose *Coupled* in the *Cap Coupling Mode* drop down list on the *Extraction* tab of the **Quantus QRC Parasitic Extraction Run** form).

  A list of the collected parasitic instances appears. Select an instance from this list to highlight the component symbol associated with this parasitic on the extracted view.

© 2023 All Rights Reserved.

Probing Buses for Parasitics

To probe parasitic values in a bussed net, you should perform the following:

1. Select *Parasitics – Report Parasitics*.
2. Choose either *Net* (continue to step 3), *Terminal to Terminal* (step 4), *Net to Net* (step 5) to display the *Parasitics for net* form.
3. For information about **parasitics on whole nets**, click a bus in the schematic view to display the bits in the Select bits from bus form. Then:
   a. Choose the bits of interest from the list (to select multiple bits, click and hold the `Shift` key while you select the bits, or choose the *Select All* option) and then click the *OK* button.
   b. The parasitics report form appears and *none* is displayed in the *Select* cyclic field (default state).
   c. Choose one or all of the bits in the *Select* cyclic field (use *Select All* to select all bits).

      An ordered list of the parasitics for the selected bus bit or bits appears.
   d. Sort the parasitics by *R*, *decoupled C*, *coupled C*, *L*, and/or *K* as required.
   e. Click *Save* to write the probed parasitics to a file.

      The Save Parasitic Probes form appears.
   f. Enter the filename into the *Filename* field and click *OK*.
4. To collect all the **parasitics between two points** (terminals), click *Terminal to Terminal*, then select bits from the first bus terminal. The bits in the second terminal are automatically matched to the bits in the first terminal. Both selected terminals must be connected to the same net in the schematic. Then:
   a. Choose the bits of interest from the *Select* bits from bus form.
   b. Choose one or *All* of the terminal pair bits from the *Select* cyclic field in the parasitic report form.

      Note: The Select Leaf Terminals form may appear if the selected terminal(s) have more than one leaf terminal. You will be asked here to firstly select the terminals you want to measure resistance values for.
   c. Click *Save* to write the probed parasitics to a file.

© 2023 All Rights Reserved.

5. To collect all the **parasitics between two nets**, click *Net to Net* and then select bits from each of the bus nets in the schematic view. Then:

   a. Choose the bits of interest from the Select bits from bus form.

   b. Choose one or all of the bits from the two *Select* cyclic fields in the parasitic report form.

      The *Select* cyclic fields contain the selected bits for the two nets.

   c. Click *Save* to write the probed parasitics to a file.

6. To probe a bus for net capacitors, click *Net Capacitors* and then click a bus in the schematic view to display the bits in the Select bits from bus form.

   a. Choose the bits of interest from the list (to select multiple bits, click and hold the `Shift` key while you select the bits).

   b. Click *OK*.

      The parasitics report form appears and *none* is displayed in the *Select* cyclic field (default state).

   c. Choose one or all of the bits in the *Select* cyclic field (use *All* to select all bits).

      An ordered list of the parasitics for the selected bus bit or bits appears.

   d. Click *Save* to write the probed parasitics to a file.

      The Save Parasitic Probes form appears.

   e. Type the filename into the *Filename* field and click *OK*.

© 2023 All Rights Reserved.

Creating a Configuration

This section explains how to set up a configuration so that the simulator will run with the `av_analog_extracted` view.

The steps listed here for using the Hierarchy Editor to create a configuration are abbreviated. For complete information, see the *Cadence Hierarchy Editor User Guide*.

To create a configuration for your design:

1. From the CIW, choose *File – New – Cellview*.

   The Create New File form appears.

2. Choose the *Library Name* for the new file.

3. Enter the *Cell Name* that you want to create the configuration for.

   The top-level cell of your design is usually appropriate for use.

4. Enter the config name you want to use into the *View Name* field.

5. Choose *Hierarchy-Editor* from the *Tool* drop-down list box.

6. Ensure that the *Library path file* field correctly specifies the `cds.lib` file that contains the paths to your libraries.

7. Click the *OK* button.

   The Hierarchy Editor along with the New Configuration form is displayed.

8. In the New Configuration form, click the *Use Template* button located at the bottom of the form.

   The Use Template form appears.

9. Select a template that is compatible with the simulator you are running from the *Name* drop-down list, for example *spectre*.

10. Click *OK* in the Use Template form.

    The New Configuration form redisplays with default data for the *Top Cell* and *Global Bindings* sections. This allows you to modify a typical *View List* and *Stop List*, rather than creating them from scratch. Templates exist for each of the simulators.

    To create templates that provide defaults for these fields, see the *Cadence Hierarchy Editor User Guide.*)

11. In the *Top Cell* section, enter the *Library*, *Cell* name, and schematic cell *View* from which to build the configuration.

© 2023 All Rights Reserved.

12. Click *OK*.

The Hierarchy Editor window displays your data.

The Hierarchy Editor window configures the design by using a default *View List* and *Stop List* in the *Global Bindings* section. You need to modify these lists for your design.

13. Use one of the following methods to specify the `extracted` view for the cells or blocks for which you want parasitics simulated.

To specify views for individual blocks	To specify views for multiple blocks
1. In the *Instance Binding* section of the Hierarchy Editor window, position the cursor in the *View To Use* column of the appropriate block. **Note:** If the *Instance Binding* section is not visible in the window, select *View – Instance Table* to display this section. **2.** Press the right mouse key to display a list of commands. **3.** Choose *Select View* to display the list of views for this block. **4.** Choose `extracted` as the view for this block.	➤ In the *Global Bindings* section of the Hierarchy Editor window, add `extracted` as the first view in the *View List* text field. This ensures that the `extracted` view is the selected view for every cell that has an `extracted` view.

Note: The `extracted` view mentioned above could be either `av_analog_extracted` or `av_extracted`.

14. Choose *View – Update* to reconfigure the design to reflect your changes.

The Update Sync-up form appears.

15. Click *OK*.

16. Choose *File – Save* to save the configuration with your changes.

17. Choose *File – Exit* to close the Hierarchy Editor.

Important

After you have created your configuration, you must descend out-of-context (descending into any cellview other than the simulation cellview) before simulating the design.

© 2023 All Rights Reserved.

Simulating the Design in ADE Explorer

Before you can run a simulator with an `extracted` view, you must first of all set up a configuration (see Creating a Configuration on page 105).

Note: Spectre and Ultrasim are the only simulators supported for back annotation.

After a successful simulation, you can select terminals and device pins on the schematic, and use the plot commands to display and probe the results in a waveform window. The resultant waveforms can then be used with ADE calculation and analysis tools.

To run the simulation:

1. In the config top-level schematic window, select *Launch – ADE Explorer*.

   The Virtuoso Analog Design Environment simulation window appears.

2. Choose *Setup – Design*.

   The Choosing Design form appears.

3. Choose the *Library Name* and *Cell Name* of your design.

4. Select the `config` *View Name* for your design.

5. Click the *OK* button.

   This view supplies configuration as well as schematic information.

6. Back in the Analog L simulation window, choose your simulator, model path, environment variables, analyses, and simulator options.

7. Choose *Simulation – Run*.

   When the simulation run completes you can select outputs and probe the design on the schematic.

8. Choose *Outputs – To Be Plotted – Select On Schematic*.

   Note: You can also use the *Results – Direct Plot*... options.

9. To select current outputs, click terminals in the schematic or in the extracted views of the blocks where parasitics were extracted.

   To select voltage outputs, click nets in the schematic or in the extracted views of the blocks where parasitics were extracted.

   Note: The only places where connections on different views are guaranteed to match are on component terminals.

© 2023 All Rights Reserved.

The names of the selected signals are listed in the *Outputs* section on the Virtuoso Analog Design Environment simulation window.

Note: Irrespective of the view from which the signals are selected, their names listed in the *Outputs* section are always taken from the schematic view.

10. Choose *Outputs – Plot Outputs*. Outputs will be plotted in the Virtuoso Visualization and Analysis XL window.

If the output contains calculator functions with net names, in case of out-of-context probing, the net names might change for different extracted views. In such case, you can use the axlMapInstTermToNet SKILL function to dynamically obtain the net connected to a particular instance terminal in the extracted view from inside calculator expressions.

Important

If a net is selected out-of-context (that is, it is selected from the schematic view of a block bound to an extracted view of a configuration), the voltage of one of the external net fragments from the corresponding parasitic net in the extracted view is plotted. The `save` statement in the netlist will contain all external net fragments. If you want to save both external and internal net fragments, set the saveOnlyExternalNodes variable to `nil`.

If a terminal is selected out-of-context and the device has a multiplication factor, the sum of the current flowing through the terminals in each parallel device is evaluated and then plotted. In this case, the `save` statement in the netlist will contain the terminals of all the devices.

For information on running a simulation in ADE Explorer or ADE Assembler see the Running a Simulation chapter in the *Virtuoso ADE Explorer User Guide* or the *Virtuoso ADE Assembler User Guide*.

Out-of-context Probing in ADE Explorer

A cellview is said to be in context when it is the view that is "bound" via configuration, that is, the view that is picked up by the simulator.

A view can therefore be classified as being out of context when it is not the current bound view. For example, if you want to use an estimated view in your simulation, you need to set up a configuration that will override the default view for one or more instances, likely to be schematic, to be the estimated view.

When you now simulate this design, any views that are bound to the estimated view will be netlisted to include the parasitics in them.

© 2023 All Rights Reserved.

Performing Out-of-Context Probing

To perform out-of-context probing in ADE Explorer, right-click on an output value and select either:

Results - Direct Plot - Transient Signal

or

Results - Plot Outputs - Transient Signal.

ADE Explorer plots the output in the Virtuoso Visualization and Analysis XL window and also opens the schematic view. Next, descend into a view that is not an estimated view, for example, the schematic view.

To probe a signal, click the voltage signal at a point close to the terminal. The probe automatically jumps to the closest terminal on that net. An X appears on the selected terminal. You can select several terminals on the same net. Each selected terminal is marked with a different color X. The associated waveform displays in the same color as the X on the schematic.

If the output contains calculator functions with net names, in case of out-of-context probing, the net names might change for different extracted views. In such case, you can use the axlMapInstTermToNet to dynamically obtain the net connected to a particular instance terminal in the extracted view from inside calculator expressions

Important

If a net is selected out-of-context (that is, it is selected from the schematic view of a block bound to an extracted view of a configuration), the voltage of one of the external net fragments from the corresponding parasitic net in the extracted view is plotted. The `save` statement in the netlist will contain all external net fragments. If you want to save both external and internal net fragments, set the saveOnlyExternalNodes variable to `nil`.

If a terminal is selected out-of-context and the device has a multiplication factor, the sum of the current flowing through the terminals in each parallel device is evaluated and then plotted. In this case, the `save` statement in the netlist will contain the terminals of all the devices.

For information on running a simulation in ADE Explorer, see the Running a Simulation chapter in the *Virtuoso ADE Explorer User Guide*.

© 2023 All Rights Reserved.

© 2023 All Rights Reserved.

2

Parasitic Aware Design in ADE Explorer and ADE Assembler

Important

The advanced Virtuoso Parasitic Aware Design (VPAD) functionality is available only in ADE Explorer and ADE Assembler, and can be enabled with the corresponding ADE licenses.

Advanced parasitic aware design functionality, includes the functionality to:

- Specify R estimates
- Specify coupled C estimates
- Specify decoupled C estimates
- Specify L (inductance) and K (mutual inductance) estimates
- Browse parasitic estimates
- Build parasitic/LDE views
- Sweep parasitics during simulation
- Sweep device parameters and perform sensitivity analysis in the presence of estimated parasitics
- Set *Parasitic Mode* toolbar settings to allow smooth re-binding of tests to parasitic/LDE (estimated) or extracted views, and control parasitic and device sweeps
- Set parasitic filters to provide simpler and more sophisticated control over which parasitics to include when refining the extracted view
- Generate reports on parasitics in parasitic/LDE (estimated) and extracted views
- Compare parasitic/LDE (estimated) schematic views with extracted parasitic views
- Investigate parasitics utilizing selections on the schematic or in the Navigator assistant

© 2023 All Rights Reserved.

For more information on parasitic aware design features seeAvailability of Parasitic Aware Design Features.

Chapter Contents

This chapter contains the following main topics:

© 2023 All Rights Reserved.

Accessing Parasitic Aware Design Functionality

You can access parasitic aware design functionality using one of the following methods:

- From the *Parasitics/LDE* menu (select *Menu – Parasitics/LDE*).

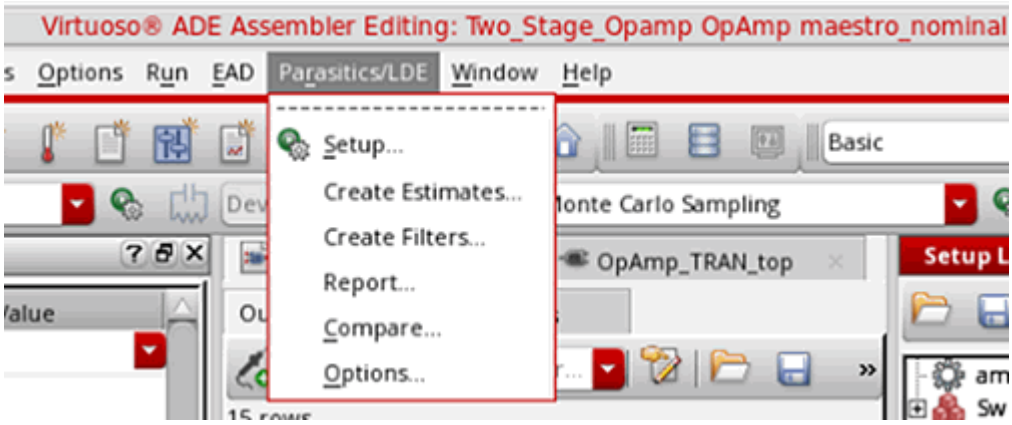


Figure 2-1 The Parasitics/LDE Menu

The *Parasitics/LDE* menu contains the following parasitic sub-menu options:

- *Setup* (see *Parasitics/LDE – Setup*)
- *Create Estimates* (see Parasitics/LDE – Create Estimates)
- *Create Filters* (see Parasitics/LDE – Create Filters)
- *Report* (Parasitics/LDE – Report)
- *Compare* (see Parasitics/LDE – Compare)
- *Options* (see Parasitics/LDE – Options)

- From The Parasitic Mode Toolbar (select *Window – Toolbars – Parasitic Mode*).

  This toolbar can be used to simplify simulation with parasitic/LDE (estimated) or extracted parasitics.

  Note: The *Parasitic Mode* toolbar is visible by default in all workspaces (including parasitics). You can also access the *Parasitic Mode* toolbar from the session window toolbar context-menu.

- From the following parasitic assistant panes (select *Window – Assistants* toolbar pull-down):
  - *Parasitic Filters Assistant*

© 2023 All Rights Reserved.

- *Parasitics & Electrical Setup Assistant*
- *Parasitic Report Assistant*

You can also access these assistants from the session window toolbar context-menu.

Note: Edits made in any of the parasitic assistants, or the *Constraint Manager*, may subsequently require an update to the parasitic/LDE (and schematic) view. If this is the case, a warning message will prompt you if you want to rebuild the impacted views. See also Building the Parasitic/LDE View.

Customizing Parasitic Assistant Display

You can customize the look and feel of each of the parasitic assistants in the following manner to best suit your needs:

- Right-clicking over the toolbar area of any parasitic assistant allows to you select/clear the component parts that you want to display, including the toolbar icons.

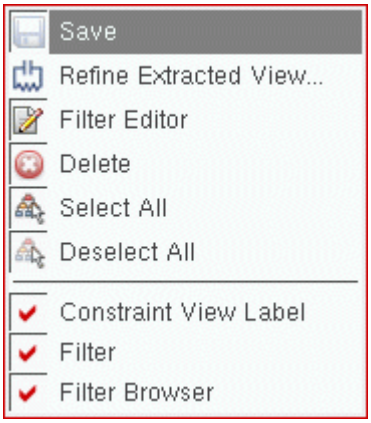


Figure 2-2 Right-clicking over the Parasitic Filters Assistant Toolbar

© 2023
All Rights Reserved.

- You can also control the information that you want to display in parasitic assistants by choosing what information order and column headers that you want to have on view. To do this, right-click over the column headers to view the column/information display options available with each assistant.

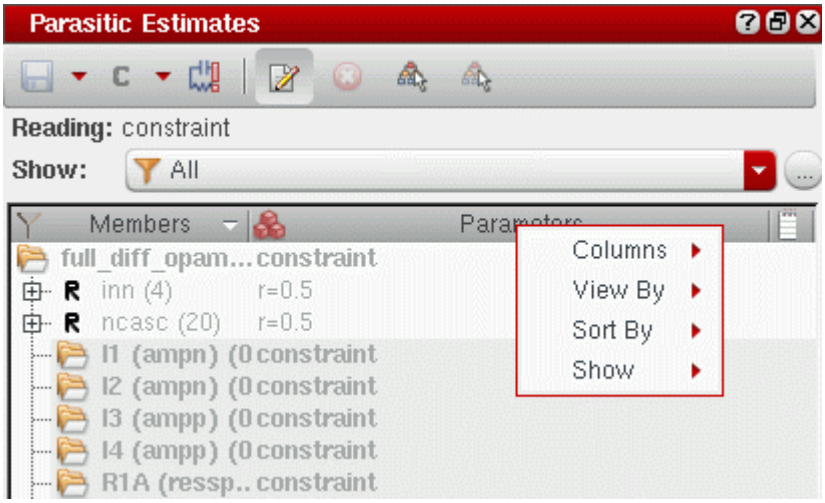


Figure 2-3 Right-clicking over the Parasitics & Electrical Setup Assistant Column Headers

Note: Customization settings persist between sessions (as stored in the .cadence directory).

© 2023
All Rights Reserved.

Parasitic Aware Design Workspace Configurations

The Virtuoso Schematic Editor provides three workspace options in the *Window* menu:

Parasitics-Estimates: The *Parasitics-Estimates* workspace is used for the specification of parasitic estimates. This workspace contains the Parasitics & Electrical Setup Assistant.

Parasitics-Filters: The *Parasitics-Filters* workspace is used to generate refined extracted views. It comprises of the Parasitic Filters Assistant.

Parasitics-Report: The *Parasitics-Report* workspace is used to report parasitic estimated/ extracted values and compare parasitic views. This workspace contains the Parasitic Report Assistant.

Important

You will only be able to access parasitic workspaces when viewing a schematic. When viewing the *Welcome to ADE Assembler* tab, you will not be able to access parasitic workspaces. Parasitic assistants will however be available through standard menu selection.

The *Parasitics-Estimates* workspace focuses on estimated parasitics while *Parasitics-Filters* is used in the extracted parasitics flow. The *Parasitics-Report* workspace can be used with both the parasitics/LDE (estimated) and extracted flows.

- For general information on workspaces and assistants, including how to create your own customized workspaces, see the *Virtuoso Studio Design Environment User Guide*.

© 2023 All Rights Reserved.

Setting Up and Using Parasitics

This section contains information on:

- An Introduction to Parasitic Estimates

And, the contents of the *Parasitics/LDE* menu:

- Parasitics/LDE – Setup
- Parasitics/LDE – Create Estimates
- Parasitics/LDE – Create Filters
- Parasitics/LDE – Report
- Parasitics/LDE – Compare
- Parasitics/LDE – Options

© 2023 All Rights Reserved.

An Introduction to Parasitic Estimates

The parasitic aware design flow provides the functionality to generate estimated parasitics, pre-layout, for simulation. Here, parasitic values are specified through *estimates* which are stored in parasitic estimate cell views separate from, but still associated with, the schematic view.

Note: R, C, L and K parasitic estimates are supported.

After layout is done, and has been extracted using Quantus QRC/RCX to create an extracted view, parasitic aware design can compare these parasitic estimates against the actual parasitic values, highlighting any parasitics that exceed their estimated values or differ from them by more than any optional specified *tolerances*. To achieve this you can control the comparison method which can treat estimates as either *limits* or *targets,* with a specified tolerance expressed as a percentage.

Note: See Extracted Parasitics for information on how to compare generated estimate views against existing extracted views.

Estimated parasitics, through the performance of a pre-layout simulation, will help ensure that your design will function correctly with specific estimated parasitic values. Once the design has been laid out, the actual extracted parasitic values can then be compared against the estimates.

Some benefits to using parasitic estimates include:

- ADE integration via The Parasitic Mode Toolbar, for example automatic test rebinding and control of parasitic versus device parameters for sensitivity analysis and optimization.
- Parasitic estimates are an overlay over “golden” schematics that may be read-only, avoiding the need to edit the schematic.
- Different parasitic estimates can be applied to different instances in the design of the same cell. This cannot be done by directly editing the schematic.
- Different parasitic estimates can be overlaid onto the schematic for “what if” experiments, process variations, and so on.

© 2023 All Rights Reserved.

Parasitics/LDE – Setup

You must set up parasitics before starting any additional parasitic tasks. Choose *Parasitics/ LDE – Setup* to display the **Parasitics & LDE Setup** form, as shown below.

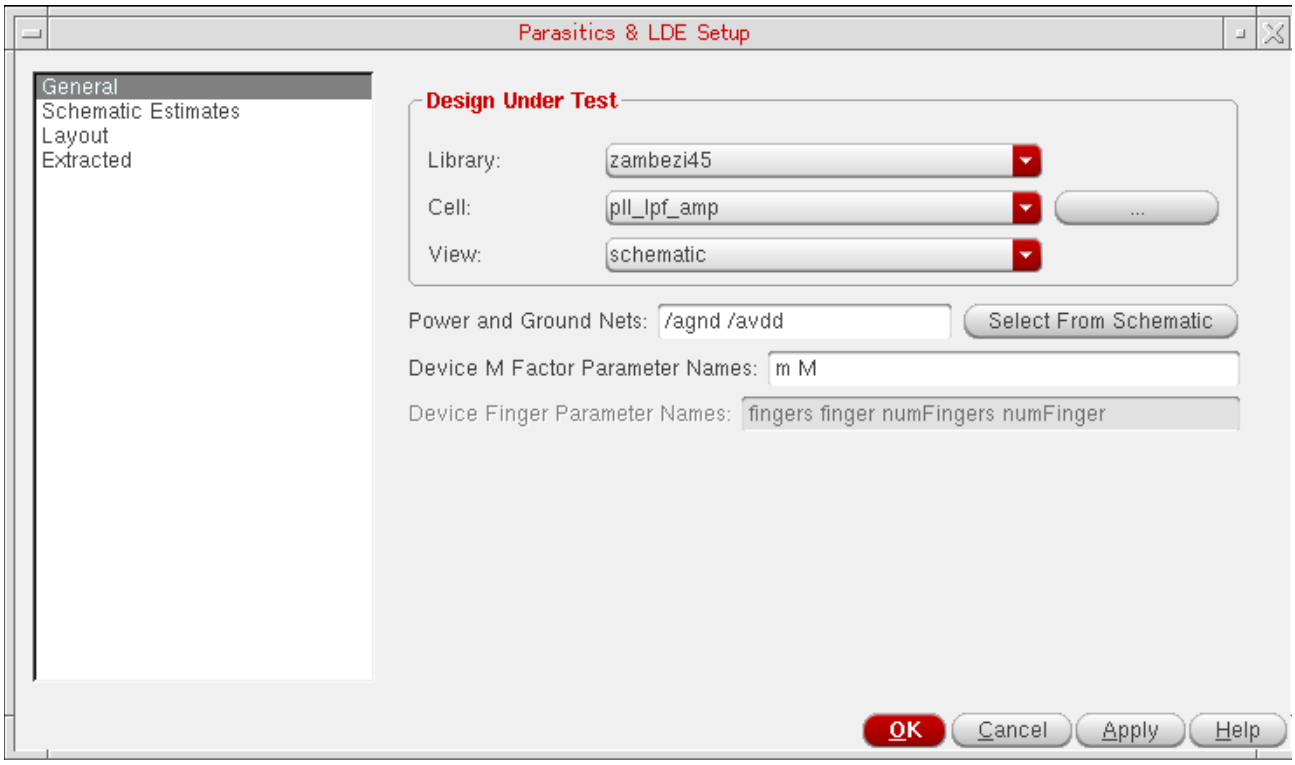


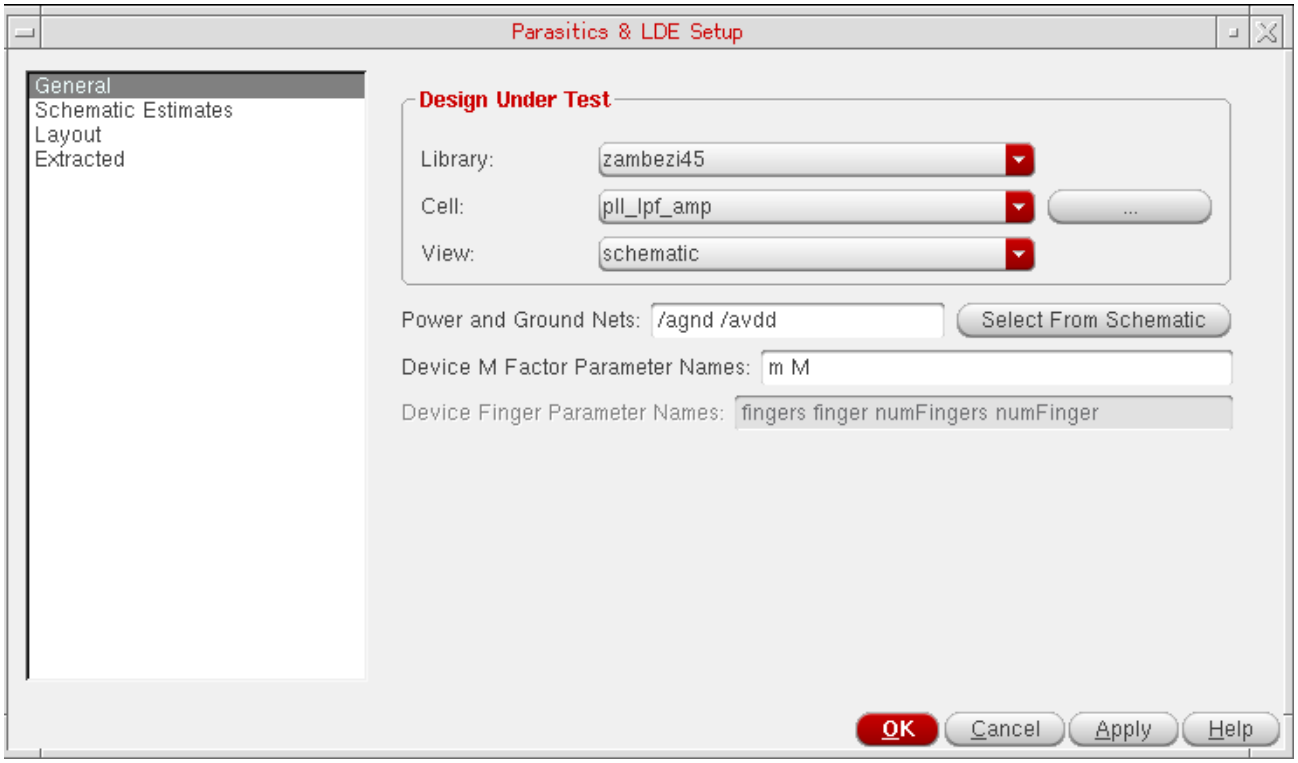
Figure 2-4 Setup Parasitics and LDE Form

The following topics describe the different tabs of the Parasitics & LDE Setup form:

- General Tab
- Schematic Estimates Tab
- Layout Tab
- Extracted Tab

© 2023 All Rights Reserved.

General Tab



Note: Click browse (…) to open the Library Browser window from where you can choose an alternative schematic design for use.

The library, cell, and view name will also be displayed in the Build Parasitic/LDE View form (see Building the Parasitic/LDE View) that you use to build the parasitic/LDE view.

Tip

You can set the showAllCellViews environment variable to specify how to populate the library and cellview names on this tab.

This tab contains the following fields:

Form Field	Description
Design Under Test	Specifies the *Library*, *Cell*, and *View* names of the top-level schematic design that parasitic estimates should be associated with.
- *Library*	Specifies the library name.

© 2023 All Rights Reserved.

- Cell	Specifies the cell name.
-View	Specifies the view name.
Power and Ground Nets	Specifies the power and ground nets. Parasitic aware design uses these nets to determine which parasitic capacitances are decoupled (one side of the capacitor is connected to a power or ground net) or coupled (the capacitance is between two other nets). **Note:** If this field is left blank, all capacitances will be coupled. See also Capacitance Estimates. You can either enter the net names or select the power and ground nets directly from the schematic. To select the nets from schematic, click *Select from Schematic*. The schematic view is displayed in a new tab if it is not already open. If the schematic is already open in another tab, that tab will be displayed. Now you can select the supply nets on this tab. After selecting the supply nets, press the `Esc` key. The Setup Parasitics and LDE form will be displayed again and the names of the selected nets are shown in the field.
Device M-Factor Parameter Names	While generating a netlist with parasitics, if the tool finds any cell with parameter names listed in field, it expands the cell if the *Expand Devices with M-Factor* check box is selected. However, when using LDE parameters, cells will be expanded according to the multiple factors used in the layout or MODGEN constraint. In both cases, the m-factor value on the individual expanded devices in the netlist is reset to `1`.
Device Finger Parameter Names	While generating a netlist with parasitics, if the tool finds any cell with the given finger parameter names, it expands the cell if the *Expand Devices with M-Factor* check box is selected. However, when using LDE parameters, cells will be expanded according to the multiple fingers used in the layout or MODGEN constraint.

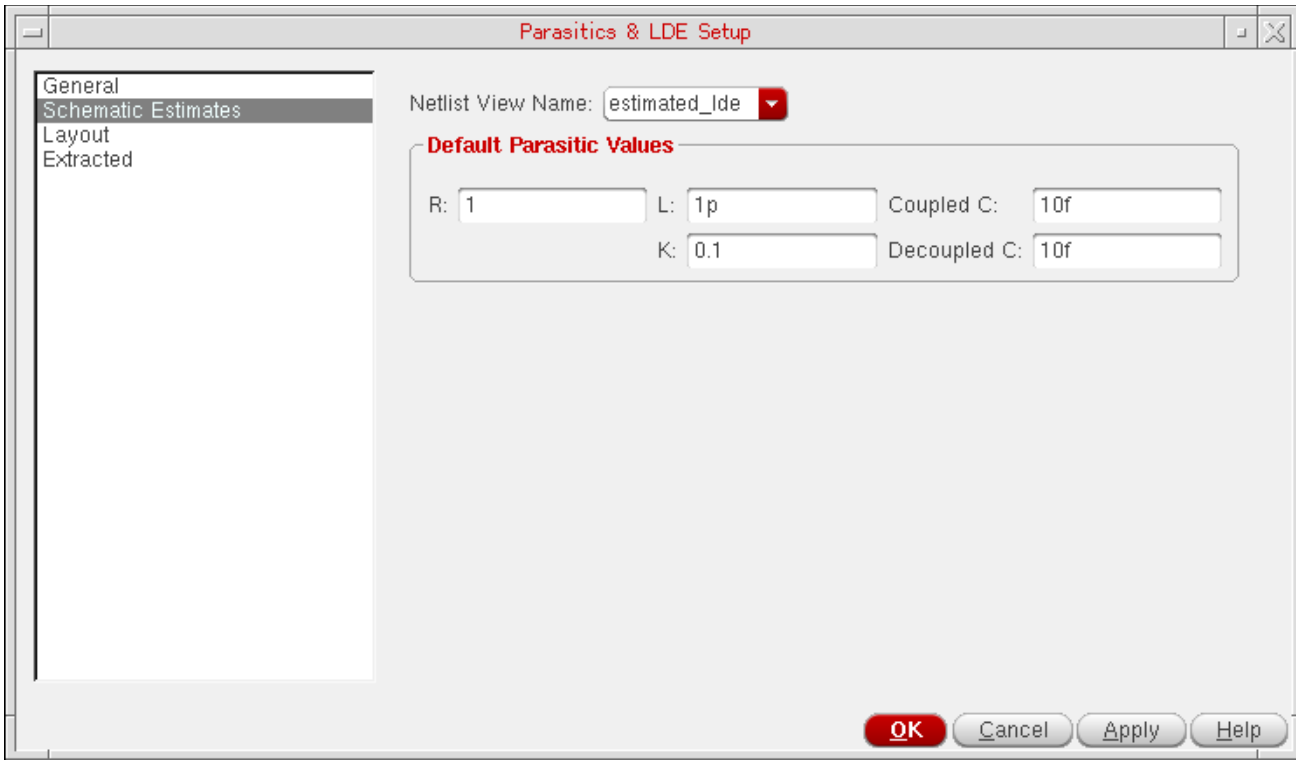
Important Points to Note

- Once the setup information has been added, it is preserved for future sessions.
- It is recommended that you set the parasitic/LDE view at the DUT (design under test) level. A parasitic/LDE view created at the DUT level allows you to run simulations with multiple testbenches, each pointing to the same parasitic/LDE view.

© 2023 All Rights Reserved.

- Parasitic estimate functionality is available only from a configuration or a schematic view; it is not available form an extracted view.
- The information from this form is shared across all tabs in the current session window. Therefore, once you have set up parasitics in one session window tab, parasitics functionality will be set up for use across all other tabs open in the current session. For more information about the use of tabs, see Tabs in a Session Window in the Virtuoso Studio Design Environment User Guide.
- Certain parasitics functionality, such as backannotation and reporting, is available only when you open the schematic view specified in the Parasitic & LDE Setup form (or descend below it).

Schematic Estimates Tab



This tab sets the name of the schematic view and the default resistance and capacitance values that are to be used when estimate constraints are first created. These values can be constants or expressions that contain ADE variables.

Note: Expression syntax must conform to the standards laid out in the Analog Expression Language Reference.

© 2023 All Rights Reserved.

The tab contains the following fields:

Form Field	Description
Netlist View Name	Specifies the name of the view to be used to create a netlist that includes schematic parasitic estimates.
R	Sets the default parasitic estimate value for resistance.
L	Sets the default parasitic estimate value for inductance.
K	Sets the default parasitic estimate value for mutual inductance.
Coupled C	Sets the default parasitic estimate value for coupled capacitance.
Decoupled C	Sets the default parasitic estimate value for decoupled capacitance.

Note: Use this tab when you want to include schematic parasitics in the simulation netlist.

You can specify the initial values of the fields described above by setting the corresponding variables in the `.cdsenv` file as shown in the following examples:

```
msps.estimates defaultR string "1.0"
msps.estimates defaultL string "1.0p"
msps.estimates defaultK string "0.1"
msps.estimates defaultCC string "10f"
msps.estimates defaultDC string "10f"
```

Setting Up Parasitics Summary

To setup parasitics:

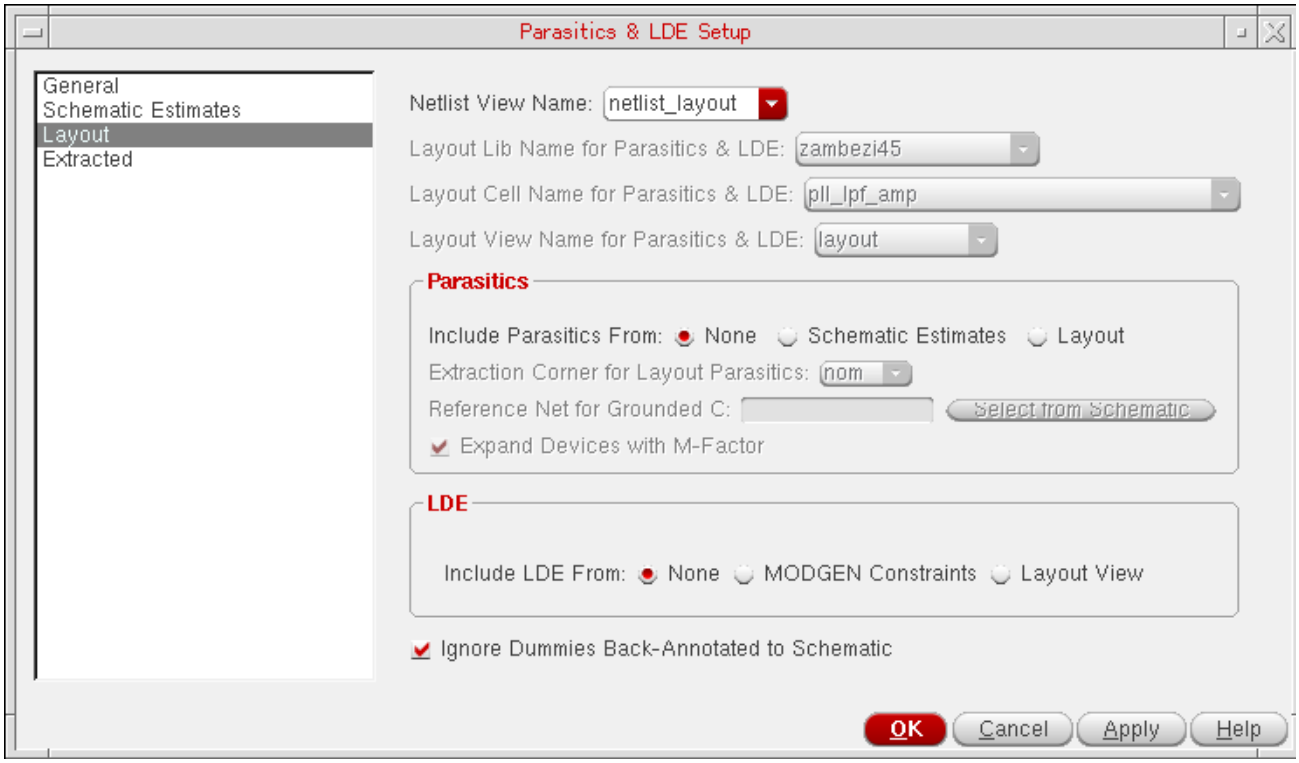
1. Select *Parasitics/LDE – Setup* to display the Parasitics & LDE Setup form.
2. Complete the *Design Under test* section on the *Genera*l tab as required.
3. Enter the names of the power and ground nets in the *Power and Ground Nets* field.

   Note: You can enter the net names manually or click the *Select From Schematic* option. Then, select the nets directly from the cellview.
4. On the *Schematic Estimates* tab, set the *Default Estimate Values* for *R*, *L*, *K*, *CoupledC*, and *DecoupledC*.
5. Click *OK* to complete parasitic setup and close the Parasitics & LDE Setup form.

© 2023 All Rights Reserved.

Parasitic functionality will now be available in all tabs open in the current session window.

Layout Tab



This tab specifies the options to include parasitics from a layout view, or to include layout dependent effects (LDEs) from Modgen constraints or from a layout view (partial or complete).

For more information about simulating designs with LDE, see the Simulating Designs with LDE section in *Virtuoso ADE Assembler User Guide*. For information on the electrically aware design flow, see Virtuoso EAD Flow user guide.

This tab contains the following fields:

© 2023 All Rights Reserved.

Form Field	Description
Netlist View Name	Specifies the name of the view to be used to create a netlist that includes parasitics or LDE parameters from the layout view. **Note:** Both LDE parameters and layout parasitics can be added to the same netlist view.
Layout Lib Name for Parasitics & LDE	Name of the library of the design that contains a layout cellview with layout parasitics.
Layout Cell Name for Parasitics & LDE	Name of the design cell that contains a layout view with layout parasitics.
Layout View Name for Parasitics & LDE	Name of the layout view that contains layout parasitics to be used in resimulation of a design. This can be a partial or a complete layout view. When you specify a layout name in this field, select *Include Parasitics From Layout* option in the *Parasitics* section on the *Layout* tab or the *Layout View* option in the *LDE* section on the *Layout* tab. This is helpful in the EAD or LDE flow. **Note:** You can choose to include both the LDE parameters and the layout parasitics from the same layout view.
Parasitics	This section contains fields to be used while including parasitics from a layout view. This is required while simulating a design with layout dependent effects or in the Electrically Aware Design flow. For more details, refer to Virtuoso Electrically Aware Design user guide.

© 2023 All Rights Reserved.

- Include parasitics from	Specifies the source of parasitics. You can select any one of the following three sources: ■ *None* - No parasitics are used. This is the default option. ■ *Schematic Estimates* - Includes estimates from the Parasitics & Electrical assistant. These are the same estimates used in the Schematic Estimates mode. This option allows the use of LDE with estimated parasitics ■ *Layout* - Includes parasitic estimates from the layout view specified in the *Layout View Name for Parasitics & LDE* field. **Note:** When you choose to include layout parasitics in simulation, you also need to change the parasitic mode to `Layout (Parasitics/LDE)`.
- Extraction Corner for Layout Parasitics	When you choose to include parasitics from the specified layout view, names of the all the extracted corners for that layout are listed in this field. Select name of the desired corner.

© 2023 All Rights Reserved.

- Reference Net for Grounded C	Specifies name of the ground net to be used for grounded capacitance. You can either type a name in this field or click *Select from Schematic* to select a net from the design schematic. **Important Notes** ■ A reference net for grounded C is not required if you are extracting only R estimates or if you are stitching only R estimates from RC extraction. ■ You can specify name of a global net that does not exist in the design under test. The net will be created in the netlist view as an inherited connection so that you can use a netSet property on the cell instance to specify the net to identify it with. The netSet property name is the same as the net name. ■ You can use he *msps.layout referenceNet* environment variable to set the initial default value of the reference net field. However, after you specify a value for this in the LDE and Parasitics Setup form, the value for reference net field is saved in the design setup and the referenceNet environment variable is not considered for the design. This allows different designs to use different reference nets without interference.
- Expand Devices with M-Factor	Specifies if it is required to expand the devices with multiple factors before generating a netlist. When this option is selected, the m-factor parameter on the schematic device is then used to create multiple parallel devices in the netlist view. This allows the parasitics for the nets between each m-factor instance in the layout to be brought into the netlist view for simulation. If this check box is disabled, the device is represented as a single instance in the netlist view, and the parasitics between m-factor instances will not be included.
LDE	This section contains fields to be used while including layout dependent effects from a layout view. This is required while re-simulating a design in the Layout Dependent Effects flow. **Note:** When you choose to include LDE parameters in simulation, you also need to change the parasitic mode to `Layout (Parasitics/LDE)`.

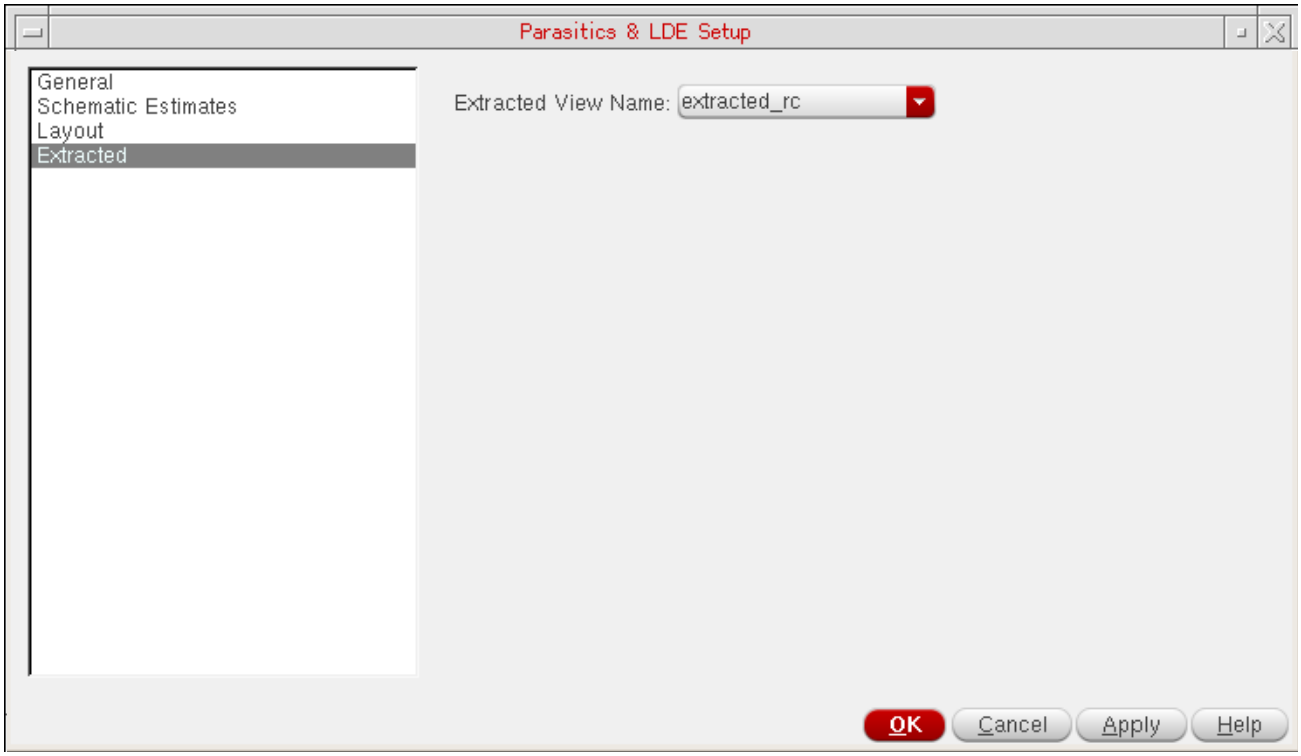
© 2023 All Rights Reserved.

- *Include LDE From*	Specifies the source of LDEs. You can choose any one of the following options: ■ *None* - This is the default option, which specifies that no parameters from layout or MODGEN constraints are to be used. ■ *MODGEN Constraints* - Includes LDEs extracted from the Modgen constraints defined for the given constraint cellview. ■ *Layout View* - Includes estimates from the layout view specified in the *Layout View Name for Parasitics & LDE* field.  *Important* If you include parasitics from layout in the *Parasitics* section, you cannot choose to use LDEs defined in MODGEN constraints. This is because parasitics and LDE parameters must come from the same source.
Ignore Dummies Back-Annotated to Schematic	Specifies that if there are any dummy cells backannotated from layout to the schematic view, they need to be ignored while generating a netlist for simulation. For more details on backannotated dummy instances, refer to Dummy Instances Backannotation.

Note: When simulating designs with LDEs, Virtuoso runs PVS-LVS to read the Modgen constraints and to extract the values of LDE parameters. The Virtuoso IPVS log file is saved in the `/tmp/<username>_pvs_*` directory and is automatically deleted when you exit the Virtuoso session. However, for debugging purposes, you can preserve the Virtuoso IPVS log file. For this, set the Virtuoso_IPVS_log environment variable before starting Virtuoso. For more details, refer to Debugging Problems in Virtuoso IPVS in the *Virtuoso IPVS User Guide*.

© 2023 All Rights Reserved.

Extracted Tab



This section specifies name of the extracted view that contains the extracted parasitic details.

See also: Creating an Analog Extracted View.

© 2023
All Rights Reserved.

Parasitics/LDE – Create Estimates

Selecting *Parasitics/LDE – Create Estimates* displays the Parasitics & Electrical Setup Assistant.

Amongst other features, from here you can create estimated parasitics and build a parasitic/LDE view.

Note: When this option is selected, the schematic specified in the setup form will be opened automatically in a new tab or raised if it is already open.

Parasitics/LDE – Create Filters

Selecting *Parasitics/LDE – Create Filters* displays the Parasitic Filters Assistant.

Amongst other features, from here you can create parasitic filters and refine the extracted view.

Note: When this option is selected, the schematic specified in the setup form will be opened automatically in a new tab or raised if it is already open.

Parasitics/LDE – Report

Selecting *Parasitics/LDE – Report* displays the Parasitic Report Assistant.

Amongst other features, from here you can create a range of parasitic reports (including comparing the parasitic/LDE and extracted views), export this report information, and cross-probe with the extracted view.

Note: When this option is selected, the schematic specified in the setup form will be opened automatically in a new tab or raised if it is already open.

Parasitics/LDE – Compare

Selecting *Parasitics/LDE – Compare* displays the Compare Parasitics form (see Parasitic Comparisons and Comparison Reports).

From here you can setup parasitic comparisons for the parasitic/LDE schematic and extracted views.

© 2023 All Rights Reserved.

Parasitics/LDE – Options

Selecting *Parasitics/LDE – Options* displays the Options form.

The Options form contains a selection of options related to backannotation and parasitic probing for estimated and extracted parasitics (see Extracted Parasitics).

Values entered here will determine some of the content of other parasitic aware design forms.

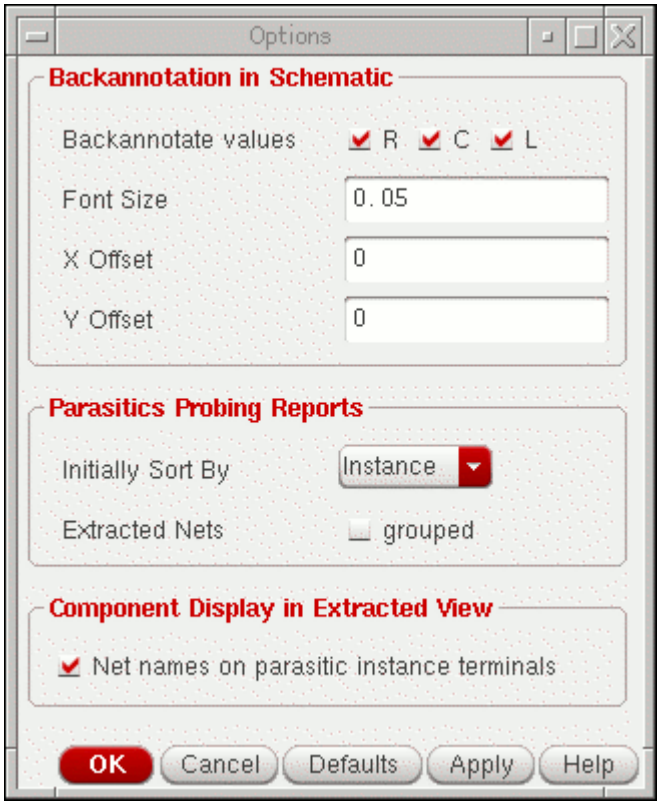


Figure 2-5 The Options Form

GUI Item	Description
Backannotation in Schematic	
- *Backannotate values*	Specify the backannotation values that you want to view (*R*, *C* and/or *L*).
- *Font Size*	Specify the label font size for displaying parasitic backannotation.
- *X Offset*	Sets the horizontal offset from the centre of the net when displaying parasitics.

© 2023 All Rights Reserved.

GUI Item	Description
- *Y Offset*	Sets the vertical offset from the centre of the net when displaying parasitics.
Parasitics Probing Reports	(See also the Extracted Parasitics).
- *Initially Sort By*	Specify the initial sorting method (*Instance*, *Type*, *Value*, *From*, or *To*) to be used when probing parasitics on nets
- *Extracted Nets*	Check the *grouped* option to report parasitics for the whole design net when probing an extracted net (this will work as if probing in the schematic). For detailed debugging however, you should clear the *grouped* option. In this case the report will be specific to the selected extracted net fragment. **Note:** The effective R results will be shown if you have setup effective R using the *Setup ~R Reporting* option in the *Parasitic Report* assistant.
Component Display in Extracted View	(See also the Extracted Parasitics).
- *Net names on parasitic instance terminals*	Enables the display of the names of nets connected to terminals of parasitic instances in the extracted view.

© 2023 All Rights Reserved.

Extracted Parasitics

This section on extracted parasitics contains the following information:

- An Introduction to Extracted Parasitics
- An Introduction to Parasitic Filters
- Extracted Parasitics Flow Overview
- Optimizing an Extracted or Layout View

Related Information

- For information on how parasitic aware design interacts with Quantus QRC and RCX to create an extracted view see An Analog Simulation Flow using PVS on page 73.
- For information on refined extracted parasitics interaction with the *Parasitics Filter* assistant, see the Parasitic Filters Assistant.

© 2023 All Rights Reserved.

An Introduction to Extracted Parasitics

Parasitic aware design also provides functionality that enables you to *filter* parasitics from an extracted view to create a new, refined, extracted view so that you can debug the sensitivities of various nets to parasitics and speed up simulation.

Note: See also the Parasitic Filters Assistant.

To achieve this, parasitic aware design uses the Virtuoso unified constraints system (see *Virtuoso Unified Custom Constraints User Guide*) to create and edit *filters* (special parasitic aware design constraints).

The refined extracted parasitics options can be accessed via the *Parasitics/LDE - Create Filters* menu option or the *Parasitics - Extracted* workspace.

An Introduction to Parasitic Filters

You can make selections from the schematic view to populate a *filter*.

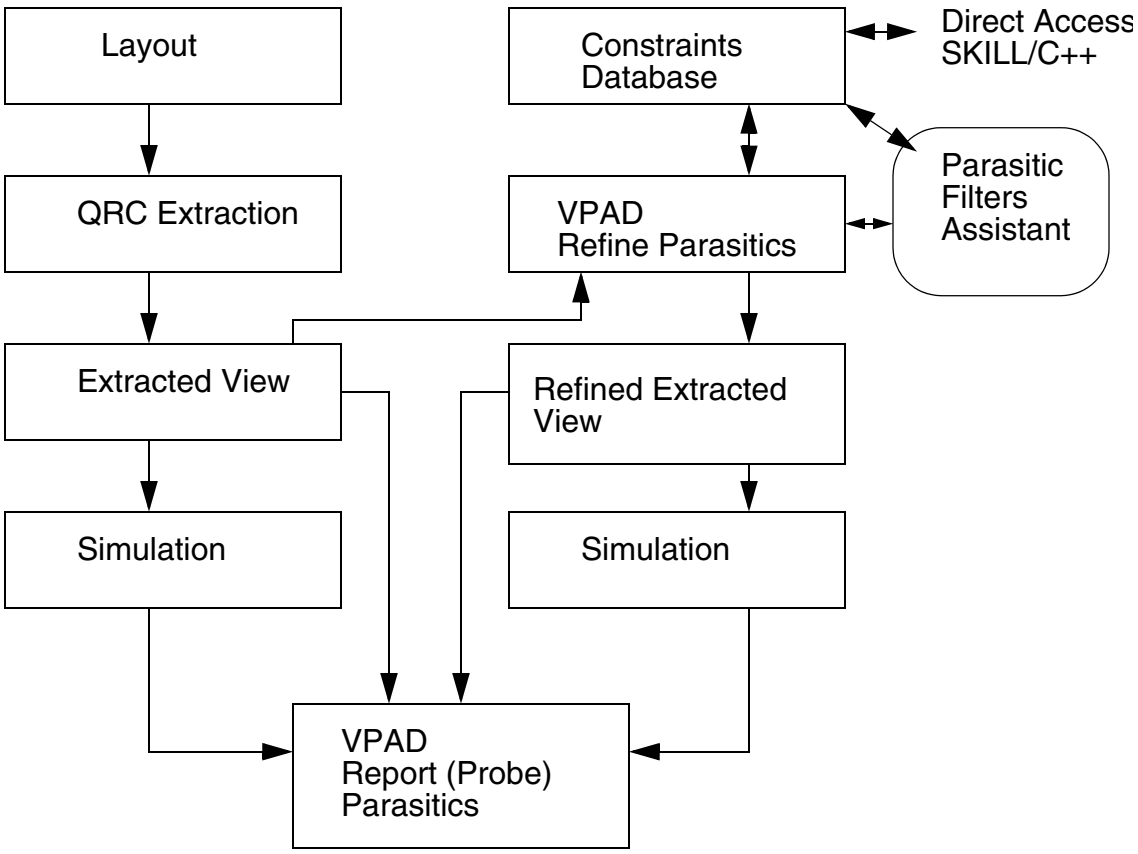
These filters are stored in the constraints database, along with other constraints, but are only retrievable using parasitic aware design. Parasitic aware design will use these filters to determine which parasitics should be maintained in any resultant refined extracted view.

The Parasitic Filters Assistant can be used to filter parasitics from the entire extracted view using a *threshold value*, or you can refine parasitics on a net by net or block by block basis, to simulate the new extracted view.

© 2023 All Rights Reserved.

Extracted Parasitics Flow Overview

A brief overview of the extracted parasitics flow, in the back end, is described as follows:



Optimizing an Extracted or Layout View

You can sweep the parameters of an extracted design, run simulations, and optimize the design to meet the desired specifications. The simulation results are reported in the *Results* tab and the optimized parameter values can be backannotated to the schematic.

To simulate an extracted or parasitic/LDE view, do the following:

1. Click *Parasitic/LDE – Setup*.

   The Setup Parasitics and LDE form is displayed.

2. Specify the name of the extracted view in the *Extracted View Name* field.
3. Click *OK* to close the Setup Parasitics and LDE form.

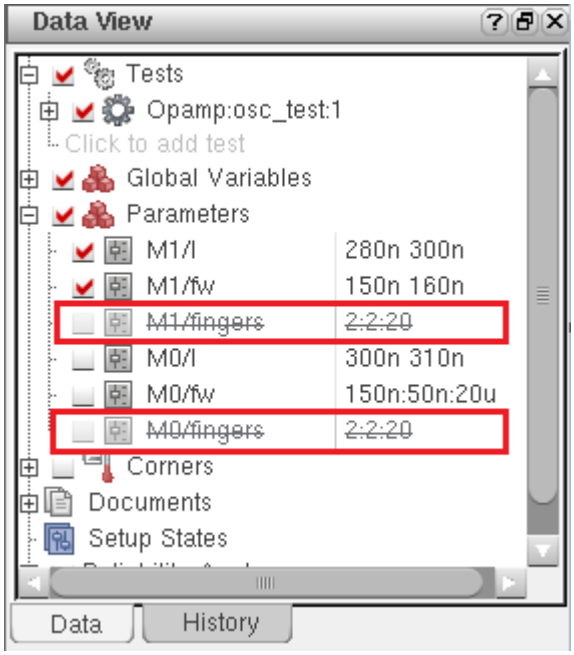
© 2023 All Rights Reserved.

4. In the Parasitic Mode toolbar, select `Extracted (Parasitic/LDE)`.

   Note: If you are considering the layout-dependent effects (LDEs) or parasitics from a layout view, which is specified in the *Layout view name for parasitics and LDEs* field in the Setup Parasitics and LDE form, select the `Layout (Parasitics/LDE)` parasitic mode.

   Note the following in the Data View pane:

   - The device parameters that define the m-factors or fingers are disabled and shown with a strikethrough, as shown in the following figure.



   In the above example figure, `M1/fingers` and `M0/fingers` are the two parameters that are defining the number of fingers for devices `M1` and `M0`, respectively. This indicates that the extracted view simulation flow will not consider these parameters because they have already been extracted out and are fixed in the extracted view. That is, the flow will not optimize the device multiplier or the number of fingers in a device.

   You can specify the names of parameters that define these properties for a device in the Layout Tab section of the Setup parasitics and LDE form.

Important

For all other device parameters, such as `M1/l` or `M0/fw` in the above figure, which are not fixed in the extracted view, the values will be taken from the instances in the extracted view. The values given in the schematic view will be ignored.

© 2023 All Rights Reserved.

- You cannot create any new parameter to define the m-factors or fingers for a device.

5. Ensure that the run mode is set to one of the following modes:
   - Single Run, Sweeps or Corners
   - Monte Carlo Sampling
   - Local Optimization
   - Global Optimization
   - Manual Tuning

   Note: Currently, this feature is not supported for other run modes.

6. On the Run toolbar, click *Run Simulation* to run the simulation.

   The simulation is run by using the extracted or the parasitic/LDE view for the testbench. After the simulation is successfully run, observe the following:

   - All the parameters added from the schematic view are mapped to the extracted view. In addition, depending on the multiplier factor, an instance in the schematic is mapped to one or more devices in the extracted view.

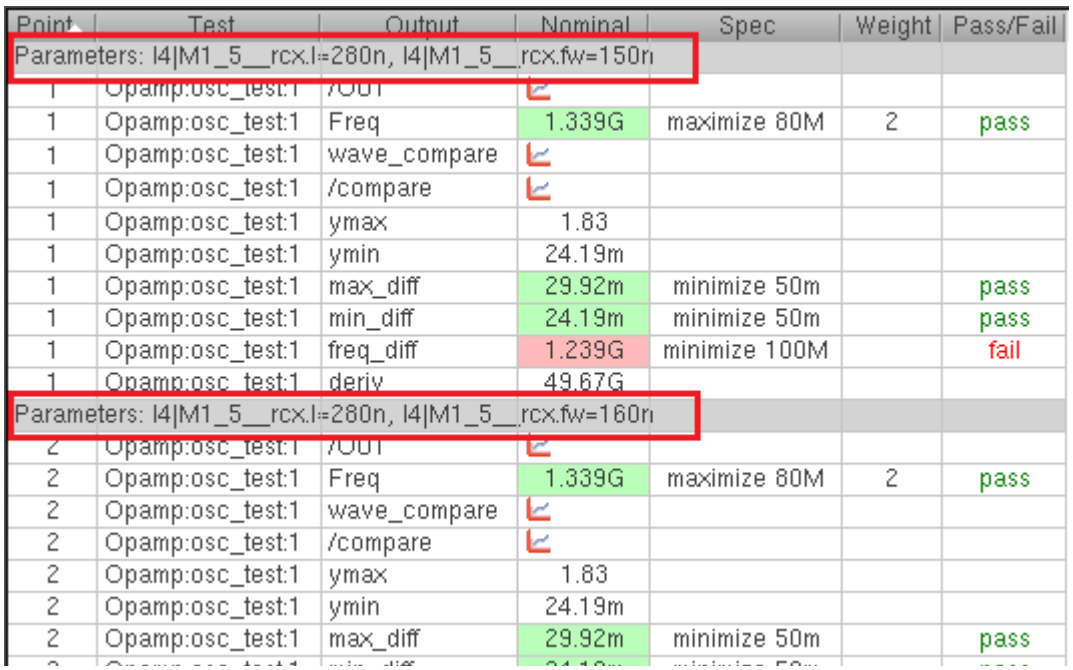
     For example, an instance M1 in the schematic may get mapped to the four devices in the extracted view that have been elaborated based on the values for the `m-factor` and `fingers` parameters. If a device parameter is `M1/fw` with values set to `5u:1u:20u`, when you switch the parasitic mode to `Extracted (Parasitic/LDE)`, the following four parameters are created:

     ```
     M1/fw = 5u:1u:20u
     M1_1__rcx/fw = M1/fw@
     M1_2__rcx/fw = M1/fw@
     M1_3__rcx/fw = M1/fw@
     ```

     On the assumption that each finger will take the same value, independent parameters are created for each finger and the parameters are matched against each other so they take the same value at each simulation point.

© 2023 All Rights Reserved.

The sweep values for parameters are shown on the parameter headers in the *Results* tab, as shown below.



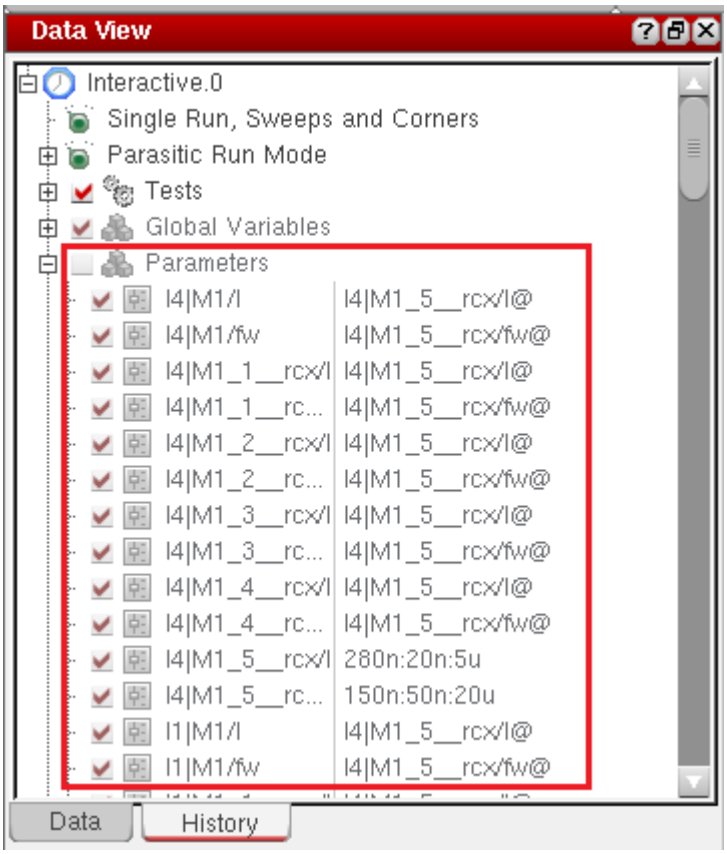
Point	Test	Output	Nominal	Spec	Weight	Pass/Fail
Parameters: I4|M1_5__rcx.l=280n, I4|M1_5__rcx.fw=150n						
1	Opamp:osc_test:1	/OUT				
1	Opamp:osc_test:1	Freq	1.339G	maximize 80M	2	pass
1	Opamp:osc_test:1	wave_compare				
1	Opamp:osc_test:1	/compare				
1	Opamp:osc_test:1	ymax	1.83			
1	Opamp:osc_test:1	ymin	24.19m			
1	Opamp:osc_test:1	max_diff	29.92m	minimize 50m		pass
1	Opamp:osc_test:1	min_diff	24.19m	minimize 50m		pass
1	Opamp:osc_test:1	freq_diff	1.239G	minimize 100M		fail
1	Opamp:osc_test:1	deriv	49.67G			
Parameters: I4|M1_5__rcx.l=280n, I4|M1_5__rcx.fw=160n						
2	Opamp:osc_test:1	/OUT				
2	Opamp:osc_test:1	Freq	1.339G	maximize 80M	2	pass
2	Opamp:osc_test:1	wave_compare				
2	Opamp:osc_test:1	/compare				
2	Opamp:osc_test:1	ymax	1.83			
2	Opamp:osc_test:1	ymin	24.19m			
2	Opamp:osc_test:1	max_diff	29.92m	minimize 50m		pass

- When you place the pointer on the parameter header, the tooltip shows the names of the expanded devices and the values of their parameters used to run simulation.

- When optimization is complete, you can backannotate the parameter values of the point that best met the specifications. For this, right-click the parameter header and choose *Backannotate*. In the Back Annotation Options form that is displayed, make sure that the *Only device parameters* option is selected and click *OK*. The corresponding device is highlighted in the schematic and the parameter value is annotated.

© 2023 All Rights Reserved.

- The expanded parameter details are saved in the history. You can view the parameter values used for a particular history by expanding the *Parameters* tree on the History tab.



Important

If you have added, modified, or removed any parameter in the extracted mode, the changes are retained after switching to the `No Parasitics` mode.

© 2023
All Rights Reserved.

Parasitics & Electrical Setup Assistant

Important

See first Setting Up and Using Parasitics on page 117 as you must correctly setup parasitic before using the *Parasitics & Electrical Setup* assistant.

Parasitic estimates comprise of a “star-shaped” model for each net.

- Manual parasitic estimate resistances and inductors are associated with the net’s instance terminals. The instance terminals may be instance terminals of instances with/without underlying hierarchy. The parasitic resistances and inductors for a specific instance terminal, or net terminal, will always be added in series.
- Manual parasitic estimate mutual inductance is associated with the two estimated parasitic inductances.
- Coupled and decoupled capacitances are associated with the net collectively and connects to the center of the star.

Note: See also Parasitic Stitching which allows for an arbitrary topology.

Estimates are stored as constraints and for simulation purposes an (parasitic/LDE) estimated schematic cellview must be generated from the design and user-selected estimates.

Note: The *Parasitics & Electrical Setup* assistant only displays parasitic estimate constraints.

This section on the *Parasitics & Electrical Setup* assistant comprises of the following:

- Accessing the Parasitics & Electrical Setup Assistant
- The Parasitics & Electrical Estimates Assistant Toolbar
- Creating Parasitic Estimates
- Saving Parasitic Estimates Created in the Current Session
- The Parasitics & Electrical Setup Assistant Context-Menu

© 2023 All Rights Reserved.

Accessing the Parasitics & Electrical Setup Assistant

The *Parasitics & Electrical Setup assistant* can be accessed using one of the following methods:

- By selecting *Window – Workspaces – Parasitics-Estimates* to display the Setting Up and Using Parasitics.
- By selecting *Window – Assistants – Parasitic Estimates.*
- By selecting *Parasitics/LDE – Create Estimates*.

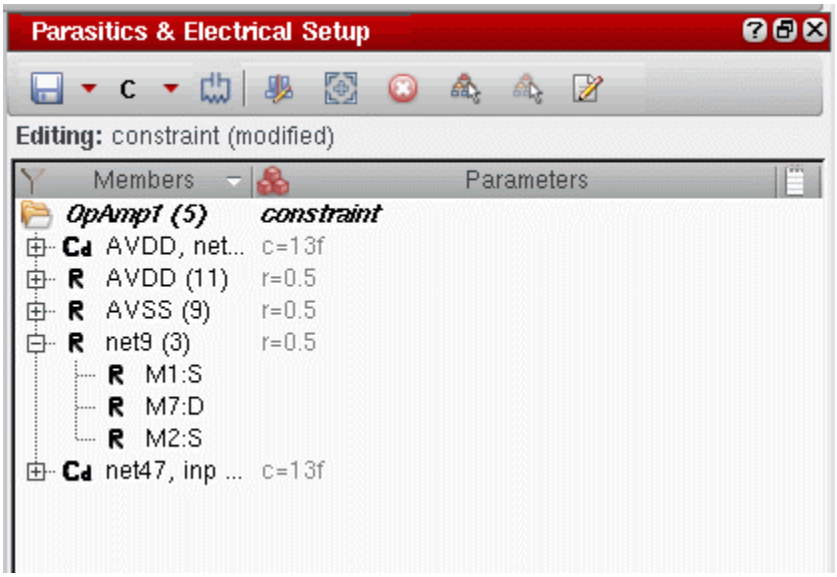


Figure 2-6 The Parasitics & Electrical Setup Assistant

The *Parasitics & Electrical Setup assistant* details the current *Parasitic Estimate* constraints (this is the only constraint type that is listed) which can be expanded to reveal the current constraint *members*.

© 2023
All Rights Reserved.

The Parasitics & Electrical Estimates Assistant Toolbar

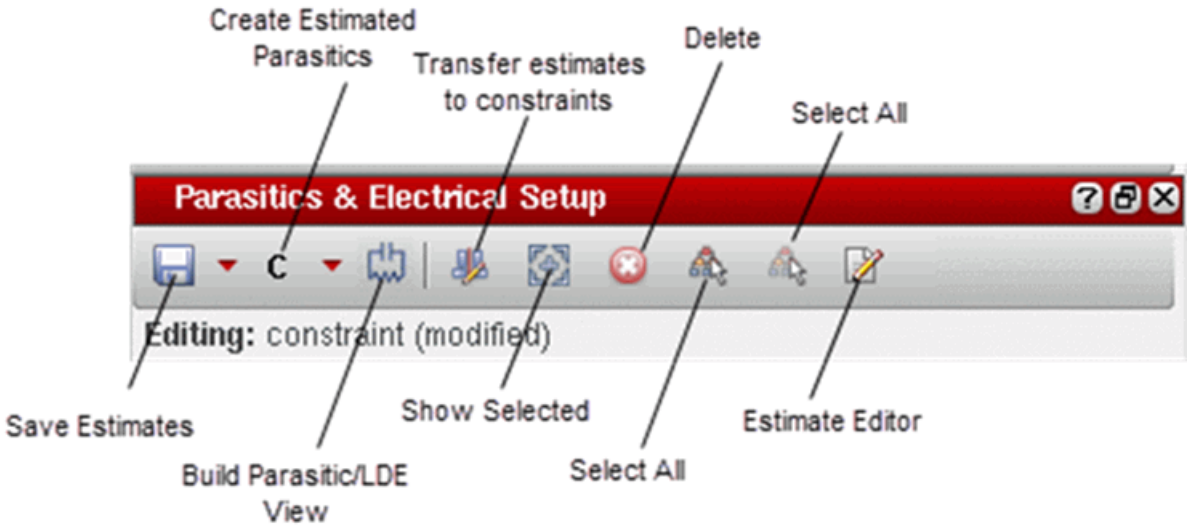


Figure 2-7 Parasitics & Electrical Setup Assistant Toolbar

The *Parasitics & Electrical Setup* assistant toolbar provides the following options:

- *Save Estimates* (see Saving Parasitic Estimates Created in the Current Session)
- *Create Estimated Parasitics* (see Creating Parasitic Estimates) with the following options available in the pull-down:
  - *Create Estimated Capacitance* (see Capacitance Estimates)
  - *Create Estimated Resistance* (see Resistance Estimates)
  - *Create Estimated Inductance* (see Inductance Estimates)
  - *Create Estimated Mutual Inductance* (see Inductance Estimates)
  - *Create Custom Estimated Parasitic* (see Custom Estimated Parasitics)
  - *Create Layout Stitching Estimates for Cell* (see Creating Layout Stitching Estimates for a Cell)
  - *Create Override*
  - *Match Estimate Parameters*
  - *Unmatch Estimate Parameters*
- *Build Parasitic/LDE View* (see Building the Parasitic/LDE View)

© 2023 All Rights Reserved.

- *Transfer estimates to constraints* transfers all the estimates to the Constraint Manager

  Note: Use the `createMaxCapDuringTransfer` environment variable to enable or disable the creation of maximum capacitance constraints when transferring coupling capacitance estimates to constraints.

- *Estimate Editor*
- *Delete* a selected parasitic estimate
- *Select All* parasitic estimates listed
- *Deselect All* parasitic estimates listed
- *Editing / Reading* displays the current mode
- *Show Filters / Customize Constraint Filters* - see Filtering Constraints in the *Virtuoso Unified Custom Constraints User Guide.*

© 2023 All Rights Reserved.

The Parasitics & Electrical Setup Assistant Estimate Editor

Click the *Estimate Editor* icon in the Parasitics & Electrical Setup Assistant and *Filter Editor* in the Parasitic Filters Assistant toolbars to display an editing table where you can edit various parameters associated with each parasitic estimate.

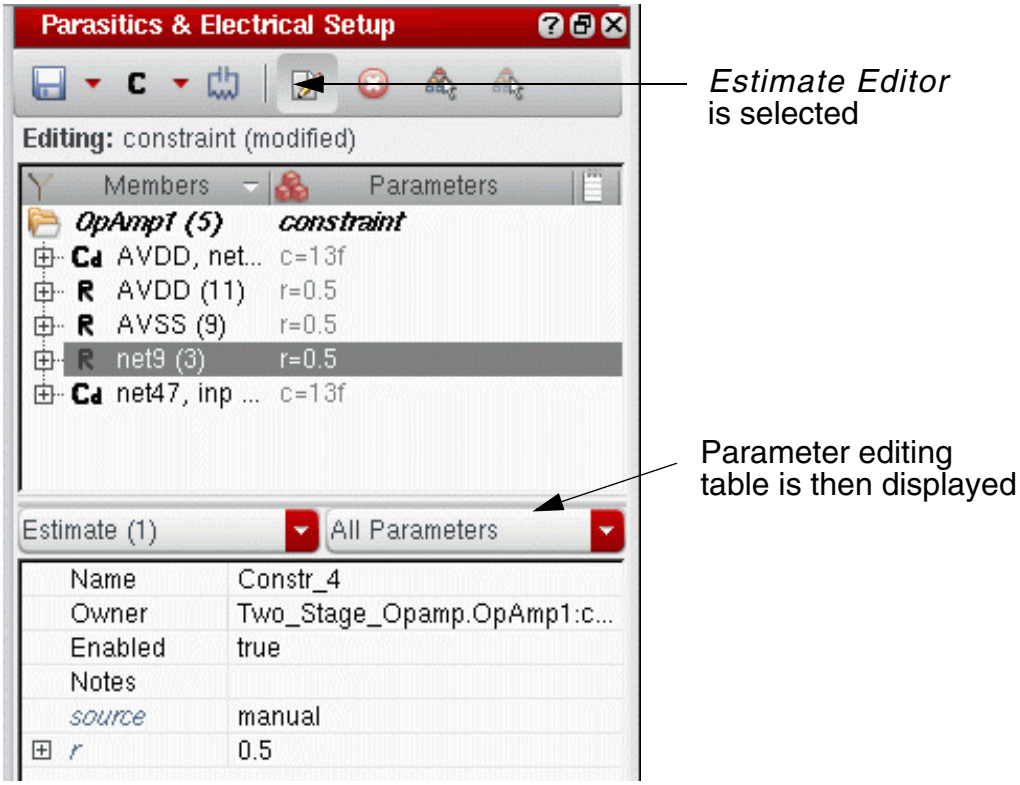


Figure 2-8 The Parasitics & Electrical Setup Assistant with the Estimate Editor Selected

For more information on how to edit parameter values, see The Constraint Editor in the *Virtuoso Custom Constraints User Guide*.

© 2023
All Rights Reserved.

Reverting To Default Values

You can restore default values for fields, for example *Sweep*, by selecting the *Revert to default* icon to the right hand-side of the edit field.

Figure 2-9 Revert to Default Icon

© 2023
All Rights Reserved.

The Parasitics & Electrical Setup Assistant Context-Menu

Right-clicking over the *Parasitics & Electrical Setup* assistant displays a context-menu. From here you can perform a variety of tasks including saving parasitic estimates and controlling object visualization on the design canvas and in the Navigator assistant.

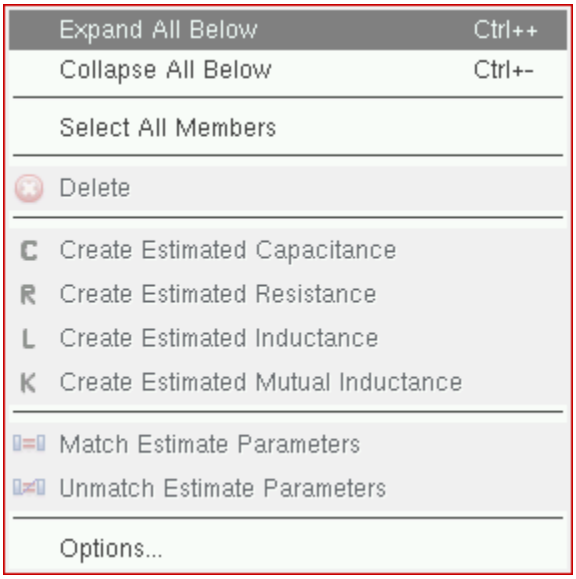


Figure 2-10 Parasitics & Electrical Setup Assistant context-menu

The full contents of the *Parasitics & Electrical Setup* assistant context menu are listed in the table below.

Context-Sensitive Menu Option	Description
Expand All Below	Expands all branches in the *Parasitics & Electrical Setup* assistant to show all estimate member objects.
Collapse All Below	Collapses all branches in the *Parasitics & Electrical Setup* assistant to hide all estimate member objects.
Select All Members	Selects all constraint members that belong to the currently selected (expanded) estimate.
Delete	Deletes the selected constraint member.

© 2023 All Rights Reserved.

Create Estimated Capacitance	Creates a capacitance estimate constraint for the selected objects. For more information see Capacitance Estimates.
Create Estimated Resistance	Creates a resistance estimate constraint for the selected objects. For more information see Resistance Estimates.
Create Estimated Inductance	Creates an inductance estimate constraint for the selected objects. For more information see Inductance Estimates.
Created Estimated Mutual Inductance	Creates a mutual inductance estimate constraint for the selected objects. For more information see Inductance Estimates.
Create Override	Creates an estimate at the current level that overrides one at a lower level. For more information see Constraint Overrides in the *Virtuoso Unified Custom Constraints User Guide.*
Match Estimate Parameters	Creates a match between two or more estimates selected in the *Parasitic Estimates* assistant. **Note:** You cannot match a resistance against a capacitance estimate. Matching will only work between resistances, and only between capacitances. The first selected instance is considered the "`master`" and the second is given a `'match(<master>)'` value. For example, if you select two terminals, `X` and then `Y`, then select *Match Estimate Parameters*, this will mean that in any simulation `Y` will always have the same value as `X`. This will be the case even if `X` has a value determined by a sweep. The parasitic net topology is unaffected. **Note**: You can match a coupledC estimate with a decoupledC estimate.
Unmatch Estimate Parameters	Removes a match and reverts the selected parasitics back to their default values. Option will be enabled if at least one of the estimate selections is currently matched.

© 2023 All Rights Reserved.

Select Inst-Terms	Selects all the terminals for the selected instances to be added to the electrical selection setup. For more information, see Selecting Signals to Save Currents in the *Virtuoso Electrically Aware Design Flow User Guide*.
Select Inst-Terms Within	Selects all the terminals of the instances within the currently selected instances. For more information, see Selecting Signals to Save Currents in the *Virtuoso Electrically Aware Design Flow User Guide*.
Select Inst-Terms Hierarchically	Selects all the terminals for all the instances in the complete design hierarchy. For more information, see Selecting Signals to Save Currents in the *Virtuoso Electrically Aware Design Flow User Guide*.
Options	Displays the Constraint Manager Options form. For more information see Options.

© 2023 All Rights Reserved.

Creating Parasitic Estimates

Important

See *Creating Constraints at Different Design Levels* in the *Virtuoso Unified Custom Constraints User Guide* for further information on creating constraints/ estimates.

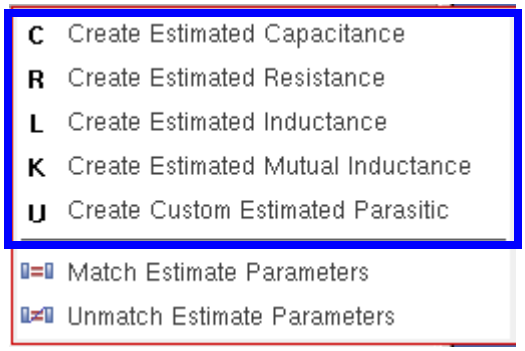
To create parasitic estimates using the *Parasitics & Electrical Setup* assistant:

1. Select the nets that you want to create parasitic estimate constraints for from either the design canvas or the Navigator assistant. Alternatively, you can select instance terminal(s) if you only want to create parasitic estimates for a particular instance's terminal(s).

   Note: Descend into an instance to create a net estimate at that level.

   If, for a given net, there is a parasitic resistance estimate that only contains some of the instance terminals, you can select the remaining instance terminals, or the net, and continue with estimate creation to include the rest of the instance terminals (see also Resistance Estimates).

2. Select the *Create Estimated Parasitics* pull-down from the The Parasitics & Electrical Estimates Assistant Toolbar at the top of the *Parasitics & Electrical Setup* assistant. This will display the following list of parasitic estimates that can be created:



   - *Create Estimated Capacitance* (see Capacitance Estimates).

     Note: If you create a capacitance estimate for a power net, "*Cd*" will be displayed against the estimate in the *Parasitic Estimates* browser to signify decoupled capacitance. Likewise, a "*C*" will be displayed for coupled capacitance for signal net estimates.

   - *Create Estimated Resistance* (Resistance Estimates)

© 2023 All Rights Reserved.

Note: The resistance estimate will only include instance terminals at the current design level. When you view the *Parasitics & Electrical Setup* assistant at a different design level, it will only display those resistance and capacitance estimates that were created at that level or below.

- *Create Estimated Inductance* (Inductance Estimates)

  Note: "*RL*" will be displayed in the *Parasitic Estimates* browser as you cannot have inductance purely on its own.

- *Create Estimated Mutual Inductance* (Inductance Estimates)

Note: See *Creating Constraints at Different Design Levels* which details that constraints/estimates need to be created on a level-by-level basis.

- *Create Custom Estimated Parasitics* (Custom Estimated Parasitics)
- *Create Layout Stitching Estimates for Cell* - For more details, refer to Creating Layout Stitching Estimates for a Cell in the *Virtuoso Electrically Aware Design user guide*.

After you choose which type of parasitic estimate you want to create, it is added to the *Parasitic Estimates* assistant and a default value is assigned to it. If required, you can change the value of an estimate.

You can now proceed to Building the Parasitic/LDE View.

Note: If you specify the estimate value as an expression using variables, after the parasitic/LDE view is built, the presence of those variables in the Data View pane is checked. If not found, the variables are added to the Global Variables list. You can modify the values of these variables from the GUI.

See also Options for Schematics Estimate Mode.

© 2023 All Rights Reserved.

Capacitance Estimates

See also: Creating Parasitic Estimates.

The method used to *Create Estimated Capacitance* for nets/buses is dependent upon the number of nets (bus or non-bus) that have been selected in the design canvas or the Navigator assistant.

Note: Only unique net names are displayed in the Create Estimated Capacitance form pull-down.

- If **one** (non-bus) net is selected, the Create Estimated Capacitance form shown below is displayed.

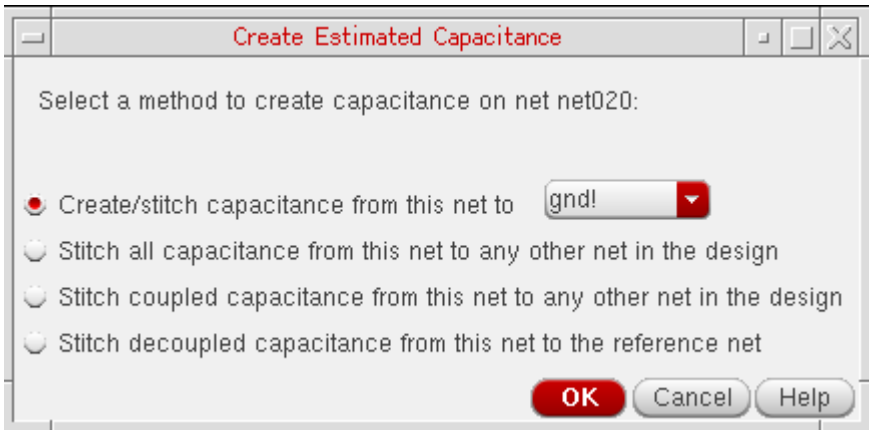


Figure 2-11 Create Estimated Capacitance form

Using this form, you can create or stitch capacitance between the net you selected and a reference net chosen on this form. The various options displayed on this form are listed below.

– *Create/stitch capacitance from this net to <supply-net>*: Creates a capacitance between the selected net and a supply net you choose from the drop-down list.

– *Stitch all capacitance from this net to any other net in the design*: Stitches all the capacitance from the selected net to any other net in the design. For example, if you select netA and select this option, any capacitance available in the layout view to netA is stitched. You do not need to explicitly stitch capacitance between the available net pairs in the design, for example, netA and netB; netA and netC; and netA and gnd!. Every capacitance available in the layout that is connected to netA will be included.

© 2023 All Rights Reserved.

– *Stitch coupled capacitance from this net to any other net in the design*: Stitches all the coupled capacitance from the selected net to any other net in the design.

– *Stitch decoupled capacitance from this net to the reference net*: Stitches only the decoupled capacitance from the selected net to the reference net.

Note: The options that are used to stitch a capacitance to other nets in the design are valid only when a layout view is available.

- If **two** (non-bus) nets are selected, estimated capacitance will be created between these two nets.
- If **three or more** (non-bus) nets are selected, the Create Estimated Capacitance form is displayed as shown below.

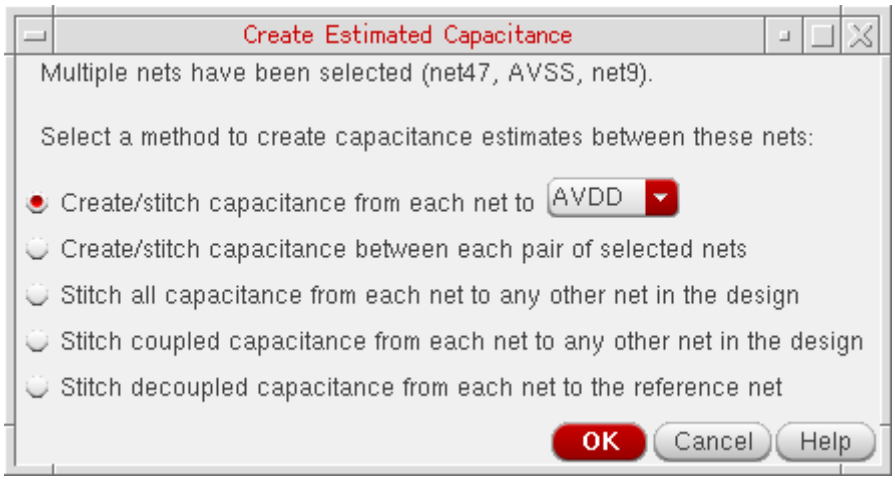


Figure 2-12 Create Estimated Capacitance form with three net selection

The various options displayed on this form are listed below.

– *Create/stitch capacitance from each net to <supply-net>*: Creates a capacitance between each selected net and a supply net you choose from the drop-down list.

– *Create/stitch capacitance between each pair of selected nets*: Creates a capacitance between every possible pair of the selected nets.

– *Stitch all capacitance from each net to any other net in the design*: Stitches all the capacitance from each selected net to any other net in the design.

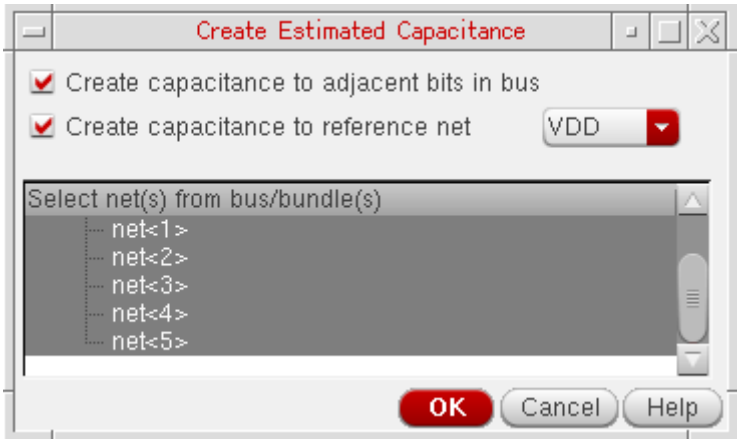
– *Stitch coupled capacitance from each net to any other net in the design*: Stitches all the coupled capacitance from each selected net to any other net in the design.

© 2023 All Rights Reserved.

– *Stitch decoupled capacitance from each net to the reference net*: Stitches only the decoupled capacitance from each selected net to the reference net. The reference net being either one of the selected nets or a supply net.

Note: The options that are used to stitch a capacitance to other nets in the design are valid only when a layout view is available.

- If **one bus net** has been selected, the Create Estimated Capacitance form allows you to create capacitance between adjacent bits in the bus and/or from each selected bit to a reference net (the possible reference nets being each bus bit or the supply nets from the Setup form).

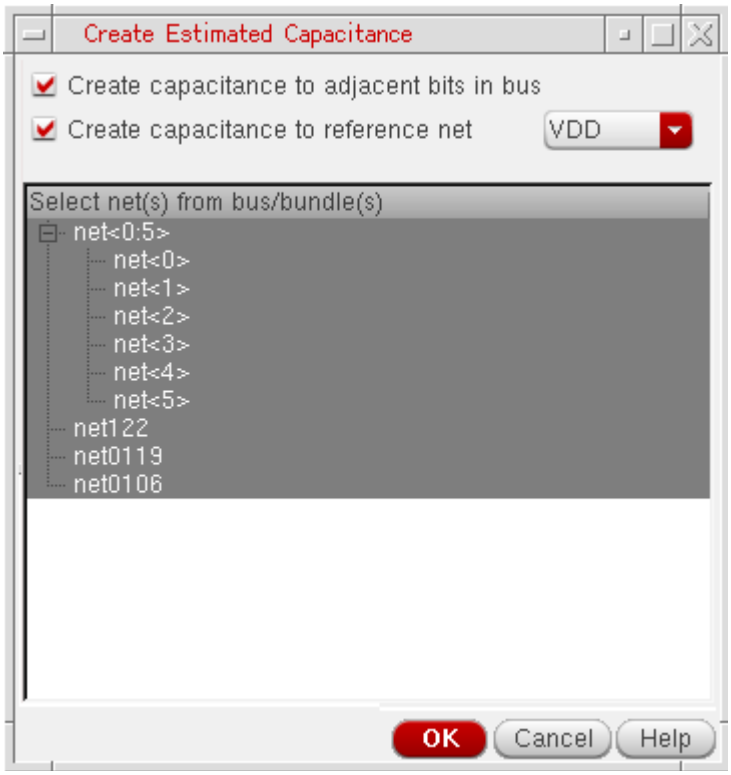


When creating capacitance to adjacent bits, the Create Estimated Capacitance form creates an estimate between selected bits and their adjacent bits.

Note: By default, the whole bus will be selected.

© 2023
All Rights Reserved.

- If you have a **multiple selection of bus and non-bus nets**, the Create Estimated Capacitance form will display each bus/net. The possible reference nets will include all bus bits and non-bus nets as well as supply nets from the Setup form.



- Creation of estimates to adjacent bits will only occur within each bus. Inter-bus estimates are not created.

© 2023
All Rights Reserved.

Resistance Estimates

See also: Creating Parasitic Estimates.

The *Create Estimated Resistance* option creates an estimate constraint for each selected net, instance terminal, or pin.

Selecting a net and choosing *Create Estimated Resistance* creates a resistor from a common central node for the net to each of its instance terminals and pins. This creates a parasitic network known as a star network.

The parameter *r* represents the resistance associated with each instance terminal or pin. A common value can be set for all parasitic resistors on a net by selecting the net name in the Parasitics & Electrical Setup assistant assistant and editing the *r* value in the parameter editing table at the bottom of the assistant. The value can be overridden for individual terminals or pins by selecting the appropriate constraint member and editing its associated *r* value in the parameter editing table.

It is not always desirable to create a resistance for every instance terminal and pin on a net. Variations of the star network can be made as follows:

- You can create a single resistance between a pin or instance terminal and the rest of the net. For this, select only the pin or instance terminal and choose *Create Estimated Resistance*.
- You can delete a member of the estimate constraint to remove the parasitic resistor for that instance terminal or pin and to connect it directly to the central node of the star network.

Note: The resistance estimate can include the instance terminals and pins at the current design level only.

© 2023 All Rights Reserved.

Inductance Estimates

See also: Creating Parasitic Estimates.

You can use the Parasitics & Electrical Setup Assistant to create both inductance and mutual inductance estimates using the *Create Estimated Inductance* and *Create Estimated Mutual Inductance* options, respectively, from the *Create Estimated Parasitics* pull-down on The Parasitics & Electrical Estimates Assistant Toolbar. Both of these options can have a number of use models, for example:

Creating *estimated (self) inductance* use models:

1. You can select a **net** in the schematic canvas (or using the *Navigator* assistant) and then select the *Create Estimated Inductance* option. In this case, the L estimate members will be all the instance terminals of the nets. R estimates will also be created, in series at each instance terminal when L is added (if not already present).
2. Alternatively, you can select **instance terminals** in the schematic and then choose the *Create Estimated Inductance* option. Here, the L estimate members would be the selected instance terminals only. If, for a net, an L estimate already exists, but with only a subset of its instance terminals, it is possible to extend the L estimate for the same net by selecting additional instance terminals on the schematic canvas, or in the *Navigator*, and then choosing the *Create Estimated Inductance* option in the browser. All of the newly selected instance terminals would now be added to the existing L estimate. R estimates will also be created, in series at each instance terminal when L is added (if not already present).

   Note: If you select instance terminals for more than one net, multiple L estimates will be created, each corresponding to a specific (unique) net.

Creating *estimated mutual inductance* use models:

1. You can select **two different inductors** (two members from one or more L estimates in the *Parasitics & Electrical Setup* assistant browser) and then choose the *Create Estimated Mutual Inductance* (*K*) option from the *Create Estimated Parasitics* pull-down. In this case, an estimated K will be created with the chosen inductors (the members of the L estimate) being its members.
2. You can select **more than two inductors** (more than two members from one or more L estimates in the *Parasitics & Electrical Setup* assistant browser) and then choose the *Create Estimated Mutual Inductance* (*K*) option.

   When you select more than two inductors, to create a K estimate, a Create Mutual Inductance form is displayed. From here, you can specify what inductor is to be the *reference* inductor. That is, where mutual inductance will be created between the reference L and all other inductors.

© 2023 All Rights Reserved.

3. You can select **two instance terminals** and then click the *K* option. Here, if any of the selected instances are not existing members of an L estimate, an L estimate will first of all be created, using the selected instance terminals, prior to creating a K estimate for the two instance terminals. R estimates will also be created, in series at each instance terminal when L is added (if not already present).

4. You can select **more than two instance terminals** on the schematic canvas, or from the *Navigator*, and then click the *Create Estimated Mutual Inductance* (*K*) option. Here, L estimate will be created for all the instance terminals that are not already members of a L estimate, prior to the creation of multiple K estimates.

   Again, the Create Mutual Inductance form is displayed to allow you to specify the reference inductor. Mutual inductance estimates will then be created between the reference L and the other inductors.

   Note: If you *Cancel* this form, neither estimated L nor estimated K will be created.

Figure 2-13 Create Mutual Inductance Form

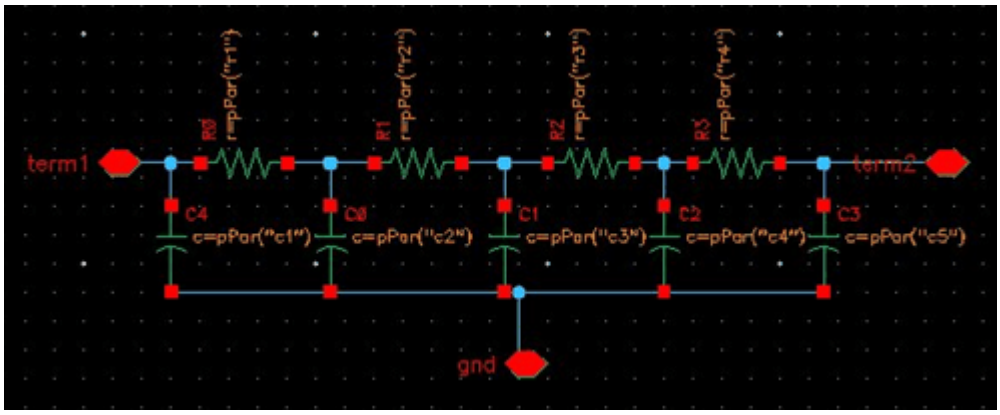
© 2023
All Rights Reserved.

Custom Estimated Parasitics

See also: Creating Parasitic Estimates.

The *Create Custom Estimated Parasitics* command on Parasitics & Electrical Setup Assistant allows you to use custom parasitic models in your design. These models are different from the standard R, C, L, or K types of parasitics, which are not always accurate enough to depict the required parasitic model. You can use custom parasitics to specify a custom topology that more closely represents the desired parasitic network on any given net.

The following example shows how a custom topology can be created for a two-terminal net. The parasitic network is created in a new schematic cellview. It defines a resistance network between two pins which represent the terminal connections on the net. Capacitance is connected from various nodes inside the network to a third pin. This pin is used to connect the capacitors to an external reference net when the model is placed.



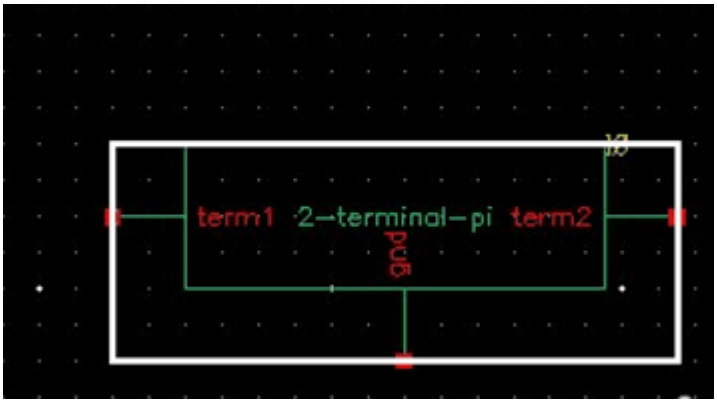
Note: It is not mandatory to define the implementation of a parasitic cell in the schematic view. Instead, you can define a parasitic cell in any HDL.

You need to associate your parasitic cell with a symbol view that has pins corresponding to the external connections required to be created. The symbol view is used for the following purposes:

- To specify which custom parasitic model to use for a net
- To replace the net in the Parasitic/LDE view. When netlisting the Parasitic/LDE view, the net will be replaced with an instance representing the parasitic network.

© 2023 All Rights Reserved.

A symbol for the example parasitic cell shown above can be created as given below.



Some important requirements for the symbol view of a custom parasitic cell are listed below:

- The number of terminals on the custom parasitic model must be equal to or greater than the number of terminals on the net to which it is to be connected.
- The custom parasitic model can be parameterized using CDF parameters. The CDF definitions should be created on the parasitic cell. The parameters will then be displayed in the Estimates Editor where the values can be modified for each instance of the model. In the implementation of the model, pPar expressions should be used to pass the parameter values down to instances inside the parasitic cell.

The following topics explain how to create and configure a custom parasitic estimate:

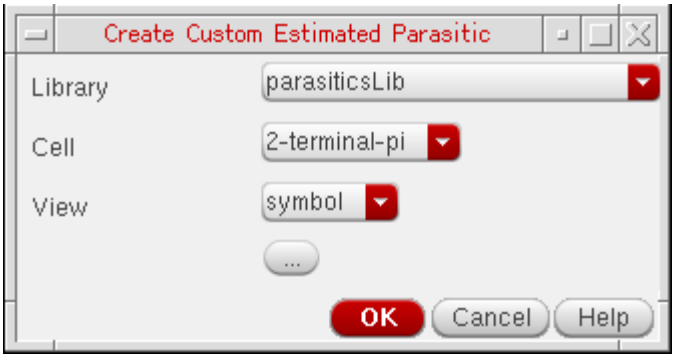
- Creating Custom Parasitic Instances
- Configuring Custom Parasitic Instances

Creating Custom Parasitic Instances

To create a custom parasitic instance, select one of more nets on the schematic and choose the *Create Estimated Parasitics – Create Custom Estimated Parasitics* command on the Parasitics & Electrical Setup Assistant.

© 2023 All Rights Reserved.

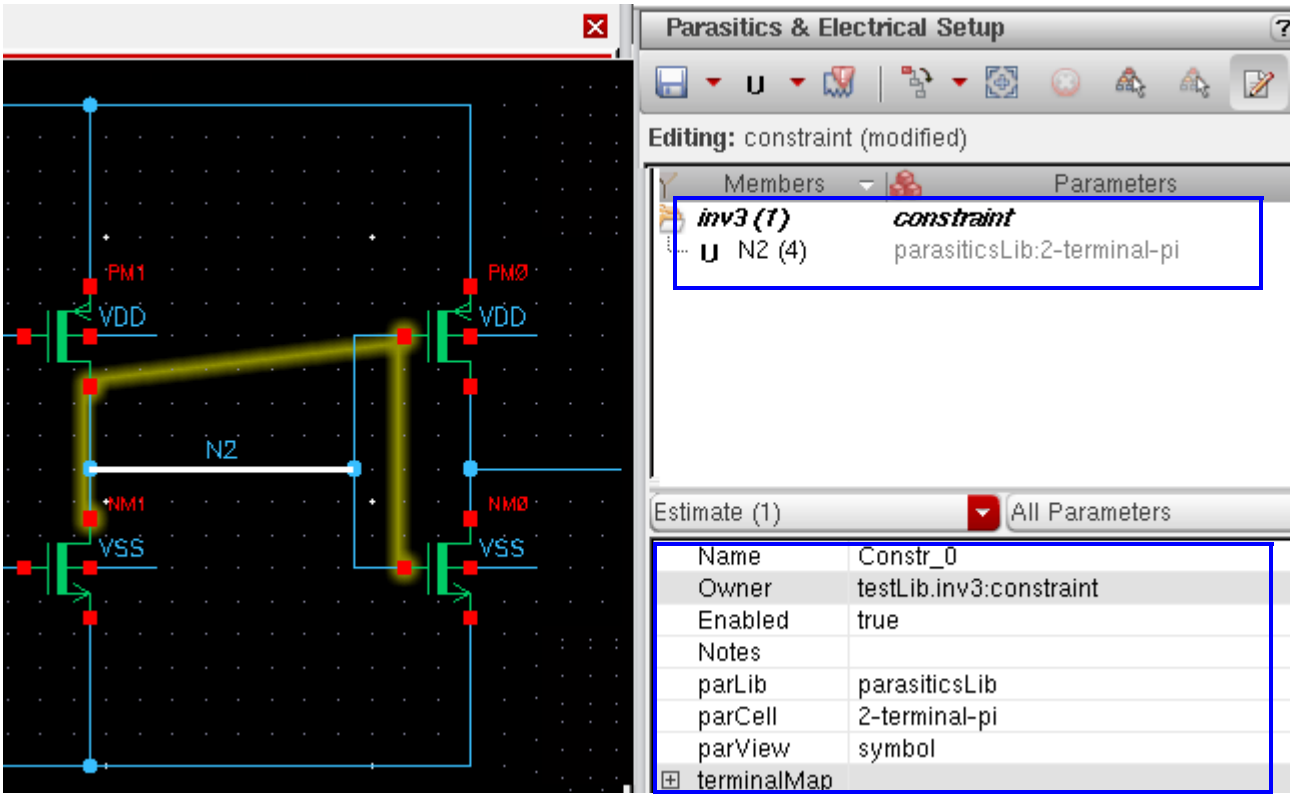
The **Create Custom Estimated Parasitic** form is displayed. In this form, select the library, cell, and view name of the custom parasitic cell, as shown in the figure below.



Alternatively, you can click the browse (...) button to browse and select the required cellview.

Click *OK* to close the form.

An instance of the custom cell is created in your design and is listed in the Parasitics & Electrical Setup assistant, as shown in the figure below.



© 2023
All Rights Reserved.

Important

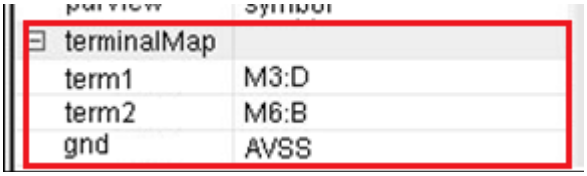
If a net has a custom parasitic instance specified on it, no other parasitic can be inserted on it.

Configuring Custom Parasitic Instances

Click the parasitic instance in the assistant and open the Estimate Editor to display its properties. Observe that the library, cell and view name of the parasitic cell are displayed in the *parLib*, *parCell*, and *parView* properties of the instance. If required, you can select a different library, cell, and view.

You can now change the values of the properties to configure the instance.

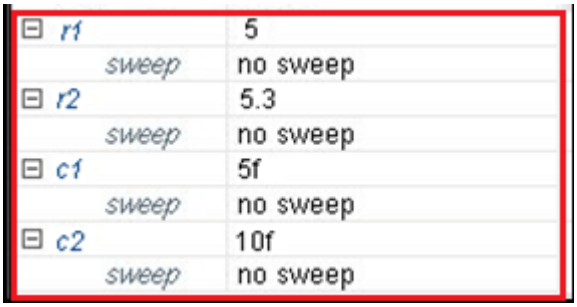
- terminalMap: The *terminalMap* property lists all the terminals in the selected net(s). Expand *terminalMap* to view the default mapping of the net terminals with the terminals of the parasitic instance. You can reconfigure this mapping as required.



You can either type in the net or terminal name or select a net or terminal on the schematic.

Note: It is important to map all the terminals of the symbol.

- CDF properties: The editable parameters of a custom parasitic are shown in the Estimate Editor. You can edit the values of these parameters and specify scalar or sweep values.



© 2023
All Rights Reserved.

Important

An example parasitic cell library and a guiding presentation is available at the following path in your Virtuoso installation:
your_install_dir/tools/dfII/samples/parasitic/automaticEstimates
You can use this library and to create a sample custom parasitic.

© 2023 All Rights Reserved.

Entering Parasitic Sweeps

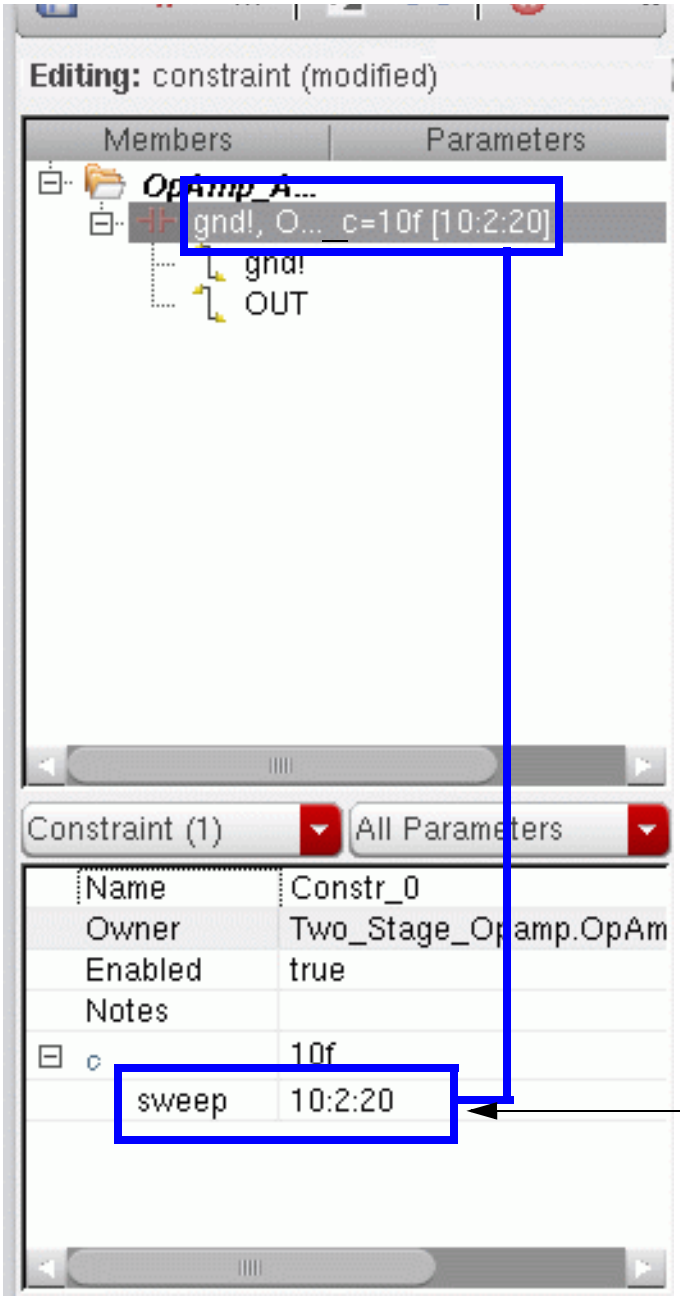
Important

You must select the *Estimate Editor* icon in the *Parasitics & Electrical Setup* assistant toolbar to display the parameter editing table.

Sweeps are set in the parameter editing table in Parasitics & Electrical Setup Assistant by expanding the *l, r*, *c* or *k* parameters.

Note: Any instance terminal rows, in the *Parasitics & Electrical Setup* assistant editor section, will be left blank unless the *r* or *l* value has been explicitly overridden.

© 2023 All Rights Reserved.



The value of 10f will only be used when parasitic sweeps are disabled in the simulation

Capacitance parasitic sweep
start: 10f
step: 2f
stop: 20f

Note: If a sweep has not been set, *"no sweep"* will appear next to the *sweep* sub-parameter.

Figure 2-14 Parasitic Sweeps set in the Parasitics & Electrical Setup Assistant

© 2023
All Rights Reserved.

The syntax is identical to that used for sweeps in global variables and parameters in ADE Explorer or ADE Assembler. For example, you can provide a comma-separated list of values: "`10, 15, 25`" or specify start, step, and stop values: "`10:2:20`".

Specifying a range or sweep will create parasitic parameters in the *Parasitics* tab of the *Variables and Parameters* assistant (for information about this assistant, see the Virtuoso Analog Design Environment XL User Guide) and allows you to sweep parasitic values during simulation runs. Parasitics can be swept at the same time as, or separately from, device parameters. See The Parasitic Mode Toolbar for running parasitic sweeps.

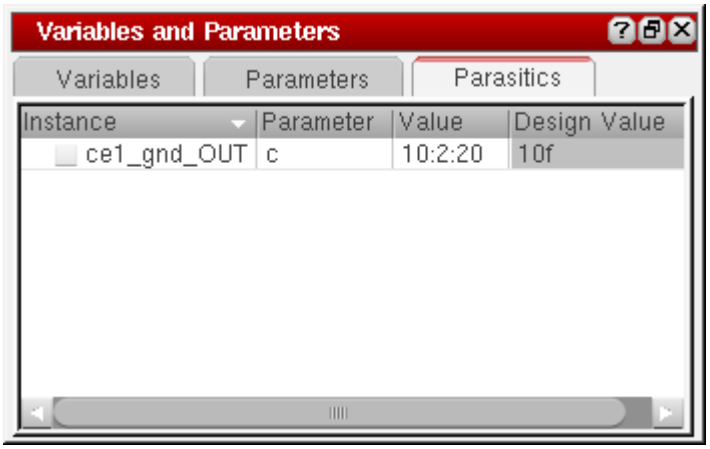
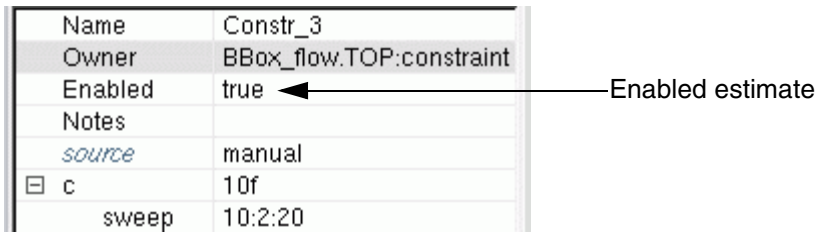


Figure 2-15 Parasitic Sweep Displayed as Parasitic Parameter in Variables and Parameters Assistant

Note the following:

- After specifying a sweep, you must run the *Build Parasitic/LDE View* option in the *Parasitic Estimates* assistant to view the parasitic parameters in the *Parasitics* tab of the *Variables and Parameters* assistant.
- The *Parasitics* tab of the *Variables and Parameters* assistant displays only the sweeps specified for the estimates that are enabled (*Enabled* field set to *true*) in the *Parasitic Estimates* assistant.



© 2023
All Rights Reserved.

Figure 2-16 Example of Enabled Estimate

Restoring Default Sweep Values

You can restore default sweep values by selecting the *Revert to default* icon to the right hand-side of the edit field.

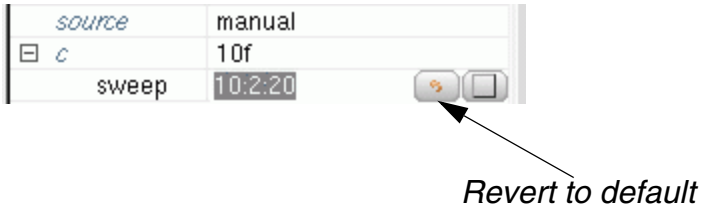


Figure 2-17 Revert to Default Icon

Note: When estimates are matched their sweeps are also matched. This means that matched estimate components will be swept together during simulation.

Enabling and Disabling Parasitic Sweeps

You can enable/disable parasitic sweeps in the *Parasitic Estimates* assistant or the *Parasitics* tab of the *Variables and Parameters* assistant.

Enabling and Disabling Parasitic Sweeps in the Parasitics & Electrical Setup Assistant

You can enable/disable parasitic sweeps directly in the *Parasitic Estimates* assistant (without having to use the *Parasitics* tab of the *Variables and Parameters* assistant).

To do this, first of all ensure that:

- The parasitic mode is set to *Schematic Estimates* on The Parasitic Mode Toolbar.
- The parasitic estimate (or a member of an estimate that does not inherit a sweep value from its parent) under consideration has a sweep recorded in the estimated view.

  Note: This is achieved by running the *Build Parasitic/LDE View* option in the *Parasitic Estimates* assistant.
- The *r*, *c*, *l*, or *k* parameter of a selected estimate, or one of its members, is expanded in the parameter editing table (so that you can view/edit the sweep parameter value).

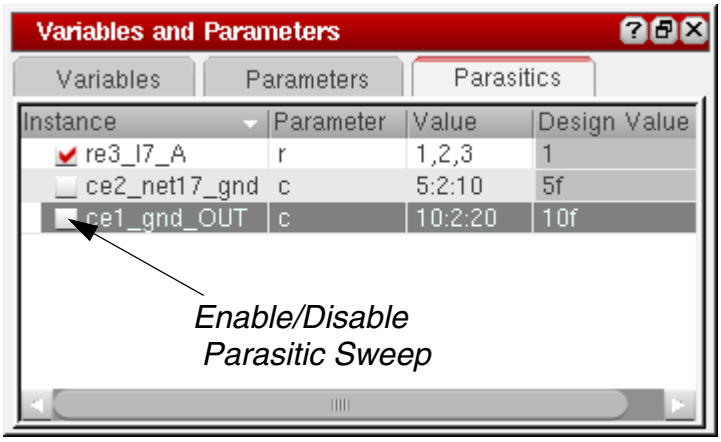
© 2023 All Rights Reserved.

From here, you can now enable/disable the current estimate sweeps by selecting or deselecting *Enable/Disable Parasitic Sweep...* from the parameter editing table context-menu or the adjacent check box.

Figure 2-18 Check box for Enabling/Disabling Parasitic Sweep

Enabling and Disabling Parasitic Sweeps in the Parasitics Tab of the Variables and Parameters Assistant

You can enable/disable parasitic sweeps in the *Parasitics* tab of the *Variables and Parameters* assistant by selecting or deselecting the check box next to each sweep parameter.



© 2023
All Rights Reserved.

Saving Parasitic Estimates Created in the Current Session

Click the *Save Estimates* icon in the *Parasitic Estimates* assistant toolbar to save (or update) the parasitic estimates that have been created or edited in the current session.

Note: You can also use the *File – Save (Constraint)* option on the main menu bar.

For more information on the other options available in the *Save Estimates* pull-down see Save Constraints in the *Virtuoso Unified Custom Constraints User Guide*.

Note: *Save a Copy* will also save parasitic estimate and parasitic filter information.

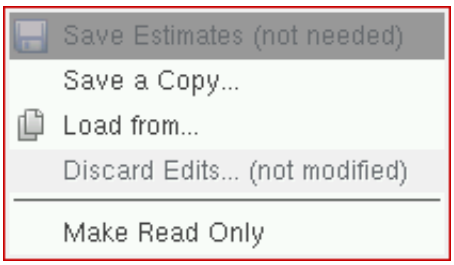


Figure 2-19 Save Estimates pull-down options in the Parasitic Estimates assistant toolbar

© 2023 All Rights Reserved.

Building the Parasitic/LDE View

Once you have finished Creating Parasitic Estimates you can now proceed to building a parasitic/LDE view.

Note: After creating one or more new parasitic estimates, the *Build Parasitic/LDE View* icon will indicate, with an exclamation mark, that a new parasitic/LDE view is required to be built to accommodate the new estimates.

Click the *Build Parasitic/LDE View* icon on the *Parasitics & Electrical Setup* assistant toolbar to display the **Build Parasitic/LDE View** form.

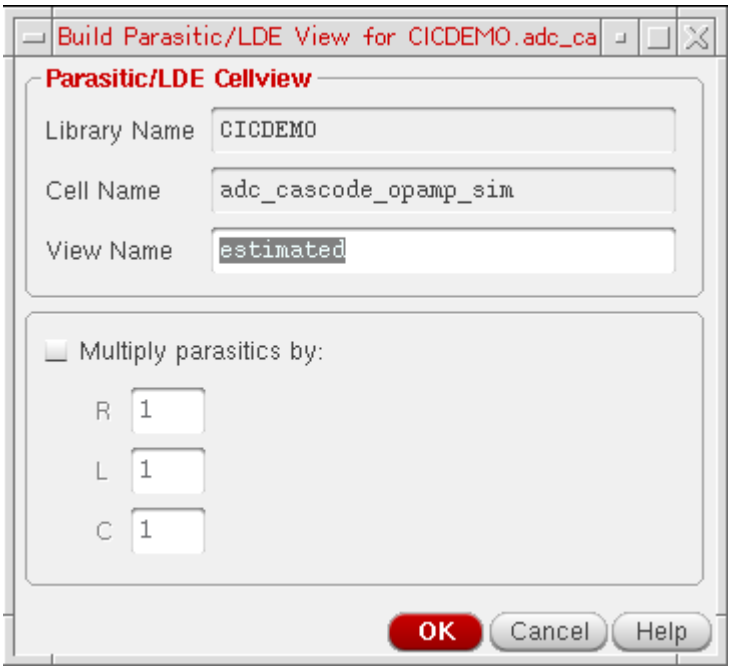


Figure 2-20 The Build Parasitic/LDE View Form

To build a parasitic/LDE view:

1. Specify the *View Name* for the parasitic/LDE view to be generated.

   The default is `estimated` but this can be changed, allowing you create multiple parasitic/LDE cellviews.

Caution

Using resistance, capacitance, or inductance values from a layout view to creates estimates checks out additional licenses.

© 2023 All Rights Reserved.

Note: The *Library Name*, *Cell Name*, and *View Name* option settings are initially automatically infilled from the *Design Under Test* settings in the Parasitic/LDE Setup form (Setting Up and Using Parasitics). If you change the *View Name* in the Build Parasitic/LDE View form, this will not however update the setup form nor, consequently, the view that is used when reporting estimates (see the Parasitic Report assistant.

2. Optionally, set scale factors for each parasitic type by activating the *Multiply estimated parasitics by* option, and entering scale values for *R*, *L* and *C*.

   Each scale factor value entered will be applied globally to all parasitic estimates of the relevant type. A scale factor of “0” should not be entered.

   Note: Default values for the Build Parasitic/LDE Schematic form can be controlled by the following `.cdsenv` entries:

   ```
   msps.estimates viewName string "estimated"
   msps.estimates scaleR float 1.0
   msps.estimates scaleL float 1.0
   msps.estimates scaleC float 1.0
   ```

3. Click the *OK* button to build the parasitic/LDE view.

4. Parasitic aware design will now create a flattened view of the schematic hierarchy, inserting the (scaled) estimates. The new parasitic/LDE view will be of a schematic view type. After the parasitic/LDE view is created, the tool displays the following message confirming the creation of the view and listing the parasitics found.

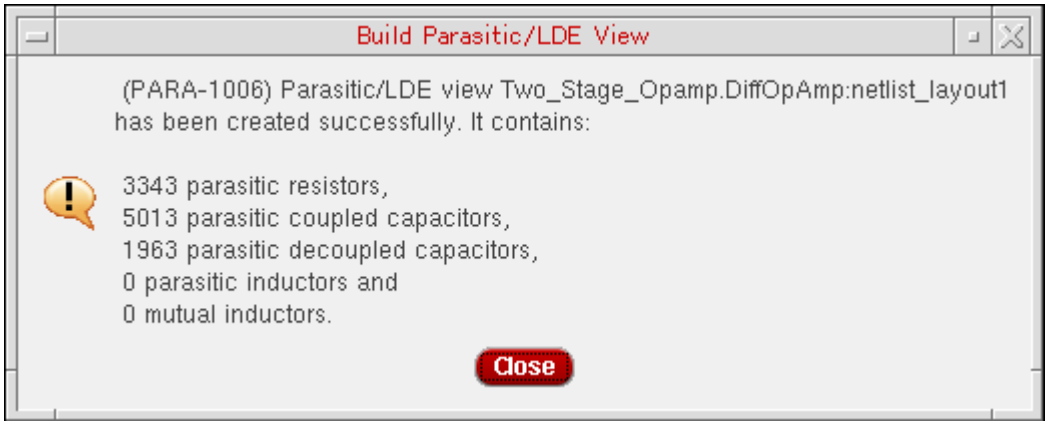


Figure 2-21 Build Parasitic/LDE View Successfully Created

© 2023
All Rights Reserved.

Tip

Before building the parasitic/LDE view, you can set the reduceParallelCaps environment variable to `t` to search for parallel capacitors in the parasitic/LDE view and to merge them into a single device.

Note: The parasitic/LDE view is used for netlisting purposes only and does not display a usable schematic if opened.

Simulating and Probing Results

To run simulations using the generated parasitic/LDE view, you must change your configuration or create a new configuration view. Once simulation has completed you can then probe the results. This task can also be performed using the schematic testbench, via a switch view list.

Note: See also The Parasitic Mode Toolbar. This toolbar can be used to bind all tests to the new parasitic/LDE view for simulation.

As the flattened schematic is only used for netlisting you will have to probe using the original schematic. This involves probing out-of-context where parasitic aware design will map between the design and estimated schematics.

© 2023 All Rights Reserved.

Parasitic Stitching

Virtuoso Parasitic Aware Design can be used to create estimates directly based on an existing layout, of a similar design, where an existing Quantus QRC extracted view can have its parasitics stitched into the parasitic/LDE view.

As parasitic stitching involves a pre-layout simulation flow, the extracted view being from an existing, similar design, you must choose a view that models the parasitics with the desired accuracy (the stitching process will not modify parasitics to compensate for differences in m-factored devices, transistor fingers, or device sizes).

Note: You must also provide schematic mapping information between the two designs if they do not match (the equivalent nets and instances).

Parasitic stitching provides for:

- The insertion of extracted parasitics, for critical nets, into an parasitic/LDE view. This allows you to choose to model parasitics using a net from an extracted view, leading to more accurate modelling where parasitic topology is critical.

  Note: Despite not being able to currently perform detailed editing of a parasitic network, stitching still provides greater accuracy at the cost of some flexibility.

- A choice of parasitic types to be stitched. For example, where a parasitic net has resistance and capacitance, you may choose to stitch only resistance, only capacitance, or both. The only restriction is that if inductance is stitched, resistance must also be stitched.

  Note: There is no explicit control over stitching of mutual inductance. Mutual inductors are included automatically where both contributing inductors have been stitched into the parasitic/LDE view. For more details on how mutual inductance is stitched based on the L estimates, refer to Important Points to Consider for Parasitic Stitching of Mutual Inductance.

- Limited reduction of stitched parasitics is available using a filtering *threshold*, similar to that of the refined flow. Here, you can change the detail in the parasitic net by modifying the thresholds and rebuilding the parasitic/LDE view.

  Note: You cannot however manually edit the values or modify the net topology.

- Reduction can also be performed using *lumped* resistance and inductance between Steiner points. Here, two or more PI models that are connected in a series are combined by adding the R and L values and replacing multiple segments with a single segment.

  Note: Choosing to *lump* the values will reduce the number of parasitics without ignoring any individual parasitic value. Some accuracy will however be sacrificed for performance by reducing the number of parasitic instances.

© 2023 All Rights Reserved.

- Although stitched parasitics cannot be swept, it is possible to sweep device parameters and run a sensitivity analysis in the presence of stitched parasitics.
- The mixing of stitched estimates with manual estimates. You cannot have a stitched and manual estimate of the same type on the same net. However, you can mix estimates where they are of different types. For example:
  - You may stitch the R and L for a net, but place a manual decoupled capacitance estimate for the same net. In this case, the manual C will be attached to the stitched network at an arbitrary point.
  - Alternatively, you may create a manual R and L estimate and stitch the parasitic capacitance. In this case, the stitched C will be connected to the central node of the manual estimate's star network.

For more details, refer to the following topics:

- Setting Parasitic Stitching
- Important Points to Consider for Parasitic Stitching of Mutual Inductance
- Important Points to Consider for Parasitic Stitching of Resistance

© 2023 All Rights Reserved.

Setting Parasitic Stitching

To set parasitic stitching:

1. Select the parasitic estimate for the net(s) to be stitched to in the *Parasitics & Electrical Setup* assistant browser.
2. Expand the *source* parameter in the parameter editor and select *extracted view* or *layout view*.

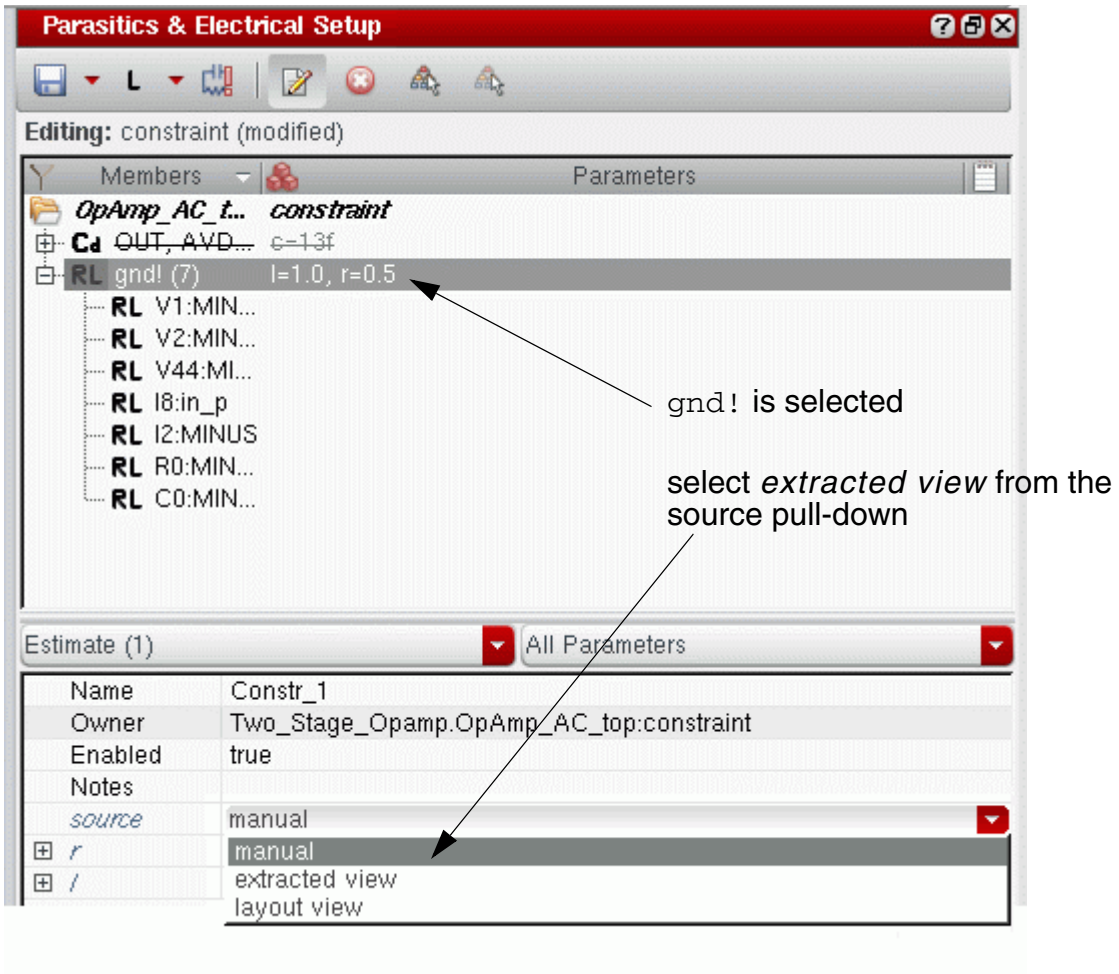


Figure 2-22 Specifying Parasitic Stitching

© 2023
All Rights Reserved.

Selecting *extracted view* as the *source* will add further stitching parameters to the parameter list.

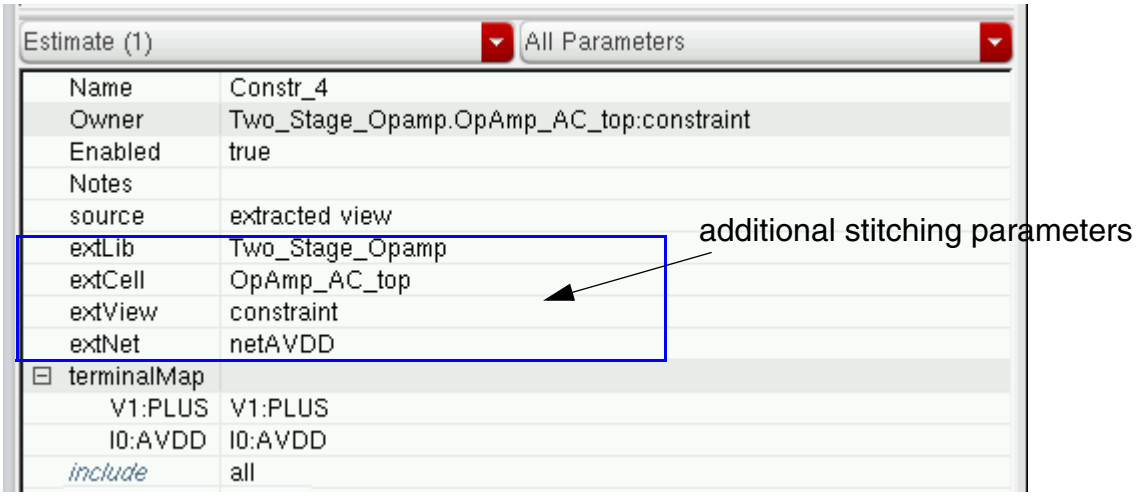


Figure 2-23 Selecting Extracted View as Source Adds Extra Stitching Parameters

Similarly, if you select *layout view* as a source for parasitic stitching, four additional parameters, *layLib*, *layCell*, *layLib*, and *extCorner* appear in the parameters list.

Note: Stitching to a layout view is useful in the EAD flow. For more information, refer to *Virtuoso Electrically Aware Design Flow Guide*.

3. If you have chosen to stitch the parasitic to an extracted view, set the *ext* parameter values as required.

   - *extLib*: list of available libraries
   - *extCell*: lists the cells in the selected library
   - *extView*: lists the available extracted views

     Note: All of the above must point to an Quantus QRC extracted view that exists.
   - *extNet*: lists all (for L or R only) extracted view net names (converted to schematic namespace). The *extNet* parameter value must exist in the extracted view.

     Similarly, if you select *layout view* as a source for parasitic stitching, set the following parameters as required:

© 2023 All Rights Reserved.

- *layLib*: name of the library that contains the layout cellview
- *layCell*: name of the cell that contains the layout view
- *layLib*: name of the layout view
- *extCorner*: name of the extracted corner

4. (for C only) Set the *netMap* from a list of pairs mapping net members to nets in the extracted view.

   Note: Extracted view stitching parameters for C differ from R and L in that a *netMap* option is available rather than an *extNet* option. There is also no *terminalMap* option for capacitances. Also, the *netMap* parameter is not available when you stitch an estimate to a layout view.

5. Set the *terminalMap* terminal values as required (for L or R only). The instance terminals must exist in both the schematic and extracted views and have complete mapping.

   This parameter field lists pairs mapping the current design's terminals to the extracted design's terminals. Both elements of the pair detail the instance and terminal name in the schematic namespace, separated by a colon.

   Note: When stitching R or L parasitics, the entire hierarchical schematic net is replaced from the current schematic down. The *terminalMap* parameter contains an entry for each instance terminal on all non-hierarchical instances connected to the net at the current design level and below. In addition, any estimates on the same net in lower levels of hierarchy will be ignored. Also, the *terminalMap* parameter is not available when you stitch an estimate to a layout view.

6. Set the *include* parameter value as required.

   Note: Selecting *threshold* will add a further parasitic type (for example *R>*) parameter value field.

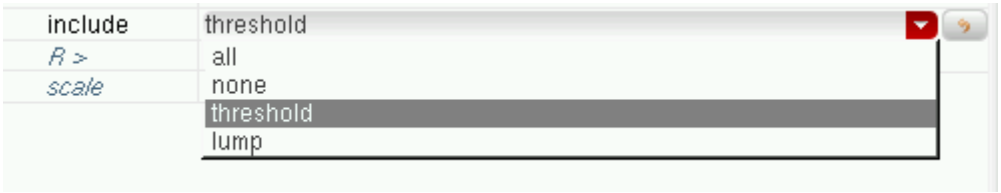


Figure 2-24 Include Parameter Value Options

7. Optionally, set the numeric *threshold* value where parasitics with values below the threshold will not be stitched into the parasitic/LDE view.

© 2023 All Rights Reserved.

8. Set the *scale* parameter value as required.

   The scale value is applied to each parasitic instance stitched by estimate. This scale value works in addition to the *Multiply parasitics by* fields in the Build Parasitic/LDE View form. For example, if you set the R multiply field in this form to `2`, and the scale value in an R stitching estimate to `1.5`, the R parasitics stitched by this estimate will be scaled by a factor of `3`, which is 2 * 1.5.

Important

For information on setting parasitic stitching using SKILL, see `parModelCreateNetL`, `parModelCreateNetR`, and `parModelCreateNetC`.

Including Instances that are Ignored in Stitched View

During netlisting, some instances are ignored and do not appear in the stitched view because the `lvsIgnore` property may be set to `true` for these instances. You can stitch these instances by setting the ignoreLVSInstForStitching environment variable to `t` and thereby, ignoring the `lvsIgnore` property. Set the `ignoreLVSInstForStitching` environment variable to `nil` to ignore the instances that have the `lvsIgnore` property set for them.

Ignoring Instances that are Included in Stitched View

During netlisting, some instances that have the `lxRemoveDevice` property set to `true` are ignored and do not appear in the stitched view. You can stitch these instances by setting the lxRemoveDeviceForStitching environment variable to `nil` and thereby, ignoring the `lxRemoveDevice` property. Set the `lxRemoveDeviceForStitching` environment variable to `t` to ignore the instances that have the `lxRemoveDevice` property set for them.

Important Points to Consider for Parasitic Stitching of Mutual Inductance

When stitching L estimates, the following points must be considered:

- When creating stitched L estimates at the top level of the DUT, any K between those Ls will also be stitched. If you use an occurrence estimate to stitch L for nets inside a named sub-block, only L and K for that occurrence of the cell will be stitched. L and K will not be stitched for other occurrences of the same cell.
- When creating stitched L estimates in a sub-cell schematic under the DUT, the L and any K between them will be copied for each occurrence of the sub-cell in the DUT. For example, if the DUT has four instances of the cell `vco2phase`, stitching L for net `n3` by

© 2023 All Rights Reserved.

creating an estimate in the `vco2phase` schematic will result in four copies of the L and K for `n3`, one for each occurrence of `vco2phase`.

- When creating a stitched L estimate at the top-level of the DUT and another stitched L estimate in a sub-cell schematic, any K between Ls on the top-level net and the net in the sub-cell will be copied for each occurrence of the sub-cell. This assumes that the sub-cell estimate uses the same extracted view as the DUT level estimate. If different extracted views are used, no K is stitched between the nets. Use of the same extracted view also implies that you must use the netMap property of the estimate in the sub-cell to map its net to one of the sub-cell occurrences in the extracted view. For example, mapping net `n3` of DUT to `I15/n3` (net `n3` of instance `I15`) will mean the L and K for that occurrence is to be copied for all occurrences in the parasitic/LDE view. If you want a different L and K for each occurrence you must create occurrence estimates at the DUT level.

Important Points to Consider for Parasitic Stitching of Resistance

When stitching R estimates, the following points must be considered:

- Every schematic terminal in the terminalMap list must map to an extracted view terminal. This ensures that all schematic devices are connected to the parasitic network in the parasitic/LDE view.
- If multiple schematic terminals are mapped to the same extracted view terminal, the schematic device terminals will be shorted together in the parasitic/LDE view. You can avoid this by setting each schematic terminal in the terminalMap list to a unique terminal.
- If unmapped terminals are found in the extracted view, they will be left floating in the parasitic/LDE view.

© 2023 All Rights Reserved.

Parasitic Filters Assistant

Parasitic filters allow you to select the nets for which parasitics should be included in the refined extracted view (see Refining the Extracted View).

Note: The use of parasitic filters replaces the `sbaLib` component method of identifying which parasitics to include in the refined extracted view, and provides greater flexibility and control in this area, for example:

- You are not required to alter the schematic, nor are you required to have write access to create new filters (see also Creating Parasitic Filters).
- When determining what parasitics to include in the refined extracted view you can set a filter to include, *all* or *none*, or set a lower *threshold* parasitic value to specify parasitic inclusion (see also Editing Parasitic Filters Parameter Values). Any parasitics with values less than the threshold set will be omitted from the refined extracted view. You can therefore chose to ignore parasitics that will have little or no impact.
- *Scoped filters* (cell, instance, or net filters) can also assist when setting filters on all nets in a cellview, or below an instance, without having to set filters on each individual net. This *overriding facility* lets you set different values on particular nets or instances that fall within those groups.

  For example, you may set a filter on the cell at the top of a design with a certain threshold. This will be applied to every net in the design. You may however also choose to place net filters on certain critical nets that cause parasitics with lower values to also be included. The premise here being that the lower value parasitics are not expected to have an impact on the general design, but may do on critical nets.
- You can also use occurrence-based path names for nets/instances which allow you to place different filters on individual instances of cellviews. Previously, with `sbaLib`, if you included the parasitics for a net they were included for every instance of that cellview. You can now however create several instances of a cellview and place different thresholds on each in order to examine the different levels of parasitics on the design.

© 2023 All Rights Reserved.

This section on the parasitic filters assistant comprises of the following:

- Accessing the Parasitic Filters Assistant
- The Parasitic Filters Assistant Toolbar
- Saving Parasitic Filters Created in the Current Session
- Creating Parasitic Filters
- Editing Parasitic Filters Parameter Values
- Deleting Parasitic Filters
- The Parasitic Filters Assistant Context-Menu

© 2023 All Rights Reserved.

Accessing the Parasitic Filters Assistant

The *Parasitic Filters* assistant can be accessed using one of the following methods:

- By selecting *Window – Workspaces – Parasitic-Extracted* to display the Parasitic Filters Assistant
- By selecting *Window – Assistants – Parasitic Filters*
- By selecting *Parasitics/LDE – Create Filters*

Note: For more information on accessing assistant panes see Accessing Parasitic Aware Design Functionality on page 113.

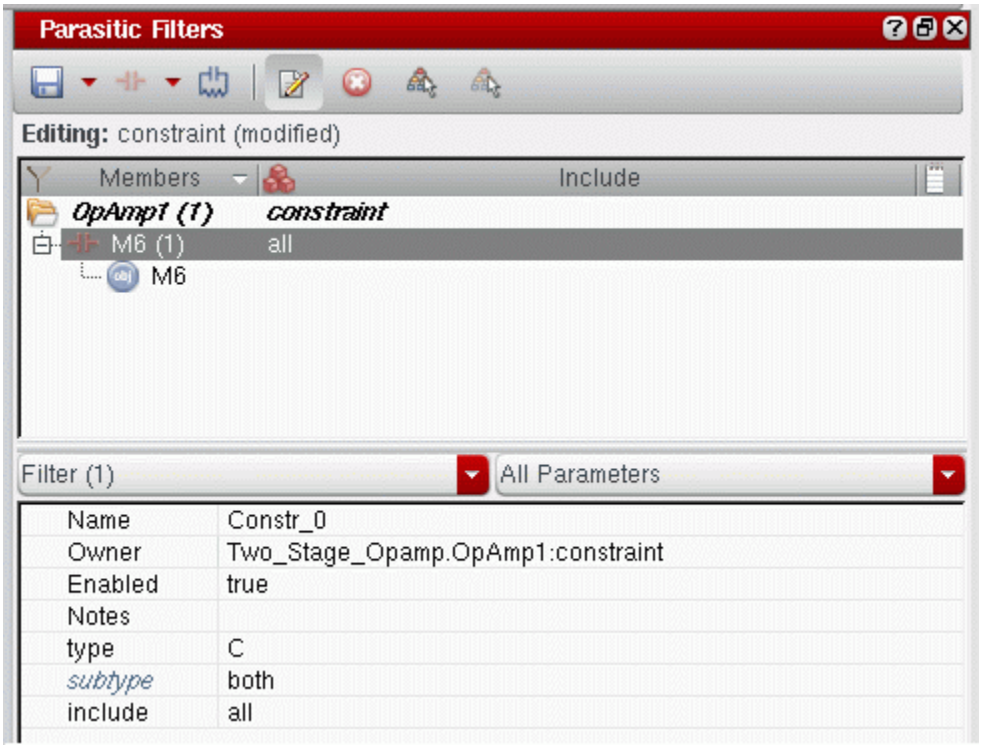


Figure 2-25 Parasitic Filters Assistant

Enabling Parasitic Filters

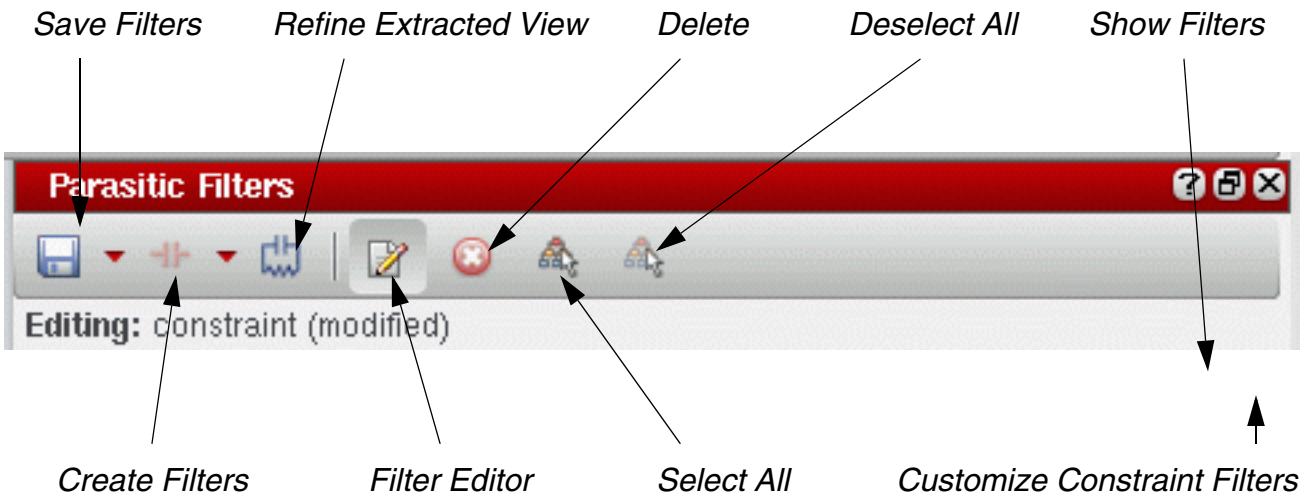
To enable the *Parasitic Filters* assistant (that is, to be in a position to start creating parasitic filters) you must setup parasitics (see Setting Up and Using Parasitics). You may then need to descend into the schematic specified in the setup form.

© 2023 All Rights Reserved.

You can cross-select between the schematic view, the Navigator, and the *Parasitic Filters* assistant so that when an object is selected, the *Parasitic Filters* assistant will highlight any filters that are associated with that object.

© 2023 All Rights Reserved.

The Parasitic Filters Assistant Toolbar



The *Parasitic Filters* assistant toolbar provides the following options:

- *Save Filters* - see Saving Parasitic Filters Created in the Current Session
- *Create Filters* - see Creating Parasitic Filters
  - *Override* - see Constraint Overrides in the *Virtuoso Unified Custom Constraints User Guide*
- *Refine Extracted View* - see Refining the Extracted View
- *Filter Editor*
- *Delete* a selected filter - Deleting Parasitic Filters
- *Select All* filters listed
- *Deselect All* filters listed
- *Show Filters / Customize Constraint Filters* selection criteria - see Filtering Constraints in the *Virtuoso Unified Custom Constraints User Guide*).

© 2023 All Rights Reserved.

The Parasitic Filters Assistant Context-Menu

Right-clicking over the *Parasitic Filters* assistant displays a context-menu. From here you can perform a variety of tasks including saving filters and controlling object visualization on the design canvas and in the *Navigator*.

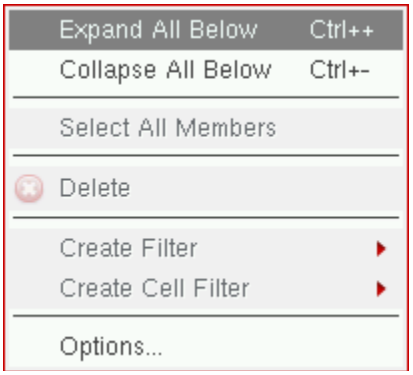


Figure 2-26 Parasitic Filters Assistant Context-Menu

The full contents of the *Parasitic Filters* assistant context menu are listed in the table below.

Context-Sensitive Menu Option	Description
Expand All Below	Expands all branches in the assistant to show all filter member objects.
Collapse All Below	Collapses all branches in the assistant to hide all filter member objects.
Select All Members	Selects all parasitic filter members that belong to the currently selected (expanded) filter.
Create Filter	Provides sub-menu options to create **net** filters for parasitic *Capacitance*, *Resistance*, and *Inductance*. See Creating Parasitic Filters.
Create Cell Filter	Provides sub-menu options to create **cell** net filters for parasitic *Capacitance*, *Resistance*, and *Inductance*. See Creating Parasitic Filters.
Create Override	See Constraint Overrides in the *Virtuoso Unified Custom Constraints User Guide.*

© 2023 All Rights Reserved.

Options	Displays the Constraint Manager Options form. For more information see Options in the *Virtuoso Unified Custom Constraints User Guide*.

© 2023 All Rights Reserved.

Saving Parasitic Filters Created in the Current Session

Click the *Save Filters* icon to save (or update) the parasitic filters that have been created or edited in the current session.

Note: You can also use the *File – Save (Constraint)* option on the main menu bar.

For more information on the other options available in the *Save Filters* pull-down see Save Constraints in the *Virtuoso Unified Custom Constraints User Guide*.

Note: *Save a Copy* will also save parasitic estimate and parasitic filter information.

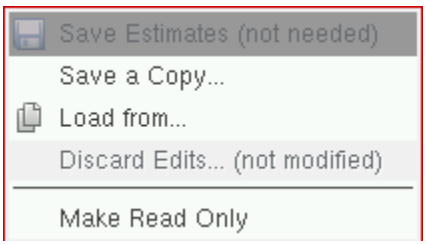


Figure 2-27 Save Filters pull-down options in the Parasitic Filters assistant toolbar

© 2023
All Rights Reserved.

Creating Parasitic Filters

The following filter creation options are available on the The Parasitic Filters Assistant Toolbar (initially prior to filter creation) and in the The Parasitic Filters Assistant Context-Menu:

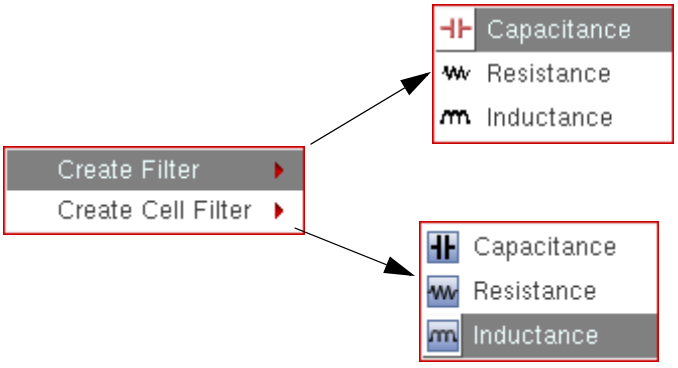


Figure 2-28 Creating Filter options available from the Parasitic Filters assistant toolbar

- *Create Filter*
  - *Capacitance*
  - *Resistance*
  - *Inductance*

  The above options are used to create filters on **instances** and **nets** that have been selected in the canvas.
- *Create Cell Filter*
  - *Capacitance*
  - *Resistance*
  - *Inductance*

  The above options are used create filters that apply to the current **cell** view (canvas selections are ignored).

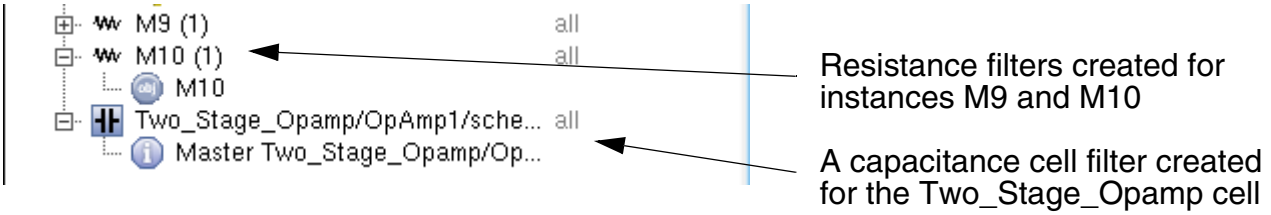
  Note: Cell filters have a blue icon background.

© 2023 All Rights Reserved.

To create a parasitic filter (overview):

1. Select the design objects in the Navigator or design canvas that you want to create a parasitic filter/cell filter for.
2. Select the type of parasitic filter/cell filter that you want to create from the The Parasitic Filters Assistant Toolbar or The Parasitic Filters Assistant Context-Menu.

   The parasitic filter/cell filter will be displayed in the constraint table in the *Parasitic Filters* assistant.



Important

For more information on parasitic constraints see parasitic filter and parasitic estimate in the *Virtuoso Unified Custom Constraints Configuration Guide*.

© 2023 All Rights Reserved.

Creating a Capacitance Filter for a Single Net

When a *Capacitance* (C) filter is created it will have an additional (to the other parasitic filter types) *subtype* parameter that indicates whether the filter applies to *coupled* capacitance, *decoupled* capacitance, or *both*.

If you set the *subtype* parameter to *coupled* or *decoupled* then you can create a second C filter on the same design object to filter the other subtype.

If *subtype* is set to *both*, only one C filter can exist for a particular design object and this will apply to both decoupled and decoupled C.

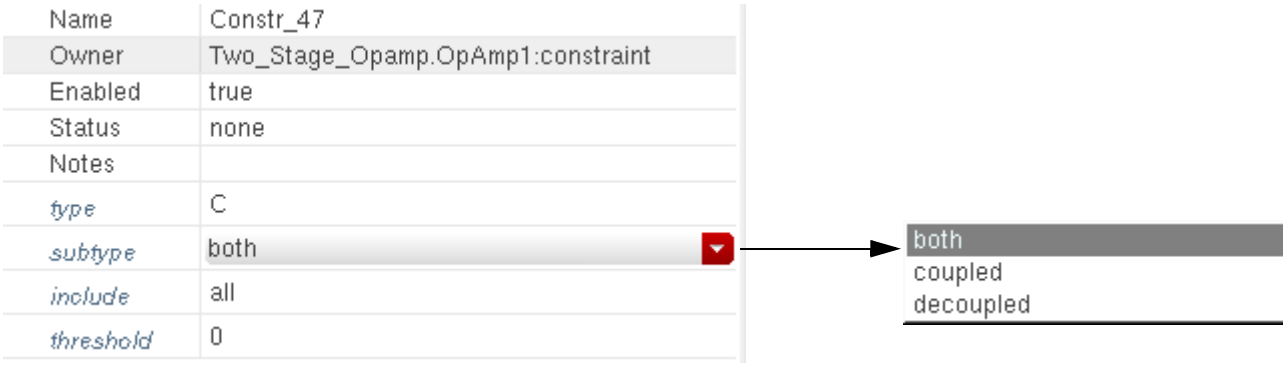


Figure 2-29 Capacitance filter Subtype parameter pull-down

Note: The *subtype* parameter does not exist for resistance or inductance filtering.

© 2023
All Rights Reserved.

Creating a Capacitance Filter for Two Nets

If you attempt to create a capacitance filter when two nets are selected the following version of the Create Parasitic Capacitance filter form will be displayed:

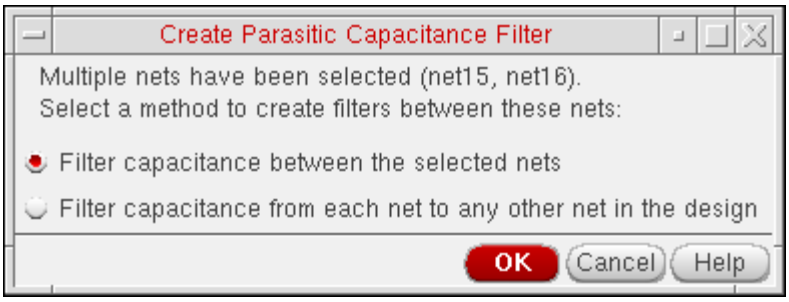


Figure 2-30 Create Parasitic Capacitance Filter form with two nets selected for filter creation

- The *Filter capacitance between the selected nets* option will create a filter that applies to the capacitance between the two selected nets only. This can be a coupled or decoupled capacitance (parasitic aware design will determine which based on the supply net information in the Setup form).
- The *Filter capacitance from each net to any other net in the design* option will create two separate filters as if you had created them one net at a time.

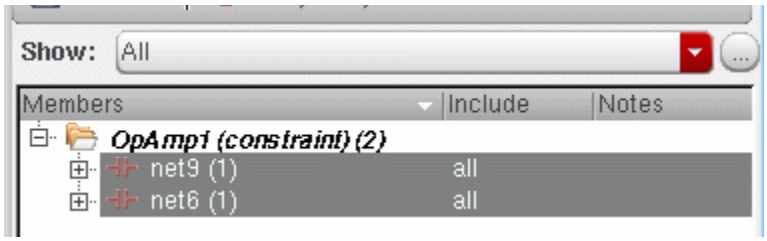


Figure 2-31 Parasitic capacitance filter created between net9 and net6

© 2023
All Rights Reserved.

Creating Multiple Filters

You can select multiple objects then apply parasitic filters to them using the Parasitic Filters Assistant.

Creating Multiple Capacitance Filters

For capacitance (C) the filter creation process differs depending on whether two nets have been selected or whether more than two nets have been selected.

For information on creating capacitance filters for two selected nets see Creating a Capacitance Filter for Two Nets.

When three or more nets are selected the following version of the Create Parasitic Capacitance filter form will be displayed when you choose *Create Filter* option:

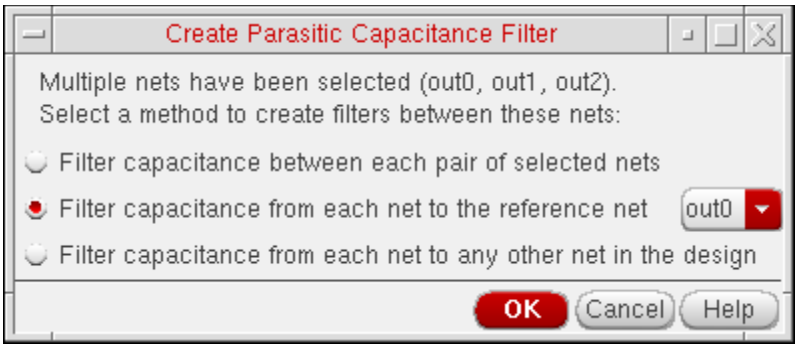


Figure 2-32 Creating parasitic capacitance filter for three or more nets

- The *Filter capacitance between each pair of selected nets* option filters capacitance between every possible pair combination of the selected nets.
- The *Filter capacitance from each net to the reference net* option will create net-to-net filters between each selected net and the chosen reference net. For example, if `out0` was the reference net and there were three selected nets (`out0`, `out1`and `out2`), two filters would be created `out1-out0` and `out2-out0`.

  Note: The nets listed in the drop-down are the selected nets.
- The *Filter capacitance from each net to any other net in the design* option will create separate filters for each selected net as if you had created them one net at a time.

© 2023 All Rights Reserved.

Creating Multiple Resistance and Inductance Filter

When applying multiple resistance (R) or inductance (L) filters you should firstly select the multiple objects then select the *Create Filter* option to create one filter for each selected object.

© 2023 All Rights Reserved.

Editing Parasitic Filters Parameter Values

You can edit one or more filter parameter values using the *Parasitic Filters* assistant.

Note: The ability to editing multiple values also applies to parasitic estimates.

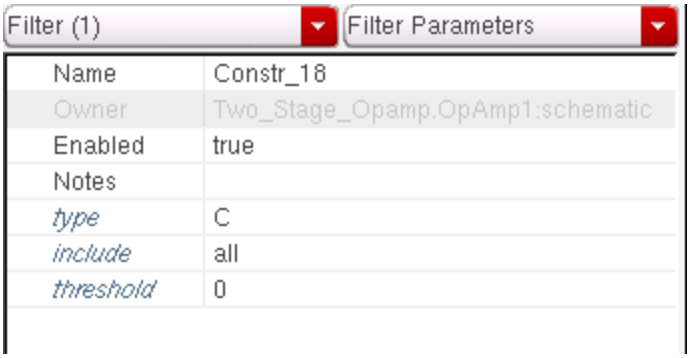


Figure 2-33 Parasitic Filter assistant's parameter editing table

- The parameter editing table, in the lower section of the *Parasitic Filters* assistant, contains details of the parasitic filter *type* (R, C, or L) as well as other parameters used in The Constraints Manager Assistant such as *Name*, *Enabled*, and *Notes*.

  Note: The parameter *type* can be changed as long as there is no other filter with the same scope and type.

  See also Creating a Capacitance Filter for a Single Net which also discusses the parameter *subtype*.

- There are two parameters that also control parasitic filter threshold values:
  - *include* - which takes *none*, *all*, or *threshold* as parameter values.
  - *threshold* - specifies the numeric threshold value if *threshold* is selected for *include* above. Threshold applies to the magnitude of the co-efficient ("-" sign is ignored).

To edit parasitic filter threshold values:

1. Select one or more filters from the upper section of the *Parasitic Filters* assistant.
2. Edit single or multiple parameter values in the lower section of the *Parasitic Filters* assistant

© 2023 All Rights Reserved.

Note: If multiple filters are selected, common parameter values can be set simultaneously in the same manner as using the Property Editor assistant. They can be set as a group by editing the value in the same row as the parameter name, or individually by expanding the parameter name row to display the values for each filter.

© 2023 All Rights Reserved.

Deleting Parasitic Filters

To delete a parasitic filter:

1. Select the filter to be deleted in the upper section of the *Parasitic Filter* assistant.
2. Click the *Delete* option on the The Parasitic Filters Assistant Toolbar or in the The Parasitic Filters Assistant Context-Menu.

Note: There is no filter for K as K will be included for any L pairs included in the refine extracted view.

© 2023 All Rights Reserved.

Refining the Extracted View

Click the *Refine Extracted View* icon in the *Parasitic Filters* assistant toolbar to display the Refine Extracted View form.

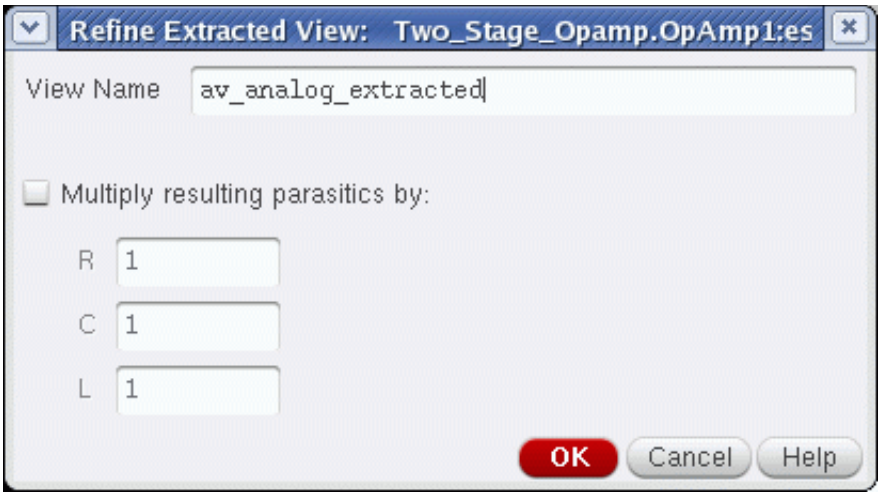


Figure 2-34 The Refine Extracted View Form

Note: Parasitic filters control the content of the refined view.

The purpose of *Refine Extracted View* is to generate an extracted view that can be used for circuit simulation. The refined extracted view generated is based on the full extracted view created in Quantus QRC or RCX.

The default full extracted view is `av_extracted`, while the default parasitic aware design refined extracted view name is `av_analog_extracted`.

After the refine extracted view has been created, a window is displayed summarizing the parasitics in the new view. This information is also displayed in the CIW.

The refined extracted view will not remove any resistance that is associated with inductors as long as the inductors are not filtered out. This will prevent the creation of zero-resistance loops in the refined parasitic net.

GUI Item	Description
View Name	Specifies the name of the refined extracted view to be created.

© 2023 All Rights Reserved.

GUI Item	Description
Multiply resulting parasitics by:	Allows you to simulate with different parasitic values. For example, the minimum and maximum expected parasitic values (based on the same extracted view).
R	Set the multiplication factor for resistance parasitics.
C	Set the multiplication factor for capacitance parasitics.
L	Set the multiplication factor for inductance parasitics.

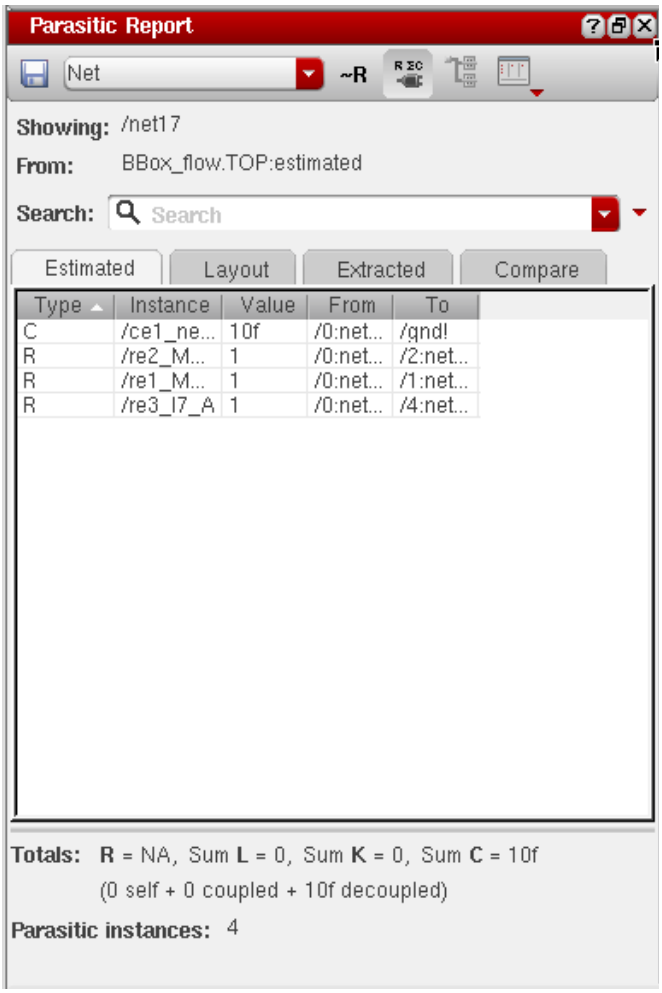
© 2023 All Rights Reserved.

Parasitic Report Assistant

The *Parasitic Report* assistant can be used to readily view and compare parasitic reports for estimated, extracted, layout, and Smart Views.

It can be accessed in ADE Explorer and ADE Assembler by selecting one of the following options:

- *Windows – Assistants – Parasitics Report*
- *Windows – Workspaces – Parasitics – Report*
- *Parasitics/LDE – Report*



Report Summary (number of each parasitic instance in current view)

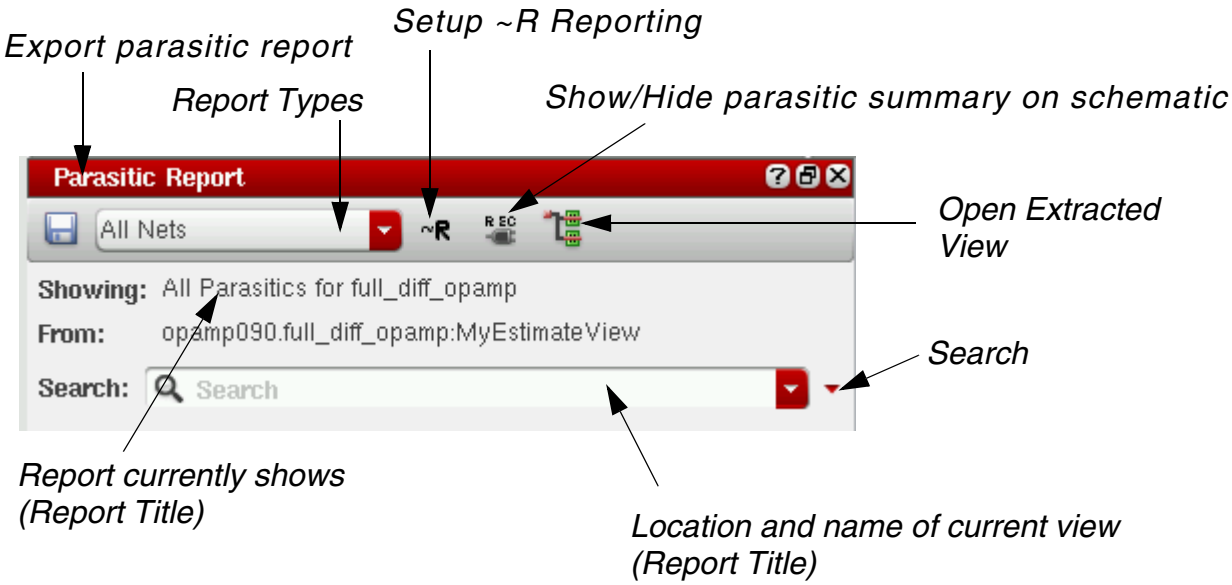
Figure 2-35 Parasitic Report Assistant

© 2023 All Rights Reserved.

Note: The *Parasitic Report* assistant replaces the Parasitic Report forms that are still available in the parasitics.

The *Parasitic Report* assistant comprises of the following sections:

- A *toolbar* that can be used to setup and specify your report choice



  - *Export Parasitic Report* (see Exporting Parasitic Reports)
  - *Report types* (see Parasitic Report Assistant Available Reports)
  - *Setup ~R reporting* (see Setting Up Effective R Reports)
  - *Show/Hide parasitic summary on schematic* (see Showing and Hiding Parasitics from the Parasitic Report Assistant)
  - *Open extracted view* (see Opening an Extracted View from the Parasitic Report Assistant and Cross-Probing Parasitics)
  - *Search* (see The Search Toolbar in the *Virtuoso Schematic Editor XL User Guide*)

  Note: You can right-click over the *Parasitic Report* assistant report table or column

© 2023 All Rights Reserved.

header area to select and clear the user-interface elements that you want to view.

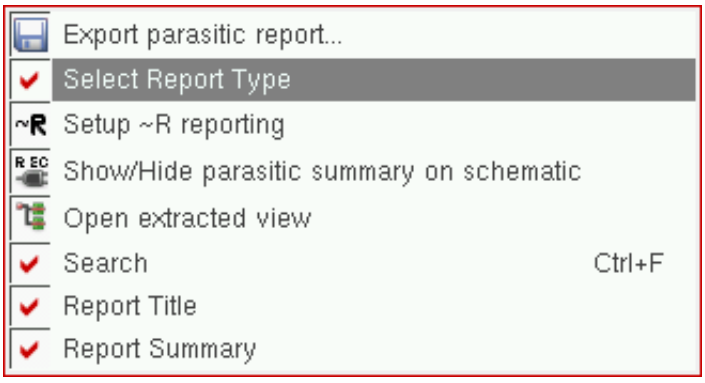


Figure 2-36 Right-Click Over the Parasitic Report Toolbar to Choose UI Elements

When the *Compare* report tab is selected in the *Parasitic Report* assistant, the *Showing* and *From* report title fields will be replaced by *Limit/Target* and *Actual* fields.

- A selection of *tabs* that can be used to view parasitics from the parasitic/LDE, layout, or extracted view. The *Compare* tab shows comparison reports..
  - The *Layout* tab shows the report of parasitics included from the layout view for the EAD flow. For more details, refer to the Resimulating Designs with Extracted Parasitics in *Virtuoso Electrically Aware Design Flow Guide*.
  - Also available from parasitic report tabs are *context-menus*, accessible with a right-mouse-button click over the appropriate report tab column headers. The context-menu-items displayed provide a selection of options related to the current report tab. From here, you specify what parasitic information is displayed (*Show*), how the displayed information is sorted (*Sort By*), and so on.

Figure 2-37 The Compare Report Context-Menu

- A *reporting table* that provides content updates dynamically based upon selections in the schematic canvas or the *Navigator* assistant.

© 2023 All Rights Reserved.

Parasitic Report Assistant Available Reports

The following reports can be accessed from the *Parasitic Report* assistant for estimated and extracted parasitics:

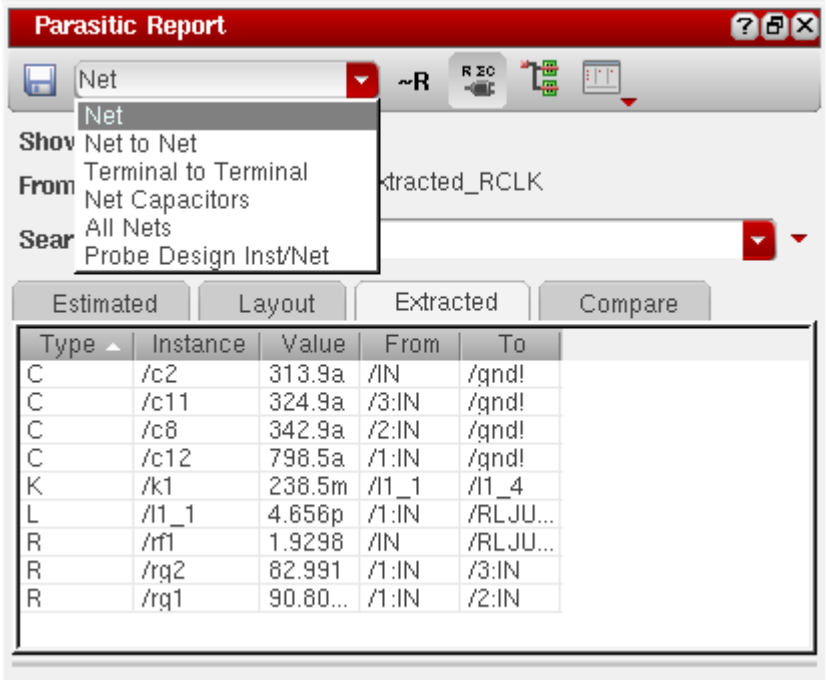


Figure 2-38 Reports Available in the Parasitic Report Assistant

- *Net* (see also Net Reports)
- *Net to Net* (see also Net to Net Reports)
- *Terminal to Terminal* (see also *Terminal to Terminal Reports*)
- *Net Capacitors* (see also Net Capacitors Reports)
- *All Nets* (see also All Nets Reports)
- *Probe Design Inst/Net* (see also *Probe Design Inst/Net Reports*)

Note: The *Probe Design Inst/Net* report is available only when the *Extracted* tab is selected.

Parasitic report results may be grouped into collapsible categories (for example, R, C, L, K) by selecting the *View By* option in the context menu.

© 2023 All Rights Reserved.

Each report table column is sortable, within a category, by column heading. Sorting criteria can be specified using the *Initially Sorted By* option in the *Parasitic Probing Reports* section of the Parasitic Options form (see Parasitics/LDE – Options).

Understanding Parasitic Report Results

The Parasitic Report assistant provides a detailed report of the parasitics in the design. The following table describes how to understand the results displayed in the assistant pane.

Report Type	Name	Description
Net, *Net to Net*, *All Nets*	*R*	Displays parasitic resistance elements.
	decoupled C	Displays capacitors that are bound to power or ground nets.
	coupled C	Displays capacitors between ordinary nets.
	self C	Displays coupled capacitors between segments of the same net.
	L	Displays parasitic inductances.
	K	Displays the mutual inductance (K) ratio. Example: For two given circuits A and B, the mutual inductance is the ratio of the flux through circuit A caused by the current in circuit B.
Terminal to Terminal, *Probe Design Inst/Net*		Reports parasitics between two instance terminals or between a terminal and a pin on the same net. **Note:** The terminal to terminal probing does not report parasitics between two pins. It lists the parasitics on all paths between the two terminals.
Net Capacitors		Generates the capacitors for the net report which displays summed capacitances (Sum C) between the selected net and all those nets to which it is connected through parasitic capacitors.

© 2023 All Rights Reserved.

Sum Totals	The *Sum* totals at the bottom of a report display the effective and summed values. The C, L, and K values are always summed, whereas the R value depends on the type of probing selected for each parasitic type that has been found for the the specified probing type. ■ Terminal to Terminal probing: The R total value displayed is calculated by reducing the parasitic R network between the two terminals. ■ Net probing: Effective R value is a measure of the power lost through the parasitic resistor network (or equivalently, the delta between the power entering the network and the power leaving the network, normalized by the total current entering (or leaving) the network. Combining `V=IR` with `P=VI`, it is calculated as `Effective R (R~) = P / I^2` where: `P` is the sum of the power through the parasitic resistors in the network. `I` is the current entering (or leaving) the network. The power and current values are read from the DCOP results. Therefore, it is expected to run a single point DC simulation.

Net Reports

For net probing, the report will initially report on the most recently selected net. If multiple nets are currently selected only the most recently selected net will be reported on. Other object types will be ignored.

Net to Net Reports

You can select the nets you want to report on prior to or after selecting the *Net to Net* report option.

© 2023 All Rights Reserved.

For net to net reporting, the report will initially report on the two most recently selected nets. If more than two nets are selected, all nets other than the last two selected will be ignored. Other object selections will also be ignored.

Note: You can also specify how to identify power and ground nets using the *Power and Ground Nets for Decoupled Capacitance Reporting* section in Parasitics/LDE – Setup.

Terminal to Terminal Reports

Using the terminal to terminal probing, you can report parasitics between two instance terminals or between a terminal and a pin on the same net.

The Parasitic Report assistant will report parasitics between the two most recent selections only. If multiple selections are done, all terminals or pins other than the last two selected will be ignored. Other object type selections are also ignored.

Note: The terminal to terminal probing does not report parasitics between two pins.

Terminal to terminal report values can differ from whole net report values

When you perform a terminal-to-terminal probe (also known as point-to-point probing) or generate a parasitic report, if two selected terminals map completely to the same extracted nets, a warning describing the situation is displayed. The resultant report will also reveal that no parasitics have been found between these terminals. This happens because the terminals are considered to be shorted, and parasitics will therefore have no influence on the interaction between the two terminals in question.

However, if you subsequently perform a whole net probe on the same net, parasitics could still be reported even if the net is a two terminal only net.

For example, if you consider two m-factored transistors (mFactor = 2) `T1` and `T2`, with the source of `T1` connected to the drain of `T2`. Selecting `T1-source` and `T2-drain`, in the schematic, will identify that `T1_m1-source` is shorted with `T2_m1-drain` and `T1_m2-source` shorted with `T2_m2-drain`. As mentioned, terminal to terminal probing would complete, reporting no parasitics, however whole net probing will identify and report any parasitics that might exist between `T1_m1-source` and `T2_m2-drain`.

S*electing Leaf Terminals*

The Select leaf terminals form is displayed when you perform terminal to terminal probing and select hierarchical terminals that have more than one leaf terminal.

© 2023 All Rights Reserved.

Note: If both terminals selected have multiple leaf terminals, the Select leaf terminals form will be displayed twice, once for each terminal.

From the Select leaf terminals form you must select the terminals (for example, gate and drain) that you want to measure resistance values for.

The ability to select individual terminals prevents the results of point to point probing of a net, that has parallel resistors, being reported as a sum of the resistance values.

Net Capacitors Reports

Reporting on net capacitors generates a report which displays summed capacitances (*Sum C*) between the selected net and all the nets that it is connected to via parasitic capacitors.

No report will be generated for “0” capacitances, and no distinction is made between power/ground, and other nets.

All Nets Reports

The *All Nets* report lists each net in the design along with the total value for the net based on the effective value for *R*, the summed value for *C* and *L*, and the weighted sum for *K*.

Note: If you are viewing the *Estimate* or *Extracted* tabs for *All Nets* reports, you can right-click over the column headers and choose what type of parasitics (*R*, *L, K, decoupled C*, *coupled C,* and/or *self C*) that you want to *Show*.

© 2023 All Rights Reserved.

Probe Design Inst/Net Reports

Note: You should ensure that both the schematic and extracted views are open to facilitate viewing and probing.

Select *Probe Design Inst/Net* from the report pull-down in the *Parasitic Report* assistant to display the Extracted Insts report form from where you can:

- probe from a design instance or net, in the current schematic view, to extracted instances in the extracted view. The probe performed will pan to and zoom in on the extracted instances.

  or

- probe from a design instance or net in the extracted view to the schematic view. The probe performed will highlight, but not zoom in on, the schematic instance. If the schematic instance is within a hierarchical block, in an open schematic, then the block will be highlighted.

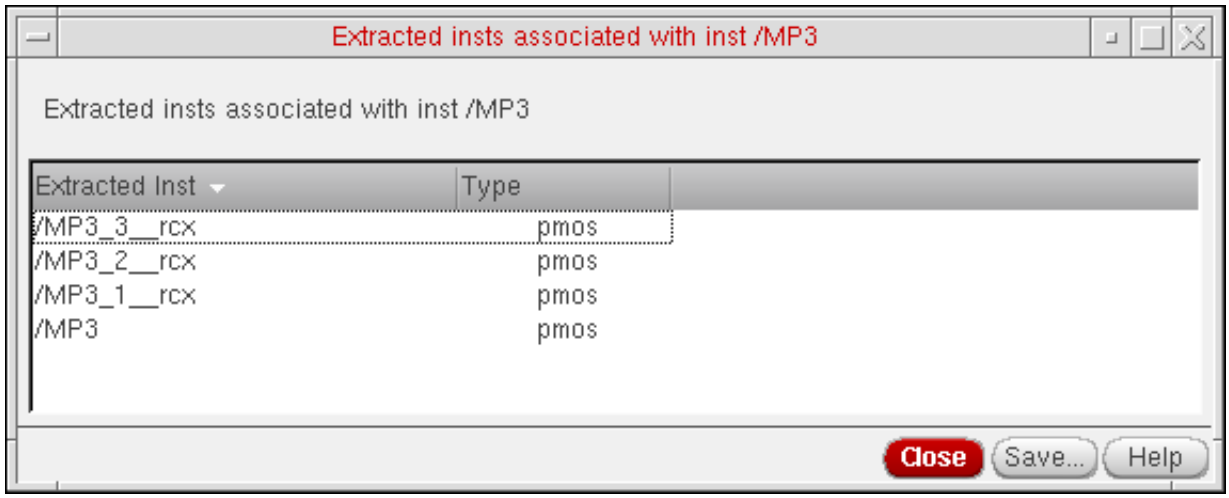


Figure 2-39 Extracted Insts Report Form

Note: When probing from the extracted view, highlighting in the schematic view will only work when the schematic is open in its own right. It will not work when the schematic is open through a configuration.

If you attempt to probe from hierarchical blocks/instances, a warning message will be displayed in the CIW. To resolve this you must descend the hierarchy (*Edit – Hierarchy – Descend Edit*) and probe at the leaf level.

© 2023 All Rights Reserved.

If you are probing a schematic net that exits the current hierarchy level, the probe will be restricted to the instances attached to that net in the current level and not the full design. Probing therefore remains localized.

After you have selected a regular device (instance or net) to probe the relevant *Extracted/ Schematic insts associated...* form is displayed.

This tabular form lists all of the *Extracted/Schematic inst* names, and their associated *Type*, that are related to the device that you selected. The results are viewed in sortable columns.

For a design net, the displayed list contains all of the extracted or schematic instances that are attached to that net in the other view type.

If you select one or more extracted instances from the list (using the `Ctrl/Shift` keys) this causes the extracted view to automatically zoom into those instances, highlighting the selected items in both the schematic and extracted views.

Note: Highlighting of instances only applies to an extracted view, not a layout view.

Clicking on the *Save* button in the *Extracted insts...* form displays the *Save Instance Report form*. For information on saving reports in parasitic aware design see Exporting Parasitic Reports.

Note: There is no corresponding zoom facility on the schematic view.

When you are performing an m-factor selection it is observed that the instances are found, and highlighted, in the same area of the chip. A m-factored device is where a single cell instance represents multiple instances of a parallel connected cell.

© 2023 All Rights Reserved.

Exporting Parasitic Reports

Selecting the *Export Parasitic Report* button on the *Parasitic Report* assistant toolbar invokes the Save Parasitic Report form.

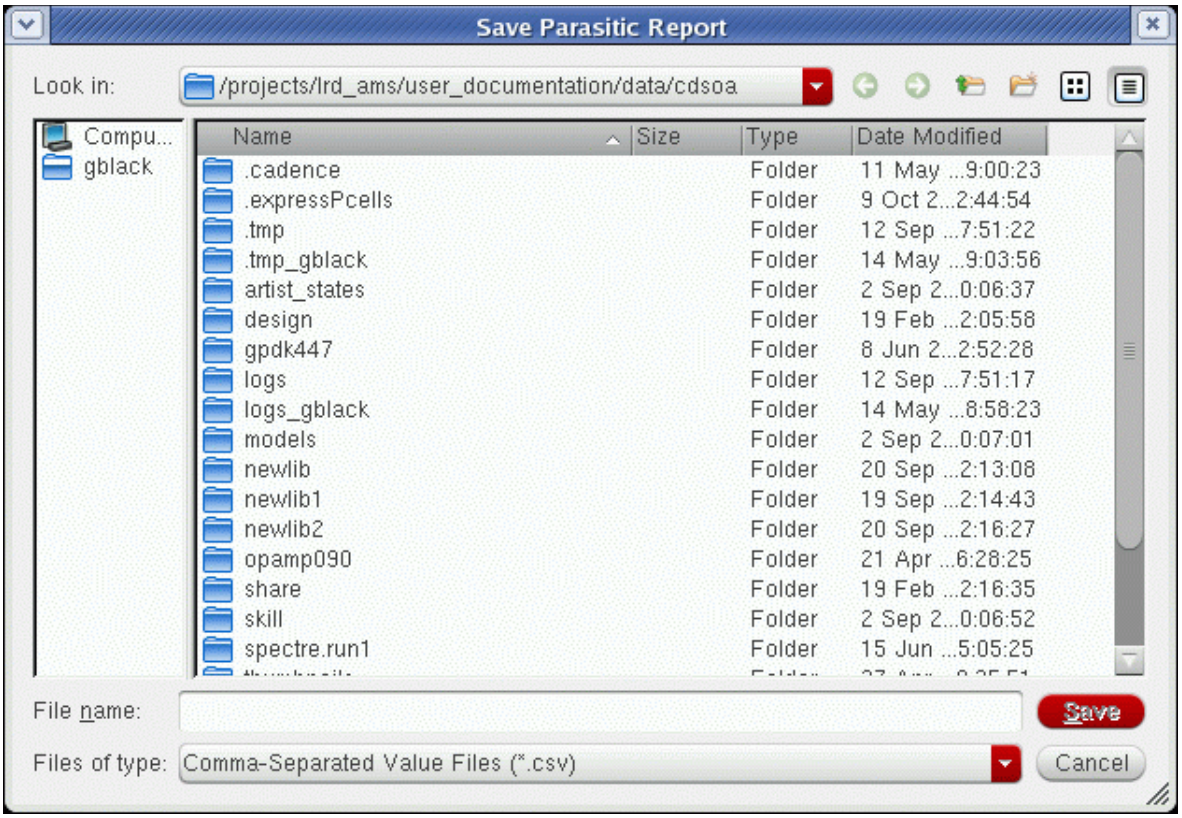


Figure 2-40 The Save Parasitic Report Form

From the Save Parasitic Report form, you can *Save* (then export) parasitic reports in `.txt` or `.csv` (comma-separated values) format.

© 2023 All Rights Reserved.

Setting Up Effective R Reports

Selecting the *~R* (effective R) option on the *Parasitic Report* assistant toolbar will display the Setup ~R Reporting form where you can specify your *~R Reporting*.

Note: See also Effective R Calculations and Preparing for Resistance Backannotation (Running a DC Analysis).

There are two option tabs, *Estimated* and *Extracted*, at the top of the form where you can choose what view type effective R information that you want to setup.

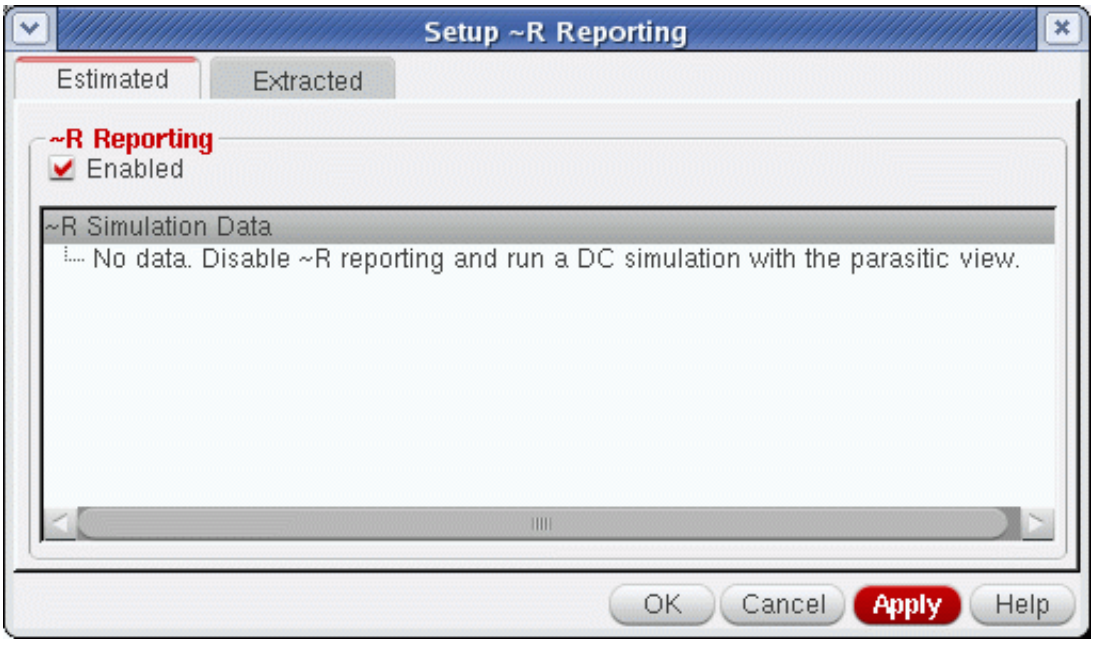


Figure 2-41 Setup ~R Reporting Form

Note: To enable the *~R Simulation Data* table, the *Enabled* option must be checked.

If you have enabled resistance reporting, you must select valid simulation results from the tree displayed. The tree will display results from the simulation history.

Note: Disable this option if DC simulation results are not available or you are yet to build the parasitic/LDE view. To ensure that DC simulation results are available, select the *Save DC Operating Point* check box on the Choosing Analysis form.

History items are filtered to only display those results that contain a simulation where the current instances have been bound to the chosen cellview. This means that if multiple instances of the DUT (design under test) exist, only those results for the one that you have

© 2023 All Rights Reserved.

descended into will be displayed. This is because results for other instances of the DUT will not be valid in this context.

However, if you open the DUT directly, you will see results for all instances of the DUT in all tests. In this scenario, you must choose the appropriate instance path.

© 2023 All Rights Reserved.

Opening an Extracted View from the Parasitic Report Assistant and Cross-Probing Parasitics

Selecting the *Open Extracted View* option on the *Parasitic Report* assistant opens (raises) the extracted view or Smart View in a new tab in your current session window. The current report content is unaffected when the new view is opened.

Note: This option is only enabled when the *Extracted* tab is currently selected in the *Parasitic Report* assistant.

With the new view raised, you can cross-probe to thisview by selecting the parasitic values in the displayed report. Cross-probing operates by selecting single or multiple items in the report. VPAD zooms and pans to the corresponding instances in the opened view.

Cross-probing from the *Parasitic Report* assistant is supported for the net, net to net, terminal to terminal, and design/net inst probe reports.

Note: Selections on the view canvas initiate probing in the view. You can switch back to the schematic view tab, and reselect the object you want to cross-probe.

Consider that *Extracted View Name* on the *Extracted* tab of the Parasitics/LDE Setup form is set to a Smart View.

To cross-probe this view:

1. Choose *Parasitics/LDE – Report*.

   The schematic and the Parasitic Report assistant opens.

2. Do one of the following:

   - Open the Navigator assistant and select a net.
   - Select a net from the schematic.

   The *Extracted* tab in the Parasitic Report assistant is populated with the parasitics on the selected net.

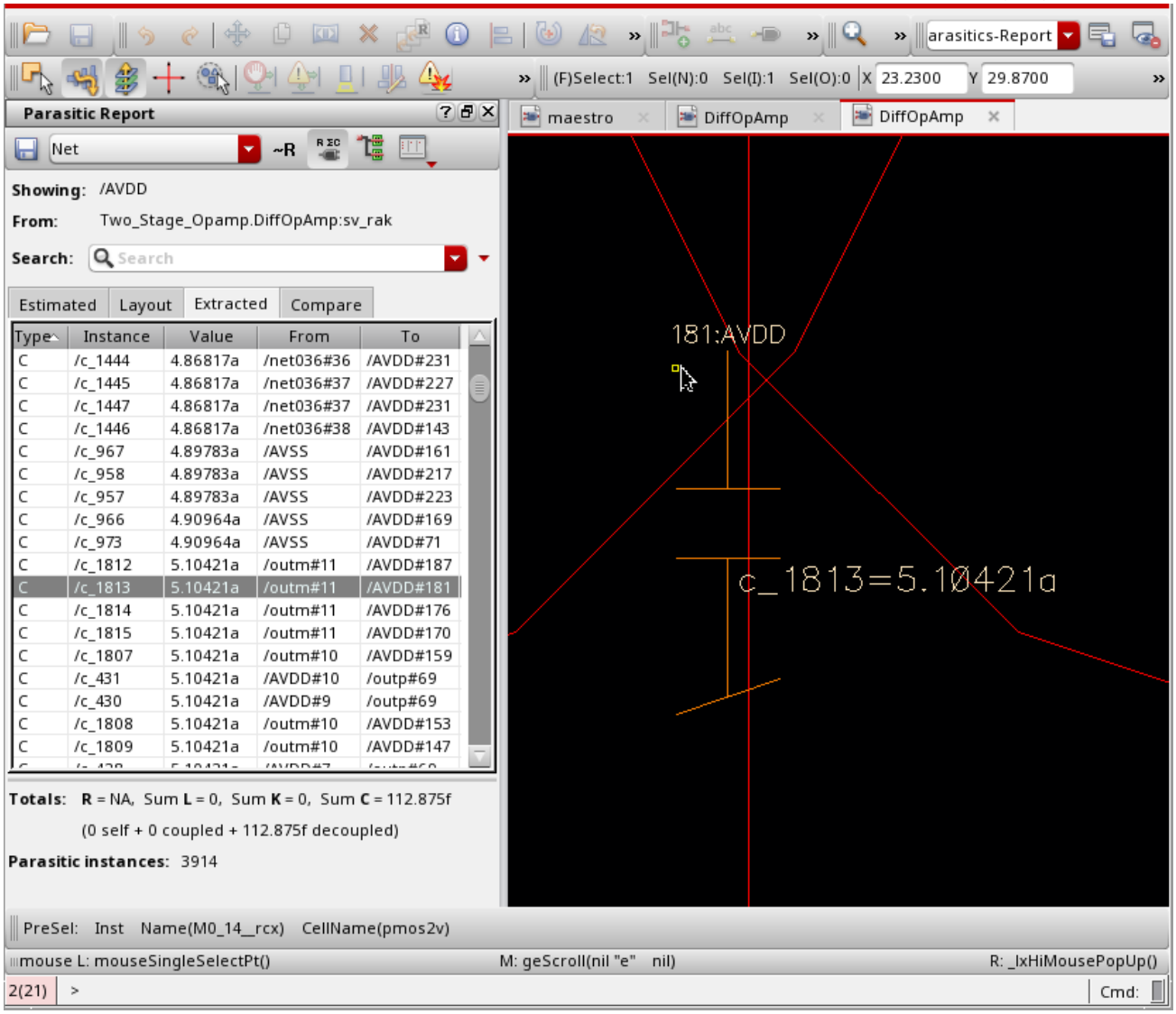
3. Click the *Open Extracted View* icon from the toolbar of the Parasitic Report assistant.

   The Smart View appears.

4. Select a parasitic resistor or capacitor from the report on the *Extracted* tab.

© 2023 All Rights Reserved.

The canvas zooms in to display the selected parasitic as shown in the following figure:



© 2023
All Rights Reserved.

Showing and Hiding Parasitics from the Parasitic Report Assistant

Selecting the *Show/Hide* button option on the *Parasitic Report* assistant toolbar will control the display of parasitic values on the design canvas.

The parasitic values displayed on the canvas will be dependent upon which tab is currently raised in the *Parasitic Report* assistant. If the *Estimated* tab is current, only parasitic/LDE view parasitics will be annotated on the canvas. Similarly, for the *Extracted* tab, should that be current. Annotations will also only remain visible when the *Parasitic Report* assistant is raised.

Note: No parasitic values are displayed if the *Compare* tab is on view.

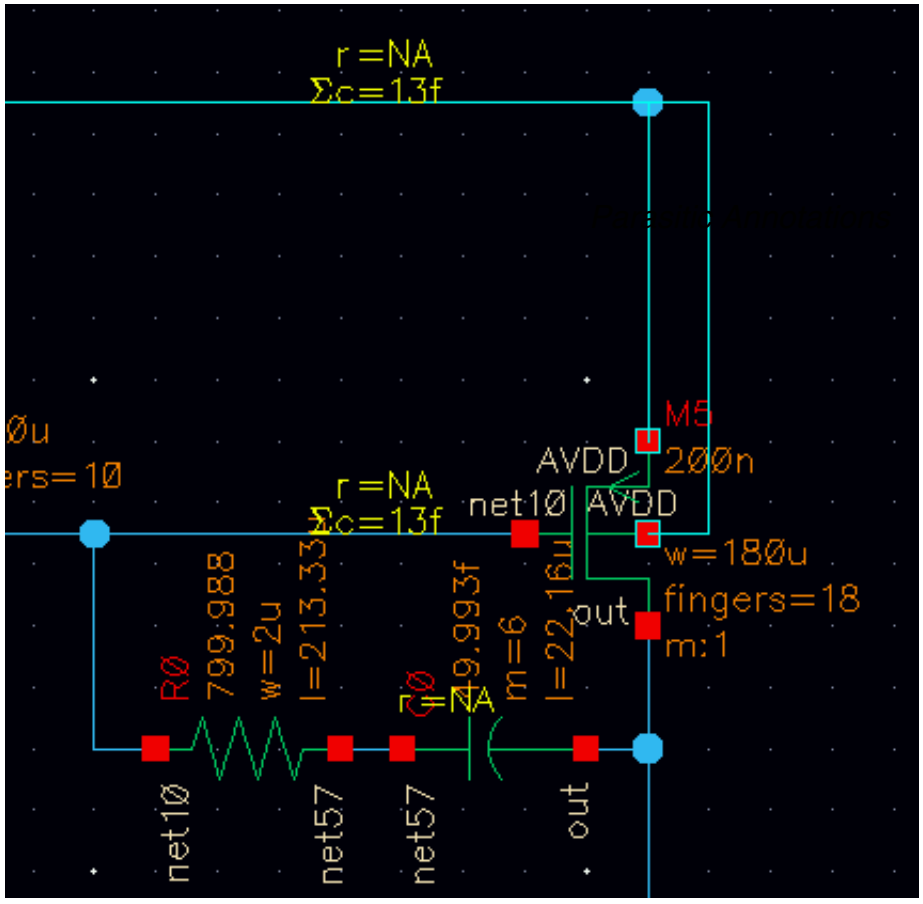


Figure 2-42 Annotations on Nets in a Schematic

© 2023
All Rights Reserved.

Comparison Reports

The *Parasitic Report* assistant can also be used to compare estimated and extracted parasitic views as well as being able to compare two extracted views and two parasitic/LDE views.

To populate this report, select *Parasitics/LDE – Compare* and complete the Compare Parasitics form (see Parasitics/LDE – Compare and Parasitic Comparisons).

The *Compare* tab report shows summaries for each net, grouped at the top-level by type. You can choose to see those nets that met (*Pass*) or exceeded (*Fail*) the set estimates, and also what parasitic values (*R*, *decoupled C*, *coupled C*, *self C*) that you want to display in the table.

If you select a net, in the report table, the *Estimated* and *Extracted* tabs will show details of that net.

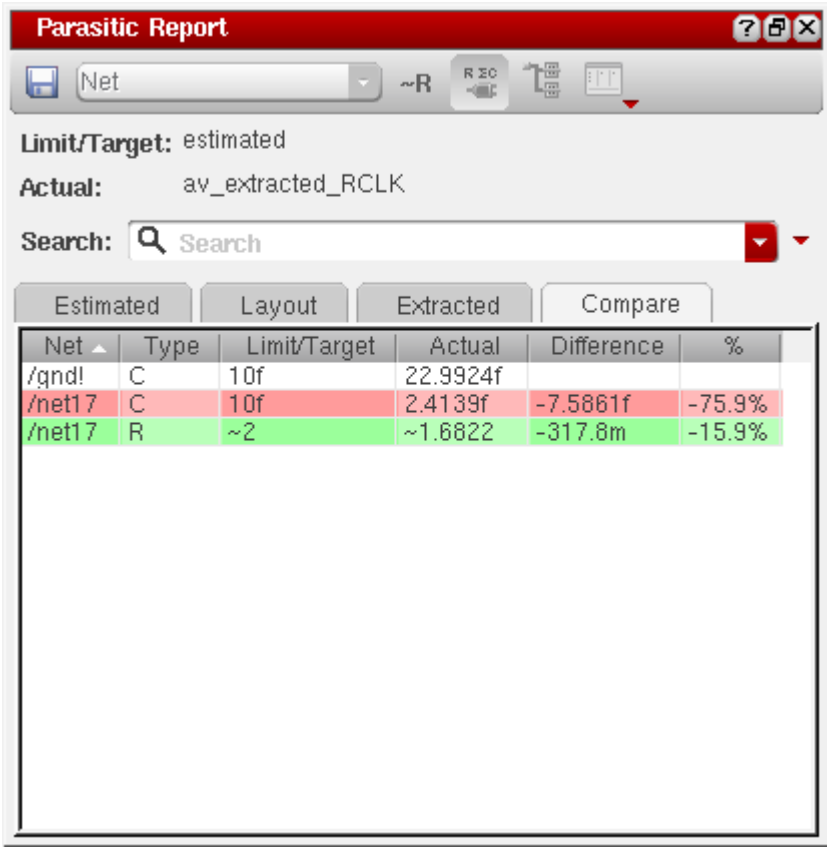


Figure 2-43 The Compare Tab in the Parasitic Report Assistant

© 2023
All Rights Reserved.

The following table describes the status information being relayed in the *Compare* tab for each color setting.

Color	Status
Red	Fail
Green	Pass
Yellow	Fail, but within 10% of the limit or 10% of the tolerance (see Parasitic Comparisons)
White	A supply net. These are not compared (but totals are reported)

© 2023 All Rights Reserved.

Parasitic Comparisons

Parasitic comparisons are used to compare parasitics from the estimated schematic and extracted views. You can use these views to compare values for *~R*, *Coupled C*, and *Decoupled C* on a net by net basis.

Note: If you need to compare *~R*, both estimated and extracted views need to be simulated before performing a comparison.

Important

For information on viewing a comparison report see Comparison Reports.

Some features and requirements when using parasitic comparisons include:

- Parasitic comparisons provide for a comparison of two parasitic views, for example a comparison of estimated parasitics and actual parasitics that are based on the result of an extraction.
- When comparing *capacitances* (decoupled and coupled) you have to provide directions to both parasitic views.
- When comparing *resistances*, you have to provide DC operating point simulation data for both views.
- You can generate a parasitic comparison report (Comparison Reports) using one of the following criteria:
  - The parasitic values from one view are used as the upper limit for the actual parasitic value.

    or

  - You can specify a threshold (+/- %) around the target value to set limits for the actual value.
- The parasitic comparison report will contain information on: the status of each net (PASS or FAIL), the type of parasitic (R, Decoupled C, or Coupled C), the target/limit value, the actual value, the parasitic value difference, and the % parasitic value difference.
- To compare estimated parasitics against extracted parasitics you must use the *original* schematic design (not the parasitic/LDE view created using Parasitics/LDE – Create Filters) and create an extracted view.

© 2023 All Rights Reserved.

Accessing the Compare Parasitics Form

To access the Compare Parasitics form:

1. Open the design under test (DUT) schematic that was used for simulation, or descend into the DUT, before opening the Compare Parasitics form.

   Note: In the case of the DUT, you must add tests which simulate the two parasitic views. These can be separate tests or different instances in the same test. The parasitic comparison flow is highly flexible in that it will search the simulation results to retrieve the required views.

2. Select *Parasitics/LDE – Compare* to display the Compare Parasitics form.

© 2023 All Rights Reserved.

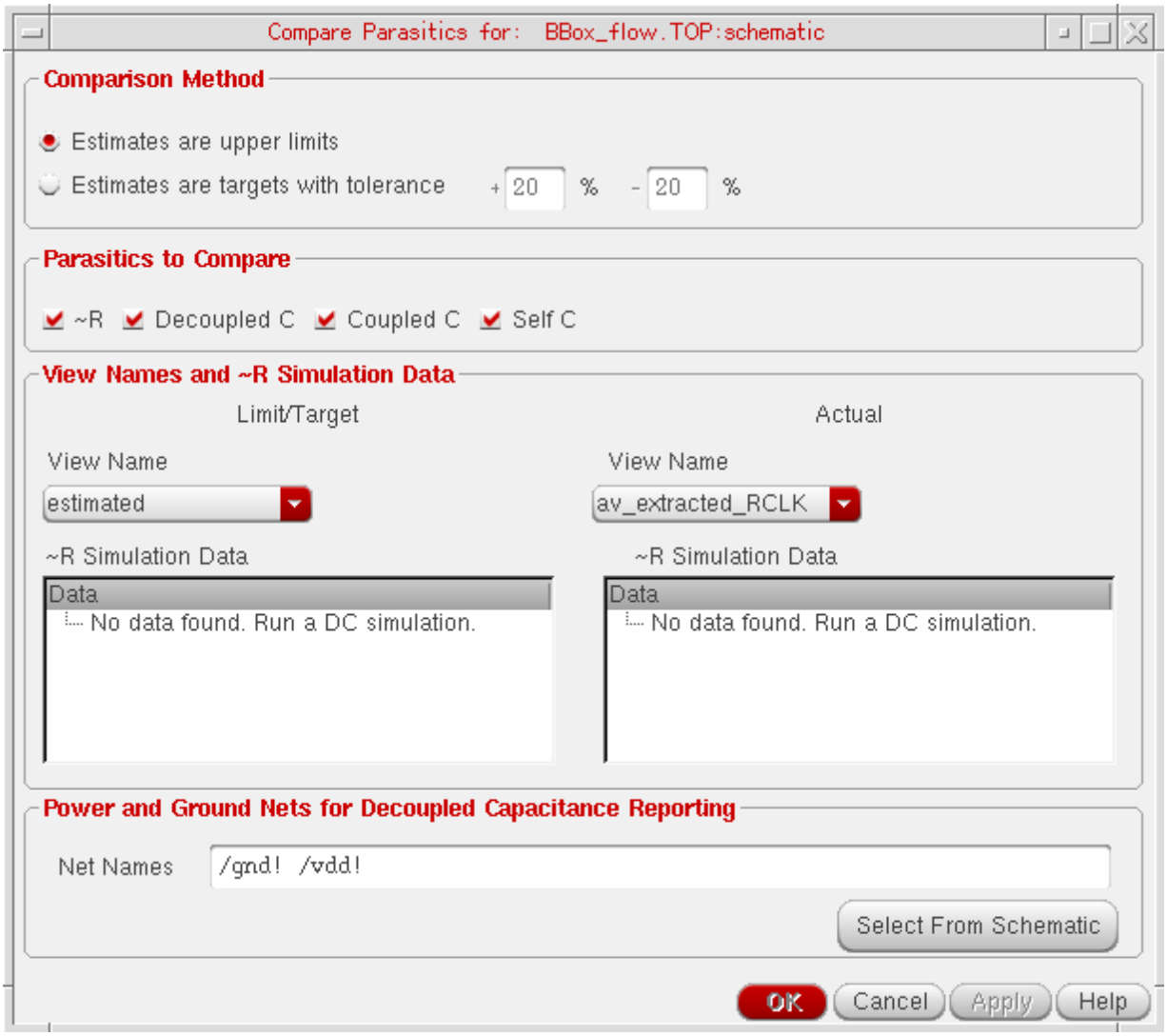


Figure 2-44 The Compare Parasitics form

Note:

- The *Target/Limit* is set to the parasitic/LDE view. The extracted view is used for the *Actual* view.
- You can select the estimated or extracted parasitics for both views.
- One view will require to be treated as the *Target/Limit* for comparisons.

© 2023 All Rights Reserved.

GUI Item	Description
Comparison Method	Controls how the estimates and extracted parasitics are compared.
- Estimates are upper limits	Specifies that estimates must represent an upper limit.
- Estimates are targets with tolerance +/-	Specifies that estimates represent a target. Here, you can specify a positive (or upper) and negative (or lower) percentage tolerance within which extracted parasitics must fall.
Parasitics to Compare	Selects which classes of parasitics should be validated and compared.
- ~R	Specifies that ~R (effective resistance) parasitics are to be compared. **Note:** If you do not check this option the *~R Simulation Data* fields will be disabled. Not checking this option also means that you are not required to run any DC oppoint simulations.
- Coupled C	Specifies that coupled capacitance parasitics are to be compared.
- Decoupled C	Specifies that decoupled capacitance parasitics are to be compared.
- Self C	Specifies that coupled capacitance parasitics between segments of the same net are to be compared.
View Names and ~R Simulation Data	Specifies what view is to be the *Target/Limit* for comparisons. See also Simulation Data.
Target/Limit	Specifies the parasitic/LDE view name to be compared.

© 2023 All Rights Reserved.

GUI Item	Description
~R Simulation Data	Specifies the simulation results to be used for the parasitic/LDE view (only enabled if *~R* values are to be compared). **Note:** The table is populated with entries from the result history. From here, you should locate the results for the simulation made using the parasitic/ LDE view. To help locate this only those results from a Spectre or UltraSim DC oppoint simulation of the current design is included in the table. The form will also filter results so that only those results for simulations containing the chosen estimate view will be displayed.
Actual	Specifies the extracted view name to be compared.
~R Simulation Data	Specifies the simulation results to be used for the extracted view (only enabled if *~R* values are to be compared). **Note:** The table is populated with entries from the result history. From here, you should locate the results for the simulation made using the parasitic/ LDE view. To help locating this, only those results from a Spectre or UltraSim DC oppoint simulation of the current design is included in the table. The form will also filter results so that only those results for simulations containing the chosen estimate view will be displayed.
Power and Ground Nets for Decoupled Capacitance	Use this field to set power and ground nets. **Note:** This field will initially be populated from the estimated/extracted setup. Parasitic aware design uses these to determine which parasitic capacitances are decoupled (one side of the capacitor is connected to a power or ground net) or coupled (the capacitance is between two other nets).
- Net Names	Identify those power and ground nets that you want to use when differentiating between coupled and decoupled nets. **Note:** When entering a net name, you can optionally add a “/” at the beginning of each net name. If omitted, the “/” is automatically added when comparing nets.
- Select From Schematic	Select the power and ground nets directly from the schematic rather than physically typing them in.

© 2023 All Rights Reserved.

Running DC Oppoint Simulation for Comparisons

Before attempting to perform a parasitic comparison you have to run a DC oppoint simulation for both the estimated parasitic/LDE view and the extracted layout. This enables parasitic aware design to calculate and compare effectiveR values for the two views. If you do not want to compare *~R* resistance values a DC oppoint simulation is not required.

Note: For information on running a DC oppoint analysis see Setting Up Analyses in the *Virtuoso ADE Explorer User Guide*.

When the Compare Parasitics form is opened, parasitic aware design searches all views of the current cellview (the DUT) looking for the parasitic/LDE and extracted views. The *Estimated View Name* and *Extracted View Name* pull-downs are populated so that only the parasitic/LDE views are listed in the estimated pull-down and extracted views listed in the extracted pull-down. Parasitic aware design then searches the data history for simulation results that contain the estimated and extracted views. If you change these views the simulation results will be re-filtered to show results from these views.

Simulation Data

The *~R Simulation Data* sections in the Parasitic Comparison form contains a tree level structure that displays information on three levels:

1. The top level contains the **history entry name** (for example `Interactive.4`).

   Note: See also the Data assistant in the *Virtuoso ADE Assembler User Guide* which will also lists the history entry name.

2. The top level expands to detail a **list of tests** that have been run (again this information is detailed in the *Data* assistant).

   Note: If a single test is used to simulate both parasitic/LDE and extracted views, but only one view had one instance of the design in the test then you will have to simulate twice, changing the configuration in between (which is done using the *Run Simulation* assistant - see the Virtuoso ADE Explorer User Guide. This is so that the first simulation will be bound to the parasitic/LDE view, and the second bounded to the extracted view or vice versa.

3. Expanding each test displays a **list of instance paths** to the estimate/extracted views (normally this will only be one instance path as there will only be one instance of a particular cellview in a design).

   Note: It is not necessary for instance paths to match (between estimated and extracted *~R Simulation Data*).

© 2023 All Rights Reserved.

Comparing Estimated Parasitics With Extracted Parasitics - Summary

1. If you want to compare ~R parasitics you need to perform a DC oppoint simulation for both the estimated and extracted views to be used.
2. Open the schematic (*design under test* (DUT)) or the configuration used for simulation.
3. Descend to where the parasitic estimates were placed.
4. Select *Parasitics/LDE – Compare*.

   This will display the Compare Parasitics form.
5. Specify a parasitic *Comparison Method* so that the estimates will either represent a limit (*Estimates are upper limits*) or a target (*Estimates are targets with tolerance*).

   If you select estimates to be a target you must specify a positive (upper) limit percentage and a negative (lower) percentage tolerance within which extracted parasitics must fall.
6. Select which classes of *Parasitics to Compare* (*~R*, *Coupled C*, *Decoupled C,* and/or *self C*).

   Note: If you do not select *~R*, the *~R Simulation Data* fields will be disabled.
7. Specify the estimated and extracted views to be used for parasitic comparisons and the *~R Simulation Data*, if necessary (see also Simulation Data).
8. Click *OK* to perform the parasitic comparison.

   This will generate a compare report (see Comparison Reports) which will list the estimates along with extracted value and status (*PASS* or *FAIL*).

© 2023 All Rights Reserved.

Comparing Against a Refined Extracted View

Rather than use the original extracted view you can also compare estimates against a refined extracted view.

In this case, the *Estimated View Simulation Data for R Calculation* section in the Compare Parasitics form requires that you specify the refined view to be simulated rather than the original, extracted view.

Comparing Capacitance

Capacitance is compared by summing all coupled capacitors and summing all decoupled capacitors on each net. Nets that have no estimates will be ignored.

- For **limit** estimates, if the actual parasitics are less than or equal to the estimated capacitance then the *Status* is marked green (*PASS)*.
- For **target** estimates, if the actual parasitics are within the specified tolerance of the estimated capacitance then the *Status* is marked green (*PASS)*.
- Otherwise, *Status* will be marked red (*FAIL)*.

Comparing Resistance

Parasitic effectiveR values are calculated for all nets in the estimated and extracted views.

- For **limit** estimates, if the actual effectiveR is less than or equal to the estimated value, the *Status* is marked as green (*PASS)*.
- For **target** estimates, if the actual effectiveR is within the specified tolerance of the estimated effectiveR, then the *Status* is marked as green (*PASS)*.
- Otherwise, *Status* will be marked as red (*FAIL)*.

© 2023 All Rights Reserved.

The Parasitic Mode Toolbar

Use the *Parasitic Mode* toolbar to change the parasitic mode so that you can update all tests to point to the parasitic view for that mode. This means that you can simulate with estimated parasitics, extracted parasitics, or no parasitics, just by switching modes.

In addition, all device parameters are modified to use the parasitic/LDE view when in *Schematic Estimates (Parasitics)* mode. This means that you can sweep device parameters in the presence of your estimated parasitics.

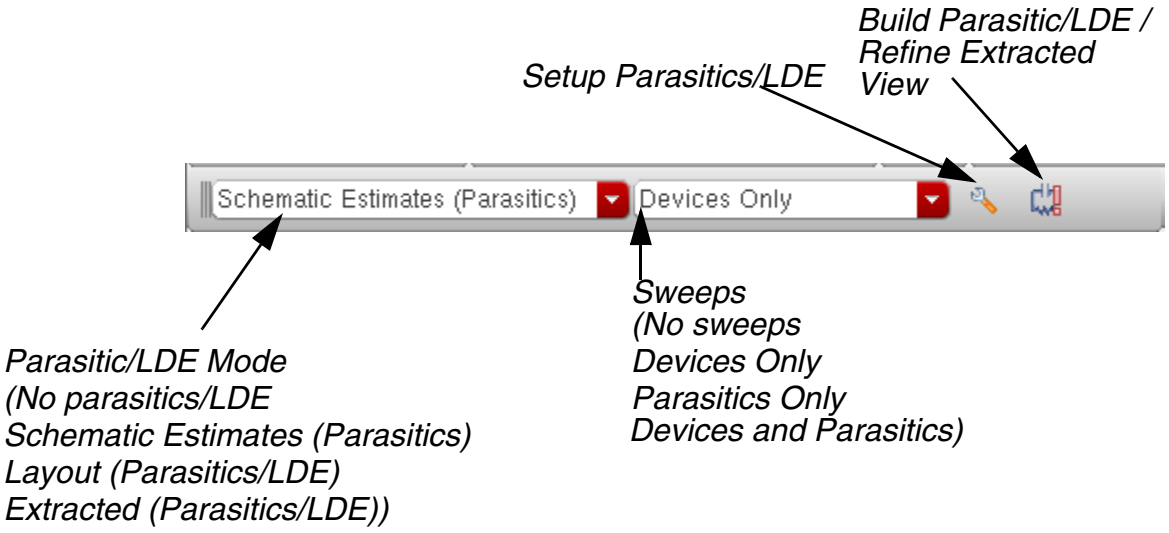


Figure 2-45 The Parasitic Mode Toolbar

Note: Any signals to be saved or plotted are automatically mapped to the estimated/extracted view when you change to either the *Schematic Estimates* or *Extracted* run modes.

Whenever you change the parasitic mode, the config view of the testbench is automatically updated to bind the cell to an appropriate view. For example, for the Extracted parasitic mode, the cell is bound to the available extracted view. Similarly, for the Layout parasitic mode, it is bound to the available netlist_layout view. A corresponding warning message is also displayed in the CIW.

- When *Parasitics Mode* is set to ***Schematic Estimates (Parasitics)***, the *Setup Parasitics* option will display the Setup form (Parasitics/LDE – Setup), and the *Build Parasitic/LDE View* button will become active. Selecting the *Build Parasitic/LDE View* button will display the Build Parasitic/LDE View form.

  Note:

© 2023 All Rights Reserved.

- A parasitic/LDE view will be built automatically if one does not already exist (when *Schematic Estimates* mode is set in the *Parasitics Mode* toolbar).
- A *Parasitic/LDE* view will also automatically be built (if it does not already exist) when the Setup form is OK'd, or if you choose to *Run Simulation* from the *Simulation* toolbar. For more information on *Run Simulation* see the *Virtuoso Analog Design Environment User Guide*.
- A parasitic/LDE view will be removed and rebuilt when you agree to overwrite it after it has been flagged as being out of date.

- When *Parasitic/LDE Mode* is set to ***Layout (Parasitics/LDE)***, depending on the setup in the Setup Parasitics and LDE form, one or more of the following are extracted from the layout or constraint view:
  - The layout dependent effects from a layout view or from modgen constraints
  - Parasitics from the layout view
- When *Parasitic/LDE Mode* is set to ***Extracted (Parasitics/LDE)***, the *Refine Extracted View* button will become active. Selecting the *Refine Extracted View* button will display the Refine Extracted View form (Refining the Extracted View).

  Note: The extracted flow is only available when you are at or below the schematic specified in the Setup *Parasitic/LDE* form.

For example, to simplify the simulation of parasitic estimates:

1. Select *Window – Toolbars – Parasitic Mode* to display the *Parasitic Mode* toolbar if it is not already on view.
2. Select *Schematic Estimates (Parasitics)* from the *Parasitics* pull-down.

   This will enable the other options on the *Parasitic Mode* toolbar.
3. Select the *Setup Parasitics* button to display the Setup form.
4. Complete the Setup form as required (see Parasitics/LDE – Setup).

   You can now create the parasitic/LDE view.
5. Select the *Build Parasitic/LDE view* button to display the Build Parasitic/LDE View form.
6. Complete the Build Estimated View form as required (see Parasitics/LDE – Create Filters).

Once you have successfully built the estimated view, you can open the testbench configuration to confirm that it has been re-bound to the estimated view. You can then choose

© 2023 All Rights Reserved.

to *Run Simulation* from the *Simulation* toolbar. For more information on *Run Simulation* see the *Virtuoso Analog Design Environment XL User Guide*.

© 2023 All Rights Reserved.

Applying Sweeps from the Parasitic Mode Toolbar

The *Sweeps* drop-down option, in the *Parasitic Mode Toolbar*, allows you to choose whether to sweep parasitics and/or device parameters during simulation. This allows you, for example, to sweep estimates during a sensitivity analysis to find out how sensitive your design is to parasitics.

Figure 2-46 Sweeps on the Parasitic Mode Toolbar

Note: Parasitic sweeps are only supported in *Schematic Estimates (Parasitics)* mode. It is not possible to sweep extracted parasitics.

Virtuoso Parasitic Aware Design creates parasitic parameters in the *Parasitics* tab of the *Variables and Parameters* assistant (for information about this assistant, see the Virtuoso Analog Design Environment XL User Guide) for each parasitic sweep that you have entered (see Entering Parasitic Sweeps).

Selecting a value from the *Sweeps* drop-down, on the *Parasitic Mode Toolbar*, controls device and parasitic parameters as follows:

- *No sweeps*

  All parasitic and device parameters are disabled. This option disables all sweeps.

- *Parasitics Only*

  Device parameters are disabled, parasitic parameters are enabled. Use this option to sweep parasitic estimates only.

- *Devices Only*

  Device parameters are enabled, parasitic parameters are disabled. Use this option to sweep device parameters only.

- *Devices and Parameters*

  All parameters are enabled.

For information about enabling or disabling individual parasitic parameters, see Enabling and Disabling Parasitic Sweeps on page 166.

© 2023 All Rights Reserved.

Note: Global variables are also enabled or disabled as appropriate if they specify a sweep. Global variables that appear in estimates are treated as parasitic parameters. Variables that have been used as a device parameter in the schematic are enabled and disabled along with other device parameters.

© 2023 All Rights Reserved.

Parasitic Probing and Ultrasim

It should be noted that from MMSIM 7.1 onwards, Ultrasim includes, by default, certain reductions on parasitics in the netlist. While this can provide an improvement on the simulation time, a potential side-effect is that parasitic probing may not work on all parts of a design.

This issue can be avoided by controlling Quantus QRC parasitic reduction in the QRC Parasitic Extraction form, ensuring that the reductions are done at this point. To achieve this, you should enable the *Reduce Parasitics* option and specify a *Reduction Frequency*. For more information see the Quantus QRC Extraction Users Manual.

Note: An additional method to avoid this issue is to prevent Ultrasim performing the reductions by requiring it save all the node information. This can be done using *Outputs – Save All* in the Test Editor.

Important

If you set "`.usim_opt preserve=1`", in Spectre, this can force Ultrasim to preserve all resistors (for more information, see the *Excluding Resistors and Capacitors from RC Reduction* section of the *Virtuoso Ultrasim User Guide*.

© 2023 All Rights Reserved.

© 2023 All Rights Reserved.

A

Parasitic Aware Design Environment Variables

This appendix describes the environment variables that control the characteristics of Virtuoso Parasitic Aware Design. You can customize the operation and behavior of Virtuoso Parasitic Aware Design features and forms by changing the values of particular environment variables. The default value of each variable appears in the syntax descriptions.

See the following sections for more information:

- msps
  - elaborateUsingConfig
  - stopViewList
  - switchViewList
  - userDefinedCDFUpdater
- msps.backAnnotate
  - effectiveRWarnLimit
  - fontSize
  - xOffset
  - yOffset
  - sortBy
  - parasiticFile
- msps.buildAnalog
  - setParasitics
  - analogExtractedViewName
- msps.mode

© 2023 All Rights Reserved.

- disableAutoCreate
- disableAutoUpdate

- msps.parProbe
  - maxListSize
  - sortBy
  - parasiticProbeFile
- mspsAv.backAnnotate
  - sortBy
  - parasiticFile
- mspsAv.buildAnalog
  - setParasitics
  - analogExtractedViewName
- mspsAv.instProbe
  - instProbeFile
- mspsAv.options
  - sortBy
  - extNetGrouped
  - fontSize
  - xOffset
  - yOffset
  - maxConsecutiveMessages
  - displayExtNetNames
  - pResCompName
  - pCapCompName
  - pIndCompName
  - pMindCompName
- mspsAv.parProbe

© 2023 All Rights Reserved.

- parasiticProbeFile
- showR
- showCoupledC
- showDecoupledC
- showSelfC
- showL
- showK

- mspsAv.p2p
  - useReducer
- mspsAv.refine
  - libraryName
  - scaleR
  - scaleL
  - scaleC
  - stitchParasitics
- mspsAv.simProbe
  - saveOnlyExternalNodes
  - saveOnlyOneGroupInstNode
- msps.layout
  - expandSchematicDevices
  - fingeringNames
  - gndCoupledCapToUnboundNets
  - ignoreBackAnnotatedDummyDevices
  - includeLayoutParasitics
  - includeLdeParameters
  - includeSchematicEstimates
  - individualInstCdfCallbacks

© 2023 All Rights Reserved.

- layerMapFile
- ldeIgnoreModels
- ldeParameterCacheEnabled
- ldeParameterSource
- ldeParameterTool
- ldeSwitchSourceDrainTerms
- lvsRuleFile
- mfactorNames
- mixSchEstWithLayoutParasitics
- netlistView
- referenceNet
- singleFactorExpansion

■ msps.setup

- showAllCellViews
- netlistViewType
- useNewSetupForm
- ignoreLVSInstForStitching
- lxRemoveDeviceForStitching

■ msps.stitch

- reduceParallelCaps

■ msps.estimates

- customRName
- customRLib
- customRCell
- customRView
- customLName
- customLLib

© 2023 All Rights Reserved.

- customLCell
- customLView
- customKName
- customKLib
- customKCell
- customKView
- customCName
- customCLib
- customCCell
- customCView
- defaultR
- defaultR
- defaultL
- defaultK
- defaultCC
- defaultDC
- viewName
- scaleR
- scaleL
- scaleC
- detailReport
- reportFile

■ layoutXL

- svDisplayResistance
- svDisplayInductance
- svDisplayCapacitance
- svDisplayNodeName

© 2023 All Rights Reserved.

- svDotWidth
- svResistanceThresholdMin
- svResistanceThresholdMax
- svCapacitanceThresholdMin
- svCapacitanceThresholdMax
- svInductanceThresholdMin
- svInductanceThresholdMax
- svInductorStyle

- maestro.test
  - autoSyncMPC

© 2023 All Rights Reserved.

msps

elaborateUsingConfig

Specifies that the design under test is to be elaborated by using the config view of the ADE Assembler test. By default, the design under test (DUT) is elaborated by using the switchViewList and stopViewList environment variables. Set the *elaborateUsingConfig* variable to `t` to use the test config view.

Note: Even if this variable is set to `t`, the test config view is used for elaboration only if all the tests in the ADE Assembler views point to the same config view. This helps in avoiding potential conflicts from different config views. For example, configA may elaborate the DUT in one way, but configB may have a different binding or switch view list which would elaborate the DUT in a different way. If the tests point to different config views or if any test is not bound to a config view, the design is elaborated in the default way.

In `.cdsenv`:

```
msps elaborateUsingConfig boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps" "elaborateUsingConfig" 'boolean nil)
```

Valid Values:

`t`	Elaborates the DUT by using the config view of the ADE Assembler test.
`nil`	Elaborates the DUT by using the switchViewList and stopViewList environment variables.

Default Value: `nil`

stopViewList

Specifies a space-separated list of cellview names that is used while elaborating the design under test to identify the view at which the traversal has to be stopped. While traversing a

© 2023 All Rights Reserved.

design hierarchy, the tool stops moving further when it finds any one of the cellviews given in this list.

In `.cdsenv`:

```
msps stopViewList string "spectre symbol"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps" "stopViewList" 'string "spectre symbol")
```

Valid Values: A space-separated list of cellview names

Default Value: `"spectre symbol"`

switchViewList

Specifies a space-separated list of cellview names that is used while elaborating the design under test to in control the order in which the design hierarchy is traversed.

In `.cdsenv`:

```
msps switchViewList string "schematic verilog symbol"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps" "switchViewList" 'string "spectre cmos_sch cmos.sch
     schematic veriloga ahdl symbol")
```

Valid Values: A space-separated list of cellview names

Default Value: `"spectre cmos_sch cmos.sch schematic veriloga ahdl symbol"`

© 2023 All Rights Reserved.

userDefinedCDFUpdater

Specifies a callback function to be called after the creation of each instance in the netlist_layout view. You can define the callback function to modify certain CDF parameters of the newly created instances.

In `.cdsenv`:

```
msps userDefinedCDFUpdater string "myFunction"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps" "userDefinedCDFUpdater" 'string "myFunction")
```

Valid Values:

A string value, which is the name of an existing the user-defined function.

Default Value: `""`

GUI Equivalent: None

© 2023 All Rights Reserved.

msps.backAnnotate

effectiveRWarnLimit

Specifies the limit beyond which effective R values should be considered invalid. The invalid effective R values will be displayed in a different color on the schematic.

In `.cdsenv`:

```
msps.backAnnotate effectiveRWarnLimit float 10000.0
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.backAnnotate" "effectiveRWarnLimit" 'float 10000.0)
```

Valid Values: Any floating point number

Default Value: `10000.0`

fontSize

Specifies the label font size for displaying parasitic backannotation.

In `.cdsenv`:

```
msps.backAnnotate fontSize float 0.05
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.backAnnotate" "fontSize" 'float 0.05)
```

Valid Values: Any floating point number

Default Value: `0.05`

© 2023 All Rights Reserved.

xOffset

Sets the horizontal offset from the center of the net when displaying parasitics.

In `.cdsenv`:

```
msps.backAnnotate xOffset float 0.01
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.backAnnotate" "xOffset" 'float 0.01)
```

Valid Values: Any floating point number

Default Value: `0.01`

yOffset

Sets the vertical offset from the center of the net when displaying parasitics.

In `.cdsenv`:

```
msps.backAnnotate yOffset float 0.01
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.backAnnotate" "yOffset" 'float 0.01)
```

Valid Values: Any floating point number

Default Value: `0.01`

sortBy

Specifies the sorting method to be used when probing parasitics on nets.

In `.cdsenv`:

```
msps.backAnnotate sortBy string "C"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.backAnnotate" "sortBy" 'string "C")
```


© 2023 All Rights Reserved.

Valid Values:	C – Sorts by capacitance parasitics R – Sorts by resistance parasitics
Default Value:	`0.01`

parasiticFile

pecifies the file name for saving the parasitic reports.

In `.cdsenv`:

```
msps.backAnnotate parasiticFile string "parasitic_file"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.backAnnotate" "parasiticFile" 'string
     "parasitic_file")
```

Valid Values:	A string value.
Default Value:	`parasitic_file`

© 2023 All Rights Reserved.

msps.buildAnalog

setParasitics

Specifies the extracted prasitics to be included while building a refined extracted view.

In `.cdsenv`:

```
msps.buildAnalog setParasitics string "Include All"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.buildAnalog" "setParasitics" 'string "Include All")
```

Valid Values:
`None`: No extracted parasitics will be included

`Select From Schematic`: Extracted parasitics will be manually selected from the schematic

`Include All`: All the extracted parasitics will be included

Default Value: `parasitic_file`

analogExtractedViewName

Specifies the name of the extracted view.

In `.cdsenv`:

```
msps.buildAnalog analogExtractedViewName string "analog_extracted"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.buildAnalog" "analogExtractedViewName" 'string
     "analog_extracted")
```

Valid Values: Any string value.

Default Value: `analog_extracted`

© 2023 All Rights Reserved.

msps.mode

disableAutoCreate

By default, the specified parasitic/LDE view is automatically created if a view with the same name does not exist, and you do any of the following:

- Select the *Schematic Estimates* mode in the *Parasitic Mode* toolbar.
- Click *OK* or *Apply* in the Setup Parasitics form, or run a simulation when the *Schematic Estimates* mode is selected in the *Parasitic Mode* toolbar.

Use this environment variable to control whether the parasitic/LDE view is automatically created.

In `.cdsenv`:

```
msps.mode disableAutoCreate boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal( "msps.mode" "disableAutoCreate" 'boolean t)
```

Valid Values:

`t`	Disables automatic creation of the parasitic/LDE view.
`nil`	Enables automatic creation of the parasitic/LDE view.

Default Value: `nil`

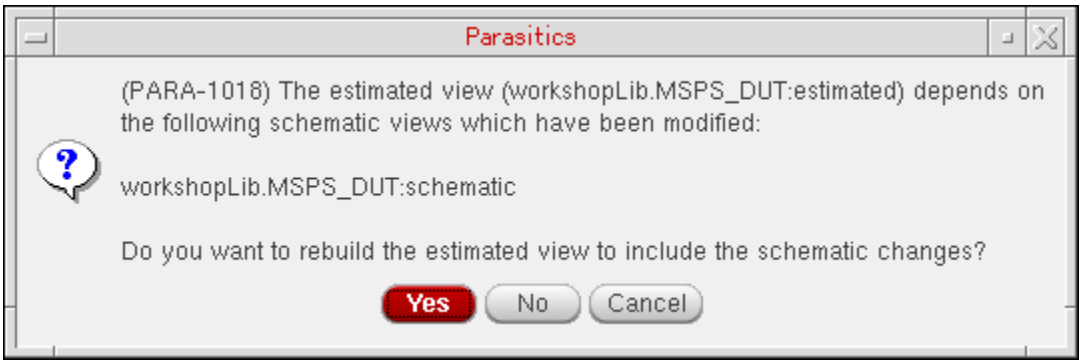
disableAutoUpdate

By default, the following message box appears prompting you to rebuild an parasitic/LDE view that it is out-of-date with respect to the schematic or constraint view it depends on, when you do any of the following:

- Select the *Schematic Estimates* mode in the *Parasitic Mode* toolbar
- Click *OK* or *Apply* in the Setup Parasitics form, or run a simulation when the *Schematic Estimates* mode is selected in the *Parasitic Mode* toolbar.

© 2023 All Rights Reserved.

Use this environment variable to control whether parasitic/LDE views are automatically updated.



In `.cdsenv`:

```
msps.mode disableAutoUpdate boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal( "msps.mode" "disableAutoUpdate" 'boolean t)
```

Valid Values:

`t`	Disables automatic update of the parasitic/LDE view. The message box prompting you to rebuild the parasitic/LDE view will not appear.
`nil`	Enables automatic update of the parasitic/LDE view. The message box prompting you to rebuild the parasitic/LDE view will appear. Click *Yes* to rebuild the parasitic/LDE view.

Default Value: `nil`

© 2023 All Rights Reserved.

msps.parProbe

maxListSize

Specifies the maximum list size of parasitic reports.

In `.cdsenv`:

```
msps.parProbe maxListSize int "20"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.parProbe" "maxListSize" 'int "20")
```

Valid Values: A positive integer value.

Default Value: `20`

sortBy

Specifies whether the list of parasitic report table is to be sorted by capacitance parasitics or resistance parasitics.

In `.cdsenv`:

```
msps.parProbe sortBy string "C"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.parProbe" "sortBy" 'string "C")
```

Valid Values: C/R

Default Value: `C`

© 2023 All Rights Reserved.

parasiticProbeFile

Specifies the file name for saving the parasitic report.

In `.cdsenv`:

```
msps.parProbe parasiticProbeFile string "parasitic_probing"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.parProbe" "parasiticProbeFile" 'string
     "parasitic_probing")
```

Valid Values: A string value for the file

Default Value: `parasitic_probing`

© 2023 All Rights Reserved.

mspsAv.backAnnotate

sortBy

Specifies the sorting method to be used when probing parasitics on nets.

In `.cdsenv`:

```
mspsAv.backAnnotate sortBy string "C"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.backAnnotate" "sortBy" 'string "C")
```

Valid Values:
- `C` – Sorts by capacitance parasitics
- `R` – Sorts by resistance parasitics

Default Value: `includeAll`

parasiticFile

Specifies the file name for saving the parasitic reports.

In `.cdsenv`:

```
mspsAv.backAnnotate parasiticFile string "parasitic_file"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.backAnnotate" "parasiticFile" 'string
     "parasitic_file")
```

Valid Values: A string value.

Default Value: `parasitic_file`

© 2023 All Rights Reserved.

mspsAv.buildAnalog

setParasitics

Specifies the extracted parasitic to be included while building a refined extracted view.

In `.cdsenv`:

```
mspsAv.buildAnalog setParasitics string "includeAll"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.buildAnalog" "setParasitics" 'string "includeAll")
```

Valid Values: A string value

Default Value: `includeAll`

analogExtractedViewName

Specifies the name of the extracted view.

In `.cdsenv`:

```
mspsAv.buildAnalog analogExtractedViewName string
    "av_analog_extracted"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.buildAnalog" "analogExtractedViewName" 'string
     "av_analog_extracted")
```

Valid Values: A string value

Default Value: `av_analog_extracted`

© 2023 All Rights Reserved.

mspsAv.instProbe

instProbeFile

Specifies the file name for saving the parasitic report of a design instance.

In `.cdsenv`:

```
mspsAv.instProbe instProbeFile string "instance_probing"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.instProbe" "instProbeFile" 'string
     "instance_probing")
```

Valid Values: A string value.

Default Value: `instance_probing`

© 2023 All Rights Reserved.

mspsAv.options

sortBy

Specifies the initial sorting method (Instance, Type, Value, From, or To) to be used when probing parasitics on nets.

In `.cdsenv`:

```
mspsAv.options sortBy string "Instance"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.options" "sortBy" 'string "Include")
```

Valid Values:
- `Instance`
- `Value`
- `From`
- `To`

Default Value: `Instance`

extNetGrouped

Specifies whether parasitics not to be reported for the whole design net when probing an extracted net (this will work as if probing in the schematic).

In `.cdsenv`:

```
mspsAv.options extNetGrouped boolean nil
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.options" "extNetGrouped" 'boolean nil)
```

Valid Values:
- `t` – Reports parasitics for the whole design net
- `nil` – Otherwise

Default Value: `nil`

© 2023 All Rights Reserved.

fontSize

Specifies the label font size for displaying parasitic backannotation.

In `.cdsenv`:

```
msps.options fontSize float "0.05"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.options" "fontSize" 'float "0.05 ")
```

Valid Values:	A floating-point number.
Default Value:	`0.05`

xOffset

Sets the horizontal offset from the center of the net when displaying parasitic aware design options.

In `.cdsenv`:

```
mspsAv.backAnnotate xOffset float "0.0"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.backAnnotate" "xOffset" 'float "0.0")
```

Valid Values:	A floating-point number.
Default Value:	`0.01`

© 2023 All Rights Reserved.

yOffset

Sets the vertical offset from the center of the net when displaying parasitic aware design options.

In `.cdsenv`:

```
mspsAv.options yOffset float "0.0"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.options" "yOffset" 'float "0.0")
```

Valid Values: A floating-point number.

Default Value: `0.0`

maxConsecutiveMessages

Sets the number of error messages displayed while using the *Parasitics* menu options.

In `.cdsenv`:

```
mspsAv.options maxConsecutiveMessages int "5"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.options" "maxConsecutiveMessages" 'int "5")
```

Valid Values: A positive integer.

Default Value: `5`

© 2023 All Rights Reserved.

displayExtNetNames

Controls the display of the names of nets connected to terminals of parasitic instances in the extracted view.

In `.cdsenv`:

```
msps.options displayExtNetNames boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.options" "displayExtNetNames" 'boolean t)
```

Valid Values:
- `t` - Displays the names of the nets
- `nil` - The names are hidden

Default Value: `t`

pResCompName

Specifies a user-defined parasitic resistance component name.

In `.cdsenv`:

```
mspsAv.options pResCompName string "resparc28hpcp"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.options" "pResCompName" 'string "resparc28hpcp")
```

Valid Values: A string value.

Default Value: `""`

© 2023 All Rights Reserved.

pCapCompName

Specifies a user-defined parasitic inductance component name.

In `.cdsenv`:

```
mspsAv.options pCapCompName string "pcapacitor"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.options" "pCapCompName" 'string "pcapacitor")
```

Valid Values:	A string value.
Default Value:	`""`

pIndCompName

Specifies a user-defined parasitic inductance component name.

In `.cdsenv`:

```
mspsAv.options pIndCompName string "inductance"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.options" "pIndCompName" 'string "inductance")
```

Valid Values:	A string value.
Default Value:	`""`

© 2023 All Rights Reserved.

pMindCompName

Specifies a user-defined parasitic mutual inductance component name.

In `.cdsenv`:

```
mspsAv.options pMindCompName string "minductance"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.options" "pMindCompName" 'string "minductance")
```

Valid Values:	A string value.
Default Value:	`""`

© 2023 All Rights Reserved.

mspsAv.parProbe

parasiticProbeFile

Specifies the file name for saving the parasitic report.

In `.cdsenv`:

```
mspsAv.parProbe parasiticProbeFile string "parasitic_probing"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.parProbe" "parasiticProbeFile" 'string
     "parasitic_probing")
```

Valid Values: A string value.

Default Value: `parasitic_probing`

showR

Enables the display of resistance parasitics in the parasitic reports.

Note: This variable is useful only for the parasitic reports generated from VSE L and VSE XL.

In `.cdsenv`:

```
mspsAv.parProbe showR boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.parProbe" "showR" 'boolean nil)
```

Valid Values: `t`/`nil`

Default Value: `t`

showCoupledC

Enables the display of coupled capacitance parasitics in the parasitic reports.

© 2023 All Rights Reserved.

Note: This variable is useful only for the parasitic reports generated from VSE L and VSE XL.

In `.cdsenv`:

```
mspsAv.parProbe showCoupledC boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.parProbe" "showCoupledC" 'boolean nil)
```

Valid Values: `t`/`nil`

Default Value: `t`

showDecoupledC

Enables the display of decoupled capacitance parasitics in the parasitic reports.

Note: This variable is useful only for the parasitic reports generated from VSE L and VSE XL.

In `.cdsenv`:

```
mspsAv.parProbe showDecoupledC boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.parProbe" "showDecoupledC" 'boolean nil)
```

Valid Values: `t`/`nil`

Default Value: `t`

showSelfC

Enables the display of self-capacitance (coupled capacitance between segments of the same net) in the parasitic reports.

© 2023 All Rights Reserved.

Note: This variable is useful only for the parasitic reports generated from VSE L and VSE XL.

In `.cdsenv`:

```
mspsAv.parProbe showSelfC boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.parProbe" "showSelfC" 'boolean nil)
```

Valid Values: `t`/`nil`

Default Value: `t`

showL

Enables the display of parasitic inductance in the parasitic reports.

Note: This variable is useful only for the parasitic reports generated from VSE L and VSE XL.

In `.cdsenv`:

```
mspsAv.parProbe showL boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.parProbe" "showL" 'boolean nil)
```

Valid Values: `t`/`nil`

Default Value: `t`

showK

Enables the display of mutual inductance parasitics in the parasitic reports.

© 2023 All Rights Reserved.

Note: This variable is useful only for the parasitic reports generated from VSE L and VSE XL.

In `.cdsenv`:

```
mspsAv.parProbe showK boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.parProbe" "showK" 'boolean nil)
```

Valid Values: `t`/`nil`

Default Value: `t`

© 2023 All Rights Reserved.

mspsAv.p2p

useReducer

Controls the usage of parasitic resistance reducer in parasitic probing.

In `.cdsenv`:

```
mspsAv.p2p useReducer boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.p2p" "useReducer" 'boolean t)
```

Valid Values:

- `t` – Enables the parasitic resistance reducer
- `nil` – Disables the parasitic resistance reducer

Default Value: `t`

© 2023 All Rights Reserved.

mspsAv.refine

libraryName

Specifies the name of the library to be used to create a netlist that includes refine schematic parasitic estimates.

In `.cdsenv`:

```
msps.refine libraryName string "av_extracted"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.refine" "viewName" 'string "av_analog_extracted")
```

Valid Values: Name of a valid library.

Default Value: `av_analog_extracted`

scaleR

Specifies the scale factor value for parasitic resistance.

In `.cdsenv`:

```
mspsAv.refine scaleR float "1.0"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.refine" "scaleR" 'float "1.0")
```

Valid Values: A floating-point number.

Default Value: `1.0`

© 2023 All Rights Reserved.

scaleL

Specifies the scale factor value for parasitic inductance.

In `.cdsenv`:

```
mspsAv.refine scaleL float "1.0"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.refine" "scaleL" 'float "1.0")
```

Valid Values: A floating-point number.

Default Value: `1.0`

scaleC

Specifies the scale factor value for parasitic capacitance.

In `.cdsenv`:

```
mspsAv.refine scaleC float "1.0"
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.refine" "scaleC" 'float "1.0")
```

Valid Values: A floating-point number.

Default Value: `1.0`

© 2023 All Rights Reserved.

stitchParasitics

Specifies whether all the parasitics are to be stitched in the extracted view.

In `.cdsenv`:

```
mspsAv.refine stitchParasitics cyclic "None"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.refine" " stitchParasitics " 'cyclic "None")
```

Valid Values:

- `None` – Parasitics are not stitched
- `All` – All the parasitics are stitched together

Default Value: `none`

© 2023 All Rights Reserved.

mspsAv.simProbe

saveOnlyExternalNodes

Adds only external net fragments for parasitic nets in the extracted view to the netlist `save` statement.

In `.cdsenv`:

```
mspsAv.simProbe saveOnlyExternalNodes boolean nil
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.simProbe" "saveOnlyExternalNodes" 'boolean nil)
```

Valid Values: `t`/`nil`

Default Value: `t`

Note: You can set this variable to `nil` when you want to save selected net signals in ADE and then to probe internal nodes in the extracted view.

saveOnlyOneGroupInstNode

Saves all fragments of the saved nets as only one fragment in the netlist.

Note: This environment variable is applicable only for extracted views and Smart Views.

By default all fragments of the saved nets are printed in the netlist. When `saveOnlyOneGroupInstNode` is set to `t`, all these fragments of saved nets are printed as only one fragment in the netlist. When the variable is set to its default value `nil`, all fragments are saved in the netlist.

Consider a sample netlist that contains the following fragments of saved nets:

```
save OUTP OUTM I0.net044 I0.net044\#1 I0.net044\#10 I0.net044\#100 \
I0.net044\#101 I0.net044\#102 I0.net044\#103 I0.net044\#104 \
I0.net044\#105 I0.net044\#106 I0.net044\#107 I0.net044\#108 \
I0.net044\#109 I0.net044\#11 I0.net044\#110 I0.net044\#111 \
….
I0.net044\#95 I0.net044\#96 I0.net044\#97 I0.net044\#98 I0.net044\#99 \
```


© 2023 All Rights Reserved.

Now, set `saveOnlyOneGroupInstNode` to `t` and create a netlist. The netlist now shows only one fragment instead of the multiple fragments of saved nets:

```
save OUTP OUTM I0.net044\#68 I0.net035\#13 I0.net22\#204 I0.net14\#127
I0.net036\#18
```

The result of this environment variable also depends on the settings that you specify for the *Select signals to output (save)* option on the Save Options form. On setting this option to:

- `all`: all fragments are saved in the Results Browser.
- `selected`: only one netlisted fragment is saved in the Results Browser.

Additional Information

- You can access the Save Options form by using the *Save All* command from the context-sensitive menu of an output.
- You can set this variable to `nil` when you want to save selected net signals in ADE and then to probe internal nodes in the extracted view.

In `.cdsenv`:

```
mspsAv.simProbe saveOnlyOneGroupInstNode boolean nil
```

In `.cdsinit` or the CIW:

```
envSetVal("mspsAv.simProbe" "saveOnlyOneGroupInstNode" 'boolean t)
```

Valid Values: `t` / `nil`

Default Value: `nil`

© 2023 All Rights Reserved.

msps.layout

Important

This set of variables is used only with the LDE re-simulation flow.

- createMaxCapDuringTransfer
- expandSchematicDevices
- fingeringNames
- ignoreBackAnnotatedDummyDevices
- includeLayoutParasitics
- includeLdeParameters
- includeSchematicEstimates
- individualInstCdfCallbacks
- ldeParameterCacheEnabled
- ldeParameterSource
- ldeParameterTool
- stitchFloatingNets
- lvsRuleFile
- mixSchEstWithLayoutParasitics
- mfactorNames
- netlistView
- referenceNet

Note: This following variables configure only the initial state of their corresponding fields in the *Options for Layout Mode* section of the Parasitics and LDE Setup form:

- expandSchematicDevices
- ignoreBackAnnotatedDummyDevices
- includeLayoutParasitics
- includeLdeParameters

© 2023 All Rights Reserved.

- includeSchematicEstimates
- ldeParameterSource
- netlistView
- referenceNet

After you set up the values in the Parasitic and LDE Setup form for a particular design under test, these variables have no effect on that DUT. This helps you to save and use different setup for each design. After the setup is saved for a design, the values of these environment variables do not have any effect for that.

All the remaining environment variables in the list are global across all designs under test and ADE Assembler sessions.

© 2023 All Rights Reserved.

createMaxCapDuringTransfer

Specifies if the maximum capacitance for constraints should be created during the transfer of estimates to constraints.

In `.cdsenv`:

```
msps.layout createMaxCapDuringTransfer boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "createMaxCapDuringTransfer" 'boolean nil)
```

Valid Values:

t	The maximum capacitance will be created.
nil	The maximum capacitance will not be created.

Default Value: `t`

expandSchematicDevices

Specifies if it is required to expand the devices with multiple factors before generating a netlist. Expanding m-factor devices allows the parasitic network to each individual device in the layout to be connected to a corresponding device in the parasitic/LDE view. If the option is disabled, the network to each device will be shorted to a single device in the parasitic/LDE view.

This option is used to gain improved accuracy while simulating with layout parasitics when LDE parameters are not enabled.

Note: When LDE parameters are extracted, this option is not used because devices will automatically be expanded according to the fingers and m-factor used in the layout.

In `.cdsenv`:

```
msps.layout expandSchematicDevices boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "expandSchematicDevices" 'boolean t)
```

Valid Values:

`t`	Expands the devices with multiple factors or fingers before generating a netlist.

© 2023 All Rights Reserved.

	`nil`	Does not expand the devices with multiple factors or fingers before generating a netlist.

Default Value: `t`

GUI Equivalent:

Command: *Parasitics/LDE – Setup Parasitics and LDE*

Field: *Expand devices with m-factor/fingers*

fingeringNames

Specifies the names of the CDF parameters that define fingers for a cell. When LDE parameters are enabled and the expandSchematicDevices variable is set to t, parallel devices are created in the parasitic/LDE view for each finger used in the layout or MODGEN constraint. The fingering parameters, identified using the names specified in this variable, on the expanded devices are reset to `1`.

If the *`fingeringNames`* variable is set to a blank string (default value), the fingering parameter names are taken from the *`layoutXL.lxFingeringNames`* environment variable.

In `.cdsenv`:

```
msps.layout fingeringNames string "numFingers numFinger"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "fingeringNames" 'string "numFingers
     numFinger")
```

Valid Values:

A string with space-separated finger parameter names.

Default Value: `""`

GUI Equivalent:

Command: *Parasitics/LDE – Setup Parasitics and LDE*

Field: *Device finger parameter names*

© 2023 All Rights Reserved.

gndCoupledCapToUnboundNets

Specifies whether the coupled capacitance in an unbound floating net in the parasitic/LDE view must be grounded.

In `.cdsenv`:

```
msps.layout gndCoupledCapToUnboundNets boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "gndCoupledCapToUnboundNets" 'boolean t)
```

Valid Values:

t	Grounds the coupled capacitance parasitics.
nil	Otherwise

Default Value: `t`

ignoreBackAnnotatedDummyDevices

Specifies if the dummy cells backannotated from the layout view to the schematic view are to be ignored while generating a netlist for simulation.

For more details on backannotated dummy cell instances, refer to Back Annotating Dummy Instances.

In `.cdsenv`:

```
msps.layout ignoreBackAnnotatedDummyDevices boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "ignoreBackAnnotatedDummyDevices" 'boolean
     t)
```

Valid Values:

`t`	Ignores backannotated dummy cells while generating a netlist.
`nil`	Includes the backannotated dummy cells form the layout view in the netlist.

Default Value: `t`

GUI Equivalent:

© 2023 All Rights Reserved.

Command: *Parasitics/LDE – Setup Parasitics and LDE*

Field: *Ignore dummies back-annotated to schematic*

includeLayoutParasitics

Specifies if the parasitics in the layout view are to be included in the netlist. The layout view is specified the *Layout view name for parasitics and LDE* field in the *Setup Parasitics and LDE* form.

In `.cdsenv`:

```
msps.layout includeLayoutParasitics boolean nil
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "includeLayoutParasitics" 'boolean nil)
```

Valid Values:

`t`	Includes parasitics from the layout view.
`nil`	Does not include parasitics from the layout view.

Default Value: `nil`

GUI Equivalent:

Command: *Parasitics/LDE – Setup Parasitics and LDE*

Field: *Include parasitics from – Layout*

includeLdeParameters

Specifies if the LDE parameters are to be included in the netlist. If this variable is set to `t`, Virtuoso includes the LDE parameters specified by the ldeParameterSource variable or the .

In `.cdsenv`:

```
msps.layout includeLdeParameters boolean nil
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "includeLdeParameters" 'boolean nil)
```

Valid Values:

© 2023 All Rights Reserved.

	`t`	Includes LDE parameters specified by ldeParameterSource in the netlist.
	`nil`	Does not include LDE parameters.

Default Value: `nil`

GUI Equivalent:

Command: *Parasitics/LDE – Setup Parasitics and LDE*

Field: *Include LDE from – None*

includeSchematicEstimates

Specifies if the parasitic estimates from the Parasitics & Electrical Setup assistant are to be included in the netlist. These are the same estimates that are used in the Schematic Estimates (Parasitics) mode.

When you set both includeSchematicEstimates and includeLdeParameters to `t`, with this combination, you can use LDE parameters with estimated parasitics.

In `.cdsenv`:

```
msps.layout includeSchematicEstimates boolean nil
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "includeSchematicEstimates" 'boolean nil)
```

Valid Values:

	`t`	Includes parasitic estimates from the schematic view of the design. These estimates are created in the Parasitics & Electrical Setup assistant.
	`nil`	Does not include parasitic estimates from the schematic view.

Default Value: `nil`

GUI Equivalent:

Command: *Parasitics/LDE – Setup Parasitics and LDE*

Field: *Include parasitics from – Schematic Estimates*

© 2023 All Rights Reserved.

individualInstCdfCallbacks

When expanding an m-factor or fingered device in the parasitic/LDE view, the tool executes a CDF callback associated with that device. This variable specifies if the CDF callback is to be executed for each instance of a device or only once for a device.

By default, this variable is set to `nil` and while building a parasitic/LDE view, Virtuoso does not execute the CDF callback for each instance of a device. Instead, it executes the callback only once for a device and the result is applicable to all the instances. This helps in optimizing the build process.

To execute the callback for each cell instance separately, set this variable to `t`.

In `.cdsenv`:

```
msps.layout individualInstCdfCallbacks boolean nil
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "individualInstCdfCallbacks" 'boolean nil)
```

Valid Values:

`t`	Executes the CDF callback for each instance of a device separately.
`nil`	Executes the CDF callback for a device only once.

Default Value: `nil`

GUI Equivalent: None

layerMapFile

Specifies the path to the layer map file required by PVS to extract LDE device parameters.

Note: This variable is required for the LDE re-simulation flow.

In `.cdsenv`:

```
msps.layout layerMapFile string "./tech/GPDK045/gpdk045/
     gpdk045.layermap"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "layerMapFile" 'string "./tech/GPDK045/
     gpdk045/gpdk045.layermap")
```


© 2023 All Rights Reserved.

Valid Values:	
	A string path to the layer map file.
Default Value:	""
GUI Equivalent:	None

© 2023 All Rights Reserved.

ldeIgnoreModels

Specifies a space-separated list of model names or name patterns to be checked by the LDE flow to ignore devices extracted by PVS. If any device extracted by PVS has a model name that matches an entry specified in this list, it is ignored from the netlist view to be used for simulation.

Note: This variable is required for the LDE re-simulation flow.

In `.cdsenv`:

```
msps.layout ldeIgnoreModels string "model1_* model2_* model3_*"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "ldeIgnoreModels" 'string "model1_*
     model2_* model3")
```

Valid Values:

A space-separated list of model names or name patterns. You can use * as a wildcard in the name pattern.

Default Value: `""`

GUI Equivalent: None

ldeParameterCacheEnabled

Keeps the extracted LDE parameters from MODGEN constraints and the layout view in cache so that they can be reused while building the parasitic/LDE view next time, if not changed. This saves time and optimizes the netlist generation process.

Every time Virtuoso builds a parasitic/LDE view, it checks for the constraints that have changed since the last extraction and extracts only the changed or new ones.

For the LDE parameters from the layout view, if the layout has been modified, it re-extracts all the parameters again. Otherwise, it uses the parameters from the cache.

This variable is useful only for the LDE re-simulation flow.

© 2023 All Rights Reserved.

Important

The cache is used only when an automatic re-build is triggered. For example, when you start an ADE Assembler simulation and the tool detects that the parasitic/LDE view is out-of-date. The cache is ignored when you click *Build Parasitic/LDE view* to forcefully re-extract the parameters from MODGEN constraints or the layout view.

In `.cdsenv`:

```
msps.layout individualInstCdfCallbacks boolean nil
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "individualInstCdfCallbacks" 'boolean nil)
```

Valid Values:

`t`	Keeps the extracted LDE parameters in cache and reuses them for the next netlist generation.
`nil`	Does not maintain a cache.

Default Value: `nil`

GUI Equivalent: None

ldeParameterSource

Specifies the source of the LDE parameters. While generating a netlist, Virtuoso extracts the parameters from the specified source and adds them to the netlist.

Note: This variable is required for the LDE re-simulation flow.

In `.cdsenv`:

```
msps.layout ldeParameterSource cyclic "MODGEN"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "ldeParameterSource" 'cyclic "MODGEN")
```

Valid Values:

`"MODGEN"`	Extracts the LDE parameters from MODGEN constraints in the constraints view

© 2023 All Rights Reserved.

	`"layout"`	Extracts the LDE parameters from the layout view specified in the *Layout View* field in the Setup Parasitics and LDE form.
Default Value:	`"MODGEN"`	
GUI Equivalent:		
	Command:	*Parasitics/LDE – Setup Parasitics and LDE*
	Field:	*Include LDE from – Scratch layout(s) using MODGEN constraints from <constraints-view-name>* *Include LDE from – Layout View*

ldeParameterTool

If you have the environment variable, `CDS_LEA_EXTRACTION`, set so that the option appears on the Parasitics/LDE Setup form, then this variable will control which tool is selected by default.

In `.cdsenv`:

```
msps.layout ldeParameterTool cyclic "pvs"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "ldeParameterTool" 'cyclic "pvs")
```

Valid Values:

`pvs`	PVS will be selected.
`lea`	LEA will be selected.

Default Value: `pvs`

© 2023 All Rights Reserved.

stitchFloatingNets

When there are coupled capacitance in a floating net in the layout, setting this variable to `t` creates an equivalent for the floating net in the parasitic/LDE view and stiches all parasitics for it.

In `.cdsenv`:

```
msps.layout stitchFloatingNets boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "stitchFloatingNets" 'boolean t)
```

Valid Values:

`t`	Creates an equivalent floating net in the parasitic/LDE view.
`nil`	Does not create an equivalent floating net in the parasitic/ LDE view.

Default Value: `t`

ldeSwitchSourceDrainTerms

Specifies whether the source and drain terminals are to be swapped in the estimated netlist view of the LDE flow.

In `.cdsenv`:

```
msps.layout ldeSwitchSourceDrainTerms boolean nil
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "ldeSwitchSourceDrainTerms" 'boolean nil)
```

Valid Values:

`t`	The source and drain terminals are swapped.
`nil`	The source and drain terminals are not swapped.

Default Value: `nil`

lvsRuleFile

Specifies the path to the rule deck file required by PVS to extract LDE device parameters.

© 2023 All Rights Reserved.

Note: This variable is required for the LDE re-simulation flow.

In `.cdsenv`:

```
msps.layout lvsRulesFile string "./tech/GPDK045/gpdk045/pvs/
     pvlLVS.rul"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "lvsRulesFile" 'string "./tech/GPDK045/
     gpdk045/pvs/pvlLVS.rul")
```

Valid Values:

A string path to the rule deck file.

Default Value: `""`

GUI Equivalent: None

mfactorNames

Specifies the names of the CDF parameters that define m-factor for a cell.

While generating a netlist with parasitics, if the tool finds any cell with the given parameter names, it creates multiple copies of the cell if the expandSchematicDevices variable is set to `t`.

Alternatively, when includeLdeParameters is set to `t`, parallel devices are created in the parasitic/LDE view for each m-factor used in the layout or MODGEN constraint.

In both cases, the tool uses the names specified in this variable to identify the m-factor parameters on the expanded devices and resets them to `1`.

If the mfactorNames variable is set to a blank string "", the fingering parameter names are instead taken from the *`layoutXL.mFactorNames`* environment variable.

In `.cdsenv`:

```
msps.layout mfactorNames string "m M"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "mfactorNames" 'string "m M")
```

Valid Values:

A string with space-separated m-factor names.

Default Value: `""`

© 2023 All Rights Reserved.

GUI Equivalent:

Command: *Parasitics/LDE – Setup Parasitics and LDE*

Field: *Device m-factor parameter names*

© 2023 All Rights Reserved.

mixSchEstWithLayoutParasitics

Specifies that while creating a netlist by using the layout view, the estimated parasitics from the schematic view will be merged with the layout parasitics. When merging the parasitics from the two views, Virtuoso gives preference to the parasitics from the layout. For example, if you have specified both resistance and capacitance for a net in the schematic view, whereas, only capacitance for that net in the layout view, the netlist will get the capacitance value from the layout and the resistance value from the schematic.

In `.cdsenv`:

```
msps.layout mixSchEstWithLayoutParasitics boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "mixSchEstWithLayoutParasitics" 'boolean
     t)
```

Valid Values:

`t`	Parasitic estimates from both the schematic and layout views are merged in the netlist created using the layout view.
`nil`	Only the parasitics from the layout view are included in the netlist created using the layout view.

Default Value: `t`

GUI Equivalent: None

netlistView

Name of the netlist view to be created for re-simulation after extracting the specified LDE parameters or layout parasitics.

In `.cdsenv`:

```
msps.layout netlistView string  "netlist_layout" nil
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "netlistView" 'string "netlist_layout")
```

Valid Values:

Name of the reference net.

Default Value: `"netlist_layout"`

© 2023 All Rights Reserved.

GUI Equivalent:

Command: *Parasitics/LDE – Setup Parasitics and LDE*

Field: *Netlist view name*

referenceNet

Specifies name of the ground net to be used for grounded capacitance.

In `.cdsenv`:

```
msps.layout referenceNet string ""
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "referenceNet" 'string "")
```

Valid Values:

Name of the reference net.

Default Value: `""`

GUI Equivalent:

Command: *Parasitics/LDE – Setup Parasitics and LDE*

Field: *Reference net for grounded C*

singleFactorExpansion

Specifies how to use the multiplier parameters for the instances copied from the schematic view to the netlist_layout view. By default, while creating the netlist_layout view, the instances are expanded by using the CDF multiplier parameters, such as m. When more than one multiplier parameter are specified, a compound multiplier factor is calculated by combining all the parameters.

However, in certain specific scenarios, different cells use different multiplier names for the same purpose, for example, m or M. In such cases, set this variable to `t` to specify that only the first found parameter from the specified list of multiplier parameter names is to be used

© 2023 All Rights Reserved.

for the expansion of each cell instance. Other parameter names in the list are ignored. You can specify this list for a cell by using the mfactorNames environment variable.

In `.cdsenv`:

```
msps.layout singleFactorExpansion boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.layout" "singleFactorExpansion" 'boolean t)
```

Valid Values: `t` or `nil`

`t`	Only the first found parameter from the specified list of multiplier parameter names is used for the expansion of each cell instance.
`nil`	A compound multiplier factor, which is calculated by combining all multiplier parameters names, is used for the expansion of each cell instance.

Default Value: `t`

GUI Equivalent: None

© 2023 All Rights Reserved.

msps.setup

showAllCellViews

By default, the drop-down lists in the *Cellviews for Design Under Test* section of the Setup Parasitics and LDE form show the names of only those libraries and cellviews that are netlisted by the testbench defined in ADE Assembler. This helps in avoiding selection of a DUT that is not simulated by the defined testbenches.

You can set this variable to `t` to list all the libraries defined in the `cds.lib` file and their respective cellviews even if they are not used in the ADE Assembler view testbench.

In `.cdsenv`:

```
msps.setup showAllCellViews boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.setup" "showAllCellViews" 'boolean t)
```

Valid Values: `t` or `nil`

`t` Shows all the library and cellview names listed in `cds.lib` and their cellviews

`nil` Shows only the library and cellview names that are simulated by the ADE Assembler testbench

Note: When this variable is set to `nil` and there is no testbench defined in ADE Assembler, all the libraries and cells are listed.

Default Value: `nil`

netlistViewType

Specifies the default name for the netlist view.

In `.cdsenv`:

```
msps.setup netlistViewType cyclic "netlist"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.setup" "netlistViewType" 'cyclic "netlist_layout")
```

Valid Values: A string value.

© 2023 All Rights Reserved.

Default Value: `netlist_layout`

useNewSetupForm

Specifies the whether the new Parasitic/LDE Setup form is to be displayed.

In `.cdsenv`:

```
msps.setup useNewSetupForm boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.setup" "useNewSetupForm" 'boolean t)
```

Valid Values: `t` or `nil`

`t` Opens the new Parasitic/LDE Setup form by default.

`nil` Opens the older version of the Parasitic/LDE Setup form.

Default Value: `t`

ignoreLVSInstForStitching

Specifies the setting for stitching those instances in the netlist which have the `lvsIgnore` property set on them. To know more, see Including Instances that are Ignored in Stitched View.

In `.cdsenv`:

```
msps.setup ignoreLVSInstForStitching boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.setup" "ignoreLVSInstForStitching" 'boolean t)
```

Valid Values: `t` or `nil`

`t` Stitches the instance in the netlist while ignoring the `lvsIgnore` property.

`nil` Ignores the instance in the netlist for stitching by acknowledging the `lvsIgnore` property.

Default Value: `t`

© 2023 All Rights Reserved.

lxRemoveDeviceForStitching

Specifies the setting for stitching those instances in the netlist which have the `lxRemoveDevice` property set on them. To know more, see Ignoring Instances that are Included in Stitched View.

In `.cdsenv`:

```
msps.setup lxRemoveDeviceForStitching boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.setup" "lxRemoveDeviceForStitching" 'boolean t)
```

Valid Values: `t` or `nil`

`t` Ignores the instances in the netlist for stitching by acknowledging the `lxRemoveDeviceForStitching` property.

`nil` Stitches the instances in the netlist while ignoring the `lxRemoveDeviceForStitching` property.

Default Value: `t`

© 2023 All Rights Reserved.

msps.stitch

reduceParallelCaps

Controls a capacitance reduction phase after the parasitic/LDE view is created. If the variable is set to `t`, after all the capacitance instances are stitched and the manual capacitance instances are inserted, the tool searches for parallel capacitors in the parasitic/LDE view and merges them into a single device. For example, parallel capacitors connected to multi-fingered devices will be merged. This can help reduce the size of the netlist.

In `.cdsenv`:

```
msps.stitch reduceParallelCaps boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.stitch" "reduceParallelCaps" 'boolean t)
```

Valid Values: `t` or `nil`

Default Value: `nil`

© 2023 All Rights Reserved.

msps.estimates

You can define the cellview to be used for estimated parasitics by setting the msps.estimates environment variables for R, L, K, or C elements.

Note: The custom name environment variable for R, L, K, or C must have a corresponding CDF parameter with the same name so that the parasitic aware design flow works correctly.

customRName

Specifies the name of the resistor to be used for resistance parasitics.

In `.cdsenv`:

```
msps.estimates customRName string "r"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customRName" 'string "r")
```

Valid Values: A string value.

Default Value: `r`

customRLib

Specifies the name of the library where the parasitic resistor is saved.

In `.cdsenv`:

```
msps.estimates customRLib string "analogLib"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customRLib" 'string "analogLib")
```

Valid Values: A valid library name.

Default Value: `analogLib`

© 2023 All Rights Reserved.

customRCell

Specifies the name of the cell to be used as resistor parasitics.

In `.cdsenv`:

```
msps.estimates customRCell string "presistor"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customRCell" 'string "presistor")
```

Valid Values: A valid cell name.

Default Value: `presistor`

customRView

Specifies the view name for the resistor parasitics.

In `.cdsenv`:

```
msps.estimates customRView string "symbol"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customRView" 'string "symbol")
```

Valid Values: A valid library name.

Default Value: `symbol`

customLName

Specifies the name of the inductor to be used for inductance parasitics.

In `.cdsenv`:

```
msps.estimates customLName string "l"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customLName" 'string "l")
```

Valid Values: A string value.

© 2023 All Rights Reserved.

Default Value: `l`

customLLib

Specifies the name of the library where the parasitic inductor is saved.

In `.cdsenv`:

```
msps.estimates customLLib string "analogLib"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customLLib" 'string "analogLib")
```

Valid Values: A valid library name.

Default Value: `analogLib`

customLCell

Specifies the name of the cell to be used as inductor parasitics.

In `.cdsenv`:

```
msps.estimates customLCell string "pinductor"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customLCell" 'string "pinductor")
```

Valid Values: A valid cell name.

Default Value: `pinductor`

© 2023 All Rights Reserved.

customLView

Specifies the view name for the inductor parasitics.

In `.cdsenv`:

```
msps.estimates customLView string "symbol"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customLView" 'string "symbol")
```

Valid Values: A valid library name.

Default Value: `symbol`

customKName

Specifies the name of the inductor to be used for mutual inductance parasitics.

In `.cdsenv`:

```
msps.estimates customKName string "k"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customKName" 'string "k")
```

Valid Values: A string value.

Default Value: `k`

customKLib

Specifies the name of the library where the parasitic mutual inductor is saved.

In `.cdsenv`:

```
msps.estimates customKLib string "analogLib"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customKLib" 'string "analogLib")
```

Valid Values: A valid library name.

© 2023 All Rights Reserved.

Default Value: `analogLib`

customKCell

Specifies the name of the cell to be used as the mutual inductor parasitics.

In `.cdsenv`:

```
msps.estimates customKCell string "pmind"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customKCell" 'string "pmind")
```

Valid Values: A valid cell name.

Default Value: `pmind`

customKView

Specifies the view name for the mutual inductor parasitics.

In `.cdsenv`:

```
msps.estimates customKView string "symbol"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customKView" 'string "symbol")
```

Valid Values: A valid library name.

Default Value: `symbol`

© 2023 All Rights Reserved.

customCName

Specifies the name of the capacitor to be used for capacitance parasitics.

In `.cdsenv`:

```
msps.estimates customCName string "c"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customCName" 'string "c")
```

Valid Values: A string value.

Default Value: `c`

customCLib

Specifies the name of the library where the parasitic capacitor is saved.

In `.cdsenv`:

```
msps.estimates customCLib string "analogLib"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customCLib" 'string "analogLib")
```

Valid Values: A valid library name.

Default Value: `analogLib`

customCCell

Specifies the name of the cell to be used as the capacitor parasitics.

In `.cdsenv`:

```
msps.estimates customCCell string "pcapacitor"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customCCell" 'string "pcapacitor")
```

Valid Values: A valid cell name.

© 2023 All Rights Reserved.

Default Value: `pcapacitor`

customCView

Specifies the view name for the capacitor parasitics.

In `.cdsenv`:

```
msps.estimates customCView string "symbol"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "customCView" 'string "symbol")
```

Valid Values: A valid library name.

Default Value: `symbol`

defaultR

Sets the initial parasitic estimate value for resistance.

In `.cdsenv`:

```
msps.estimates defaultR string "1"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "defaultR" 'string "1")
```

Valid Values: Value of a valid parasitic resistance element.

Default Value: `1`

© 2023 All Rights Reserved.

defaultL

Sets the initial parasitic estimate value for inductance.

In `.cdsenv`:

```
msps.estimates defaultL string "1p"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "defaultL" 'string "1p")
```

Valid Values: Value of a valid parasitic inductance element.

Default Value: `1p`

defaultK

Sets the initial parasitic estimate value for mutual inductance.

In `.cdsenv`:

```
msps.estimates defaultK string "0.1"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "defaultK" 'string "0.1")
```

Valid Values: Value of a valid parasitic mutual inductance element.

Default Value: `0.1`

defaultCC

Sets the initial parasitic estimate value for coupled capacitance.

In `.cdsenv`:

```
msps.estimates defaultCC string "10f"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "defaultCC" 'string "10f")
```

Valid Values: Value of a valid parasitic coupled capacitance element.

© 2023 All Rights Reserved.

Default Value: `10f`

defaultDC

Sets the initial parasitic estimate value for decoupled capacitance.

In `.cdsenv`:

```
msps.estimates defaultDC string "10f"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "defaultDC" 'string "10f")
```

Valid Values: Value of a valid parasitic decoupled capacitance element.

Default Value: `10f`

viewName

Specifies the name of the view to be used to create a netlist that includes schematic parasitic estimates.

In `.cdsenv`:

```
msps.estimates viewName string "estimated"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "viewName" 'string "estimated")
```

Valid Values: A valid view name.

Default Value: `estimated`

© 2023 All Rights Reserved.

scaleR

Specifies the scale factor value for parasitic resistance.

In `.cdsenv`:

```
msps.estimates scaleR float "1.0"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "scaleR" 'float "1.0")
```

Valid Values: A floating-point number.

Default Value: `1.0`

scaleL

Specifies the scale factor value for parasitic inductance.

In `.cdsenv`:

```
msps.estimates scaleL float "1.0"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "scaleL" 'float "1.0")
```

Valid Values: A floating-point number.

Default Value: `1.0`

scaleC

Specifies the scale factor value for parasitic capacitance.

In `.cdsenv`:

```
msps.estimates scaleC float "1.0"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "scaleC" 'float "1.0")
```

Valid Values: A floating-point number.

© 2023 All Rights Reserved.

Default Value: `1.0`

detailReport

Specifies whether a parasitic comparison report is to be generated.

In `.cdsenv`:

```
msps.estimates detailReport boolean nil
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "detailReport" 'boolean nil)
```

Valid Values: `t` – generates a detailed report

`nil` – does not generate a detailed report.

Default Value: `nil`

reportFile

Specifies a name for the parasitic comparison report.

In `.cdsenv`:

```
msps.estimates reportFile string "compare_report"
```

In `.cdsinit` or the CIW:

```
envSetVal("msps.estimates" "reportFile" 'string "compare_report")
```

Valid Values: Any string value.

Default Value: `compare_report`

© 2023 All Rights Reserved.

layoutXL

You can define the settings to display and analyze R or C elements in a layout cellview.

Note: These settings are only available in the extracted views of type *Smart View* in the parasitic aware design flow.

svDisplayResistance

Enables or disables the display of parasitic resistances in the Smart View.

In `.cdsenv`:

```
layoutXL svDisplayResistance boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svDisplayResistance" 'boolean t)
```

Valid Values: `t` or `nil`.

Default Value: `t`

svDisplayInductance

Enables or disables the display of parasitic inductances in the Smart View.

In `.cdsenv`:

```
layoutXL svDisplayInductance boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svDisplayInductance" 'boolean t)
```

Valid Values: `t` – Enables the display of parasitic inductances.

`nil` – Disables the display of parasitic inductances.

Default Value: `t`

© 2023 All Rights Reserved.

svDisplayCapacitance

Enables or disables the display of parasitic capacitances in the Smart View.

In `.cdsenv`:

```
layoutXL svDisplayCapacitance boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svDisplayCapacitance" 'boolean t)
```

Valid Values: `t` or `nil`.

Default Value: `t`

svDisplayNodeName

Controls the display of the net fragment names of parasitic nodes in the Smart View.

In `.cdsenv`:

```
layoutXL svDisplayNodeName boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svDisplayNodeName" 'boolean t)
```

Valid Values: `t` – Enables the display of the net fragment name of parasitic nodes.

`nil` – Disables the display of the net fragment name of parasitic nodes.

Default Value: `nil`

svDotWidth

Controls the size of the parasitic nodes displayed in the Smart View. The width value is calculated in database units and the nodes or dots are displayed as square shapes.

In `.cdsenv`:

```
layoutXL svDotWidth int 200
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svDotWidth" 'int 200)
```


© 2023 All Rights Reserved.

Valid Values:	A positive integer value.
Default Value:	`100`

svResistanceThresholdMin

Specifies the minimum threshold for resistance to control the display of parasitic resistances in Smart View.

In `.cdsenv`:

```
layoutXL svResistanceThresholdMin string t
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svResistanceThresholdMin" 'string "")
```

Valid Values:	A positive integer value.
Default Value:	`""`

svResistanceThresholdMax

Specifies the maximum threshold for resistance to control the display of parasitic resistances in Smart View.

In `.cdsenv`:

```
layoutXL svResistanceThresholdMax string t
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svResistanceThresholdMax" 'string "")
```

Valid Values:	A positive integer value.
Default Value:	`""`

© 2023 All Rights Reserved.

svCapacitanceThresholdMin

Specify the minimum threshold for capacitance to control the display of parasitic capacitances in Smart View.

In `.cdsenv`:

```
layoutXL svCapacitanceThresholdMin string t
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svCapacitanceThresholdMin" 'string "")
```

Valid Values:	A positive integer value.
Default Value:	""

svCapacitanceThresholdMax

Specify the maximum threshold for capacitance to control the display of parasitic capacitances in Smart View.

In `.cdsenv`:

```
layoutXL svCapacitanceThresholdMax string t
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svCapacitanceThresholdMax" 'string "")
```

Valid Values:	A positive integer value.
Default Value:	""

© 2023 All Rights Reserved.

svInductanceThresholdMin

Specify the minimum threshold for inductance to control the display of parasitic inductances in Smart View.

In `.cdsenv`:

```
layoutXL svInductanceThresholdMin string ""
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svInductanceThresholdMin" 'string "")
```

Valid Values:	A positive integer value.
Default Value:	`""`

svInductanceThresholdMax

Specify the maximum threshold for inductance to control the display of parasitic inductances in Smart View.

In `.cdsenv`:

```
layoutXL svInductanceThresholdMax string ""
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svInductanceThresholdMax" 'string "")
```

Valid Values:	A positive integer value.
Default Value:	`""`

© 2023 All Rights Reserved.

svInductorStyle

Specify the style for parasitic inductances displayed in Smart View.

In `.cdsenv`:

```
layoutXL svInductorStyle string "ROUND"
```

In `.cdsinit` or the CIW:

```
envSetVal("layoutXL" "svInductorStyle" 'string "SHARP")
```

Valid Values: `"ROUND"` or `"SHARP"`.

Default Value: `"ROUND"`

© 2023 All Rights Reserved.

maestro.test

autoSyncMPC

Controls the detection of a Smart View with multi-process corners in a `maestro` design block.

In `.cdsenv`:

```
maestro.test autoSyncMPC boolean t
```

In `.cdsinit` or the CIW:

```
envSetVal("maestro.test" "autoSyncMPC" 'boolean "")
```

Valid Values:	`t`: Enables searching for a Smart View with multi-process corners when opening the `maestro` cellview or setting up the design for a test. This may impact performance. `nil`: Disables searching for a Smart View with multi-process corners when opening the `maestro` cellview or setting up the design.
Default Value:	`nil`

© 2023 All Rights Reserved.

B

Backannotation of dcOp / Transient Values for M-Factor Devices

This appendix looks at the backannotation of dcOp and transient values for multiple factor devices. It describes the process of schematic annotation of operational data associated with m-factor devices.

Note: A device with a multiplication factor (m-factor device) is where a single schematic cell instance represents multiple instances of a parallel connected cell.

There are two alternatives open to you in achieving this:

1. Using the `?mfactorR` and `?mfactorW` qrcParameters when extracting a view. Here, you will instruct Cadence Quantus QRC Extraction to merge m-factor devices.

   For more information on this option see Grouping M-Factor Devices At Extraction Time on page 309.

2. Using `opParamExprList` to change the CDF of the cell to make use of the new parameters.

   For more information on this option see Using opParamExprList Functionality on page 311.

   Note: This is the recommended option.

© 2023 All Rights Reserved.

Identifying M-Factor Devices

When a backannotation has an m-factored device, and it only processes the first device, a ",.." will be added to the backannotated value and the following, once only, warning message will be issued in the CIW:

```
"WARNING* Labels with suffix ",.." are on devices which map to multiple devices.
Only the value from one device is currently annotated. For more information on
displaying alternative values, see the Backannotation of dcOp / Transient Values
for M-Factor Devices section in Chapter 1 of the "Virtuoso Parasitic Aware Design
User Guide"
```

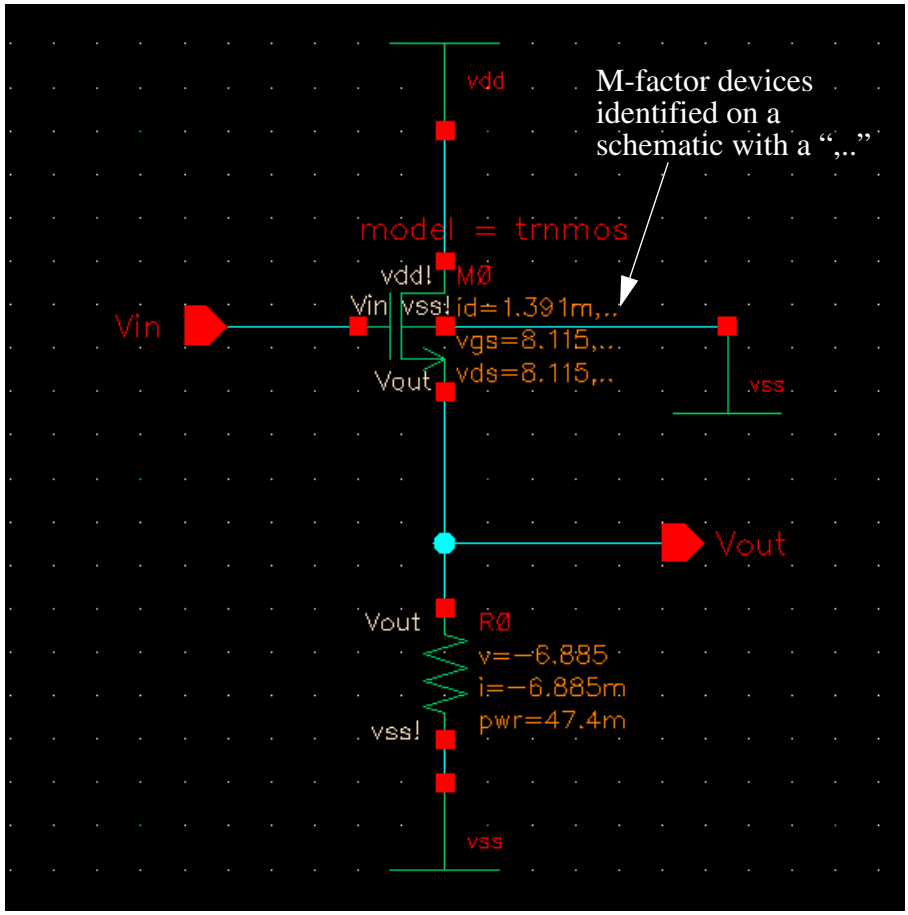


Figure B-1 M-Factor Devices On A Schematic

Once the M-Factor devices have been identified on a schematic, you can choose to display the m-factor devices by Grouping M-Factor Devices At Extraction Time or Using opParamExprList Functionality.

© 2023 All Rights Reserved.

Grouping M-Factor Devices At Extraction Time

This approach relies on the capabilities of Quantus QRC for merging m-factor devices at extraction time. It involves merging m-factor devices so that there would only be one extracted instance per schematic instance.

This means that when backannotation is used, only one value will be displayed, as no m-factor groups are found. You will therefore direct Quantus QRC extraction to merge m-factor devices, which will require the modification of the RSF file.

Modifying the RSF File

A run specification file (RSF) is used to direct the extraction of parasitics during a Quantus QRC extraction run.

In the RSF, you will find a `qrcParameter` section, which you can modify to provide `?mFactorR` and/or `?mFactorW` qrcParameters. These parameters are used to group m-factor devices as follows:

- `?mFactorR` - reduces the number of `MOS` and `LDD` transistors in the output netlist by merging parallel transistors in the layout.
- `?mFactorW` - changes the default behavior of `?mFactorR` so that merged transistor devices are output to the netlist with widths summed, and no m-factor parameters added.

To modify the RSF file, with the new parameter settings, you edit the *M Factor R* and *M Factor W* options in the QRC Parasitic Extraction Run Form.

1. Select *QRC – Setup QRC*.
2. Click the *Filtering Options* tab.

   The *Filtering Options* tab has *M Factor R*(eduction) and *M Factor W*(idth) options that correspond to `?mFactorR` and `?mFactorW`.
3. Edit the *M Factor R*(eduction) and *M Factor W*(idth) options as required.

   M Factor R lets you reduce the number of `MOS` and `LDD` transistors in the output netlist by merging parallel transistors in the layout. The M-Factor is annotated to a transistor in schematic capture, and the resultant layout should contain “m” transistors laid out in parallel. These parallel transistors are designed so that the parasitics from gate-to-gate, source-to-source, and drain-to-drain are minimal. *M Factor R* uses a specified resistance value to merge all transistors in which the shortest path resistance between the source/drain/gate of adjacent devices is less than the value specified in the `?mFactorR` value.

© 2023 All Rights Reserved.

Activating the *M Factor W* command will change the default behavior of *M Factor R*, in that width values will always be summed, regardless of their equivalence, and no M-Factor (m=n) parameter is output to the netlist.

For more information, see the *Filtering Options Tab* section in the *Quantus QRC Extraction Users Manual*.

Note: After `?mFactorR` and `?mFactorW` settings/values have been provided, Quantus QRC extraction will need to be re-run.

Limitations of the Edit RSF Approach

Editing the `?mFactorR` and `?mFactorW` parameters may not group m-factor devices as required.

This could be because:

- `?mFactorR` only merges transistors that follow a set criteria, for example the transistors must be the same model type, or share the same source/drain/gate and so on.
- `?mFactorR` only supports `MOS`/`LDD` transistors. However, m-factor can be found in other types of devices, for example `CAP` and `RES`. An alternative solution is therefore required in these cases, such as Using opParamExprList Functionality.

© 2023 All Rights Reserved.

Using opParamExprList Functionality

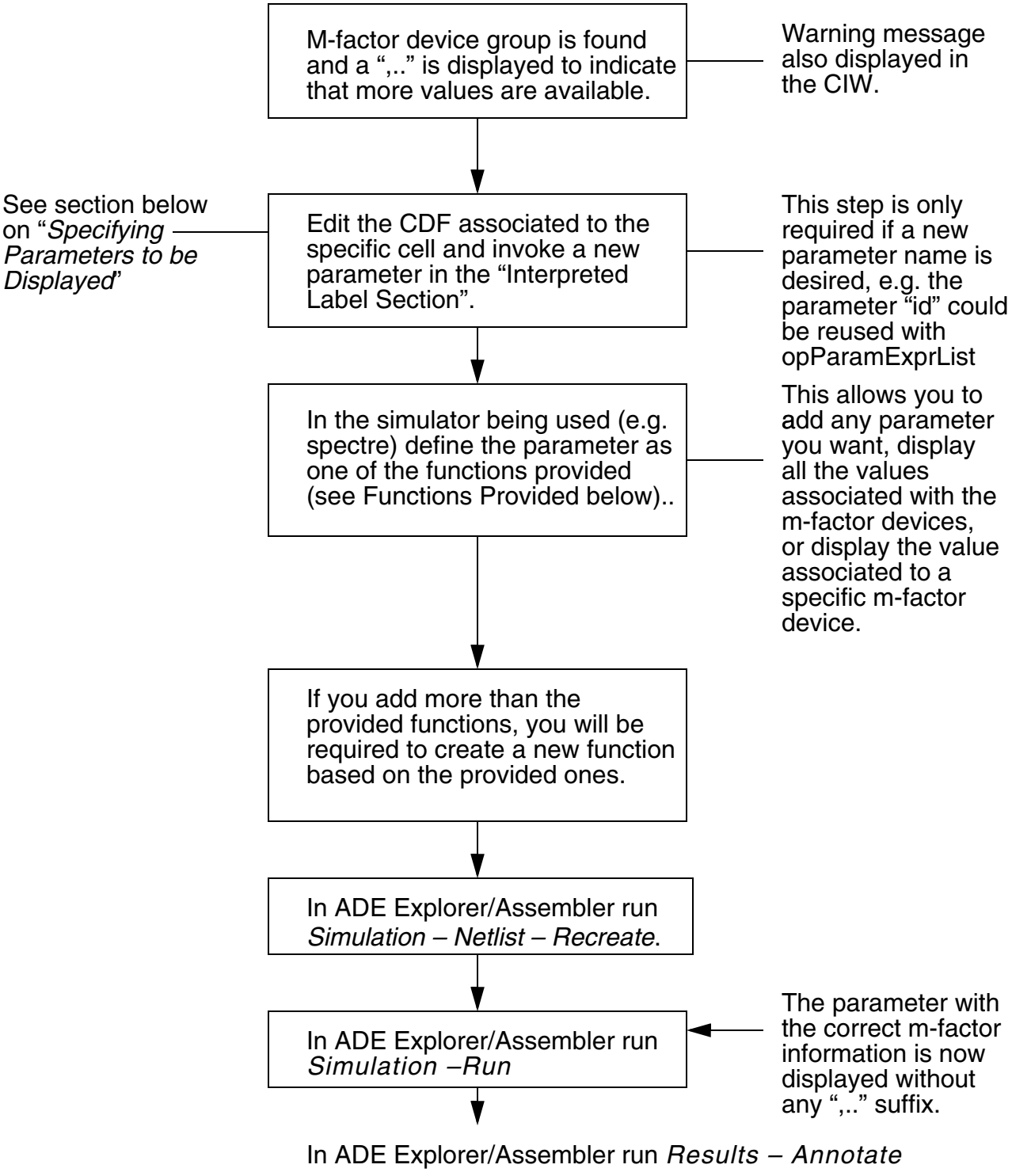
This method of backannotating m-factor device values requires the use of `opParamExprList`, and involves you changing the CDF (Component Description Format) of the cell to make use of the new parameters. Here, we process the m-factor device's parameter values, and provide a new value which summarizes their effect. This is done using a list that contains all of the values associated with the m-factor devices.

After doing this, re-running netlist and simulation again will display the m-factor devices value.

This cannot be an automatic function as the parameters do not have semantic information associated with them.

The following flowchart provides an overview of backannotation of m-factor devices using the `opParamExprList` functionality. As mentioned, this is the recommended approach.

© 2023 All Rights Reserved.



© 2023
All Rights Reserved.

Specifying Parameters to be Displayed

As mentioned, the function `opParamExprList` is a CDF parameter that allows you to save additional `op` information. For example, `opParamExprList` allows you to create a new parameter which sums the "`id`" of all the m-factor devices. This information can then be displayed using the calculator or layer display mechanism. It is the label display information that is of particular interest to us, as it allows you to display parameter values. The interpreted label `cdsParam()` allows you to display information about parameter values.

Note: For more information on the functions described in this section see Parasitic Aware Design SKILL Commands.

During automatic symbol generation, three `cdsParam` labels are usually generated. You can however define new labels if required.

To specify which parameter will be displayed in the `cdsParam()` labels, you will need to edit the CDF of the relevant library cell. To do this:

1. From the CIW, select *Tools – CDF – Edit* to display the Edit Component CDF form.
2. *Browse* to locate the cell whose parameters you want to specify for display.
3. Scroll down to the *Interpreted Labels Information* section in the Edit Component CDF Form.

   In this section you will see the *op pointLabelSet* field, which will list the parameters to be displayed if *paramDisplayMode* is set to *op point*. For example:

   You can see above that the *op pointLabelSet* only has two parameters: "`id`" which is the current of the cell, and "`mFactorF`" which is a new parameter defined in the `opParamExprList`. This new parameter takes into account the m-factor devices, and in this case adds the "`id`" values of all devices.

   Note: Each label set is limited to the number of `cdsParam` labels in the symbol.

   Note: Instead of creating a new parameter name "`mFactorF`", the existent parameter name "`id`" could have been redefined in function of the `opParamExprList`. This has the advantage that the existent parameter will handle single devices as well as m-factor devices in a transparent manner.
4. You now need to define the new `mFactorF` parameter or, as mentioned above, redefine the existent parameter "`id`". The parameter may be defined using one of the Functions Provided, or by creating your own function, which may be based on the provided functionality.

   The CDF description of the cell now needs to be modified.

© 2023 All Rights Reserved.

5. In the Edit Component CDF form (accessible from the CIW by selecting *Tools – CDF – Edit*), scroll to the *Simulation Information* section and click the *Edit* button to display the Edit Simulation Information form.

   In the above screenshot, you can see that the parameter name "`id`" is defined by aelSumOPParam.

   Note: Only *spectre* and *csSpice* simulators are supported.

© 2023 All Rights Reserved.

Functions Provided

aelSumOPParam

Description Returns a number which is the result of adding the values of the parameters specified by `inst`. This `inst` can be, for example, a schematic name which maps to multiple m-factor devices, one device, or a specific extracted name which will allow you to display specific m-factor devices values. To do this, `aelSumOPParam` creates a list with all the instances being considered. The instance may be a schematic instance (the result of `inst()`), or an extracted instance. For example "`I0/M0_1_qrc`". If a schematic instance is given in out-of-context, then the mapped extracted instances are considered, for example if `inst()` is given, the instances considered could be (`"/I0/M0" "/I0/M0_1_qrc" "/I0/M0_2_qrc" "/I0/M0_2_qrc" "I0/M0_3_qrc" "I0/M0_4_qrc"`). Once the list is created, the "`param`" specified for each instance is added. This "`param`" can be any of the simulation parameters, but if not specified, is "`id`" by default.

Inputs `instName`: A string that can be a schematic instance. The result of the method `inst()`, or an extracted instance name.

`simParam`: Can be any simulation parameter, for example "`id`".

`labelParam`: An optional parameter that is required when the name of the label parameter defined by `opParamExprList` is different from the simulation parameter being processed. For example, if the label parameter is `mFactorF` and the simulation parameter being processed is `id`, then `labelParam` must be given with the value `mFactorF`.

`resname`: Another optional string parameter used to select the type of results from a particular analysis, for example `dcOpInfo-info`. The type of results available can be obtained using the following command: `results(?noAlias t)`. As a default, this input is set to the current type of results.

Outputs A number which is the result of adding all of the "param" available in the specified "`instName`", or `nil` if the instance fails to map.

© 2023 All Rights Reserved.

Definition

```
procedure( aelSumOPParam(instName simParam @optional (labelParam nil) (resName nil))
  let((insts (total 0.0))
        unless( resName
                resName = strcat(asiMapFuncToLogicalName(
                                  asiGetTool(asiGetCurrentDataContext()->simulator)
                                  'OP))
        )

        if(auLvsParSimContext() then
                unless(labelParam labelParam = simParam)
                insts = auLvsMapSchInstNameParInstName(instName "" labelParam)
                if(!insts then insts = list(instName))
                if(!listp(insts) then insts = list(insts))
        else
                insts = list(instName)
        )

        when((and insts member(resName results(?noAlias t)))
                foreach( x insts
                        total = total + pv(x simParam ?result resName)
                )
        )
        total
    )
)
```

sumOPParamV2

Description As `aelSumOPParam`.

This method is not provided, therefore you will have to introduce it by copy and pasting the code. This function does effectively the same as `aelSumOPParam`, but differs in the following:

- `sumOPParamV2` requires that the label parameter name is different than the processed simulator name. For example it does not allow you to have a label parameter name "`id`" processing the simulator parameter "`id`".
- Using this method is approximately 50% faster than using `aelSumOPParam`.

Note: You should not use `aelSumOPParam` and `sumOPParamV2` at the same time, in the same library/cell.

© 2023 All Rights Reserved.

Inputs

`instName`: A string that can be a schematic instance; the result of the method `inst()`, or an extracted name instance.

`simParam`: A string that can be a simulation parameter, for example: `id`.

`labelParam`: A string that is the label parameter defined by `opParamList`, for example `mFactorF`.

Outputs

A number which is the result of adding all of the “param” available in the specified “`instName`”, or `nil` if the instance fails to map.

Definition

```
defun( sumOPParamV2 (instName simParam labelParam)
   let((insts (total 0.0))

   if(auLvsParSimContext() then
      insts = auLvsMapSchInstNameParInstName(instName "*"
labelParam)
      if(!listp(insts) then insts = list(insts))
      else insts = list(instName)
   )

   unless( null(insts)
     foreach( x insts total = total + OP(x simParam))
   )

   total
  )
)
```


© 2023 All Rights Reserved.

aelDisplayOPParam

Description This function returns a list whose elements are the "`param`" of each of the instances being processed. The instances being processed depend on the given "`instName`". The function `aelDisplayOPParam` creates a list with all of the instances being considered. The instance may be a schematic instance (the result of `inst()`), or an extracted instance. For example "`/I0/M0_1_qrc`". If a schematic instance is given in out-of-context, then the mapped extracted instances are considered, for example if `inst()` is given, the instances considered could be (`"/I0/M0" "/I0/M0_1_qrc" "/I0/M0_2_qrc" "/I0/M0_2_qrc" "I0/M0_3_qrc" "I0/M0_4_qrc"`). Once the list is created, the "param" specified for each instance is added to the return list. This "`param`" can be any of the simulation parameters and, if not specified, the default is "`id`".

Inputs `instName`: A string that can be a schematic instance. The result of the method `inst()`, or an extracted instance name.

`simParam`: Can be any simulation parameter, for example "`id`".

`labelParam`: An optional parameter that is required when the name of the label parameter defined by `opParamExprList` is different than the simulation parameter being processed. For example, if the label parameter is `mFactorF` and the simulation parameter being processed is `id`, then `labelParam` must be given with the value `mFactorF`.

`resname`: Another optional string parameter used to select the type of results from a particular analysis, for example `dcOpInfo-info`. The type of results available can be obtained using the following command: `results(?noAlias t)`. As a default, this input is set to the current type of results.

Outputs A string with a list of numbers separated by commas, or `nil` if the instance fails to map.

© 2023 All Rights Reserved.

Definition

```
procedure( aelDisplayOPParam(instName simParam  @optional (labelParam nil) (resName nil))
   let((insts values)
   unless( resName
           resName = strcat(asiMapFuncToLogicalName(
                            asiGetTool(asiGetCurrentDataContext()->simulator)
                            'OP))
   )

   if(auLvsParSimContext() then
      unless(labelParam labelParam = simParam)
      insts = auLvsMapSchInstNameParInstName(instName "" labelParam)
      if(!insts then insts = list(instName))
      if(!listp(insts) then insts = list(insts))
   else
                insts = list(instName)
   )

   when((and insts member(resName results(?noAlias t)))

        values = foreach( mapcar x insts aelSuffixNotation(pv(x simParam ?result resName)))
   )

   buildString( values ",")
   )
)
```


© 2023 All Rights Reserved.

© 2023 All Rights Reserved.

C

Parasitic Aware Design and Diva Verification

This appendix describes parasitic aware design flows for analog and mixed-signal circuits. It comprises of the following sections:

© 2023 All Rights Reserved.

Diva Flow: Simulating Analog Circuits with Parasitic Aware Design

This section describes how you can use Cadence® tools to investigate the effect of parasitics on analog circuits. By accounting for the effect of parasitics, you can improve the accuracy of your circuit simulations. If your design includes digital or mixed-signal circuits, skip this section and use Diva Flow: Simulating Mixed-Signal Circuits with Parasitic Aware Design.

This section comprises of the following topics:

Note: For more information on Diva see the *Diva Reference*.

© 2023 All Rights Reserved.

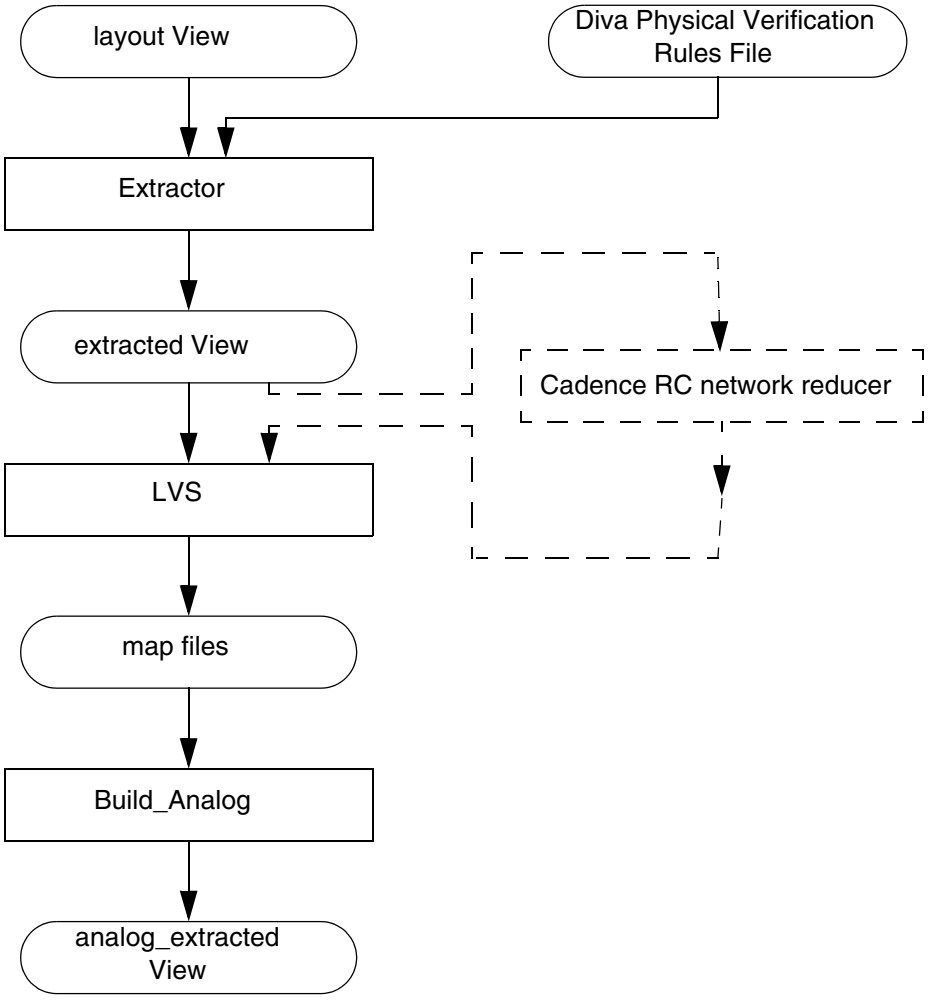
Overview

Simulating an analog circuit with parasitics requires these steps.

1. Preparing cell libraries
2. Creating an analog_extracted view of your design

   In this step, the tool calculates parasitics from information in the layout view of your circuit.

   The following flow diagram illustrates the substeps in creating an analog_extracted view using the Diva® physical verification tool. The substeps for the Cadence RC network reducer are shown with dotted lines because they are optional.



3. Creating a configuration for your design.

© 2023
All Rights Reserved.

4. Simulating the design with parasitics included

After a successful simulation, you can select terminals and device pins on the schematic and use plot commands to display the results in a waveform window. The resulting waveforms can be used with all Virtuoso analog design environment (ADE) calculation and analysis tools.

© 2023 All Rights Reserved.

Preparing Cell Libraries

Before you can follow the flow outlined in this chapter, you need to provide the following views and component description format (CDF) information for analog primitives and parasitic cells.

Analog primitives must have	Parasitic cells (such as presistors and pcapacitors) must have
symbol cellview	symbol cellview
layout cellview or extraction rules that the extractor will recognize	auLvs cellview for analog primitives
Model matching the simulator you use	CDF simulation information for auLvs
auLvs cellview that provides parameter values used by LVS	CDF component parameters for resistance (r) and capacitance (c)

The `analogLib` library contains examples of analog primitives and parasitic cells that you can copy to create your cell library.

Note: You can find the `analogLib` in the following hierarchy:

```
$CDS_INST_DIR/tools/dfII/etc/cdslib/artist/analogLib
```

Transferring Schematics from Diva to Assura

Any schematics, with placed estimated parasitic devices, that have been used with Diva will not be recognized if transferred to the Assura flow.

This is because the LVS (layout versus schematic) checks will ignore the following (shorted) devices: `presistor`, `pinductor`, `tline2`, `tline3`, `tline4`, `tline4x`, `tline4p`.

Note: `pcapacitor`, `pdiode`, and `tline1` are opened as expected.

© 2023 All Rights Reserved.

Preparing Technology Files

To prepare a library for parasitic extraction,

1. Describe the technology layers.

   For details about the technology layers, refer to the *Incremental Technology Databases and Display Resources User Guide*.

2. Add or modify the verification rules used by the Diva processes DRC, Extract, and LVS.

   Refer to the *Diva Reference* for details about creating verification and extraction rules.

© 2023 All Rights Reserved.

Adding Component Description Format Simulation Information

Refer to the *Component Description Format User Guide* for more details about the steps in this section.

To netlist primitives correctly, you must verify the auLvs CDF parameters for each primitive.

1. Start the Cadence software by typing `virtuoso&` at the command prompt.

   For more information about the options you can use with the command to start the software, refer to the *Virtuoso Design Environment User Guide*.

2. In the command interpreter window (CIW), choose *Tools – CDF – Edit*.

   The Edit Component CDF form appears.

3. In the upper portion of the form, choose *Cell* for *CDF Selection* and *Base* for *CDF Type*.

   You must edit the base-level CDF for changes to be effective.

4. Fill in the *Library Name* and *Cell Name* fields, or click the *Browse* button to select the cell.

   The Edit Component CDF form expands to display additional information.

5. In the *Simulator Information* area of the expanded Edit Component CDF form, click *Edit*.

   The Edit Simulation Information form appears, displaying existing CDF information about auLvs netlisting.

6. Select *auLvs* in the *Choose Simulator* drop-down list box.

7. Ensure that the *netlistProcedure* field specifies `ansLvsCompPrim`. This is the internal auLvs procedure for netlisting primitives.

8. In the *instParameters* field, specify the parameters you want in the netlist.

   A component can have several parameters, such as temperature coefficients, that do not apply to LVS netlist comparison. You LVS comparison rules tell LVS how to handle such parameters.

   If `model` is included in the *instParameters* field, auLvs uses the value of the model property in the instance instead of the value of *componentName* in the netlist.

9. In the *componentName* field, type the component name you want included in the netlist.

   This optional field allows you to use a common name in the netlist for different cells. For example, 3-terminal cellviews (with programmable bulk nodes) and 4-terminal cellviews

© 2023 All Rights Reserved.

(with a bulk node as a pin) that have distinct names like `nmos3` and `nmos4` can be netlisted with the same component name like `nmos`.

The component names `pcapacitor`, `presistor`, `pinductor`, and `pdiode` are used for parasitic devices. All these devices are removed from the netlist before layout versus schematic (LVS) runs, but are used in simulation and backannotation. For presistors and pinductors, the nets are shorted together.

10. In the *termOrder* field, type the names of the device terminals as they appear in the symbol cellview.

    This is the order in which the terminals are netlisted.

11. If *termOrder* uses programmable nodes, type the names of the terminals in *deviceTerminals*.

    The input is the same as for *termOrder*, but programmable nodes are replaced by names in this field.

12. For existing designs that use older databases, use *prop Mapping* to change the name of an instance parameter.

    This allows instParameter names that use lowercase letters to be mapped to LVS rules that are defined in uppercase letters.

    Note: Do not use this feature for new designs.

13. In the *Permute Rule* field, specify the LVS permute rule used to define equivalent pins.

14. Click *OK* on the Edit Simulation Information form and *Apply* on the Edit Component CDF form to accept your changes.

Note: The CDF *parseAsNumber* property distinguishes strings from numbers in numeric parameters. String parameters without the *parseAsNumber* property set to *true* are netlisted as strings beginning and ending with "\". This feature is *not* compatible with releases before 4.4.2.

© 2023 All Rights Reserved.

Creating Designs

If you intend to extract parasitic components from the layout view and run a simulation with parasitics, use the following guidelines to avoid problems as you plan your design.

- Devices with the *componentName* parameter set to *pcapacitor*, *presistor*, *pinductor*, and *pdiode* are automatically removed from the netlist. Do not use these names for your components.
- Nets are shorted together for LVS on presistors and pinductors.
- Do not use the LVS *permuteDevice* parameter to match groups of components in a series because that makes it impossible to determine which device is which for waveform probing.

© 2023 All Rights Reserved.

Creating Extracted Views

You use the Diva physical verification tool to extract parasitics from the layout view of a block. Then you use LVS to compare the extracted view to the schematic view to identify areas that are not consistent between the views. After a successful LVS run, you create an analog_extracted view of the design.

Extracting Parasitics

To extract parasitics from the layout view of a cell or block,

Important

Ensure that the environment variable `CDS_Netlisting_Mode` is set to `Analog`:

1. Choose *Verify – Extract* from the layout cellview of the cell.

   The Extractor form appears.

2. Choose *flat* for *Extract Method*.

   You need to use flat extraction because parasitic capacitance values can vary between different instances of the same cell. Each cell, therefore, must be extracted.

3. (Optional) Choose *Join Nets With Same Name*.

   This ensures that nets with the same name are joined automatically.

4. To select the types of parasitics you want extracted, click *Set Switches*.

   The Set Switches form appears. The parasitics displayed vary, depending on the extraction rules file defined for your design. In some cases, you do not need to make any selections.

   To select more than one item, click your first selection, then hold down the `Control` key and make the rest of your selections.

5. When you have specified the parasitics you want, click *OK*.

   The Extractor form reappears with the parasitics you selected in the *Switch Names* field.

6. Click *OK* or *Apply* to create the extracted views.

   A message in the command interpreter window (CIW) tells you when the extraction process is complete.

© 2023 All Rights Reserved.

Comparing Schematic and Extracted Views

To compare the schematic view with the extracted view created earlier, follow these steps.

1. From a window displaying the extracted view, choose *Verify – LVS*.

   The LVS form appears.

   Note: For more information on the LVS form see Verify Menu Commands in the *Diva Reference*.

2. Depending on which views are open, use one of the following procedures to identify the schematic and extracted views that you want to compare.

If both the schematic and extracted views are open	If only the extracted view is open
1. Click the *Sel by Cursor* button below the schematic detail, then click the cursor in the open schematic view. 2. Do the same for the extracted view.	1. Click the *Sel by Cursor* button below the extracted detail, then click the cursor in the open extracted view. 2. Click the *Browse* button below the schematic detail and select the schematic view.

3. Enter the names of the rules file and rules library for the Diva LVS rules.
4. Click the *Run* button to begin the comparison.
5. When the comparison finishes, click *Info*.

   The Display Run Information form appears.

6. Click *Log File*.

   Scroll through the log file to the netlist comparison section near the end of the file. This section identifies any mismatches between the two files. Each error is described in the sections following the comparison results.

   Not all mismatches are fatal. Look over the comparison results to determine if you need to correct one of the files and redo the extraction and comparison or if you can proceed with the views as they are.

7. Choose *File – Close Window* in the log file window.
8. Click *Cancel* in the Display Run Information form.
9. Correct any problems in the schematic or extracted views.

© 2023 All Rights Reserved.

10. If necessary, rerun the comparison.

Building an analog_extracted View

When the comparison between schematic and extracted views is acceptable, you need to select the parasitics to use for simulation. You also need to build the analog_extracted view.

1. In the LVS form, click *Build Analog*.

The Build Analog Extracted View form appears.

Note: For more information on the LVS form see Verify Menu Commands in the *Diva Reference*.

2. Select one of the following choices to specify the analog parasitics that you want to use for simulation.

Select	If you want to
Include All	Simulate with all the parasitics that have been extracted.

© 2023 All Rights Reserved.

Select	If you want to

Set From Schematic	Select parasitics to include in the simulation by placing special symbols (spresistor, spcapacitor, spinductor, and spcapacitor2) on nets in the schematic view. The spcapacitor2 device is used to include parasitics in the simulation that appear between two specified nets. If you add or remove symbols from the schematic, click *Check* and *Save* to save the modified view.
	
	The parasitics you select by placing these symbols (which are provided in `sbaLib`) are the only ones included in the simulation.
	
	Note: The `sbaLib` can be found at the following location in the Cadence installation:
	
	`<installation_dir>/tools/dfII/etc/cdslib/artist/sbaLib`
	
	To define this library in your `cds.lib` you need to set the following:
	
	`DEFINE sbaLib <installation_dir>/tools/dfII/etc/cdslib/artist/sbaLib`
	
	If you choose *Set From Schematic* and click *OK* without identifying any nets on the schematic, the Extracted Parasitics Selective Annotation form asks you to confirm your choice.

None	Simulate with none of the parasitics.

3. Click *OK* to accept your settings and build the analog_extracted view.

© 2023 All Rights Reserved.

Creating and Using a Configuration

This section explains how to set up a configuration so that the simulator runs with the analog_extracted view created in the previous step. The steps given here for using the Cadence Hierarchy Editor to create a configuration are abbreviated. For complete information, see the *Cadence Hierarchy Editor User Guide*.

To create a configuration for your design,

1. From the CIW, choose *File – New – Cellview*.

   The Create New File form appears.

2. Choose the library for the new file.

3. Type the name of the cell for which you want to create the configuration.

   The top-level cell for your design is usually the appropriate cell to use.

4. If you do not want to use `config` as the view name, type the name you want into the *View Name* field.

5. Choose *Hierarchy-Editor* from the *Tool* drop-down list box.

6. Ensure that the *Library path file* field correctly specifies the `cds.lib` file that contains the paths to your libraries.

7. Click *OK*.

   The New Configuration form appears.

8. Click the *Use Template* button located at the bottom of the form.

   The Use Template form appears.

9. Select a template that is compatible with the simulator you are running from the *Name* drop-down list box.

10. Click *OK* in the Use Template form.

    The New Configuration form redisplays with default data for the *Top Cell* and *Global Bindings* sections. This allows you to modify a typical view list and stop list, rather than creating them from scratch.

    Templates exist for each of the simulators. To create templates that provide defaults for these fields, see the *Cadence Hierarchy Editor User Guide*.

11. In the *Top Cell* section, enter the library, cell name, and schematic cellview from which to build the configuration.

© 2023 All Rights Reserved.

Be sure to specify *schematic* for the view type because the configuration is built from the original schematic of your design.

12. Click *OK*.

The Hierarchy Editor window displays your data.

The Hierarchy Editor window configures the design by using a default *View List* and *Stop List* in the *Global Bindings* section. You need to modify these lists for your design.

13. Use one of the following methods to specify the analog_extracted view for the cells or blocks for which you want parasitics simulated.

To specify views for individual blocks	To specify views for multiple blocks
1. In the *Instance Binding* section of the Hierarchy Editor window, position the cursor in the *View To Use* column of the appropriate block. **Note:** If the *Instance Binding* section is not visible in the window, choose *View – Instance Table* to display this section. **2.** Press the right mouse key to display a list of commands. **3.** Choose *Select View* to display the list of views for this block. **4.** Choose *analog_extracted* as the view for this block.	➤ In the *Global Bindings* section of the Hierarchy Editor window, add `analog_extracted` as the first view in the *View List* text field. This ensures that the analog_extracted view is the selected view for every cell that has an analog_extracted view.

14. Choose *View – Update* to reconfigure the design to reflect your changes.

The Update Sync-up form appears.

15. Click *OK*.

16. Choose *File – Save* to save the configuration with your changes.

17. Choose *File – Exit* to close the Hierarchy Editor.

© 2023 All Rights Reserved.

Simulating the Design

To run the simulation,

1. In the CIW, choose *Tools – Analog Environment – Simulation*.

   The Analog Design Environment Simulation window appears.

2. Choose *Setup – Design*.

   The Choosing Design form appears.

3. Select the library and cell name of your design.

4. Select *config* from the *View Name* drop-down list box.

5. Click *OK*.

   This view supplies configuration as well as schematic information.

6. In the Analog Design Environment Simulation window, choose your simulator, model path, environment variables, analyses, and simulator options.

7. Choose *Simulation – Run*.

   When complete, the schematic appears so that you can select outputs and probe the design.

8. Choose *Outputs – Set from Schematic*.

9. Select the terminals in the schematic, or in the layout views of the blocks where parasitics were extracted, to define outputs.

   Note: The only places where connections on different views are guaranteed to match are on component terminals.

© 2023 All Rights Reserved.

Probing Parasitic Values

By probing the schematic or extracted view, you can examine the instances of parasitic components. To probe parasitic values, follow these steps.

Important

Parasitic probing in the Diva flow only works for simple nets. For full probing on iterated instances, bus wires, and wire bundles, you should use the parasitic's Assura flow. For more information, see

1. In the LVS form, click the *Parasitic Probe* button.

   The Parasitic Probing form appears.

2. In the *Max list size* field, specify how many parasitic instances to display.
3. Sort parasitics by resistance or capacitance by selecting *R* or *C*.
4. Click the appropriate button to specify which parasitics should be collected.

   - Click *Whole Net* and then select a net in the schematic or extracted view to display an ordered list of all the parasitics on the net. The largest resistances or capacitances appear at the top of the list.

     Note: You can click the *Save* button in the displayed Parasitics for net... form to save a text file of the parasitic results.

   - Click *Point to Point* and then select two pins or instance pins (on the same net) in the schematic or extracted view to collect all the parasitics between two points.
   - Click *Net to Net* and then select two nets in the schematic or extracted view to collect parasitic capacitances between two different nets.

   A list of the collected parasitic instances appears. Select an instance from this list to highlight the component symbol associated with this parasitic on the extracted view.

Probing Using the Diva Probing Form

If you choose to probe on a net by selecting *Tools – Diva – Verify – Probe*, you should be aware that the only probing option supported here, by parasitic aware design in the Diva flow, is where you choose to select *Probing Method: single w/o parasitics*, and then click the *Add Devs for Net* option.

Note: For information on using the Probing form, see the Verify Menu Commands chapter in the *Diva Reference*.

© 2023 All Rights Reserved.

Note: Iterated instances, bus wires, and wire bundles are only partially supported using Diva probe functionality. For full support of these, you should use the Assura flow as described in chapters 1 and 2.

Out of Context Probing

A cell view is said to be *in context* when it is the view that is "bound" via the configuration, that is, the view that is picked up by the simulator.

A view can therefore be classified as being *out of context* when it is not the current bound view. For example, if you want to use an analog_extracted view in your simulation, you need to set up a configuration that will override the default view for one or more instances, likely to be schematic, to be "analog_extracted".

When you now simulate this design, any views that are bound to "analog_extracted" will be netlisted to include the parasitics in them.

Performing Out of Context Probing

To perform out of context probing, using the above example, you would therefore have to descend into a view that is not analog_extracted, e.g. the schematic view.

From here, you would perform simulation probing within that view, for example from the Analog Development Environment (ADE). Within the ADE, you would select either:

Results – Direct Plot – Transient Signal

or

Results – Plot Outputs – Transient Signal.

Important

If you are attempting a full chip simulation of an `analog_extracted` view, this can occasionally fail during the reading of the design with an "out of memory" error. If this happens, it can be due to the chip being too big to simulate. The recommendation, if this occurs, is to break the chip up into smaller blocks and then simulate partially with extracted views and partially with schematic views.

© 2023 All Rights Reserved.

Backannotating Parasitic Values

1. Click the *Backannotate* button on the LVS form to backannotate the resistances and capacitances to the schematic.

   The Parasitic Backannotation form appears.

2. Select the font size and label offsets that you want and click the *Add Parasitics* button.

   Resistance and capacitance labels appear on the schematic view. To see them, you might need to zoom in on a portion of the schematic.

Note: The new information displayed on the schematic is for viewing only. Using the *Add Parasitics* button does not include the parasitics in the schematic.

3. Click the *Remove Parasitics* button to remove these labels.
4. Choose *Print All* to write all of the parasitics to a file.

   The Print All Parasitics form appears.

5. Click the appropriate *Sort Parasitics by* button.

   Select *R* for a list sorted by resistance or *C* for a list sorted by capacitance.

6. Specify the filename for the printed listing.

© 2023 All Rights Reserved.

Diva Flow: Simulating Mixed-Signal Circuits with Parasitic Aware Design

The information in this section describes how you can use Cadence® tools to investigate the effect of parasitics on mixed-signal circuits. By accounting for the effect of parasitics, you can improve the accuracy of your circuit simulations. If your design includes only analog circuits, go to Diva Flow: Simulating Analog Circuits with Parasitic Aware Design.

This section comprises of the following topics:

Note: For more information on Diva see the *Diva Reference*.

© 2023 All Rights Reserved.

Overview

The flows in this section describe two ways to calculate delays for mixed-signal circuits.

- You can estimate delays before layout by using timing library format (TLF) and fan-in and fan-out information.
- You can use layout information to determine delays with increased accuracy.

The pre-layout flow is discussed in Estimating Delays (Pre-Layout) For information on using layout information to calculate delays, see Calculating Delays (Post-Layout).

Important

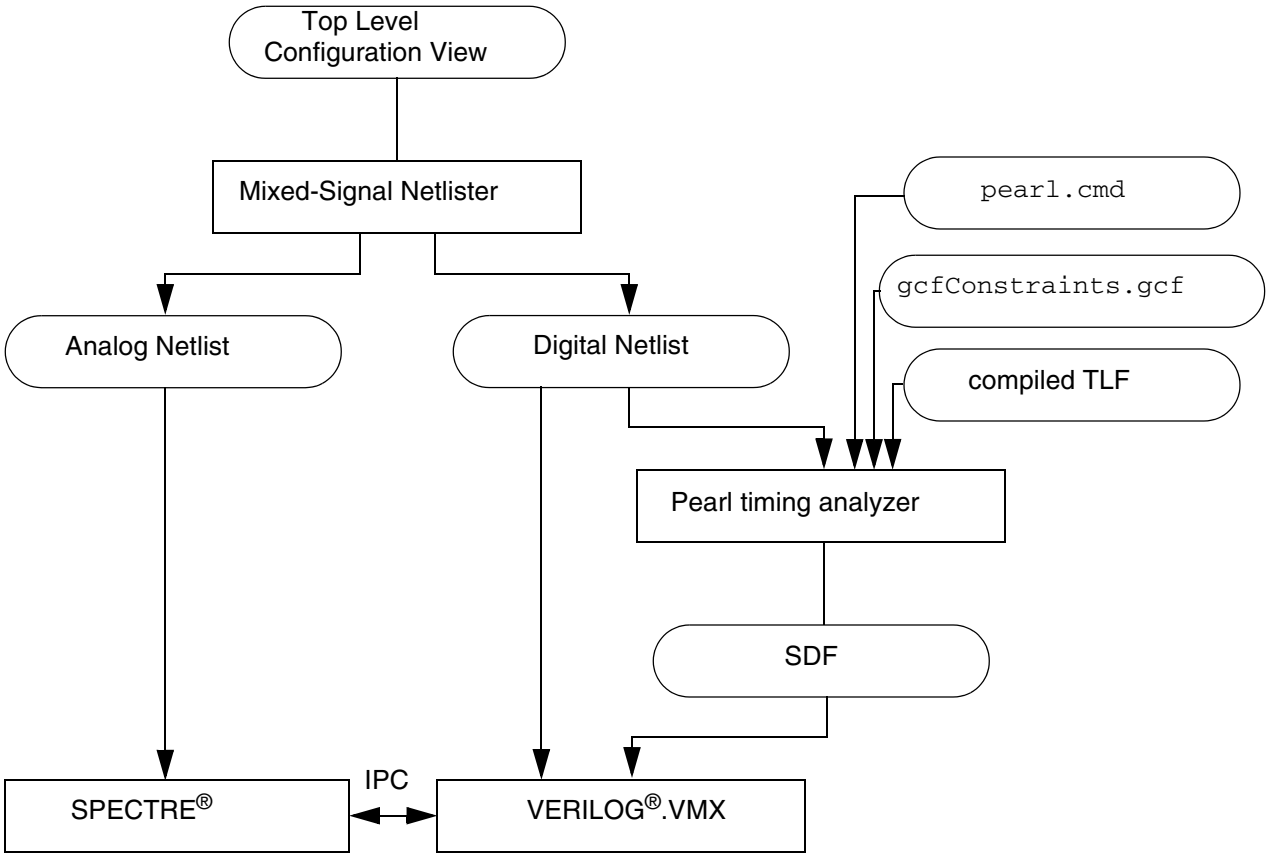
Before following any of the flows in this chapter, be sure that the environment variable `CDS_Netlisting_Mode` is set to `Analog`. To ensure that all the tools for the flow are available, start your session with the command `virtuoso`.

For more information about the options you can use with the command to start the software, refer to the *Virtuoso Design Environment User Guide*.

© 2023 All Rights Reserved.

Estimating Delays (Pre-Layout)

Even without layout information, you can obtain useful delay estimates of digital partitions by following the pre-layout mixed-signal parasitic aware design. The figure below illustrates how the Pearl timing analyzer operates on the digital netlist to produce a standard delay format (SDF) file. The parasitic aware design flow then annotates the SDF file to the top-level cell instance.



Setting Up for Pre-Layout Delay Estimation

To specify that delays are to be estimated, set up the Mixed Signal Options form as described in the following steps.

1. Choose *Simulation – Options – Mixed Signal* in the Analog Design Environment Simulation window.

   The Mixed Signal Options form appears.

2. If necessary, set the *DC Interval* and *Max DC Iterations* fields.

© 2023 All Rights Reserved.

3. Choose the *Estimate (Pre-Layout)* button.

   The MIxed Signal Options form expands to reveal related options.

4. Edit the delay calculator files as necessary.

   For guidance, see Preparing the pearl.cmd and gcfConstraints.gcf Files.

5. Ensure that you have a correctly set up the SDF annotator file (`sdf.cfg`).

   For more information, see Editing the SDF Annotator File.

Preparing the pearl.cmd and gcfConstraints.gcf Files

The Pearl timing analyzer requires two control files: `pearl.cmd` and `gcfConstraints.gcf`. The `pearl.cmd` file is the command initialization file for the Pearl timing analyzer. The `gcfConstraints.gcf` file specifies the boundary and operating conditions for the analysis and lists the compiled timing library format (CTLF) file to be used. You can do either of the following:

- Provide these files in one of the locations listed in Locations Searched for the pearl.cmd and gcfConstraints.gcf Files

  If you provide the files, clicking on the *Command* and *Constraints* buttons opens the files for editing.

- Create these files from templates by clicking on the *Command* and *Constraints* buttons

  If the files do not exist, clicking the buttons copies templates to your run directory and opens the copies for editing.

Locations Searched for the pearl.cmd and gcfConstraints.gcf Files

The Pearl timing analyzer searches for the `pearl.cmd` and `gcfConstraints.gcf` files in the following locations, which are searched in the order given.

1. The run directory

   For example, if the simulation directory is `$HOME/simulation`, the run directory is `$HOME/simulation/`*`topLevelCellName`*`/`*`simulatorName`*`/`*`viewName`*`/netlist/digital`

2. Your working directory (where you start `virtuoso` or `icms`)
3. Your home directory `($HOME)`
4. Your installation path (`$CDS_INST_DIR/tools/dfII/etc/tools/mmsimenv`)

© 2023 All Rights Reserved.

Editing the pearl.cmd and gcfConstraints.gcf Files

You can change the contents of the `gcfConstraints.gcf` and `pearl.cmd` files as necessary.

1. To change the `gcfConstraints.gcf` file, click the *Constraints* button in the Mixed Signal Options form. Your default text editor opens, displaying the contents of the file. For example, the file might contain information llike this.

```
(GCF
(HEADER
(VERSION "1.2")
    (DESIGN "ccadc")
    (TIME_SCALE 1.0E-09)
)
    (GLOBALS
        (GLOBALS_SUBSET ENVIRONMENT
            (OPERATING_CONDITIONS "" 1.00 3.13 100.00)
            (EXTENSION "CTLF_FILES" ("./timing.ctlf"))
                (VOLTAGE_THRESHOLD 10.0 90.0)
        )
    )
    (CELL()
        (SUBSET TIMING
            (ENVIRONMENT
                (INPUT_SLEW 1.60 1.60)
            )
        )
    )
)
```

Change this to the name of your top-level design.

Specify the path to the compiled CTLF file here.

Important

Change the design name to the name of your top-level design. Ensure that the path to the CTLF files is specified with one of the following.

- An absolute path
- A relative path defined with respect to the run directory

For more information about the run directory, see Locations Searched for the pearl.cmd and gcfConstraints.gcf Files.

Do not use a tilde (~) to specify the path.

When you finish editing the file, save it.

2. To change the contents of the `pearl.cmd` file, click the *Command* button in the Mixed Signal Options form. The Command Options form appears.
3. Set the options as required.

© 2023 All Rights Reserved.

4. Click *OK*.

Editing the SDF Annotator File

The simulator uses the `sdf.cfg` file to control the SDF annotation. An existing `sdf.cfg` file that you want the simulator to use must be located in one of the following locations, which are searched in the following order. These are the same locations searched for the `pearl.cmd` and `gcfConstraints.gcf` files.

1. The run directory

   For example, if the simulation directory is `$HOME/simulation`, the run directory is `$HOME/simulation/`*`topLevelCellName`*`/`*`simulatorName`*`/`*`viewName`*`/netlist/digital`
2. Your working directory (where you start `virtuoso` or `icms`)
3. Your home directory `($HOME)`
4. Your installation path (`$CDS_INST_DIR/tools/dfII/etc/tools/mmsimenv`)

To edit the `sdf.cfg` file, or to copy a template so that you can create a new `sdf.cfg` file,

1. Click *Config* on the Mixed Signal Options form.

   The SDF Annotator Config File form appears.
2. Change the values as necessary.

Simulating a Design with Pre-Layout Estimation

After you set up the mixed-signal simulation options, you are ready to simulate. Follow the standard mixed-signal simulation process.

© 2023 All Rights Reserved.

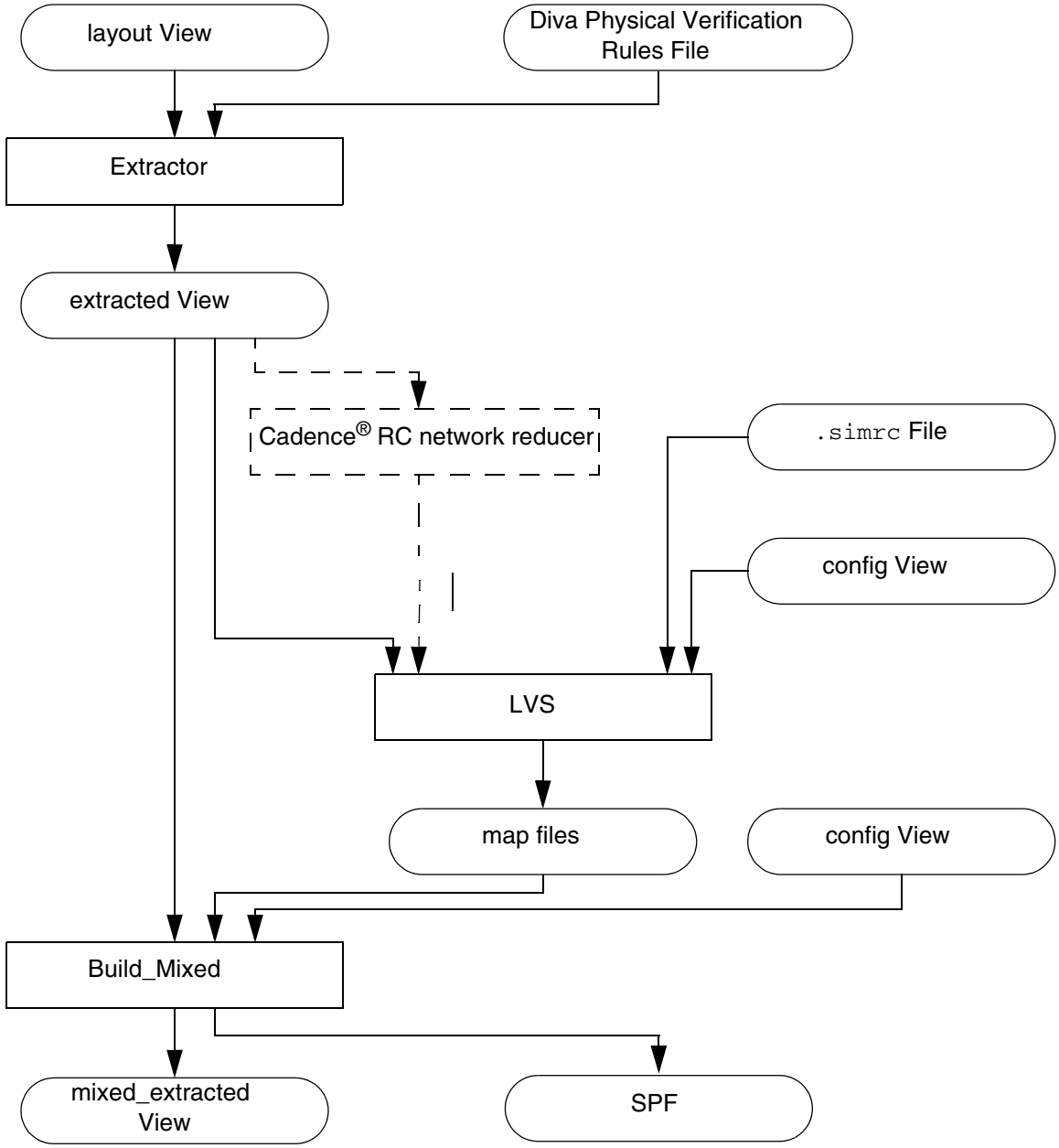
Calculating Delays (Post-Layout)

Simulating a mixed-signal design with parasitics calculated from layout information involves the following major steps:

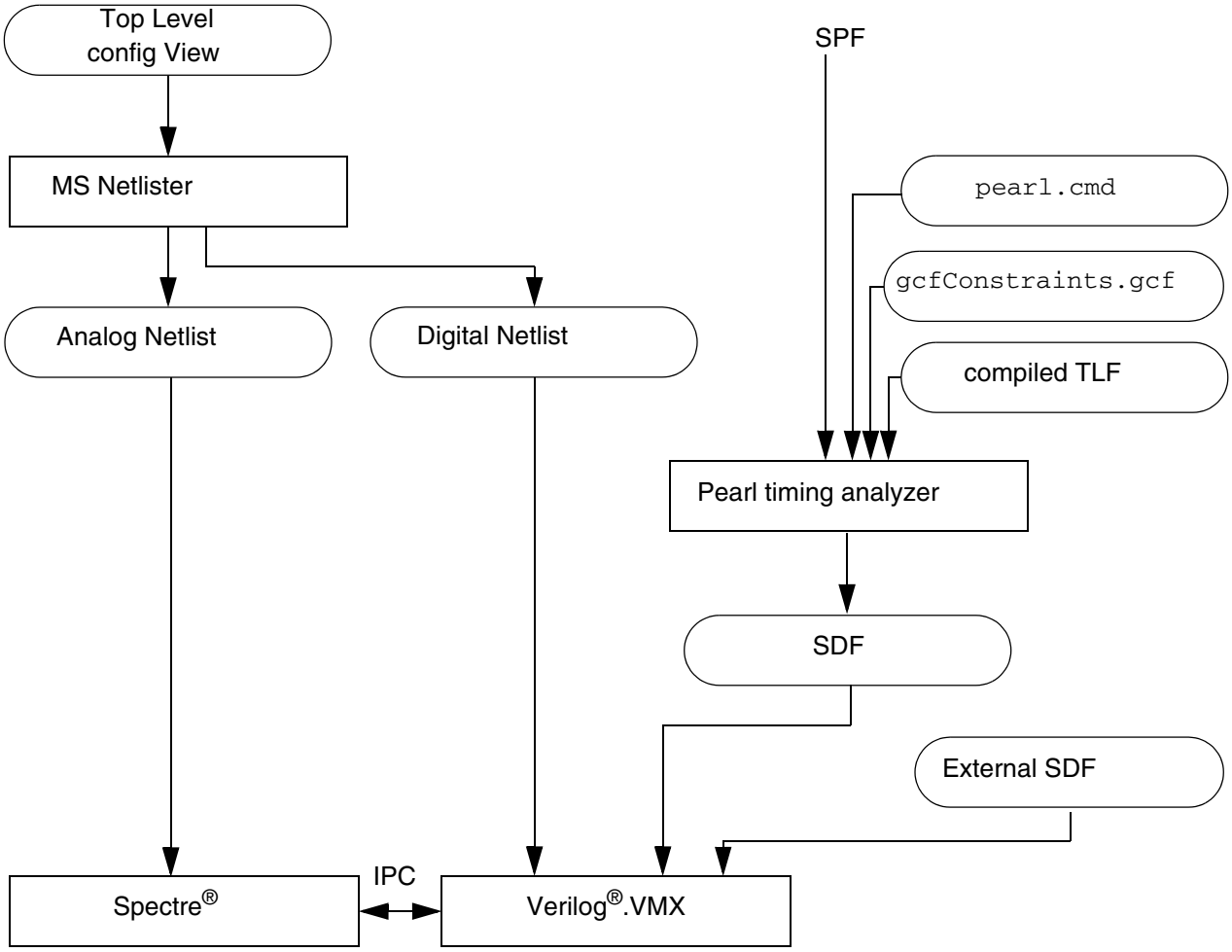
1. Preparing cell libraries
2. Creating a mixed_extracted view of your design
3. Creating or modifying a configuration for the design so that mixed_extracted views are used for the mixed-signal simulation
4. Using one of the mixed-signal simulators to simulate the configured schematic with parasitics included

© 2023 All Rights Reserved.

The figures below shows the flow for steps 2 and 3 in graphical format. The Cadence RC network reducer steps are shown with dotted lines because they are optional.



© 2023
All Rights Reserved.



Digital parasitics are calculated by the Pearl timing analyzer or can be imported from an external calculator. The SDF file created by the timing analyzer is annotated to the netlist at simulation time.

© 2023 All Rights Reserved.

Preparing for Post-Layout Mixed-Signal Parasitic Aware Design

Before you can run a post-layout mixed-signal parasitic aware design, you must ensure that the necessary preliminary steps are complete. The following sections describe the tasks.

- Preparing Libraries for Post-Layout Mixed-Signal Parasitic Aware Design
- Preparing Layout Views for Analog and Digital Cells
- Updating View and Stop Lists for LVS
- Preparing to Create the top.spf File

Preparing Libraries for Post-Layout Mixed-Signal Parasitic Aware Design

Ensure that the cells and primitives that you plan to use in a post-layout parasitic aware design have the following required views and information.

Analog cells must have	Analog primitives must have
schematic view	simulation stopping view
symbol view	auLvs view (the default stopping view for auLvs)
layout view (with ivCellType = “graphic” for analog layouts with pins)	CDF simulation information

Digital cells must have	Digital primitives must have
symbol view	parasitic aware design view
logic view (verilog, for example)	CDF simulation information for LVS
schematic view	If you are using the Pearl timing analyzer, an entry in a compiled timing library format (CTLF) file
layout view (with ivCellType = “graphic” for hierarchical digital blocks)	

The following sections describe how to prepare some of this information.

© 2023 All Rights Reserved.

Creating an parasitic View for a Digital Primitive

Each digital primitive must have an msps stopping view, which is required for layout versus schematic (LVS).

To create parasitic views:

1. In the CIW, choose *Tools – Mixed Signal Environment – Prepare Library for MSPS.*

   The Create msps views & auLvs CDF simInfo form appears.

2. Select the primitives for which you want to create msps views. As described below, you can either select the primitives manually or select primitives that have certain specified views.

Selecting Primitives Manually

To select primitives manually,

1. Choose a cell from the *Not in the Selected List* list box, and click the right-arrow button to add it to the *In the Selected List* list box.

   To select more than one cell, click your first selection, then hold down the `Control` key and click the rest of your selections.

2. To create an msps view for each cell in the *In the Selected List* list box, click *OK* or *Apply* and then click *Yes* in the create msps views confirmation form.

© 2023 All Rights Reserved.

Selecting Primitives with Specified Views

To select primitives that have specified views,

1. Click *Select Cells.*

   The Select Cell Views form appears.

2. Change the *View Choice List* field as necessary.

   The views specified in the *View Must List* field and the *View Choice List* field become the selection criteria for digital primitives. To be selected, a cell must have all the views specified in the *View Must List* field and at least one of the views specified in the *View Choice List* field.

   For example, assume that *View Must List* contains `layout` and `symbol` and that *View Choice List* contains `behavioral` and `auLvs`. Then any cell that has a layout view, a symbol view, and either a behavioral view or an auLvs view meets the search criteria.

3. Click *OK* or *Apply*.

   The search results appear in the *In the Selected List* list box on the Create msps views & auLvs CDF simInfo form.

4. In the Create msps views & auLvs CDF simInfo form, click *OK* or *Apply* to create an msps view for each cell in *In the Selected List*.

5. Confirm your actions by clicking *Yes* in the create msps views confirmation dialog box.

6. If any of the selected cells have existing auLvs CDF siminfo, the create auLvs Siminfo confirmation dialog box asks you to confirm the overwrite.

Preparing Layout Views for Analog and Digital Cells

In macro mode, the extractor treats any cell with pins as a macro cell and stops expanding it. If a block is an analog block or a hierarchical digital block and requires further expansion, you need to add the property `ivCellType = "graphic"` to the layout master of the block. With this property set, the extractor expands the cell even though pins exist.

You can set the `ivCellType` property at the instance level or for multiple cells in the `macroCellFile`. Refer to the *Diva Reference* for details on either of these methods.

For example, the following procedure sets the `ivCellType` property at the instance level for a cell. With this method, every instance of this cell in the design has the same setting.

1. Open in edit mode the layout view of the instance you want expanded to the transistor level.

© 2023 All Rights Reserved.

2. From the Layout window, choose *Design – Properties*.

   The Edit Cellview Properties form appears.

3. Choose *Property*.

   The Edit Cellview Properties form expands.

4. Click the *Add* button.

   The Add Property form appears.

5. Type `ivCelltype` in the *Name* field.

6. Set the *Type* drop-down list box to *String*.

7. Type `graphic` in the *Value* field.

8. Click *OK* to add the new property and its value.

Updating View and Stop Lists for LVS

The `.simrc` file contains the view lists and stop lists for Diva® LVS. For mixed-signal parasitic aware design, you must update these lists with the msps view before you run LVS.

To update the lists,

1. Open the `.simrc` file using any text editor.

2. Add or update the following variable definitions in the `.simrc` file so that the msps view appears at the beginning of each list. For example, after you update the file, the definitions might look like this:

```
lvsSchematicViewList =
    '( "msps" "auLvs"
    "schematic" "symbol")
lvsSchematicStopList =
    '( "msps" "auLvs")
lvsLayoutViewList =
    '( "msps" "auLvs"
    "extracted")
lvsLayoutStopList =
    '( "msps" "auLvs")
```

   For standard settings for these variables, refer to the *Diva Reference*.

3. Save the `.simrc` file.

© 2023 All Rights Reserved.

Preparing to Create the top.spf File

The Pearl timing analyzer uses a standard parasitic format (SPF) file called `top.spf`, which contains the parasitic information for your design. In preparation for creating this file, you must ensure that the property names for resistance and capacitance are set to *r* and *c*.

➤ In the CDF Simulation Information section of the presistor or pcapacitor component, specify the resistance and capacitance parameter names as *r* and *c*.

Creating mixed_extracted Views

For mixed-signal blocks, the extraction process consists of

1. Verifying consistent pin direction in schematic and layout views
2. Extracting parasitics and creating extracted views
3. Comparing the schematic and extracted views
4. Creating mixed_extracted views and (optional) using the Pearl timing analyzer to generate delay calculation files

The mixed_extracted views and the optional SDF files become input to the simulation of the top-level design.

You can run the extraction process on selected blocks within the design or on the entire design.

Verifying Consistent Pin Direction

To verify that pin directions on the schematic and layout views are consistent,

1. From a window displaying the layout or extracted view, choose *Verify – MSPS Check Pins*.

   The MSPS Check Pins form appears.

2. Click *OK*.

   The CIW displays a list of any discrepancies. Fix them before you extract the parasitics.

Extracting Parasitics and Creating Extracted Views

To extract parasitics and create extracted views,

© 2023 All Rights Reserved.

1. From a window displaying a layout view of the cell, choose *Verify – Extract*.

   The Extractor form appears.

2. Choose *macro cell* for *Extract Method*.

   This allows the digital cells to be extracted at the macro level.

   Be sure that any analog blocks have the `ivCellType` property set to `graphic`. This ensures that the analog blocks are flattened. For more information, see "Preparing Layout Views for Analog and Digital Cells" on page 351.

3. Choose *Join Nets With Same Name* (optional).

   This ensures that nets with the same name are joined automatically.

4. Click *Set Switches* to select the type of parasitics you want extracted.

   The Set Switches form appears. The parasitics displayed vary, depending on the extraction rules file defined for your design. In some cases, you do not need to make any selections.

   To select more than one item, click your first selection, then hold down the `Control` key and click the rest of your selections.

5. Click *OK*.

   The Extractor form reappears with the parasitics you selected in the *Switch Names* field.

6. Click *OK* or *Apply* to create the extracted views.

   A message in the CIW tells you when the extraction process is complete.

© 2023 All Rights Reserved.

Comparing Schematic and Extracted Views

To compare the schematic view with the extracted view created earlier, follow these steps.

1. From a window displaying the extracted view, choose *Verify – LVS*.

   The LVS form appears.

   Note: For more information on the LVS form see Verify Menu Commands in the *Diva Reference*.

2. Depending on which views are open, use one of the following procedures to identify the schematic and extracted views that you want to compare.

If both the schematic and extracted views are open	If only the extracted view is open
1. Click the *Sel by Cursor* button below the schematic detail, then click the cursor in the open schematic view.	1. Click the *Sel by Cursor* button below the extracted detail, then click the cursor in the open extracted view.
2. Do the same for the extracted view.	2. Click the *Browse* button below the schematic detail and select the schematic view.

3. Fill in the names of the rules file and rules library for the Diva LVS rules.
4. Click the *Run* button near the bottom of the form to begin the comparison.
5. When the comparison finishes, click *Info*.

   The Display Run Information dialog box appears.

6. Click *Log File*.

   Scroll through the log file to the netlist comparison section near the end of the file. This section identifies any mismatches between the two files. Each error is described in the sections following the comparison results.

   Not all mismatches are fatal. Look over the comparison results to determine if you need to correct one of the files and redo the extraction and comparison, or if you can proceed with the views as they are.

7. Choose *File – Close Window* in the log file window.
8. Click *Cancel* in the Display Run Information dialog box.
9. Correct any problems in the schematic or extracted views.

© 2023 All Rights Reserved.

10. If necessary, rerun the comparison and compare the results.

Building a Mixed_Extracted View

When the comparison between schematic and extracted views is acceptable, you need to select the parasitics to use for simulation. You also need to build the mixed_extracted view.

1. In the LVS form, click *Build Mixed*.

The Build Mixed Extracted View form appears.

2. Verify that the *Library*, *Cell*, and *View* fields correctly specify the configuration view that you want to use.

If your design does not have a configuration view associated with it, refer to the *Cadence Hierarchy Editor User Guide* and create a configuration.

Note: The msps view, used as the digital stopping view for LVS, is also used as the internal stopping view for SPF generation when the build mixed process runs. Be sure the configuration stopping view stops at digital cells that have an msps view.

3. Select one of the following options to specify the analog parasitics that you want to use for simulation.

Select	If you want to
Include All	Simulate with all the parasitics that have been extracted
Set From Schematic	Select parasitics to include in the simulation by placing special symbols (spresistors and spcapacitors) on nets in the schematic view. The parasitics you select are the only ones included in the simulation. The special symbols are available in the `sbaLib` library. If you choose *Set From Schematic* and click *OK* without identifying any nets on the schematic, the Analog Parasitics Selective Annotation form asks you to confirm your choice.
None	Simulate with none of the parasitics

4. Ensure that the `pearl.cmd` and `gcfConstraints.gcf` files are ready and available in one of the following locations, which are searched in the order given.

© 2023 All Rights Reserved.

- The run directory

  library_path/*cell*/*view*/mixed_extracted/layout_msb
- Your working directory (where you start `virtuoso` or `icms`)
- Your home directory
- Your installation path (`$CDS_INST_DIR/tools/dfII/etc/tools/mmsimenv`)

For guidance on using the *Command* and *Constraints* buttons to view or change these files, see Preparing the pearl.cmd and gcfConstraints.gcf Files. When the files are ready, turn on the *Calculate* button in the *Digital Delays* section.

If you click *OK* without editing the `pearl.cmd` and `gcfConstraints.gcf` files or without ensuring that the files are available in the searched directories, the Delay Calculator Option Files message window is displayed.

5. Click *Yes* if you want to use the default templates for the option files.
6. To build the mixed_extracted view, click *OK* in the Build Mixed Extracted View form.

The build mixed process removes all digital parasitics and places them in the SPF file. The Pearl timing analyzer uses the SPF file to calculate the delays and generate an SDF file. The mixed_extracted view contains analog parasitics and analog and digital instances for netlisting and simulation.

The build mixed process creates or places the following files in the `layout_msb` directory.

Filename	Description
`msbCheckFile`	Timestamp file specifying the SPF creation time
`msbEnableFlag`	Zero-length file that indicates the Pearl timing analyzer is on, and enables sdfAnnotate
`pearl.cmd`	Command initialization file for the Pearl timing analyzer
`gcfConstraints.gcf`	File that defines constraints, operating conditions, and compiled timing library format (CTLF) files used by the Pearl timing analyzer
`top.spf`	Detailed SPF file with digital parasitics
`top.tmp.sdf`	SDF file generated by the Pearl timing analyzer delay calculation
`annotate.com`	File that contains a `$sdf_annotate` command with the location of the `top.sdf` file

© 2023 All Rights Reserved.

Filename	Description
`runPearl.log`	Error and log file

Modifying the Configuration

To use parasitic aware design, you must specify mixed_extracted as the *View To Use* for each cell in your top-level design for which you want the extracted parasitics simulated.

To specify mixed_extracted as the *View To Use,* you modify the configuration for your top-level design. If your design does not have a configuration, refer to the *Cadence Hierarchy Editor User Guide* to create one.

To modify a configuration,

1. Open the Hierarchy Editor and specify the configuration for your top-level design.
2. Use one of the following methods to specify the mixed_extracted view for the cells or blocks for which you want parasitics simulated.

To specify views for individual blocks	To specify views for multiple blocks
1. In the *Instance Binding* section of the Hierarchy Editor window, position the cursor in the *View To Use* column of the appropriate block. **Note:** If the *Instance Binding* section is not visible in the window, choose *View – Instance Table* to display this section. 2. Press the right mouse key to display a list of commands. 3. Choose *Select View* to display the list of views for this block. 4. Select *mixed_extracted* as the view for this block to update the *View Found* and *View To Use* fields.	1. Add `mixed_extracted` as the first view in the *Global Bindings View List* text field. This ensures that the mixed_extracted view is the selected view for every cell that has a mixed_extracted view.

3. Choose *View – Update* to reconfigure the design to reflect your changes.
4. To save the configuration with your changes, choose *File – Save*.
5. To close the Hierarchy Editor, choose *File – Exit*.

Partitioning is done automatically by comparing the *Global Bindings Stop List* field and the Analog and Digital Stop View Sets.

© 2023 All Rights Reserved.

Setting the Mixed-Signal Simulation Options

To set up the Mixed Signal Options form for post-layout delay calculations,

1. Choose *Simulation – Options – Mixed Signal* in the Analog Design Environment Simulation window.

   The Mixed Signal Options form is displayed.

2. If necessary, set the *DC Interval* and *Max DC Iterations*.

3. Click the *Use Existing (Layout)* radio button.

   The form expands to allow you to use SDF files created during the build mixed process and to import SDF files.

4. To use the SDF files created during the build mixed process, turn on the *SDF From Mixed Extracted View* button.

5. To import SDF files created by a different tool, turn on the *Import SDF Files* button and fill in the associated fields.

   - In the *File* field, type the path to and filename of the SDF file that you want to import. The name you enter must be a legal Verilog® language name.
   - In the *Scope* field, type the hierarchical scope of the instance for which the delay file is to be annotated during simulation. For example, you might type something like `I1/I3` to indicate an instance one level down in the hierarchy.

6. If you want to import more SDF files, click the *Import More* button and fill in the Import SDF Files form as described in Importing Additional SDF Files.

Importing Additional SDF Files

The Mixed Signal Options form provides space for you to enter the name of one SDF file to be imported. If you want to import more than one SDF file, click the *Import More* button to open the Import SDF Files form.

To use the form,

1. Type a number from 2 to 10 in the *Number of Additional Files To Import* field.

   The form expands to accommodate the information that you want to enter.

2. For each additional file, type the name of the SDF file to be imported.

3. For each additional file, type the hierarchical scope of the instance for which the delay file is to be annotated during simulation.

© 2023 All Rights Reserved.

Simulating a Design (Post-Layout)

After you set up the configuration for the parasitic cells, you are ready to simulate the configured schematic using one of the mixed-signal simulators. Follow the standard mixed-signal simulation process.

© 2023 All Rights Reserved.

Probing Parasitic Values

Although it is not required, you might want to probe the instances of parasitic components.

To probe parasitic values in the schematic or extracted views,

1. In the LVS form, click the *Parasitic Probe* button.

   The Parasitic Probing form appears.

2. In the *Max list size* field, specify how many parasitic instances to display.

3. Sort parasitics by resistance or capacitance by selecting *R* or *C*.

4. Click the appropriate button to specify the parasitics to be collected.

   - Click *Whole Net* and then select a net in the schematic or extracted view to display an ordered list of all the parasitics on the net. The largest resistances or capacitances appear at the top of the list.

     Note: You can click the *Save* button in the displayed Parasitics for net... form to save a text file of the parasitic results.

   - Click *Point to Point* and then select two pins or instance pins (on the same net) in the schematic or extracted view to collect all the parasitics between two points.

   - Click *Net to Net* and then click two nets in the schematic or extracted view to collect parasitic capacitances between two different nets.

   A list of the collected parasitic instances appears.

   Select an instance from this list to highlight the component symbol associated with this parasitic on the extracted view.

5. To backannotate the resistances and capacitances to the schematic view, click the *Backannotate* button on the LVS form.

   The Parasitic Backannotation form appears.

6. Select the font size and label placement that you want and click the *Add Parasitics* button.

   Resistance and capacitance labels appear on the schematic view. To see them, you might need to zoom in on a portion of the schematic. The new information displayed on the schematic is for viewing only. Using the *Add Parasitics* button does not include the parasitics in the schematic.

7. To remove these labels, click the *Remove Parasitics* button.

8. To write all of the lumped parasitics to a file, click *Print All*.

© 2023 All Rights Reserved.

The Print All Parasitics form appears:

9. Click the appropriate *Sort parasitics by* button.

Select *R* for a list sorted by resistance or *C* for a list sorted by capacitance.

10. Specify the filename for the printed listing.Even though you are simulating from the mixed_extracted view of the design, you can probe signals from the schematic view. This is called out-of-context probing. You can also probe the `mixed_extracted` view directly for analog and mixed nets.

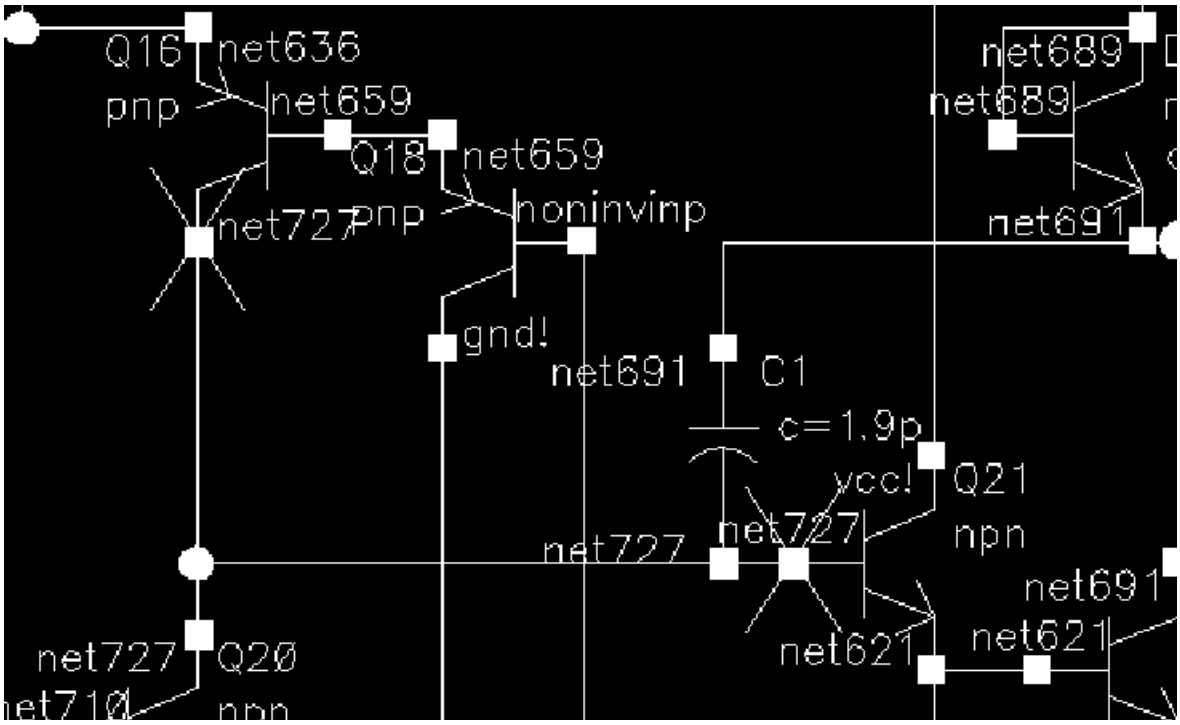
Important

When you probe different types of signals (analog, digital, or mixed), keep in mind which nets exist in the mixed_extracted view. Probing a net or a terminal at a level of the schematic that does not have a simulation waveform causes a probing error.

Because analog components are flat in the mixed_extracted view, you cannot probe nets connected to terminals of hierarchical analog blocks. You must descend to the transistor level of the schematic to probe these nets. It is only at the transistor level that the program can map terminals from the schematic to the mixed_extracted view.

© 2023 All Rights Reserved.

To probe a signal, click the wire at a point close to the terminal. The probe automatically jumps to the closest terminal on that net. An X appears on the selected terminal as shown in the following figure.



You can select several terminals on the same net. Each selected terminal is marked with a different color X. The associated waveform displays in the same color as the X on the schematic.

You can select and probe only real geometries from a mixed_extracted view. For example, if the metal layer is broken up into resistors, the geometries do not have connectivity. In this case, you need to probe the metal layer at contacts and vias.

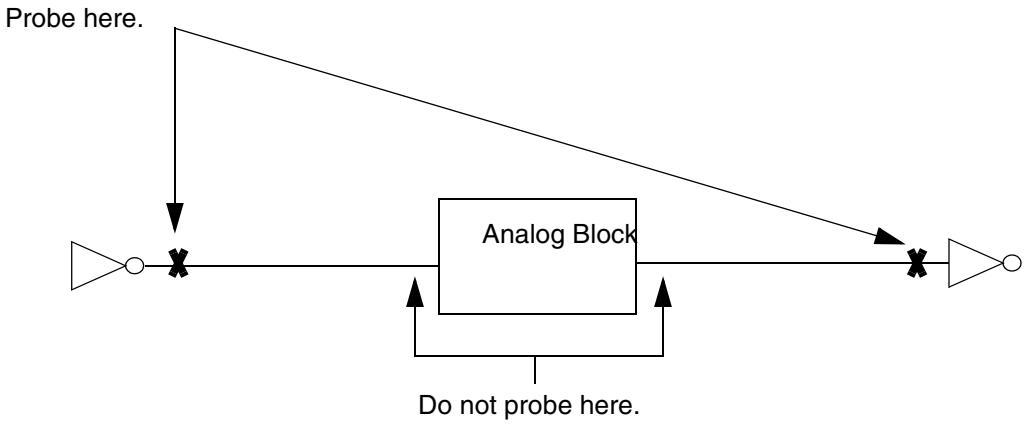
If you are unable to select a geometry on the mixed_extracted view, the layer might be invalid. Set valid layers from the *Edit* menu of the LSW form.

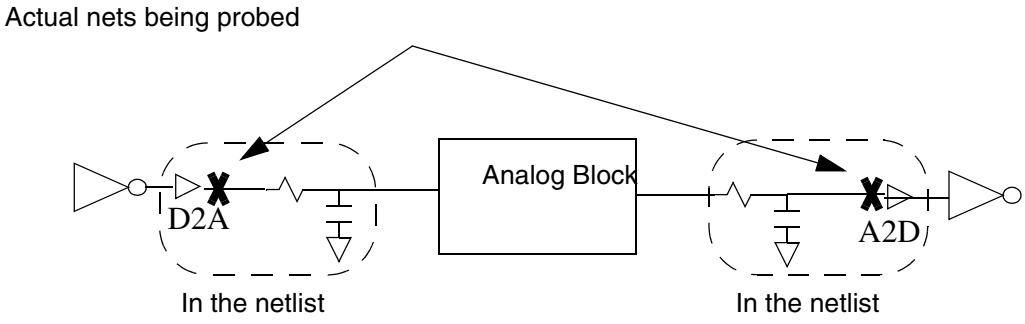
If you probe a net that cannot be mapped to a terminal in the mixed_extracted view, warnings similar to the following appear:

```
*WARNING* Could not obtain the external name
*WARNING* Unable to map net 'VBG'
*Warning* no valid full path name for net "VBG", selection not taken
```


© 2023 All Rights Reserved.

You can probe mixed-nets connected to terminals of hierarchical analog blocks near the digital terminal, but not near analog terminals, as shown in the following figure.





The D2A and A2D elements are attached to the digital components in the netlist. The output of the D2A element or the input of the A2D element, therefore, is a valid analog net in the extracted view.

Because digital parasitics are removed from the mixed_extracted view, digital nets can be probed anywhere and do not have to be associated with terminals. An X is placed in the middle of the net indicating its selection.

Note: For information on probing views that are out of context, see Out of Context Probing.

© 2023 All Rights Reserved.

D

Parasitic Aware Design Workspace Configurations

Important

Parasitic aware design workspace configurations are only available as part of ADE Explorer or ADE Assembler.

When you access ADE Explorer or ADE Assembler functionality, the *Workspace Configuration* toolbar updates to include three Cadence-provided parasitic aware design window workspaces, *Parasitics - Estimated, Parasitics - Compare* and *Parasitics - Extracted*, that comprise of a combination of parasitic aware design and ADE Explorer or ADE Assembler assistant panes.

- For more information on window workspaces, including the ability to customize and save your own, see the *Virtuoso Design Environment User Guide*.
- For more information on the *Parasitics - Estimated* workspace, see Setting Up and Using Parasitics
- For more information on the *Parasitics - Extracted* window workspace, see Parasitic Filters Assistant
- For more information on the *Parasitics - Compare* window workspace, see Parasitic Report Assistant

© 2023 All Rights Reserved.

Availability of Parasitic Aware Design Features

The following table details what parasitic aware design features are available in each application that parasitic aware design can be invoked from:

Feature	VSE L/XL	ADE Explorer / ADE Assembler
Show Parasitics	Yes	Yes
Report Parasitics	Yes	Yes
Out-Of-Context Probing	Yes	Yes
Refine Extracted View	Yes	Yes
Probe Instance Design/Nets	Yes	Yes
Parasitic Filters (including Parasitic Filters assistant)	No	Yes
Specify R Estimates	No	Yes
Specify Coupled C Estimates	No	Yes
Specify Decoupled C Estimates	No	Yes
Parasitics & Electrical Setup (including the Parasitic Estimates assistant)	No	Yes
Build Estimated Schematic View	No	Yes
Compare Estimates with Extracted View	No	Yes

© 2023 All Rights Reserved.

Key Bindings

For details of key binding shortcuts that can be used with parasitic aware design see the `dfII/samples/local/mspsBindKeys.il` in the Cadence installation hierarchy.

Access Keys

Menu access keys also provide keyboard access to functionality and application menus without the need to use mouse selections.

For a full description of access keys see Quick Reference - Menu Access Keys in the *Virtuoso Schematic Editor L User Guide*.

Menu Command	Access Key
Setup	`S`
Options	`O`
Parametric Variables	
Show Parasitics	`P`
Report Parasitics	`R`
Net	`N`
Net to Net	`E`
Terminal to Terminal	`M`
Net Capacitors	`C`
All Nets	`A`
Refine Extracted View	`X`
Probe Design Inst/Net	`D`

Note: The access keys to display the contents of the *Parasitics* menu are: `Alt + R`. If you initially need to add the *Parasitics* menu to the banner menu you should first select `Alt + L` (to show the contents of the *Launch* menu), then select the `R` access key to add the *Parasitics* menu to the Schematics XL/GXL banner menu (the *Parasitics* menu exists by default in the ADE Explorer/ADE Assembler banner menus).

© 2023 All Rights Reserved.

Menu Command	Access Key
Setup	S
Create Estimates	
Create Filters	
Report	
Compare	C
Options	O

© 2023 All Rights Reserved.

Setting Parasitic Aware Design Options Using .cdsenv

As well as setting parasitic aware design options via the user interface, you can also set parasitic aware design options by editing the `.cdsenv` file.

The `.cdsenv` file is automatically read when `virtuoso` and other Cadence tools are invoked. The `.cdsenv` file exists as both a system setup file and an individual user file.

- The **system setup** file is located at:

  (For QRC) `$CDS_INST_DIR/tools/dfII/etc/tools/mspsAv/.cdsenv`

  (For Diva) `$CDS_INST_DIR/tools/dfII/etc/tools/msps/.cdsenv`

  Its content is in the format:

  ```
  tool[.partition] varName type value private {choices, minmax} comment
  ```

  For example:

  ```
  mspsAv.options xOffset float 0.0 nil
  mspsAv.options yOffset float 0.0 nil
  ```

  Editing the system file will impact all users of an installation. Consequently, it can be useful when you want to set company-wide default preferences.

- For an **individual user** the `.cdsenv` file exists in their home directory:

  `~/.cdsenv`

  Its content is in the format:

  ```
  tool[.partition] varName type value
  ```

  For example you could add the following to change an individual user's default x-offset.

  ```
  mspsAv.options xOffset float 0.1
  ```


© 2023 All Rights Reserved.

Error Message Display

Parasitic aware design uses an error message counter to prevent the display of excessively repetitive messages. The `maxConsecutiveMessages` variable can be defined in `.cdsenv`.

```
mspsAv.options maxConsecutiveMessages int 5 nil
```

The above, example, variable setting will ensure that any potential errors occurring, from use of any of the *Parasitics* menu options, will only be repeated up to a maximum of 5 times in the CIW.

© 2023 All Rights Reserved.