SIEMENS EDA

xCalibrate™ Batch User's Manual

Software Version 2021.2
Document Revision 14

SIEMENS

Support Center: support.sw.siemens.com
Send Feedback on Documentation: support.sw.siemens.com/doc_feedback_form

Revision History

Revision	Changes	Status/ Date
14	Modifications to improve the readability and comprehension of the content. Approved by Lucille Woo. All technical enhancements, changes, and fixes listed in the *Calibre Release Notes* for this products are reflected in this document. Approved by Michael Buehler.	Released April 2021
13	Modifications to improve the readability and comprehension of the content. Approved by Lucille Woo. All technical enhancements, changes, and fixes listed in the *Calibre Release Notes* for this products are reflected in this document. Approved by Michael Buehler.	Released January 2021
12	Modifications to improve the readability and comprehension of the content. Approved by Lucille Woo. All technical enhancements, changes, and fixes listed in the *Calibre Release Notes* for this products are reflected in this document. Approved by Michael Buehler.	Released October 2020
11	Modifications to improve the readability and comprehension of the content. Approved by Lucille Woo. All technical enhancements, changes, and fixes listed in the *Calibre Release Notes* for this products are reflected in this document. Approved by Michael Buehler.	Released July 2020

Author: In-house procedures and working practices require multiple authors for documents. All associated authors for each topic within this document are tracked within the Siemens EDA documentation source. For specific topic authors, contact the Siemens Digital Industries Software documentation department.

Revision History: Released documents maintain a revision history of up to four revisions. For earlier revision history, refer to earlier releases of documentation which are available on https://support.sw.siemens.com/.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction to xCalibrate

The xCalibrate™ rule file generator automatically creates the necessary capacitance and resistance rules for accurately extracting parasitic devices.

The following sections provide an overview of the xCalibrate rule file generator:

xCalibrate Overview

xCalibrate™ is a software tool that lets you describe your process technology information and generate a rule file containing Standard Verification Rule Format (SVRF) capacitance and resistance statements.

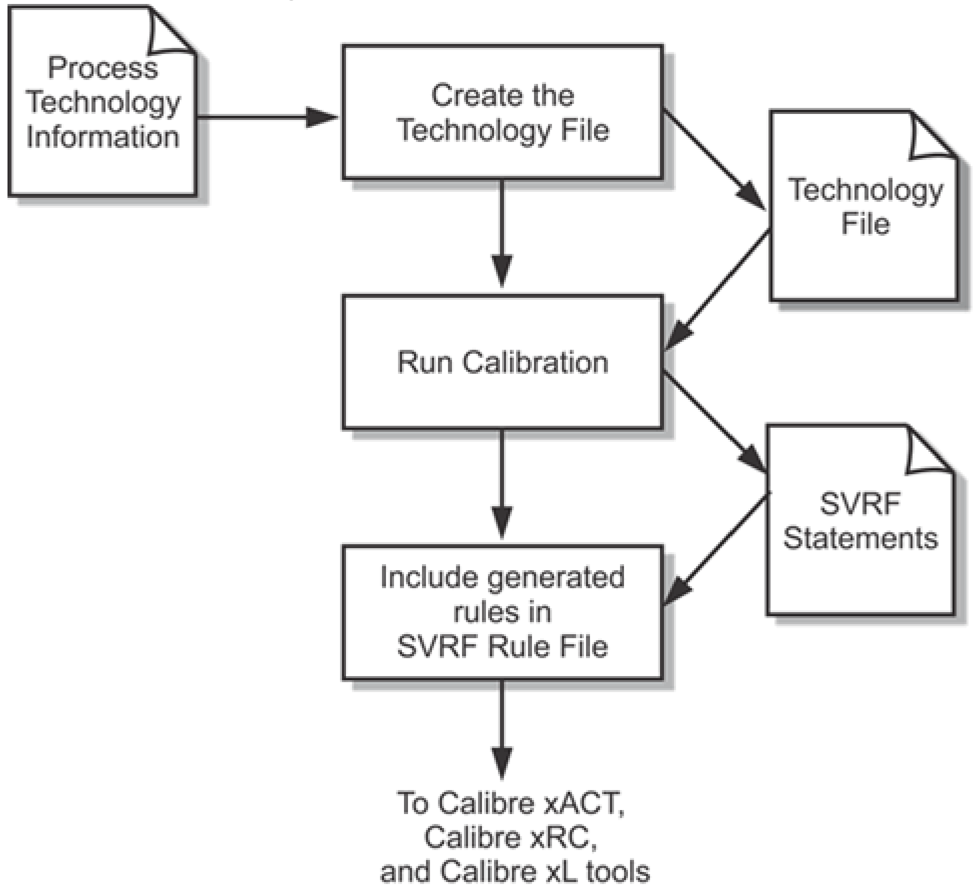
The generated rule files can be included in a larger SVRF rule file for use by the Calibre® xACT™, Calibre® xRC™, and Calibre® xL tools during parasitic extraction. For information on SVRF rules, see *Standard Verification Rule Format (SVRF) Manual*.

The xCalibrate tool automatically creates the necessary capacitance and resistance rules for accurately extracting parasitic devices. If you are not using rule files supplied by your foundry, you need to use the xCalibrate tool to set up rule files for use by the Calibre xACT, Calibre xRC, or Calibre xL tools.

xCalibrate Tool Flow

Figure 1-1 shows the xCalibrate tool flow and how it leads into the Calibre tool flow.

Figure 1-1. The xCalibrate Flow



xCalibrate Requirements

You must have the following to run the xCalibrate tool:

- You must have an xCalibrate license. If you plan to run other Calibre tools, you must have licenses for those tools. For licensing information, see the "Licensing: Parasitic Extraction Products" chapter of the *Calibre Administrator's Guide*.
- You must have set the environment variable CALIBRE_HOME to the location of your Siemens EDA software and set any other environment variables you want to use. For details on the CALIBRE_HOME environment variable, see "Setting the CALIBRE_HOME Environment Variable" in the *Calibre Administrator's Guide*.
- Before running calibration, you must have either a .mipt technology file or a .tm technology file. The .mipt and .tm files are both process technology files and functionally equivalent. The advantage of using .mipt files is that they are easier to produce programmatically. For information on creating .mipt files, refer to "The MIPT File Format" on page 27.

xCalibrate Command Line Usage

The xCalibrate tool is run in batch mode from a shell command line.

The command line usage is shown when

```
xcalibrate
```

or

```
xcalibrate -help
```

is entered on the command line. For more information on the xCalibrate command line syntax, refer to "xCalibrate Invocation Reference" on page 487.

Syntax Conventions

The command descriptions use font properties and several metacharacters to document the command syntax.

Table 1-1. Syntax Conventions

Convention	Description
Bold	Bold fonts indicate a required item.
Italic	Italic fonts indicate a user-supplied argument.
`Monospace`	Monospace fonts indicate a shell command, line of code, or URL. A bold monospace font identifies text you enter.
<u>Underline</u>	Underlining indicates either the default argument or the default value of an argument.
UPPercase	For certain case-insensitive commands, uppercase indicates the minimum keyword characters. In most cases, you may omit the lowercase letters and abbreviate the keyword.
[ ]	Brackets enclose optional arguments. Do not include the brackets when entering the command unless they are quoted.
{ }	Braces enclose arguments to show grouping. Do not include the braces when entering the command unless they are quoted.
‘ ’	Quotes enclose metacharacters that are to be entered literally. Do not include single quotes when entering braces or brackets in a command.
| or ||	Vertical bars indicate a choice between items. Do not include the bars when entering the command.

Table 1-1. Syntax Conventions (cont.)

<table>
<tr><th>Convention</th><th>Description</th></tr>
<tr><td>…</td><td>Three dots (an ellipsis) follows an argument or group of arguments that may appear more than once. Do not include the ellipsis when entering the command.</td></tr>
<tr><td colspan="2">Example:
DEVice {<i>element_name</i> [‘(’<i>model_name</i>‘)’]}
<i>device_layer</i> {<i>pin_layer</i> [‘(’<i>pin_name</i>‘)’] …}
[‘<’<i>auxiliary_layer</i>‘>’ …]
[‘(’<i>swap_list</i>‘)’ …]
[BY NET | BY SHAPE]</td></tr>
</table>

Chapter 2
Getting Started with xCalibrate

xCalibrate takes process technology information specified in an MIPT file and generates calibrated rule files used by the Calibre parasitic extraction tools.

Creating a Technology File

To create calibrated rule files you must first create the MIPT file that xCalibrate accepts as input. An MIPT file is a technology file that contains a text description of the process technology stack.

Prerequisites

- Process technology data obtained from your foundry for the desired process.
- A valid xCalibrate license.

Procedure

1. Open a file in your favorite ASCII text editor and name it with a *.mipt* file name extension. Certain rules must be followed when creating or editing an MIPT file. See "Rules for Creating an MIPT File" on page 29 and "MIPT File Elements" on page 43 for a details.

2. Specify the header information. The global parameters mipt_version, process, and calibration_type are required.

   The mipt_version parameter must always be set to 2.0.

   ```
   mipt_version = 2.0
   ```

   The process parameter is a user specified name for the process. For example:

   ```
   process = miptexample
   ```

   The calibration_type parameter specifies what type of calibrate rule files to generate. The choices are RONLY, FIELDSOLVER, RULEBASED, and ALL. More than one type can be specified. For example:

   ```
   calibration_type = {RONLY RULEBASED}
   ```

   For a complete list of global parameters and their descriptions see "Global Parameters" on page 43.

3. Using the process technology data obtained from your foundry specify the layers in the process stack. Each layer type has a minimum set of required parameters used to describe the layer. Define the layers in their relative order in the stack.
4. Use Table Syntax to input process variation information if needed.
5. Save and close the MIPT file.
6. You can check for syntax errors using the xCalibrate syntax checker:

   ```
   xcalibrate -check myfile.mipt
   ```

   Correct any syntax errors before running xCalibrate.

Results

Your MIPT file is now ready for calibration.

You can also use the stack viewer to visualize the stack. See "Stack Viewer" on page 481 for more information about the stack viewer.

Running Calibration

Calibrated rule files are generated by xCalibrate using the MIPT file as input.

Prerequisites

- A syntactically correct MIPT file.
- A valid xCalibrate license.

Procedure

1. Verify the MIPT file has no syntax errors:

   ```
   xcalibrate -check process.mipt
   ```

   Check the transcript for errors and warnings. If no errors were encountered, xCalibrate finishes with the following message:

   ```
   xCalibrate finished successfully ...
   ```

   Review any warnings that appear in the transcript and make adjustments to the MIPT file if needed.
2. Run calibration:

   ```
   xcalibrate -exec process.mipt
   ```

   If no errors were encountered, xCalibrate finishes with the following message:

   ```
   xCalibrate finished successfully ...
   ```

Results

The current working directory contains the calibrated rule files rules.C, rules.R, and rules.xact. A large part of each file is encrypted to protect IP. The calibrated rule files contain the modeling information needed for extraction. Include these files in the top-level rule file defined for Calibre xRC or Calibre xACT when performing extraction.

Chapter 3
MIPT File Format

The MIPT file format enables you to create an ASCII file containing process technology information.

This chapter includes the following sections that provide information on the MIPT file format:

The MIPT File Format

The MIPT file format enables you to create an ASCII file containing process technology information. It is a master profile containing all of the layers, with optional parallel device and interconnect stacks if needed.

The MIPT 2.0 file format includes:

```
# header
mipt_version = 2.0
process, date and contact information
calibration_type = ALL # where ALL is one of 4 keyword choices

# base layer:
base = base_name { # Layers must be named
   properties # Layers have required and optional properties
}

# dielectric layers:
dielectric = d1 { # where d1 is the user specified name of the dielectric
   properties
}
```

```
dielectric = d2 { # where d2 is the user specified name of the dielectric
   properties
}

# metal layer
conductor = m1 {
   properties
}

dielectric = d3 {
   properties
}

# metal layer
conductor = m2 {
   properties
}

# via layers:
via = via1 {
   properties
}

via = via2 {
   properties
}
```

MIPT File Data

Before creating an MIPT file, you need the conductor and dielectric information for your process technology.

The MIPT file should include the following data for each layer:

- Thickness
- Conductor's minimum width
- Conductor's minimum spacing
- Distance between conductor layer and substrate
- Conductor layers' biasing
- Conductor layers' scaling
- Dielectric constant

For calibration, the MIPT file must define at least one metal layer.

Rules for Creating an MIPT File

Certain rules must be followed when creating or editing MIPT files.

Follow these guidelines:

- Type the keyword as the first token on the line, followed by a space (optional), an equal sign, a space (optional), and then the argument or values on the same line. Keywords are not case sensitive. For example, the following three cases are valid and equivalent:

  ```
  thickness = 0.085
  THICKNESS =0.085
  Thickness=0.085
  ```

  The following is not valid:

  ```
  thickness =
  0.085
  ```

  For more information on keywords, refer to Keywords.

- Valid layer type keywords are base, dielectric, conductor, poly, diffusion, li, contact, via, and seed.
- Keywords may not be used as layer names or any other reference names.
- When you define a layer, the layer type keyword must come first, followed by an equal sign then the layer name.

  A layer must be given:

  - a name.
  - a list of layer properties enclosed in braces ({ }). The opening brace must follow the layer name on the same line. Layers have both required and optional properties. These properties can be specified in any order. Failure to specify a required property for a layer will generate an error during calibration.

  The following is an example of a valid layer definition:

  ```
  conductor = metal1 {
     min_width = 0.065
     min_spacing = 0.065
     r_sheet =0.38
     thickness = 0.13
     extension = 0.01
  }
  ```

The following layer definition is not valid because the opening brace is not on the same line as the layer name:

```
conductor = metal1
{
   min_width = 0.065
   min_spacing = 0.065
   r_sheet =0.38
   thickness = 0.13
   extension = 0.01
}
```

xCalibrate issues the following message when parsing the MIPT file:

```
Error XCAL_2_002: MIPT Parser Error:Expected "<layer_type> =
<layer_name> {" near line 9 in file tech_45nm.mipt
```

The following layer definition is not valid because it is missing the required parameter thickness:

```
conductor = metal1 {
   min_width = 0.065
   min_spacing = 0.065
   r_sheet =0.38
   extension = 0.01
}
```

xCalibrate issues the following message when parsing the MIPT file:

```
Error XCAL_2_002: MIPT Parser Error:Missing conductor layer
parameter "thickness" for layer near line 94 in file tech_45nm.mipt
```

- Layer definitions may appear in any order in the file.
- Use the comment character '#' at the start or middle of a line to indicate the text that follows is a comment. All characters after '#' are ignored. For more information on comments, refer to Comments.
- Numbers can be floating point and integers. Negative numbers must be prefixed with a negative sign (-).
- The units for distance, capacitance, and inductance can be changed using the global parameters distance_unit, cap_unit, and inductance_unit.
- Use the identical capacitance unit in the technology file as the CAD person will use in the Unit Capacitance SVRF statements.
- If possible, use the identical conductor layer names in the technology file as the CAD person will use in the Connect SVRF statement.
- You must use a *.mipt* file name extension when naming your MIPT file.
- It is recommended that you define the layers in their relative order as much as possible.

MIPT Syntax Conventions

The MIPT language follows certain syntax conventions.

Keywords

Keywords are assigned a value by using the equals sign (=).

All keywords must have:

- A value assigned to them or a parsing error is generated.
- Values specified on same line as the keyword.
- A single value assigned as a single one line string without spaces. Multiple string values may be assigned, but must be enclosed within quotes.

Keywords cannot be used as reference names. For more information on available MIPT keywords see MIPT File Elements.

Units

Unit values for different MIPT parameters are defined by the tool.

The default units used for MIPT values are shown in Table 3-1:

Table 3-1. Default Units for MIPT Values

Property	Units
Distance	microns (um)
Temperature	Celsius
Resistivity	ohms
Capacitance	femtofarads (1e-15 farads)

Table 3-1. Default Units for MIPT Values (cont.)

Property	Units
Inductance	picohenrys (1e-12 henrys)

Distance

The default unit for distance is micron, also referred to as micrometer, um, or 1e-06. To specify a different unit for distance, the distance_unit parameter may be specified in your MIPT file:

```
distance_unit = unit
```

This optional parameter affects the global setting for all distance values and must be specified as a global parameter and not as part of another keyword definition.

Table 3-2 lists the unit values that may be specified for distance_unit:

Table 3-2. Distance Units

Unit Setting	Description
a	angstrom (1e-10)
nm	nanometer (1e-09)
um	micrometer or micron (1e-06)

For example, to specify all distance units in nanometers, include the following parameter specification in your MIPT file:

```
distance_unit = nm
```

Temperature

The unit for temperature is degrees Celsius. It is not possible to change the temperature unit.

Resistivity

The unit for resistivity is ohm * distance_unit. The unit for sheet resistance is ohm. It is not possible to change the resistivity or sheet resistance units. Resistance is a function of temperature.

Capacitance

The default unit for capacitance is femto-farads (1e-15 farads). To specify a different unit for capacitance, the cap_unit parameter may be specified in your MIPT file:

```
cap_unit = unit
```

This optional parameter affects the global setting for all capacitance values and must be specified as a global parameter and not as part of another keyword definition or block.

Table 3-3 lists the unit values that may be specified for cap_unit:

Table 3-3. Capacitance Units

Unit Setting	Description
af	attofarad (1e-18 farads)
ff	femtofarad (1e-15 farads)
pf	picofarad (1e-12 farads)
nf	nanofarad (1e-09 farads)
uf	microfarad (1e-06 farads)
mf	millifarad (1e-03 farads)
f	farad (1 farad)

For example, to specify all capacitance units in picofarads, include the following parameter specification in your MIPT file:

```
cap_unit = pf
```

Inductance

The default unit for inductance is picohenry (1e-12 henrys). To specify a different unit for inductance, the inductance_unit parameter may be specified in your MIPT file:

```
inductance_unit = unit
```

This optional parameter affects the global setting for all inductance values and must be specified as a global parameter and not as part of another keyword definition.

Table 3-4 lists the unit values that may be specified for inductance_unit:

Table 3-4. Inductance Units

Unit Setting	Description
ah	attohenry (1e-18 henry)
fh	femtohenry (1e-15 henry)
ph	picohenry (1e-12 henry)
nh	nanohenry (1e-09 henry)
uh	microhenry (1e-06 henry)
mh	millihenry (1e-03 henry)
h	henry (1 henry)

For example, to specify all inductance units in nanohenrys, include the following parameter specification in your MIPT file:

```
inductance_unit = nh
```

Case Sensitivity

The MIPT syntax is case sensitive. The case of keywords and values is preserved and output. This includes layer names and any other specified labels.

Comments

There are two types of comments supported, block and single line.

Single line comments begin with a pound sign (#) character. This character may appear in any position on the line. All subsequent characters appearing after the pound sign (#) are ignored. For example:

```
# This is a single line comment.
This is a comment from # this point on.
```

Block comments begin with the /* string and end with the */ string. These may surround a single line or multiple lines. For example:

```
/* This is a single line block comment.
*/

/* This is a multiple
   line
   block
   comment. */
```

Block comments can not be nested within other block comments.

Use Directive

The use keyword is an optional text string that enables you to reuse named objects to create new objects. This is a special keyword that may be used more than once.

The use keyword syntax is:

```
# <keyword> = <reference_name> use <name of object to be duplicated>.
dielectric = mydiel1 use poly_diel
```

An optional modifier called with is permitted in the use command. Properties from the re-used object may be overridden or added to the new object by including the with modifier followed by the list of properties enclosed in braces ({ }). This modifier may be applied to any use statement. For example:

```
# <keyword> = <reference_name> use <name of duplicated object> with {
#  properties
#}
dielectric = mydiel1 use poly_diel with {
   thickness = 0.299
   eps = 3.95
}
```

The following is an example of how to define a new layer pc_par from an original layer pc:

```
# original layer is pc, but we want to rename it to pc_par
poly = pc_par use pc {
      //thickness for pc_par, overwrites the thickness for pc
      thickness = 0.343
      r_sheet = 0.23
}
```

Encryption

#ENCRYPT and #DECRYPT are directives used to control the encryption of SVRF statements.

The SVRF statements are defined within the MIPT file using the optional global parameter svrf_verbatim. These optional directives may only be used when specifying multiple lines within braces.

Add the #ENCRYPT directive at the start of the block and the #DECRYPT after the last line of the block you want to encrypt:

```
svrf_verbatim = {
   PEX REDUCE ANALOG YES
   #ENCRYPT
   // capacitance controls
      CAPACITANCE ORDER poly m1 m2
      N4_PEX = M4 NOT GATE
   #DECRYPT

}
```

Statements specified outside of the directives appear in the header section of the calibrated rule file. Statements specified within the block are encrypted:

```
...
PEX REDUCE ANALOG YES
#DECRYPT %'+~$IA7I@3^UE:P%1QMT4:*!H;J;1Q!!"[X<'@?U.`WA<L7,O4"R02A!!"
R*E"CSA\+0^Y:"RO@VLX4!!!"E^W["MS@>7R,=2Q$9L>X5@N3-\/>>%CEC7)4!AXFL:Q!#
,L#I!]@%B%5E2.(%Q,^$DQ!!")9OAR4C?L!@9&WA#J']E[1!!"
#ENDCRYPT
...
```

Hidden Layer Directives

The #HIDDEN and #ENDHIDDEN are directives used to control the encryption of layers in the calibrated rule file.

The layers defined within the scope of the #HIDDEN and #ENDHIDDEN directives will be encrypted in the output calibrated rule files. If both directives are not specified a warning is generated during calibration and the layers remain visible.

Add the #HIDDEN directive at the start of the block and the #ENDHIDDEN after the last line of the block you want to hide:

```
...

#HIDDEN
base = field_base {
   zbottom = -1
   thickness = 1
}

dielectric = field_base_diel {
   diel_type = planar
   # zbottom = 0
    thickness = 0.4
   eps = 3.29
}
#ENDHIDDEN

conductor = poly {
   # zbottom = 0.4
   thickness = 0.1
   min_width = 0.046
   min_spacing = 0.048
   max_width = 0.184
   max_spacing = 5
   r_sheet = 9
}

...
```

Statements specified within the block are encrypted and layers specified outside of the directives appear in the calibrated rule file, for example:

```
...

#DECRYPT %?)~5@~
SQC#A;=D'\OI'/E=T:%D$HW2TW,5~1KT(UWI1)\M!#DZMXS<#)#,<JT>DV)$07QA! !
"_Y4PT;/3P?OCL63J"$YM@A! !"SJ6IG,2X$R,~)?]4Z?N,'/ZU"F3GT--
L'_NU=MP1R2>7OI&10VTB6RO,+^"*ML=!F*V"+&.B-5F-Q;W:2+-JWL!!"])J#J*;6T^-
4;4F:+SIN-!!!"
#ENDCRYPT

XCALIBRATE__poly CONDUCTOR 0.4 0.5 MINWIDTH 0.046 MINSPACING 0.048
EXTENSION 0 MAXWIDTH 0.184 MAXSPACING 5

...
```

C Preprocessor Directives

The C preprocessor directives and the -cpp command line option can be used to control which portions of the MIPT file are processed.

The MIPT file can be preprocessed to contain specific blocks of syntax using conditional compilation with the -cpp command line option in two ways.

- Generate the preprocessed MIPT file (*filename_cpp.mipt*).

  ```
  xcalibrate -cpp mytech_file.mipt
  ```

- Generate the preprocessed MIPT file and run calibration.

  ```
  xcalibrate -exec -cpp mytech_file.mipt
  ```

If the MIPT file is calibrated without the -cpp command line option, then the C preprocessor directives such as #IFDEF, #ENDIF, and #DEFINE directives are treated as comments. For more information on the -cpp command line option, see "xcalibrate -exec" on page 489 or "xcalibrate -cpp" on page 503.

The svrf_verbatim parameter is ignored during C preprocessor execution to avoid potential conflicts with macro syntax that may be used to specify SVRF statements. The svrf_verbatim specifications are written directly to the *filename_cpp.mipt* file.

The syntax for the #IFDEF directive is:

#IFDEF *name*

where ***name*** is a single alpha numeric string. More than one #IFDEF directive can be specified. Each #IFDEF directive must have a corresponding #ENDIF directive.

MIPT syntax specified within the scope of the #IFDEF and #ENDIF directives is only recognized when the #DEFINE for the #IFDEF condition is specified. The syntax for #DEFINE is:

#DEFINE *name*

If the #DEFINE is not specified, then the syntax within the #IFDEF and #ENDIF directives is ignored.

Consider the following example file called *mytech.mipt*, which contains two #IFDEF directives called TECH1 and TECH2, and only one #DEFINE for TECH1:

```
mipt_version = 2.0
process = cmos
author = Foundry
calibration_type = ALL
#DEFINE TECH1
...

base = base_diel {
   zbottom = -1
   thickness = 1
}

dielectric = tech_base_diel {
   diel_type = planar
    thickness = 0.4
   eps = 3.29
}

#IFDEF TECH1
conductor = M1_tech1 {
   # zbottom = 0.4
   thickness = 0.1
   min_width = 0.044
   min_spacing = 0.055
   max_width = 0.19
   max_spacing = 5
   r_sheet = 9
}
#ENDIF
...
#IFDEF TECH2
conductor = M1_tech2 {
   # zbottom = 0.4
   thickness = 0.1
   min_width = 0.033
   min_spacing = 0.044
   max_width = 0.20
   max_spacing = 5
   r_sheet = 9
}
#ENDIF

...
```

The following command:

```
xcalibrate -cpp mytech.mipt
```

produces a file called *mytech_cpp.mipt* that only contains the conductor layer named M1_tech1:

```
# 1 "mytech.mipt_tmp"
# 1 "<built-in>"
# 1 "<command-line>"
# 1 "mytech.mipt_tmp"
mipt_version = 2.0
process = cmos
author = Foundry
calibration_type = ALL
...

base = base_diel {
   zbottom = -1
   thickness = 1
}

dielectric = tech_base_diel {
   diel_type = planar
   thickness = 0.4
   eps = 3.29
}

conductor = M1_tech1 {
   # zbottom = 0.4
   thickness = 0.1
   min_width = 0.044
   min_spacing = 0.055
   max_width = 0.19
   max_spacing = 5
   r_sheet = 9
}
...
```

The conductor layer definition for M1_tech2 is ignored.

You can also nest the #DEFINE, #IFDEF, and #ENDIF directives. The following example demonstrates how to define different stacks using nested directives:

```
#DEFINE BEOL_STACK4
#DEFINE PACKAGE_OPTION2
#IFDEF BEOL_STACK3
#DEFINE M3
#ENDIF
#IFDEF BEOL_STACK4
#DEFINE M3
#DEFINE M4
#ENDIF
#IFDEF BEOL_STACK5
#DEFINE M3
#DEFINE M4
#DEFINE M5
#ENDIF
...
mipt_version= 2.0
process = my_process
calibration_type= ALL
...
conductor = metal1
via = via1
...
conductor = metal2
...
#IFDEF M3
conductor = metal3
via = via2
dielectric = metal3_diel
#ENDIF

#IFDEF M4
conductor = metal4
via = via3
dielectric = metal4_diel
#ENDIF

#IFDEF M5
conductor = metal5
via = via4
dielectric = metal5_diel
#ENDIF
...
#IFDEF PACKAGE_OPTION1
“Pkg1”
#ENDIF
#IFDEF PACKAGE_OPTION2
“Pkg2”
#ENDIF
```

By specifying “#DEFINE BEOL_STACK4” and “#DEFINE PACKAGE_OPTION2”, the preprocessed MIPT file contains only the requested layers and package definition syntax:

```
# 1 "mytech.mipt_tmp"
# 1 "<built-in>"
# 1 "<command-line>"
# 1 "mytech.mipt_tmp"
mipt_version= 2.0
process = my_process
calibration_type= ALL
...
conductor = metal1 {
...
}
via = via1 {
...
}
conductor = metal2 {
...
}

conductor = metal3 {
...
}
via = via2 {
...
}
dielectric = metal3_diel {
...
}

conductor = metal4 {
...
}
via = via3 {
...
}
dielectric = metal4_diel {
...
}
...
"Pkg2"
...
```

Calibration Types

Calibration types identify the intended used for the MIPT file.

The calibration_type keyword is a required keyword used to indicate the possible uses of a MIPT file. It accepts one or more of the following values for type:

- RONLY

Specifies that the MIPT file is only intended for RONLY calibrations. When doing an RONLY calibration, only the technology header and resistance rules are calibrated. It may not be suitable for other calibrations due to insufficient data or accuracy of data.

If the stack specified in your MIPT file does not contain any dielectric layers, then it may only be used for calibration of RONLY rules.

- FIELDSOLVER

  Specifies that the MIPT file is only intended for FIELDSOLVER calibrations. When doing a FIELDSOLVER calibration, only the technology header and resistance rules are calibrated. It may not be suitable for other calibrations due to insufficient data or accuracy of data.

- RULEBASED

  Specifies that the MIPT file is only intended for RULEBASED calibrations. This type produces the same output as is currently expected for traditional calibrations. It may not be suitable for other calibrations due to insufficient data or accuracy of data.

- ALL

  Specifies that the MIPT file is intended for ALL calibrations. It may be used for other calibrations as specified on the command line.

If multiple types are specified they must be enclosed in braces ({ }). For example:

```
# the following are all valid examples
of how to specify calibration_type
# but remember you may only specify this keyword once in your MIPT file.
calibration_type = RONLY

# or with braces
calibration_type = {ALL}

# or a list of types, this is equivalent to specifying ALL.
calibration_type = {RONLY FIELDSOLVER RULEBASED}
```

Chapter 4
MIPT File Elements

MIPT files are composed with certain file elements.

This chapter provides a summary of information on elements used to create MIPT files.

Global Parameters

The global parameters specify global information about the process. Specify this information before using the layer keywords.

At minimum the following global parameters must be specified:

```
mipt_version = 2.0
process = allLAYER
calibration_type = ALL
```

Typically MIPT files also specify the background dielectric and capacitance unit settings:

```
# Required syntax
mipt_version = 2.0
process = allLAYER
calibration_type = ALL
# Optional but frequently used
cap_unit = pf
background_dielectric = 2.5
```

Table 4-1 lists the global parameters.

Table 4-1. Global Parameters

Global Parameters	Description
mipt_version	Required parameter that specifies the version number of the MIPT syntax used in the MIPT file.
process	Required parameter that specifies the name of the manufacturing technology (process) being used.
calibration_type	Required parameter that specifies the type of calibration used or possible uses of the MIPT file.
author	Optional parameter that specifies the author of the MIPT file and other related author information.
background_dielectric	Optional real number or integer that specifies the value of the default dielectric constant (permittivity).
cap_unit	Optional parameter that specifies the units for capacitance values.
comment	Optional parameter used to specify a string that describes comments and other information for the manufacturing technology (process) being used.
corner	Optional parameter that specifies the corner variation for the process.
date	Optional parameter that specifies the date the MIPT file containing the description of the process was created.
distance_unit	Optional parameter that specifies the units for distance values.
env	Optional parameter that specifies one or more environment variables used during calibration.

Table 4-1. Global Parameters (cont.)

Global Parameters	Description
half_node_scale_factor	Optional parameter that specifies the amount by which to scale the layout. Scaling affects a polygon’s width, length, and spacing properties.
inductance_unit	Optional parameter that specifies the unit of inductance to use for calculating the self inductance of vias.
li_device_model	Optional parameter that specifies the type of device model used during calibration. The default device model is CMOS if this parameter is not specified.
plate_loading	Optional parameter that specifies whether the models and rules apply loading effects to large plates or very wide wires.
process_foundry	Optional parameter that specifies the process foundry name.
rsh_type	Optional parameter that specifies how to interpret the indices for the sheet resistance (rsh) process variation table.
svrf_verbatim	Optional text string or text block that allows the insertion of SVRF statements in to the calibrated rule deck.
temperature	Optional parameter that specifies the nominal temperature for the calibration in units Celsius. The default value is 25 C.
tsv_model	Optional parameter that specifies whether the TSV radius is retrieved from the rules (device) or the layout (layout) during calibration. The default is device.
tsv_radius_type	Optional parameter that specifies whether the radius of the TSV is metal or hole. The default is metal.
version	Optional parameter that specifies the version number of the manufacturing technology (process) being used in the MIPT file.
via_r_extrapolation	Optional parameter that specifies whether or not to extrapolate the variable via_resistance table boundary values for all via and contact layers in the MIPT file.

Layer Definitions

The process technology is principally defined by the layers.

The layer definition syntax is:

```
layer_type = layer_name {
   list_of_parameters
}
```

A *layer_type* keyword is followed by an equal sign, a unique user-specified *layer_name*, and a list of parameters enclosed in required braces, ({ }). The layer parameters describe the properties of the layer. Each parameter must appear on its own line. The parameters within the braces are order independent.

There are three categories of required layers:

- Dielectric layers
- Conducting layers:
    - Conductor
    - Ground
    - Poly
    - Diffusion
    - li (Metal or Interconnect layers)
- Contact and Via layers

The following layer types are optional:

- Base
- Base_via
- Device_li
- Derived
- Multigate
- Pad
- PCaux
- Resistor
- Src_drn
- Seed

- TSV
- uBump

Base

The base layer is a layer used to define the base reference plane. All other layers should start after the base layer. The base layer should not be overlapped with any other layer.

Base layers can be specified in two different ways. In the MIPT file, it is possible to have:

- a single base plane from which all dimensions are defined.
- multiple base layers where each base layer definition represents a different possible ground plane height.

Single Base Layer

For a single base plane, the reference is established at absolute zero. This plane is defined by defining the base layer. The base keyword defines the reference plane by which all other layers are measured. All other dimensions in the file are relative to the top z-coordinate (ztop) of this reference plane. This reference plane is used consistently for the interconnect and parallel (device) stack descriptions. A base layer can not be used with the Use Directive.

Parameters ztop or zbottom, and thickness are required to establish the base plane. If ztop is used, then zbottom is optional. The recommended usage is ztop = 0 to establish a reference plane. Most users use the bottom of the shallow trench layer as their base reference plane. Typically, the base has a negative zbottom, with ztop at 0. If you are using relative ordering for calculating the z-coordinates of layers, the base definition should appear first.

The following minimum parameters are required for the single base layer definition:

```
base = <layer_name> {
   thickness = <layer thickness>
   ztop = 0
   zbottom = <bottom z-coordinate, typically a negative value>
}
```

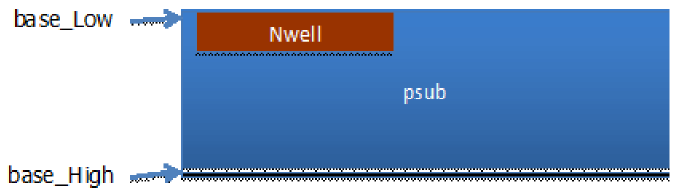
There can be only one base plane per process definition for parallel stacks.

Multiple Base Layers

Use multiple base layers in a technology stack where each base layer definition represents a different possible ground plane height. Use this in designs that integrate front-end RF devices such as power amplifiers, RF switches, and MEMs in conventional CMOS technologies where both high resistivity (low mobility) and low resistivity substrate areas exist within the same wafer (multiple substrates). Base layers must be placed below all of the conductor layers. Calibration produces a set of rules for each base layer as an enclosure layer.

Figure 4-1 describes how more than one substrate can exist inside the technology stack.

Figure 4-1. Multi-level Substrate



The following is an example of a multiple base layer definition:

```
base = base_High_R {
   zbottom = -1
   thickness = 1
}
dielectric = SUBSTRATE {
   zbottom = 0
   thickness = 700
   eps = 11.9
   diel_type = planar
}
base = base_Low_R {
   zbottom = 699
   thickness = 1
}
dielectric = STI {
   zbottom = 700
   thickness = 1
   eps = 4.1
   diel_type = planar
}
```

For information on multiple base layer extraction see "Extracting With Multiple Substrates" in the *Calibre xRC User's Manual*.

Table 4-2 describes the syntax for the base keyword.

Table 4-2. Base Keyword

Base Keyword	Description
base	Required keyword used to define a base reference plane.
layer_name	Required user-specified name that identifies the layer.

Table 4-3 lists the base parameters.

Table 4-3. Base Parameters

Base Parameters	Description
thickness	Required parameter that specifies the thickness of the base plane.
ztop	Parameter that specifies the top z-coordinate of the base layer. It is required if and only if, zbottom parameter has not be specified for the layer. If you have specified zbottom you are not required to specify ztop.
zbottom	Parameter that specifies the bottom z-coordinate of the base layer. It is required if and only if, ztop parameter has not be specified. If you have specified ztop you are not required to specify zbottom.
bulk_min_width	Optional layer parameter that specifies the minimum width of the bulk layer.
bulk_resistance	Optional layer parameter that specifies the sheet resistance of the bulk layer.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Dielectric

The dielectric layer defines the electrical and physical characteristics of insulating layers between interconnect layers.

The dielectric keyword is used when specifying dielectric layers and may appear multiple times in your MIPT file. If the stack specified in your MIPT file does not contain any dielectric layers, then it may only be used for calibration of RONLY rules.

There are seven types of dielectric:

- planar
- conformal
- rsd_conformal
- trench
- mask
- spacer
- spacer_mask

Each type has specific parameters required for accurate definition of the dielectric material.

Planar Dielectrics

The following minimum parameters are required for planar dielectrics:

```
dielectric = <layer_name> {
   diel_type = planar
   thickness = <value>
   eps = <value>
}
```

In the absence of the z-bottom or z-top settings, the dielectric layer should be defined in the order immediately after the dielectric layer which is below the current layer.

Conformal, RSD_Conformal, Trench, and Spacer Dielectrics

Conformal, rsd_conformal, trench, and spacer dielectrics require some additional parameters:

```
dielectric = <layer_name> {
   diel_type = conformal
   # or diel_type can be set to rsd_conformal, trench, or spacer
   thickness = <value>
   eps = <value>
   ref_layer = <referenced_layer_name>
   swthk = <value>
   topthk = <value>
}
```

Rsd_conformal dielectrics have the same properties as conformal dielectrics except they are applied to raised source/drain (rsd) layers instead of diffusion layers.

Spacer dielectrics are conformal dielectrics used to self align contacts near the gate or other similar operations. They look and behave just like regular conformal dielectrics, but have priority over conductors.

Mask and Spacer_mask Dielectric

Mask and spacer_mask dielectrics are used only with Calibre xACT 3D. A mask dielectric specifies a local dielectric layer typically around a device. A spacer_mask dielectric specifies a painted dielectric over a conductor and is typically used when modeling slotted metal.

Dielectric Layer Syntax

The dielectric layer syntax allows the reuse of a layer within another layer stack by using the Use Directive. MIPT syntax does not allow duplicate layer names and generates an error if duplicate name is encountered during syntax verification with -check.

By default, the minimal z-coordinate or bottom of the dielectric layer, is placed at the top of the previously defined dielectric layer. For a conformal dielectric layer, the top is the planar part. For a trench dielectric, thickness is always zero so the top is always the same as the bottom.

An optional syntax zbottom is available if the dielectric layer is not defined in the specified order. You can measure from the top of a reference layer defined by relative_to instead of measured_from. If measured_from is omitted, zbottom measures from absolute 0. The zbottom parameter defines the bottom of the layer. The ztop parameter defines the top of the layer. Both measure from the top of the previously placed dielectric layer by default. Note that in the presence of process variability, the use of an absolute zbottom or minimal z-coordinate measured from an absolute height, is not a convenient way to specify the position of a layer. In this case, it is better to use either the default minimal z-coordinate or measured from the top surface of the previously defined dielectric layer, if the layers are defined in order, or combine zbottom with measured_from to define the relative layer position if the layers are not defined in order. The same is true for ztop.

It is recommended that layers are defined (stacked) in order as much as possible, without the use of absolute placement. This is to avoid stacking and measurement confusion in the z-direction.

Table 4-4 describes the syntax for the dielectric keyword.

Table 4-4. Dielectric Keyword

Dielectric Keyword	Description
dielectric	Required syntax used to define a dielectric layer.
layer_name	Required user-specified name that identifies the layer.

Table 4-5 lists the dielectric parameters.

Table 4-5. Dielectric Parameters

Dielectric Parameters	Description
diel_type	Required parameter that specifies the type of dielectric being described.
thickness	Required parameter that specifies the thickness of the dielectric layer in the z-direction.
eps	Required parameter that specifies the relative permittivity (dielectric constant).
zbottom	Optional parameter that specifies the bottom z-coordinate when the dielectric layer is not defined in the specified order; that is, the dielectric layer is not placed relative to a previous layer's placement.
ztop	Optional parameter that specifies the top z-coordinate when the dielectric layer is not implicitly placed; that is, the dielectric layer is not placed relative to a previous layer's placement.
measured_from	Optional parameter that specifies the layer whose top surface is used as the base from which the layer's bottom is measured.

Table 4-5. Dielectric Parameters (cont.)

Dielectric Parameters	Description
thickness_type	Optional parameter used with measured_from. Either relative or absolute, specifies how thickness is measured. Default is absolute.
ref_layer	Required parameter used when diel_type is set to conformal or trench.
swthk	Required parameter used when diel_type is set to conformal or trench.
topthk	Required parameter used when diel_type is set to conformal or trench.
botthk	Optional parameter used when diel_type is set to conformal or trench.
swslope	Optional parameter that specifies the angle of the side wall in degrees. Only permitted when diel_type is set to conformal or trench.
swstep	Optional parameter that specifies the number of rectangles used to model trapezoid shapes. Only permitted when diel_type is set to conformal. Used only by the Calibre xACT 3D field solver.
airgap	Optional parameter that specifies the airgap parameters. Only permitted when diel_type is set to conformal.
damage_eps	Optional parameter that specifies the damaged relative permittivity (dielectric constant), which changes the dielectric constant in a planar dielectric where the damaged portion touches the conductor laterally or vertically. Only permitted when diel_type is set to planar and must be specified with damage_thickness.
damage_thickness	Optional parameter that specifies the damaged thickness of the dielectric layer in the z-direction where it touches the conductor laterally or vertically. Only permitted when diel_type is set to planar and must be specified with damage_eps.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Conductor

The conductor keyword is a required keyword used to define electrical and physical characteristics of conducting layers, also referred to as metal layers.

You are required to specify at least one conducting layer in your MIPT file. The z-coordinates can be either relative to the previous layer or absolute compared to a reference. By default, the z-coordinate of the bottom of the conducting layer is equal to the top z-coordinate of the previously-defined dielectric layer.

For RONLY calibrations:

- Dielectrics are not allowed. Only conductors are allowed in the RONLY conductor stack.
- Parallel structures are not allowed. Specify all the conductor and via layers in the master stack.
- The z-coordinates, ztop and zbottom, are not allowed in RONLY calibrations. Explicit placement using these parameters in the RONLY conductor stack generates an error.

Table 4-6 describes the syntax for the conductor keyword.

Table 4-6. Conductor Keyword

Conductor Keyword	Description
conductor	Required syntax used to define a conductor layer.
layer_name	Required user specified name that identifies the layer.

Table 4-7 lists the conductor parameters.

Table 4-7. Conductor Parameters

Conductor Parameters	Description
thickness	Required parameter that specifies the thickness of the conductor layer in the z-direction.
min_width	Required parameter that specifies the minimum drawn width of conductor for this layer.
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between conductors on this layer.
resistivity	Required if r_sheet parameter is not defined, this parameter specifies the nominal metal resistance for this layer.
r_sheet	Required if resistivity parameter is not defined for the layer, this parameter specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
metal_fill	Optional parameter that specifies a set of values enclosed in braces used to define virtual fill parameters.

Table 4-7. Conductor Parameters (cont.)

Conductor Parameters	Description
ztop	Optional parameter that specifies the top z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
zbottom	Optional parameter that specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
measured_from	Optional parameter that specifies the layer name whose top surface is used as the base from which zbottom is measured. If measured_from is omitted, zbottom measures from absolute 0.
airgap	Optional parameter that specifies the airgap parameters. Only applied when a marker layer is present.
thickness_type	Optional parameter used with measured_from. Either relative or absolute, specifies how thickness is measured. Default is absolute.
coplanar_min_spacing	Optional keyword that specifies the minimum spacing value between coplanar layers.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
max_width	Optional parameter that specifies the maximum allowed metal width.
max_spacing	Optional parameter that specifies the maximum allowed metal spacing.
maxwidth_for_minspacing	Optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
devices	Optional parameter that specifies a set of device names enclosed in braces that the layer definition applies to. For the conductor layer type, this parameter can only be specified in the conductor layer definition that defines a metal layer associated with a device, such as metal1. Specifying the devices parameter for conductor metal layers that are not part of a device is not allowed.

Table 4-7. Conductor Parameters (cont.)

Conductor Parameters	Description
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width of this layer.
capacitive_only_etch	Optional parameter that specifies the layer bias for capacitance.
resistive_only_etch	Optional parameter that specifies the layer bias for resistance.
cond_type	Optional parameter that specifies the conductor layer is non-planar.
trap_style	Optional parameter whose value set as top, middle, or bottom specifies where the trapezoid measurements are taken from.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
swstep	Optional parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.
extension	Optional parameter that specifies the amount the conductor layer extends past the diffusion or contact layers. For diffusion layer, this specifies the amount diffusion extends past the contact layer.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
density_window	Optional parameter that specifies a list of values to be applied to any conductor that has a thickness table.
widths	Optional conducting layer parameter that specifies a space delimited list of floating point width values enclosed in braces ({ }) used to override modeled width values.
spacings	Optional conducting layer parameter that specifies a space delimited list of floating point spacing values enclosed in braces ({ }) used to override modeled spacing values.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Resistor

The resistor keyword is an optional layer definition keyword used to define a resistor.

A resistor layer has the electrical and physical characteristics of conducting layers, also referred to as metal layers. The resistor layer represents a resistor such as RMOL and has a physical location in the layer stack. The z-coordinates can be either relative to the previous layer or absolute compared to a reference. By default, the z-coordinate of the bottom of the conducting layer is equal to the top z-coordinate of the previously-defined dielectric layer.

Table 4-8 describes the syntax for the resistor keyword.

Table 4-8. Resistor Keyword

Resistor Keyword	Description
resistor	Required syntax used to define a resistor layer.
layer_name	Required user specified name that identifies the layer.

Table 4-9 lists the resistor parameters.

Table 4-9. Resistor Parameters

Resistor Parameters	Description
thickness	Required parameter that specifies the thickness of the resistor layer in the z-direction.
min_width	Required parameter that specifies the minimum drawn width of resistor for this layer.
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between resistors on this layer.
resistivity	Required if the r_sheet parameter is not defined, this parameter specifies the nominal metal resistance for this layer.
r_sheet	Required if the resistivity parameter is not defined for the layer, this parameter specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
metal_fill	Optional parameter that specifies a set of values enclosed in braces used to define virtual fill parameters.
ztop	Optional parameter that specifies the top z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer’s placement.
zbottom	Optional parameter that specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer’s placement.
measured_from	Optional parameter that specifies the layer name whose top surface is used as the base from which zbottom is measured. If measured_from is omitted, zbottom measures from absolute 0.
airgap	Optional parameter that specifies the airgap parameters. Only applied when a marker layer is present.

Table 4-9. Resistor Parameters (cont.)

Resistor Parameters	Description
thickness_type	Optional value used with measured_from. Either relative or absolute, specifies how thickness is measured. Default is absolute.
coplanar_min_spacing	Optional parameter that specifies the minimum spacing value between coplanar layers.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
max_width	Optional parameter that specifies the maximum allowed metal width.
max_spacing	Optional parameter that specifies the maximum allowed metal spacing.
devices	Optional parameter that specifies a set of values enclosed in braces used to specify the devices the layer definition applies to.
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width of this layer.
trap_style	Optional parameter value set as top, middle, or bottom that specifies where the trapezoid measurements are taken from.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
swstep	Optional parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.
extension	Optional parameter that specifies the amount the resistor layer extends past the diffusion or contact layers. For diffusion layer, this specifies the amount diffusion extends past the contact layer.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
density_window	Optional parameter that specifies a list of values to be applied to any resistor that has a thickness table.

Table 4-9. Resistor Parameters (cont.)

Resistor Parameters	Description
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Ground

The optional ground keyword is used for extraction with field solvers to define electrical and physical characteristics of conducting layers, also referred to as metal layers. It has the same properties as a conductor layer with the following exceptions:

- It cannot be connected to a via.
- It may touch or be coplanar with other conductor layers in the interconnect stack, usually the diffusion layer.
- It may have conformal dielectrics associated with it.
- It cannot have metal_fill, extension, or raised OD parameters.
- It is only used by Calibre xACT 3D.

Table 4-10 describes the syntax for the ground keyword.

Table 4-10. Ground Keyword

Ground Keyword	Description
ground	Required syntax used to define a ground layer.
layer_name	Required user specified name that identifies the layer.

Table 4-11 lists the ground parameters.

Table 4-11. Ground Parameters

Ground Parameters	Description
thickness	Required parameter that specifies the thickness of the ground layer in the z-direction.
min_width	Required parameter that specifies the minimum drawn width of ground for this layer.
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between conductors on this layer.
resistivity	Required if r_sheet parameter is not defined, this parameter specifies the nominal metal resistance for this layer.

Table 4-11. Ground Parameters (cont.)

Ground Parameters	Description
r_sheet	Required if resistivity parameter is not defined for the layer, this parameter specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
ztop	Optional parameter that specifies the top z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
zbottom	Optional parameter that specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
measured_from	Optional parameter that specifies the layer name whose top surface is used as the base from which zbottom is measured. If measured_from is omitted, zbottom measures from absolute 0.
thickness_type	Optional parameter used with measured_from. Either relative or absolute, specifies how thickness is measured. Default is absolute.
coplanar_min_spacing	Optional parameter that specifies the minimum spacing value between coplanar layers.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
max_width	Optional parameter that specifies the maximum allowed metal width.
max_spacing	Optional parameter that specifies the maximum allowed metal spacing.
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width of this layer.
trap_style	Optional parameter set as top, middle, or bottom that specifies where the trapezoid measurements are taken from.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
swstep	Optional parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.

Table 4-11. Ground Parameters (cont.)

Ground Parameters	Description
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
virtual	Optional parameter that specifies whether or not to treat the ground layer as virtual.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
density_window	Optional parameter that specifies a list of values to be applied to any conductor that has a thickness table.
copy	Optional parameter that specifies a list of layer names to be mapped to ground layers under the gate.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Derived

The optional derived keyword defines the electrical characteristics of conducting layers. This layer type cannot alter any physical parameters of a conducting layer and may be specified more than once.

It is possible to describe derived layers within a parallel structure definition. These layers have no physical location in the main stack, and are used to modify the electrical characteristics of specific regions being mapped to PEX layers from the layout. Derived layers are optional within the parallel structure and may be one of the following:

- src_drn
- src_drn_contact
- seed
- ronly
- ronly_contact
- ronly_via

There is no requirement that a derived layer needs to be defined before its first use in the MIPT file. As with other layer types, derived layers may benefit from the **use** and **use with** keywords. For more information on the **use** keywords, see Use Directive.

Table 4-12 describes the syntax for the derived keyword.

Table 4-12. Derived Keyword

Derived Keyword	Description
derived	Required syntax used to define a derived layer.
layer_name	Required user specified name that identifies the layer.

Table 4-13 lists the derived parameters.

Table 4-13. Derived Parameters

Derived Parameters	Description
derived_type	Required parameter that specifies the type of derived layer being specified. Must be src_drn, src_drn_contact, seed, ronly, ronly_contact, or ronly_via.
resistance	Required if derived_type is src_drn_contact, ronly_contact, or ronly_via. Values specifies the resistance of the contact/via in ohms.
resistivity	Required if r_sheet parameter is not defined, this parameter specifies the nominal metal resistance for this layer.
r_sheet	Required if resistivity parameter is not defined for the layer, this parameter specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
maxwidth_for_minspacing	Optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for the given layer is ignored during extraction. May only be specified if derived_type is ronly or ronly_contact.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Poly

The poly keyword is an optional keyword used to define electrical and physical characteristics of conducting layers, also referred to as metal layers.

The poly keyword uses the same required and optional properties as conductor and may be specified more than once. The poly layer type has an additional optional layer property called seed_layers. Do not use this conducting layer keyword poly for RONLY calibrations.

The z-coordinates can be either relative to the previous layer or absolute compared to a reference. By default, the z-coordinate of the bottom of the conducting layer is equal to the top z-coordinate of the previously defined dielectric layer.

Table 4-14 describes the syntax for the poly keyword.

Table 4-14. Poly Keyword

Poly Keyword	Description
poly	Optional syntax used to define a poly layer.
layer_name	Required user specified name that identifies the layer.

Table 4-15 lists the poly parameters.

Table 4-15. Poly Parameters

Poly Parameters	Description
thickness	Required parameter that specifies the thickness of the poly layer in the z-direction.
min_width	Required parameter that specifies the minimum drawn width of poly for this layer.
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between poly for this layer.
resistivity	Required if r_sheet parameter is not defined, this parameter specifies the nominal metal resistance for this layer.
r_sheet	Required if resistivity parameter is not defined for the layer, this parameter specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
metal_fill	Optional parameter that specifies a set of values enclosed in braces used to define virtual fill parameters.
thickness_type	Optional parameter set to either relative or absolute, which specifies how thickness is measured.

Table 4-15. Poly Parameters (cont.)

Poly Parameters	Description
ztop	Optional parameter that specifies the top z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
zbottom	Optional parameter that specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
measured_from	Optional parameter that specifies the layer name whose top surface is used as the base from which zbottom is measured. If measured_from is omitted, zbottom measures from absolute 0.
airgap	Optional parameter that specifies the airgap parameters. Only applied when a marker layer is present.
coplanar_min_spacing	Optional parameter that specifies the minimum spacing value between coplanar layers.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
max_width	Optional parameter that specifies the maximum allowed metal width.
max_spacing	Optional parameter that specifies the maximum allowed metal spacing.
maxwidth_for_minspacing	Optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
min_actual_contact_length	Optional parameter that specifies to override the default minimum actual length of a diffusion contact for a specific device.
min_actual_contact_width	Optional parameter that specifies to override the default minimum actual width of a diffusion contact for a specific device.
devices	Optional parameter that specifies a set of values enclosed in braces that specify the devices the layer definition applies to.
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width of this layer.

Table 4-15. Poly Parameters (cont.)

Poly Parameters	Description
capacitive_only_etch	Optional parameter that specifies the layer bias for capacitance.
resistive_only_etch	Optional parameter that specifies the layer bias for resistance.
trap_style	Optional parameter set as top, middle, or bottom that specifies where the trapezoid measurements are taken from.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
swstep	Optional parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.
extension	Optional parameter that specifies the amount the conductor layer extends past the diffusion or contact layers. For diffusion layer, this specifies the amount diffusion extends past the contact layer.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
density_window	Optional parameter that specifies a list of values to be applied to any conductor that has a thickness table.
widths	Optional conducting layer parameter that specifies a space delimited list of floating point width values enclosed in braces ({ }) used to override modeled width values.
spacings	Optional conducting layer parameter that specifies a space delimited list of floating point spacing values enclosed in braces ({ }) used to override modeled spacing values.
src_drn_layers	Optional parameter that specifies a space delimited list of src_drn layer names.
seed_layers	Optional parameter that specifies a list of seed layer names.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

PCaux

The pcaux keyword is an optional keyword used to define a floating (non-gate) poly layer abutting the diffusion in between or at the end of a device. PCaux layers are dummy layers treated as field poly and have a physical location in the layer stack.

Table 4-16 describes the syntax for the pcaux keyword.

Table 4-16. PCaux Keyword

Pcaux Keyword	Description
pcaux	Optional syntax used to define a pcaux layer.
layer_name	Required user specified name that identifies the layer.

Table 4-17 lists the pcaux parameters.

Table 4-17. PCaux Parameters

Pcaux Parameters	Description
thickness	Required parameter that specifies the thickness of the pcaux layer in the z-direction.
min_width	Required parameter that specifies the minimum drawn width of pcaux for this layer.
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between pcaux layers for this layer.
resistivity	Optional if r_sheet parameter is not defined, this parameter specifies the nominal metal resistance for this layer.
r_sheet	Optional if resistivity parameter is not defined for the layer, this parameter specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
metal_fill	Optional parameter that specifies a set of values enclosed in braces used to define virtual fill parameters.
thickness_type	Optional parameter, either relative or absolute, that specifies how thickness is measured.
ztop	Optional parameter that specifies the top z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer’s placement.
zbottom	Optional parameter that specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer’s placement.
measured_from	Optional parameter that specifies the layer name whose top surface is used as the base from which zbottom is measured. If measured_from is omitted, zbottom measures from absolute 0.
airgap	Optional parameter that specifies the airgap parameters. Only applied when a marker layer is present.
coplanar_min_spacing	Optional parameter that specifies the minimum spacing value between coplanar layers.

Table 4-17. PCaux Parameters (cont.)

Pcaux Parameters	Description
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
max_width	Optional parameter that specifies the maximum allowed metal width.
max_spacing	Optional parameter that specifies the maximum allowed metal spacing.
maxwidth_for_minspacing	Optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
multigate parameter	Optional parameter that specifies which default multigate model for FinFETs should be used during calibration.
devices	Optional parameter that specifies a set of values enclosed in braces used to specify the devices the layer definition applies to.
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width of this layer.
contact_width2	Optional parameter that specifies the ct2 bias width for a seed layer contact.
trap_style	Optional parameter, set as top, middle, or bottom that specifies where the trapezoid measurements are taken from.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
swstep	Optional parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.
extension	Optional parameter that specifies the amount the conductor layer extends past the diffusion or contact layers. For diffusion layer, this specifies the amount diffusion extends past the contact layer.
gate_to_cont_min_spacing	Optional parameter that specifies the minimum spacing between the gate poly layer and the diffusion contact layer. See the contact from diffusion to metal1 in Figure 4-7 in "Seed" for details.

Table 4-17. PCaux Parameters (cont.)

Pcaux Parameters	Description
gate_to_LI1_min_spacing	Optional parameter that specifies the minimum spacing between the gate poly layer (PC) and the li layer touching the diffusion layer (TS). See Figure 4-7 in "Seed" for details.
gate_to_LI2_min_spacing	Optional parameter that specifies the minimum spacing between the gate poly layer (PC) and the second li layer (CA). See Figure 4-7 in "Seed" for details.
gate_to_via_min_spacing	Optional parameter that specifies the minimum spacing between the gate poly layer and the diffusion via layer. See the via from diffusion (Rx) to li1 layer (TS) in Figure 4-7 in "Seed" for details.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
ignore_diff_to_diff_under_poly	Optional parameter that controls whether or not diffusion to diffusion under poly capacitance is ignored by calibration.
ignore_endcap	Optional parameter that controls whether or not poly extension to source and drain coupling is ignored by calibration.
ignore_gate_to_diff	Optional parameter that controls whether or not the coupling capacitance between gates to diffusion is ignored by calibration.
ignore_gateext_to_diff	Optional parameter that controls whether or not the coupling capacitance between the gate extension to diffusion and raised source/drain is ignored by calibration.
ignore_gateext_to_diff_only	Optional parameter that controls whether or not the coupling capacitance between the gate extension to diffusion is ignored by calibration.
rsd_enclosure	Optional parameter that specifies raised source/drain edge bias. See the measurement labeled "Raised OD edge biasing" in Figure 4-2 in "Diffusion" for details.
rsd_thickness	Optional parameter that specifies raised source/drain height. See the measurement labeled "Raised Height" in Figure 4-2 in "Diffusion" for details.
rsd_spacing	Optional parameter that specifies raised source/drain gate spacing. See the measurement labeled "Raised OD to Gate Poly Spacing" in Figure 4-2 in "Diffusion" for details.
rsd_swslope	Optional parameter that specifies the raised source/drain swslope. This parameter must be specified with rsd_thickness and rsd_spacing parameters.

Table 4-17. PCaux Parameters (cont.)

Pcaux Parameters	Description
ronly_layers	Optional parameter that specifies a list of r-only layer names.
density_window	Optional parameter that specifies a list of values to be applied to any conductor that has a thickness table.
src_drn_layers	Optional parameter that specifies a space delimited list of src_drn layer names.
seed_layers	Optional parameter that specifies a list of seed layer names.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Diffusion

The diffusion keyword is an optional keyword used to define electrical and physical characteristics of conducting layers, also referred to as metal layers.

The diffusion keyword uses the same required and optional properties as conductor and may be specified more than once. Diffusion layer type has an additional optional layer property called src_drn_layers. Do not use the conducting layer keyword diffusion for RONLY calibrations.

The z-coordinates can be either relative to the previous layer or absolute compared to a reference. By default, the z-coordinate of the bottom of the conducting layer is equal to the top z-coordinate of the previously-defined dielectric layer.

Table 4-18 describes the syntax for the diffusion keyword.

Table 4-18. Diffusion Keyword

Diffusion Keyword	Description
diffusion	Optional syntax used to define a diffusion layer.
layer_name	Required user specified name that identifies the layer.

Table 4-19 lists the diffusion parameters.

Table 4-19. Diffusion Parameters

Diffusion Parameters	Description
thickness	Required parameter that specifies the thickness of the diffusion layer in the z-direction.
min_width	Required parameter that specifies the minimum drawn width of diffusion for this layer.

Table 4-19. Diffusion Parameters (cont.)

Diffusion Parameters	Description
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between metal for this layer.
resistivity	Required if r_sheet parameter is not defined, this parameter specifies the nominal metal resistance for this layer.
r_sheet	Required if resistivity parameter is not defined for the layer, this parameter specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
metal_fill	Optional parameter that specifies a set of values enclosed in braces used to define virtual fill parameters.
thickness_type	Optional parameter, either relative or absolute, that specifies how thickness is measured.
ztop	Optional parameter that specifies the top z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer’s placement.
zbottom	Optional parameter that specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer’s placement.
measured_from	Optional parameter that specifies the layer name whose top surface is used as the base from which zbottom is measured. If measured_from is omitted, zbottom measures from absolute 0.
airgap	Optional parameter that specifies the airgap parameters. Only applied when a marker layer is present.
coplanar_min_spacing	Optional parameter that specifies the minimum spacing value between coplanar layers.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
max_width	Optional parameter that specifies the maximum allowed metal width.
max_spacing	Optional parameter that specifies the maximum allowed metal spacing.

Table 4-19. Diffusion Parameters (cont.)

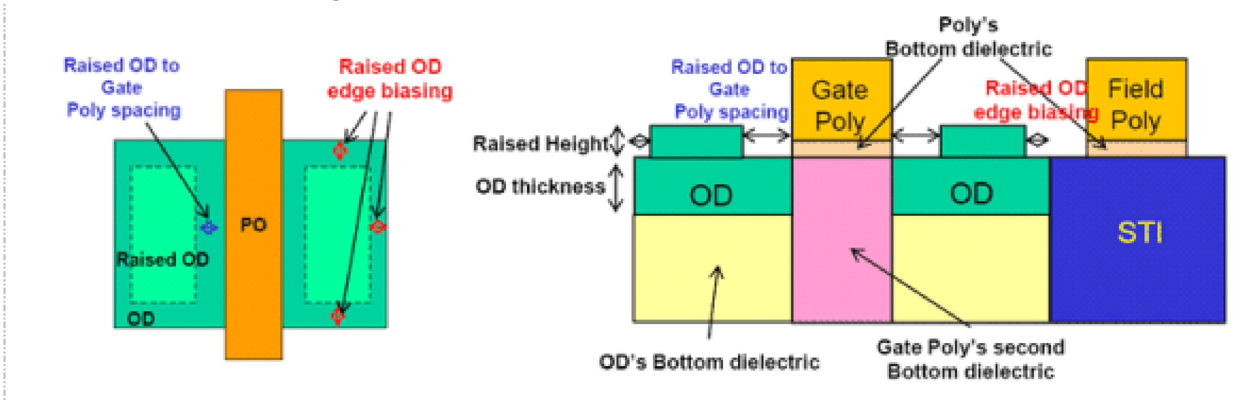
Diffusion Parameters	Description
maxwidth_for_minspacing	Optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
devices	Optional parameter that specifies a set of values enclosed in braces used to specify the devices the layer definition applies to.
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width of this layer.
trap_style	Optional parameter set as top, middle, or bottom, that specifies where the trapezoid measurements are taken from.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
swstep	Optional parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.
extension	Optional parameter that specifies the amount the conductor layer extends past the diffusion or contact layers. For diffusion layer, this specifies the amount diffusion extends past the contact layer.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
density_window	Optional parameter that specifies a list of values to be applied to any conductor that has a thickness table.
src_drn_layers	Optional parameter that specifies a space delimited list of src_drn layer names.
rsd_enclosure	Optional parameter that specifies raised source/drain edge bias. See the measurement labeled “Raised OD edge biasing” in Figure 4-2 for details.
rsd_thickness	Optional parameter that specifies raised source/drain height. See the measurement labeled “Raised Height” in Figure 4-2 for details.
rsd_spacing	Optional parameter that specifies raised source/drain gate spacing. See the measurement labeled “Raised OD to Gate Poly Spacing” in Figure 4-2 for details.
rsd_swslope	Optional parameter that specifies the raised source/drain swslope. This parameter must be specified with rsd_thickness and rsd_spacing parameters.

Table 4-19. Diffusion Parameters (cont.)

Diffusion Parameters	Description
widths	Optional conducting layer parameter that specifies a space delimited list of floating point width values enclosed in braces ({ }) used to override modeled width values.
spacings	Optional conducting layer parameter that specifies a space delimited list of floating point spacing values enclosed in braces ({ }) used to override modeled spacing values.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

The diagram in Figure 4-2 describes how the raised source/drain parameters for diffusion layers are measured.

Figure 4-2. Raised Source/Drain Parameters



li

The li keyword is an optional keyword used to define electrical and physical characteristics of conducting layers, also referred to as metal layers or local interconnect. The li keyword uses the same required and optional properties as conductor and may be specified more than once.

The z-coordinates can be either relative to the previous layer or absolute compared to a reference. By default, the z-coordinate of the bottom of the conducting layer is equal to the top z-coordinate of the previously-defined dielectric layer.

Table 4-20 describes the syntax for li layer keywords.

Table 4-20. li Keyword

li Keyword	Description
li	Required syntax used to define a local interconnect (li) layer.
layer_name	Required user specified name that identifies the layer.

Table 4-21 lists the li parameters.

Table 4-21. li Parameters

li Parameters	Description
thickness	Required parameter that specifies the thickness of the li layer in the z-direction.
min_width	Required parameter that specifies the minimum drawn width of li for this layer.
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between li on this layer.
resistivity	Required if r_sheet parameter is not defined, this parameter specifies the nominal metal resistance for this layer.
r_sheet	Required if resistivity parameter is not defined for the layer, this parameter specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
metal_fill	Optional parameter that specifies a set of values enclosed in braces used to define virtual fill parameters.
thickness_type	Optional parameter, either relative or absolute, that specifies how thickness is measured.
ztop	Optional parameter that specifies the top z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
zbottom	Optional parameter that specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
measured_from	Optional parameter that specifies the layer name whose top surface is used as the base from which zbottom is measured. If measured_from is omitted, zbottom measures from absolute 0.
airgap	Optional parameter that specifies the airgap parameters. Only applied when a marker layer is present.
coplanar_min_spacing	Optional parameter that specifies the minimum spacing value between coplanar layers.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.

Table 4-21. li Parameters (cont.)

li Parameters	Description
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
max_width	Optional parameter that specifies the maximum allowed metal width.
max_spacing	Optional parameter that specifies the maximum allowed metal spacing.
maxwidth_for_minspacing	Optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
devices	Optional parameter that specifies a set of values enclosed in braces used to specify the devices the layer definition applies to.
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width of this layer.
trap_style	Optional parameter set as top, middle, or bottom, that specifies where the trapezoid measurements are taken from.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
swstep	Optional parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.
extension	Optional parameter that specifies the amount the conductor layer extends past the diffusion or contact layers. For diffusion layer, this specifies the amount diffusion extends past the contact layer.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
device_li_layers	Optional parameter that specifies a space delimited list of device_li layer names identifying the device LI layers associated with it.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
density_window	Optional parameter that specifies a list of values to be applied to any conductor that has a thickness table.
src_drn_layers	Optional parameter that specifies a space delimited list of src_drn layer names.

Table 4-21. li Parameters (cont.)

li Parameters	Description
widths	Optional conducting layer parameter that specifies a space delimited list of floating point width values enclosed in braces ({ }) used to override modeled width values.
spacings	Optional conducting layer parameter that specifies a space delimited list of floating point spacing values enclosed in braces ({ }) used to override modeled spacing values.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Device_li

The device_li keyword is used to define the device region of the li layer. The device_li keyword uses the same required and optional properties as li and may be specified more than once.

The z-coordinates can be either relative to the previous layer or absolute compared to a reference. By default, the z-coordinate of the bottom of the conducting layer is equal to the top z-coordinate of the previously-defined dielectric layer.

Table 4-22 describes the syntax for device_li keywords.

Table 4-22. Device_li Keyword

Device_li Keyword	Description
device_li	Required syntax used to define a device local interconnect (li) layer.
layer_name	Required user specified name that identifies the layer.

Table 4-23 lists the device_li parameters.

Table 4-23. Device_li Parameters

Device_li Parameters	Description
thickness	Required parameter that specifies the thickness of the device_li layer in the z-direction.
min_width	Required parameter that specifies the minimum drawn width of device_li for this layer.
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between device_li on this layer.
resistivity	Required if r_sheet parameter is not defined, this parameter specifies the nominal metal resistance for this layer.

Table 4-23. Device_li Parameters (cont.)

Device_li Parameters	Description
r_sheet	Required if resistivity parameter is not defined for the layer, this parameter specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
metal_fill	Optional parameter that specifies a set of values enclosed in braces used to define virtual fill parameters.
thickness_type	Optional parameter, either relative or absolute, that specifies how thickness is measured.
ztop	Optional parameter that specifies the top z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
zbottom	Optional parameter that specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
measured_from	Optional parameter that specifies the layer name whose top surface is used as the base from which zbottom is measured. If measured_from is omitted, zbottom measures from absolute 0.
airgap	Optional parameter that specifies the airgap parameters. Only applied when a marker layer is present.
coplanar_min_spacing	Optional parameter that specifies the minimum spacing value between coplanar layers.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
max_width	Optional parameter that specifies the maximum allowed metal width.
max_spacing	Optional parameter that specifies the maximum allowed metal spacing.
maxwidth_for_minspacing	Optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
devices	Optional parameter that specifies a set of values enclosed in braces used to specify the devices the layer definition applies to.

Table 4-23. Device_li Parameters (cont.)

Device_li Parameters	Description
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width of this layer.
capacitive_only_etch	Optional parameter that specifies the layer bias for capacitance.
resistive_only_etch	Optional parameter that specifies the layer bias for resistance.
trap_style	Optional parameter set as top, middle, or bottom, that specifies where the trapezoid measurements are taken from.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
swstep	Optional parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.
extension	Optional parameter that specifies the amount the conductor layer extends past the diffusion or contact layers. For diffusion layer, this specifies the amount diffusion extends past the contact layer.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
density_window	Optional parameter that specifies a list of values to be applied to any conductor that has a thickness table.
src_drn_layers	Optional parameter that specifies a space delimited list of src_drn layer names. .
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Contact

The contact keyword is an optional keyword used to define electrical and physical characteristics of contact layers. Contact layers are optional.

Table 4-24 describes the contact keyword.

Table 4-24. Contact Keyword

Contact Keyword	Description
contact	Required syntax used to define a contact layer.
layer_name	Required user specified name that identifies the layer.

Table 4-25 lists the contact parameters.

Table 4-25. Contact Parameters

Contact Parameters	Description
measured_from	Required parameter that specifies lower conductor layer_name.
measured_to	Required parameter that specifies upper conductor layer_name.
area	Required if min_width is not specified for the contact. Specifies the area of the contact in distance_units squared.
min_width	Required if area is not specified for the contact. Specifies the minimum drawn width of the contact. Shape is assumed to be square.
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between li on this layer.
resistance	Required parameter that specifies the resistance of the contact in ohms.
enclosure_down	Required parameter that specifies enclosure with respect to a lower conductor.
enclosure_up	Required parameter that specifies enclosure with respect to an upper conductor.
min_length	Optional parameter that specifies the length of the shape.
bottom_thickness	Optional parameter that specifies the lower section of the diffusion contact.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
gate_to_cont_spacing_min	Optional parameter that specifies the minimum allowed gate to contact spacing.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
extrapolation	Optional parameter that specifies whether or not to extrapolate the variable via_resistance table boundary values.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.

Table 4-25. Contact Parameters (cont.)

Contact Parameters	Description
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width and length of the contact.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
src_drn_contact_layers	Optional parameter that specifies the layer or layers that make up the source/drain contact region of devices.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Via

The via layer syntax defines the electrical and physical characteristics of the connections between layers. Via layers are optional.

Table 4-26 describes the via keyword.

Table 4-26. Via Keyword

Via Keyword	Description
via	Required syntax used to define a via layer.
layer_name	Required user specified name that identifies the layer.

Table 4-27 lists the via parameters.

Table 4-27. Via Parameters

Via Parameters	Description
measured_from	Required parameter that specifies lower conductor layer_name.
measured_to	Required parameter that specifies upper conductor layer_name.
area	Required if min_width is not specified for the via. Specifies the area of the via in distance_units squared.
min_width	Required if area is not specified for the via. Specifies the minimum drawn width of the via. Shape is assumed to be square.
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between li on this layer.

Table 4-27. Via Parameters (cont.)

Via Parameters	Description
resistance	Required parameter that specifies the resistance of the via in ohms.
enclosure_down	Optional parameter that specifies enclosure with respect to a lower conductor.
enclosure_up	Optional parameter that specifies enclosure with respect to an upper conductor.
max_area	Optional parameter that specifies the area threshold for large area via connections.
max_length	Optional parameter that specifies the maximum distance between distributed vias.
min_length	Optional parameter that specifies the length of the shape.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
gate_to_via_spacing_min	Optional parameter that specifies the minimum allowed gate to via spacing.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
extrapolation	Optional parameter that specifies whether or not to extrapolate the variable via_resistance table boundary values.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width and length of the via.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
parallel_to_gate	Optional layer parameter that specifies how the via width and length are measured.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.
trim	Optional layer parameter that controls how resistance is calculated for the via.

Table 4-27. Via Parameters (cont.)

Via Parameters	Description
via_r_select_up_down	Optional layer parameter that controls whether to use the smaller or larger resistance values from the upper and lower via layer enclosure.
interpolation	An optional parameter that specifies how the area-based tables interpolate resistance values for fixed or variable sized vias.

Base_via

The base_via layer syntax defines the properties of vias between the base layer and diffusion layers. Use base_via layers to include the tap resistance from the base layer to well or substrate layers in extraction. Calibre xACT and Calibre xACT 3D extract capacitance for this layer type. This layer type does not affect capacitance calculations in Calibre xRC.

Table 4-28 describes the base_via keyword.

Table 4-28. Base_via Keyword

Base_via Keyword	Description
base_via	Required syntax used to define a base_via layer.
layer_name	Required user specified name that identifies the layer.

Table 4-29 lists the base_via parameters.

Table 4-29. Base_via Parameters

Base_via Parameters	Description
measured_from	Required parameter that specifies the base layer_name.
measured_to	Required parameter that specifies the diffusion layer_name.
area	Required if min_width is not specified for the via. Specifies the area of the via in distance_units squared.
min_width	Required if area is not specified for the via. Specifies the minimum drawn width of the via. Shape is assumed to be square.
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between li on this layer.
resistance	Required parameter that specifies the resistance of the via in ohms.
enclosure_down	Optional parameter that specifies enclosure with respect to a lower conductor.

Table 4-29. Base_via Parameters (cont.)

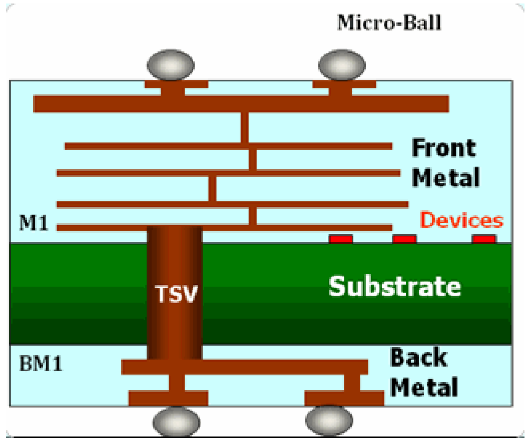
Base_via Parameters	Description
enclosure_up	Optional parameter that specifies enclosure with respect to an upper conductor.
max_area	Optional parameter that specifies the area threshold for large area via connections.
max_length	Optional parameter that specifies the maximum distance between distributed vias.
min_length	Optional parameter that specifies the length of the shape.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
extrapolation	Optional parameter that specifies whether or not to extrapolate the variable via_resistance table boundary values.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.
trim	Optional layer parameter that controls how resistance is calculated for the via.
interpolation	An optional parameter that specifies how the area-based tables interpolate resistance values for fixed or variable sized vias.

TSV

The optional TSV keyword defines the electrical and physical characteristics of through silicon vias.

Figure 4-3 shows a 3D-IC with front metal (M1 - M5), back metal (BM1 - BM2) in addition to a TSV going through the substrate.

Figure 4-3. TSV



Typically micro-balls are used to make electrical connections from one 3D-IC to another die. The TSV model must have two external pins. Pin1 is measured from the start layer and Pin2 measured to the stop layer. Multiple TSV layers may be defined.

All TSV processes have insulators surrounding the TSV as shown in Figure 4-4. You should describe the insulators using either a conformal dielectric layer or with the TSV layer insulator parameter.

Figure 4-4. TSV Insulator

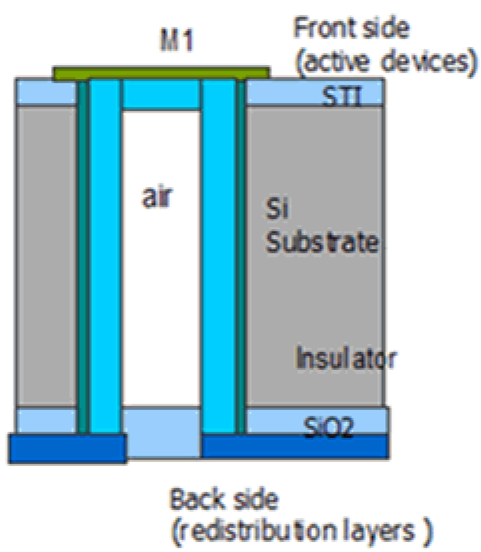


Table 4-30 describes the syntax for the TSV keyword.

Table 4-30. TSV Keyword

TSV Keyword	Description
TSV	Required syntax used to define a TSV layer.
layer_name	Required user specified name that identifies the layer.

Table 4-31 lists the TSV parameters.

Table 4-31. TSV Parameters

TSV Parameters	Description
measured_from	Required parameter that specifies the back side metal layer that the TSV connects to.
measured_to	Required parameter that specifies the front metal layer that the TSV connects to.
radius	Required parameter that specifies the radius of the TSV.
hollow_radius	Required parameter that specifies the radius of the hollow area of the TSV.
height	Required parameter that specifies the height of the TSV.
resistivity	Required parameter that specifies the resistivity of the TSV material.
depletion_width	Required parameter that specifies the depletion region width.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
top_enclosure	Optional parameter that specifies the top enclosure covered by the metal “to_layer”.
bot_enclosure	Optional parameter that specifies the bottom enclosure covered by the metal “from_layer”.
extra_width	Optional parameter that specifies the extra-width of side walls for the tapered style TSV.
insulator	Optional parameter that specifies one or more insulators for the TSVs in the process. Each insulator is described by a thickness and dielectric constant pair.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

uBump

The optional ubump keyword defines the connection between the back-side metal and a package.

Figure 4-5 shows a uBump.

Figure 4-5. uBump

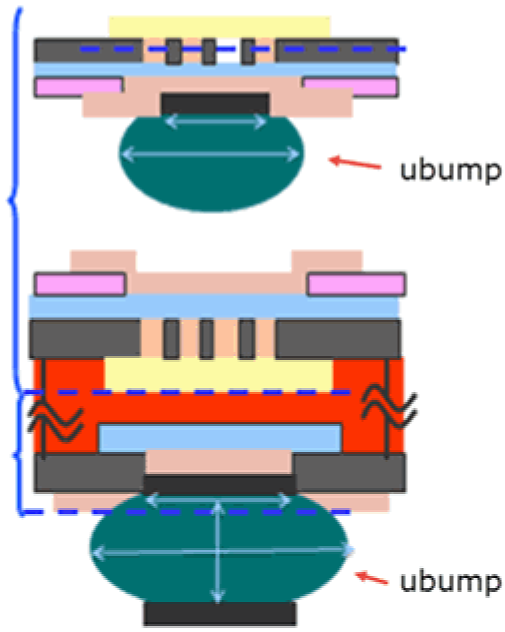


Table 4-32 describes the syntax for the uBump keyword.

Table 4-32. uBump Keyword

uBump Keyword	Description
ubump	Required syntax used to define a uBump layer.
layer_name	Required user specified name that identifies the layer.

Table 4-33 lists the uBump parameters.

Table 4-33. uBump Parameters

uBump Parameters	Description
measured_from	Required parameter that specifies the bottom layer that the uBump connects to.
measured_to	Required parameter that specifies the top layer that the uBump connects to.
min_width	Required parameter that specifies the minimum allowed metal width of the ubump.
mid_width	Required parameter that specifies the middle metal width of the ubump.
enclosure_down	Required parameter that specifies enclosure with respect to a lower conductor.

Table 4-33. uBump Parameters (cont.)

uBump Parameters	Description
enclosure_up	Required parameter that specifies enclosure with respect to an upper conductor.
resistivity	Required parameter that specifies the resistivity of the ubump material.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Pad

The pad layer is an optional layer type used to define electrical and physical characteristics of conducting layers that have no routing, and are square or round in shape.

Table 4-34 describes the syntax for the pad keyword.

Table 4-34. Pad Keyword

Pad Keyword	Description
pad	Required syntax used to define a conductor layer that has no routing.
layer_name	Required user specified name that identifies the layer.

Table 4-35 lists the pad parameters.

Table 4-35. Pad Parameters

Pad Parameters	Description
thickness	Required parameter that specifies the thickness of the conductor layer in the z-direction.
min_width	Required parameter that specifies the minimum drawn width of conductor for this layer.
min_spacing	Required parameter that specifies the minimum allowed drawn spacing between conductors on this layer.
resistivity	Required if r_sheet parameter is not defined, this parameter specifies the nominal metal resistance for this layer.
r_sheet	Required if resistivity parameter is not defined for the layer, this parameter specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
metal_fill	Optional parameter that specifies a set of values enclosed in braces used to define virtual fill parameters.

Table 4-35. Pad Parameters (cont.)

Pad Parameters	Description
ztop	Optional parameter that specifies the top z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
zbottom	Optional parameter that specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
measured_from	Optional parameter that specifies the layer name whose top surface is used as the base from which zbottom is measured. If measured_from is omitted, zbottom measures from absolute 0.
thickness_type	Optional parameter used with measured_from. Either relative or absolute, specifies how thickness is measured. Default is absolute.
coplanar_min_spacing	Optional keyword that specifies the minimum spacing value between coplanar layers.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
max_width	Optional parameter that specifies the maximum allowed metal width.
max_spacing	Optional parameter that specifies the maximum allowed metal spacing.
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width of this layer.
capacitive_only_etch	Optional parameter that specifies the layer bias for capacitance.
resistive_only_etch	Optional parameter that specifies the layer bias for resistance.
trap_style	Optional parameter whose value set as top, middle, or bottom specifies where the trapezoid measurements are taken from.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

Table 4-35. Pad Parameters (cont.)

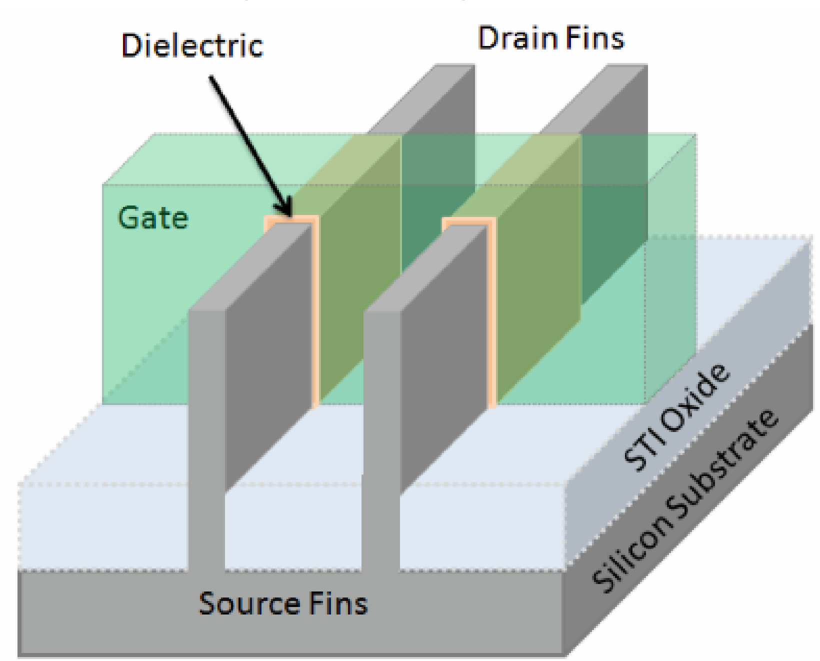
Pad Parameters	Description
extension	Optional parameter that specifies the amount the conductor layer extends past the diffusion or contact layers. For diffusion layer, this specifies the amount diffusion extends past the contact layer.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
density_window	Optional parameter that specifies a list of values to be applied to any conductor that has a thickness table.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Multigate

The multigate keyword is used to define a multigate device such as a FinFET. A multigate device is a MOSFET which incorporates more than one gate into a single device and is commonly used in microprocessor and memory cell design.

A multigate FinFET is a fin-based transistor with two or more gates built on a substrate. Figure 4-6 is an example of a multigate FinFET with two gates.

Figure 4-6. Multigate FinFET



A PEX Fin SVRF statement must be specified in your top-level extraction rule file when a multigate layer is specified in your MIPT file. Using calibrated rule files that contain a multigate layer that does not have a corresponding PEX Fin statement causes extraction to stop with an error.

Calibration adds the PEX Qualification Mode XACT statement to the calibrated rule file when a multigate layer is specified.

Table 4-36 describes the syntax for the multigate keyword.

Table 4-36. Multigate Keyword

Multigate Keyword	Description
multigate	Required syntax used to define a multigate layer.
layer_name	Required user specified name that identifies the layer.

Table 4-37 lists the multigate parameters.

Table 4-37. Multigate Parameters

Multigate Parameters	Description
fin_length	Required parameter that specifies the length of the fin.
fin_spacing	Required parameter that specifies the spacing between fins.
fin_thickness	Required parameter that specifies the thickness of the fin.
fin_width	Required parameter that specifies the width of the fin.
fin_bias	Optional parameter that specifies the biasing effect on the fin.
gate_oxide_er	Required parameter that specifies the permittivity of the gate oxide.
gate_oxide_side_t	Optional parameter that specifies the side thickness of the gate oxide.
gate_oxide_top_t	Required parameter that specifies the top thickness of the gate oxide.
gate_poly_side_t	Optional parameter that specifies the gate poly thickness between the side of the gate poly and the side of the gate oxide.
gate_poly_top_t	Required parameter that specifies the gate poly thickness between the top of the gate poly and the top of the gate oxide.
channel_er	Required parameter that specifies the difference in permittivity of the gate oxide and its lateral side.

Table 4-37. Multigate Parameters (cont.)

Multigate Parameters	Description
gate_diffusion_layer_pair	Required parameter that specifies a list of one or more layer pairs used to define which gate and diffusion layers can be paired together to describe different device types with the same device parameters.
gate_extension	Optional parameter that specifies the distance the gate layer extends outside the diffusion area.
li1_extension	Optional parameter that specifies the distance the li1 layer extends outside the diffusion area.
rsd_enclosure	Optional parameter that specifies raised source/drain edge bias. See the measurement labeled “Raised OD edge biasing” in Figure 4-2 in “Diffusion” for details.
rsd_thickness	Optional parameter that specifies raised source/drain height. See the measurement labeled “Raised Height” in Figure 4-2 in “Diffusion” for details.
rsd_spacing	Optional parameter that specifies raised source/drain gate spacing. See the measurement labeled “Raised OD to Gate Poly Spacing” in Figure 4-2 in “Diffusion” for details.
rsd_swslope	Optional parameter that specifies the raised source/drain swslope. This parameter must be specified with rsd_thickness and rsd_spacing parameters.
trench_contact_extension_length	Optional layer parameter that specifies the nominal extension of the trench contact beyond the diffusion.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

MOSFET Device Layers

Seed and src_drn layers are used to define MOSFET devices in MIPT.

Use a seed layer to create a gate from an associated poly layer and use a src_drn layer to create a source/drain from an associated diffusion layer. This gate specification triggers the use of device modeling data for calibration.

To properly define a device in MIPT you need:

- A Poly layer to use for the device. Use the seed_layers parameter to specify the seed layers that use this poly layer.
- A Diffusion layer to use for the device. Use the src_drn_layers to specify the src_drn layers that use this diffusion layer.
- One or more Seed layers that are referenced by the gate poly layer.

  The seed layer is required when defining a device and needs an associated src_drn layer.
- One or more Src_drn layers that are referenced by the gate diffusion layer.

  The src_drn layer is required when defining a device and needs an associated seed layer.
- The devices keyword specified in the poly/seed and diffusion/src_drn with the same device names.

  By default, the first seed layer definition is paired with the first src_drn layer encountered during calibration.

  Use the devices keyword to ensure that the seed and src_drn layers are paired correctly. The device names specified by the devices keyword are the names xCalibrate uses to associate the seed layer with a src_drn layer. These devices are not related to the LVS device definitions so the names do not have to be the same as the LVS device names. To complete the device definition, specify a contact layer that connects the device to metal1.
- One or more Contact layers where the diffusion layer name is specified for measured_from.

Caution

The ronly_layers parameter cannot be used to in place of the seed_layers to specify a device.

The layers used specifically for MOSFET device modeling are:

Seed

The seed keyword is an optional keyword used to define the area of the poly over the diffusion layer.

Seed layers are primary layers and have a physical location in the layer stack. The seed layer must be defined with an associated poly layer by specifying the seed_layers parameter in the poly layer definition. If a poly layer is defined without an associated seed layer the following warning is generated during calibration:

```
WARNING: No SEED layers exist in the MIPT file
```

Seed layers use the same parameters as poly layer type, except all seed layer parameters are optional, not required. The seed layer inherits parameter settings from its associated poly layer. Any parameters specified in the seed layer definition override the parameter setting specified in the parent poly layer. Do not use the conducting layer keyword seed for RONLY calibrations.

A seed layer can be placed or derived. A placed seed layer is a layer that has been specified using seed layer syntax. A derived seed layer is a layer that has been specified using derived layer syntax with derived_type parameter set to *seed*. A placed seed layer does not inherit the conformal dielectric from its parent poly layer. If a conformal dielectric layer is needed by the seed layer, then a separate conformal dielectric with the same properties as the parent poly layer must be defined. All parameters not specified are inherited from the parent poly layer.

The z-coordinates can be either relative to the previous layer or absolute compared to a reference. By default, the z-coordinate of the bottom of the conducting layer is equal to the top z-coordinate of the previously-defined dielectric layer.

Table 4-38 describes the syntax for the seed keyword.

Table 4-38. Seed Keyword

Seed Keyword	Description
seed	Optional syntax used to define a seed layer.
layer_name	Required user specified name that identifies the layer.

Table 4-39 lists the seed parameters.

Table 4-39. Seed Parameters

Seed Parameters	Description
thickness	Optional parameter that specifies the thickness of the seed layer in the z-direction.
min_width	Optional parameter that specifies the minimum drawn width of seed for this layer.

Table 4-39. Seed Parameters (cont.)

Seed Parameters	Description
min_spacing	Optional parameter that specifies the minimum allowed drawn spacing between seed layers for this layer.
resistivity	Optional if r_sheet parameter is not defined, this parameter specifies the nominal metal resistance for this layer.
r_sheet	Optional if resistivity parameter is not defined for the layer, this parameter specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
metal_fill	Optional parameter that specifies a set of values enclosed in braces used to define virtual fill parameters.
thickness_type	Optional parameter, either relative or absolute, that specifies how thickness is measured.
ztop	Optional parameter that specifies the top z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
zbottom	Optional parameter that specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement.
measured_from	Optional parameter that specifies the layer name whose top surface is used as the base from which zbottom is measured. If measured_from is omitted, zbottom measures from absolute 0.
airgap	Optional parameter that specifies the airgap parameters. Only applied when a marker layer is present.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
max_width	Optional parameter that specifies the maximum allowed metal width.
max_spacing	Optional parameter that specifies the maximum allowed metal spacing.

Table 4-39. Seed Parameters (cont.)

Seed Parameters	Description
maxwidth_for_minspacing	Optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
min_actual_contact_length	Optional parameter that specifies to override the default minimum actual length of a diffusion contact for a specific device.
min_actual_contact_width	Optional parameter that specifies to override the default minimum actual width of a diffusion contact for a specific device.
multigate parameter	Optional parameter that specifies which default multigate model for FinFETs should be used during calibration.
devices	Optional parameter that specifies a set of values enclosed in braces used to specify the devices the layer definition applies to.
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width of this layer.
contact_width2	Optional parameter that specifies the ct2 bias width for a seed layer contact.
trap_style	Optional parameter, set as top, middle, or bottom that specifies where the trapezoid measurements are taken from.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
swstep	Optional parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.
extension	Optional parameter that specifies the amount the conductor layer extends past the diffusion or contact layers. For diffusion layer, this specifies the amount diffusion extends past the contact layer.
gate_extension	Optional parameter that specifies the distance the gate layer extends outside the diffusion area.
li1_extension	Optional parameter that specifies the distance the li1 layer extends outside the diffusion area.
diel_over_gate_bottom	Optional parameter that specifies the distance (in microns) from the gate poly to the bottom of a dielectric air gap. This parameter must be specified together with diel_over_gate_width and diel_over_gate_thickness seed layer parameters.

Table 4-39. Seed Parameters (cont.)

Seed Parameters	Description
diel_over_gate_eps	Optional parameter that specifies the dielectric gap permittivity.
diel_over_gate_extrawidth	Optional parameter that specifies the extra width of the top side of a trapezoid-shaped air gap. This parameter must be specified together with diel_over_gate_bottom, diel_over_gate_width, and diel_over_gate_thickness seed layer parameters.
diel_over_gate_thickness	Optional parameter that describes the thickness (in microns) of the a dielectric air gap. This parameter must be specified together with diel_over_gate_width and diel_over_gate_bottom seed layer parameters.
diel_over_gate_width	Optional parameter that describes the width (in microns) of a dielectric air gap. This parameter must be specified together with diel_over_gate_thickness and diel_over_gate_bottom seed layer parameters.
gate_to_cont_min_spacing	Optional parameter that specifies the minimum spacing between the gate poly layer and the diffusion contact layer. See the contact from diffusion to metal1 in Figure 4-7 for details.
gate_to_LI1_min_spacing	Optional parameter that specifies the minimum spacing between the gate poly layer (PC) and the li layer touching the diffusion layer (TS). See Figure 4-7 for details.
gate_to_LI2_min_spacing	Optional parameter that specifies the minimum spacing between the gate poly layer (PC) and the second li layer (CA). See Figure 4-7 for details.
gate_to_via_min_spacing	Optional parameter that specifies the minimum spacing between the gate poly layer and the diffusion via layer. See the via from diffusion (Rx) to li1 layer (TS) in Figure 4-7 for details.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
ignore_diff_intrinsic	Optional parameter that controls whether or not capacitance from diffusion to ground is ignored by calibration.
ignore_diff_to_diff_under_poly	Optional parameter that controls whether or not diffusion to diffusion under poly capacitance is ignored by calibration.
ignore_endcap	Optional parameter that controls whether or not poly extension to source and drain coupling is ignored by calibration.

Table 4-39. Seed Parameters (cont.)

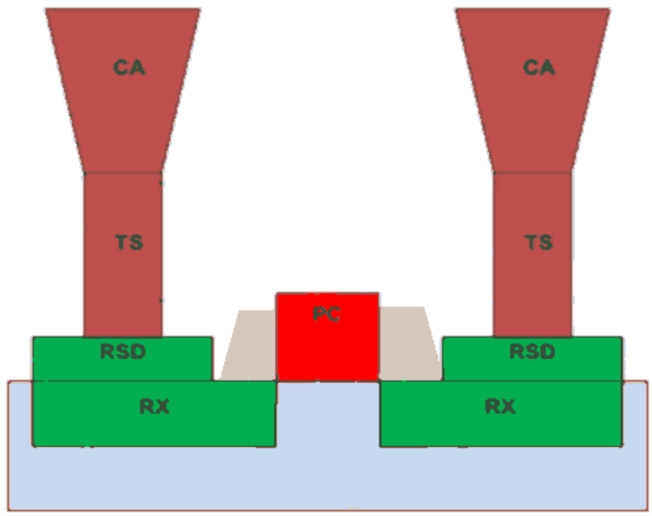
Seed Parameters	Description
ignore_gate_intrinsic	Optional parameter that controls whether or not capacitance from the gate poly layer to ground is ignored by calibration.
ignore_gate_to_diff	Optional parameter that controls whether or not the coupling capacitance between gates to diffusion is ignored by calibration.
ignore_gateext_to_diff	Optional parameter that controls whether or not the coupling capacitance between the gate extension to diffusion and raised source/drain is ignored by calibration.
ignore_gateext_to_diff_only	Optional parameter that controls whether or not the coupling capacitance between the gate extension to diffusion is ignored by calibration.
ignore_li1	Optional parameter that controls whether or not the gate to li1 and poly to li1 coupling is ignored by calibration.
ignore_rsd	Optional parameter that controls whether or not the poly to raised source/drain coupling is ignored by calibration.
ignore_rsd_intrinsic	Optional parameter that controls whether or not the capacitance between raised source/drain layer and ground is ignored by calibration.
rsd_enclosure	Optional parameter that specifies raised source/drain edge bias. See the measurement labeled “Raised OD edge biasing” in Figure 4-2 in “Diffusion” for details.
rsd_thickness	Optional parameter that specifies raised source/drain height. See the measurement labeled “Raised Height” in Figure 4-2 in “Diffusion” for details.
rsd_spacing	Optional parameter that specifies raised source/drain gate spacing. See the measurement labeled “Raised OD to Gate Poly Spacing” in Figure 4-2 in “Diffusion” for details.
rsd_swslope	Optional parameter that specifies the raised source/drain swslope. This parameter must be specified with rsd_thickness and rsd_spacing parameters.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
density_window	Optional parameter that specifies a list of values to be applied to any conductor that has a thickness table.
src_drn_layers	Optional parameter that specifies a space delimited list of src_drn layer names.
seed_layers	Optional parameter that specifies a list of seed layer names.

Table 4-39. Seed Parameters (cont.)

Seed Parameters	Description
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

The diagram in Figure 4-7 describes how the gate to contact and via minimum spacing properties are represented.

Figure 4-7. Gate to Contact/Via Min Spacing



Src_drn

The src_drn keyword is an optional keyword used to define the diffusion layer area used to form the gate.

Src_drn layers have a physical location in the layer stack. A src_drn layer must be defined with an associated diffusion layer by specifying the src_drn_layers parameter in the diffusion layer definition. If a diffusion layer is defined without an associated src_drn layer the following warning is generated during calibration:

```
WARNING: No SRC_DRN layers exist in the MIPT file
```

Src_drn uses the same parameters as diffusion layer type, except all src_drn parameters are optional, not required. The src_drn layer inherits parameter settings from its associated diffusion layer. Any parameters specified in the src_drn layer definition override the parameter setting specified in the parent diffusion layer. Do not use the conducting layer keyword src_drn for RONLY calibrations.

The z-coordinates can be either relative to the previous layer or absolute compared to a reference. By default, the z-coordinate of the bottom of the conducting layer is equal to the top z-coordinate of the previously-defined dielectric layer.

Table 4-40 describes the syntax for the src_drn keyword.

Table 4-40. Src_drn Keyword

Src_drn Keyword	Description
src_drn	Optional syntax used to define a src_drn layer.
layer_name	Required user specified name that identifies the layer.

Table 4-41 lists the src_drn parameters.

Table 4-41. Src_drn Parameters

Src_drn Parameters	Description
thickness	Optional for src_drn. Specifies the thickness of the src_drn layer in the z-direction.
min_width	Optional for src_drn. Specifies the minimum drawn width of src_drn for this layer.
min_spacing	Optional for src_drn. Specifies the minimum allowed drawn spacing between metal for this layer.
resistivity	Optional for src_drn. Specifies the nominal metal resistance for this layer.
r_sheet	Optional for src_drn. Specifies the nominal metal resistance for this layer. This value may be specified as resistivity (rho) or sheet resistance (rsh).
metal_fill	Optional parameter that specifies a set of values enclosed in braces used to define virtual fill parameters.
thickness_type	Optional parameter, either relative or absolute, that specifies how thickness is measured.
ztop	Optional parameter that specifies the top z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer’s placement.
zbottom	Optional parameter that specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer’s placement.
measured_from	Optional parameter that specifies the layer name whose top surface is used as the base from which zbottom is measured. If measured_from is omitted, zbottom measures from absolute 0.

Table 4-41. Src_drn Parameters (cont.)

Src_drn Parameters	Description
airgap	Optional parameter that specifies the airgap parameters. Only applied when a marker layer is present.
tc1	Optional parameter that specifies the first order temperature coefficient (TC1) for resistance.
tc2	Optional parameter that specifies the second order temperature coefficient (TC2) for resistance.
max_rlength	Optional parameter that specifies the maximum length of a wire when calculating how many pieces a resistor should be broken into for representing in a distributed network.
max_width	Optional parameter that specifies the maximum allowed metal width.
max_spacing	Optional parameter that specifies the maximum allowed metal spacing.
maxwidth_for_minspacing	Optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
layer_bias	Optional parameter that specifies the bias on the edge of metal objects with respect to the drawn width of this layer.
trap_style	Optional parameter set as top, middle, or bottom, that specifies where the trapezoid measurements are taken from.
devices	Optional parameter that specifies a set of values enclosed in braces used to specify the devices the layer definition applies to.
extra_width	Optional parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
swstep	Optional parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.
extension	Optional parameter that specifies the amount the conductor layer extends past the src_drn or contact layers. For src_drn layer, this specifies the amount src_drn extends past the contact layer.
ignore_caps	Optional conducting layer parameter that controls whether or not all capacitance for this layer is ignored during extraction.
ronly_layers	Optional parameter that specifies a list of r-only layer names.
density_window	Optional parameter that specifies a list of values to be applied to any conductor that has a thickness table.

Table 4-41. Src_drn Parameters (cont.)

Src_drn Parameters	Description
src_drn_layers	Optional parameter that specifies a space delimited list of src_drn layer names.
rsd_enclosure	Optional parameter that specifies raised source/drain edge bias. See the measurement labeled “Raised OD edge biasing” in Figure 4-2 in “Diffusion” for details.
rsd_thickness	Optional parameter that specifies raised source/drain height. See the measurement labeled “Raised Height” in Figure 4-2 in “Diffusion” for details.
rsd_spacing	Optional parameter that specifies raised source/drain gate spacing. See the measurement labeled “Raised OD to Gate Poly Spacing” in Figure 4-2 in “Diffusion” for details.
rsd_swslope	Optional parameter that specifies the raised source/drain swslope. This parameter must be specified with rsd_thickness and rsd_spacing parameters.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Substrate Layer Structures

Substrate layer structures are used by inductance and substrate extraction. They contain the layer model for the substrate. Substrate layers are semiconductors, so they may be both conductors and dielectrics.

Substrate layers are not part of the relational conductor stack, but do require some form of absolute placement and thickness information, such as zbottom and thickness or zbottom and ztop parameters. Their absolute placement is negative and not allowed by normal conductor layers, but may be positive for an SOI type process.

Substrate layer types include:

Well

The well keyword is an optional keyword used to define the substrate layer structure. This layer type may be specified once in the MIPT file. The zbottom and thickness parameters are required to establish the substrate.

Table 4-42 describes the well keyword.

Table 4-42. Well Keyword

Well Keyword	Description
well	Required syntax used to define a well layer.
layer_name	Required user specified name that identifies the layer.

Table 4-43 lists the well parameters.

Table 4-43. Well Parameters

Well Parameters	Description
zbottom	Required parameter that specifies the bottom z-coordinate of the well layer.
thickness	Required parameter that specifies the thickness of the well layer.
resistivity	Required parameter that specifies the well layer resistance as rho, typically greater than or equal to 0.
eps	Required parameter that specifies the relative permittivity (dielectric constant).

Table 4-43. Well Parameters (cont.)

Well Parameters	Description
bulk_min_width	Optional layer parameter that specifies the minimum width of the bulk layer.
bulk_resistance	Optional layer parameter that specifies the sheet resistance of the bulk layer.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Substrate

The substrate keyword is an optional keyword used to define the substrate layer. This layer type may be specified once in the MIPT file. Substrate layers require resistivity, eps, and thickness. They may not have tables and should not be coplanar with each other or have gaps. The ztop and zbottom are required to establish the substrate.

Table 4-44 describes the substrate keyword.

Table 4-44. Substrate Keyword

Substrate Keyword	Description
substrate	Required syntax used to define a substrate layer.
layer_name	Required user specified name that identifies the layer.

Table 4-45 lists the substrate parameters.

Table 4-45. Substrate Parameters

Substrate Parameters	Description
ztop	Required parameter that specifies the top z-coordinate of the substrate layer.
zbottom	Required parameter that specifies the bottom z-coordinate of the substrate layer.
resistivity	Required parameter that specifies the substrate layer resistance as rho, typically greater than or equal to 0.
eps	Required parameter that specifies the relative permittivity (dielectric constant).
bulk_min_width	Optional layer parameter that specifies the minimum width of the bulk layer.
bulk_resistance	Optional layer parameter that specifies the sheet resistance of the bulk layer.

Table 4-45. Substrate Parameters (cont.)

Substrate Parameters	Description
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Tap

The tap keyword is an optional keyword used to connect power and ground nets to the substrate. This layer type may be specified once in the MIPT file. The vertical and lateral resistance is modeled. Tap layers require the parameters for thickness and resistivity. They have placement information defined relative to the top of the substrate. This information is not used for capacitance calculations in Calibre xRC or Calibre xACT 3D. The ztop and zbottom parameters are required.

Table 4-46 describes the tap keyword.

Table 4-46. Tap Keyword

Tap Keyword	Description
tap	Required syntax used to define a tap layer.
layer_name	Required user specified name that identifies the layer.

Table 4-47 lists the tap parameters.

Table 4-47. Tap Parameters

Tap Parameters	Description
ztop	Required parameter that specifies the top z-coordinate of the tap layer.
zbottom	Required parameter that specifies the bottom z-coordinate of the tap layer.
resistivity	Required parameter that specifies the tap layer resistance as rho, typically greater than or equal to 0.
bulk_min_width	Optional layer parameter that specifies the minimum width of the bulk layer.
bulk_resistance	Optional layer parameter that specifies the sheet resistance of the bulk layer.
hidden	Optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files.

Chapter 5
Process Variation

Process variation occurs due to many systematic and random factors. xCalibrate provides MIPT syntax used to account for process variation when performing parasitic extraction.

Process variation factors can include:

- Design related factors such as the design pattern and homogeneity of interconnect density.
- Process related factors such as manufacturing imperfections.

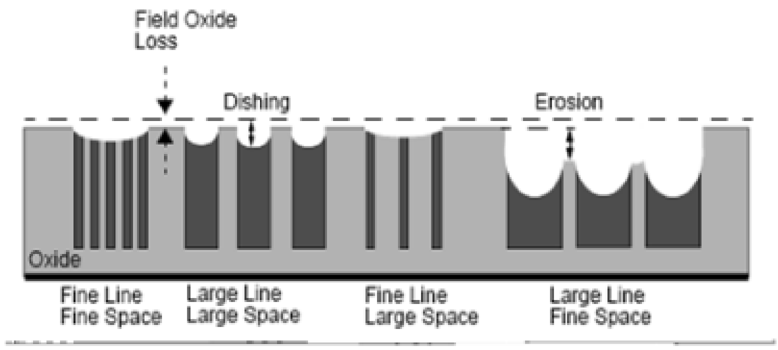
There are multiple manufacturing steps that cause process variation such as optical, etch, and polishing. This can significantly affect parametric yield, thus the impact on parasitics must be accurately predicted. The effects of systematic variations on conductor dimensions such as width and thickness, are estimated using simplified equations, tables, or models.

Various foundries have different emphasis on what to model, as well as different formats to provide variation data. Typical inputs provided by the foundries include the width, thickness, and interconnect resistivity variations. Via dimension and resistivity variation, and temperature coefficient variation dependent on conductor width are provided.

Thickness Variation

Thickness variation is due to chemical mechanical polishing (CMP), reactive ion etch (RIE), or micro loading effect. Chemical mechanical polishing causes dishing and erosion on the tops of conductors. Therefore, top thickness variation depends on conductor width, spacing, and density. Reactive ion etch and micro loading effect affect the bottom of the conductor. Consequently, bottom thickness variation depends on conductor width, spacing and density.

Figure 5-1. Chemical Mechanical Polishing



Thickness varies as a function of:

- density in one or more specified windows.
- conductor width.
- spacing to neighbors.

Thickness variation is specified by one of the following:

- polynomial equations as a function of conductor width and density.
- thickness tables as a function of conductor width and spacing.
- output of CMP and Etch simulation.

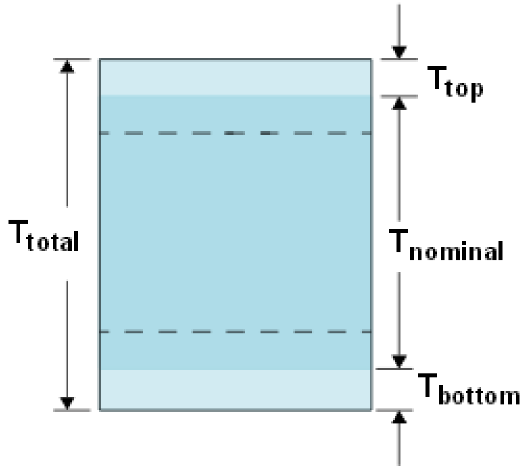
When using Table Syntax, different variation tables are needed for capacitance and resistance calculations. Capacitance depends on the conductor side dimensions, while resistance depends on conductor cross section, or effective thickness and width. Use THICKNESS Table to define actual thickness for a conducting layer. In order to use CMP data for thickness, the conducting layer must have a RHO_T or RSH_T table and a THICKNESS table.

To calculate actual thickness, T_{total}, use the following equation:

$$T_{total} = T_{nominal} + T_{top} + T_{bottom}$$

Nominal thickness, $T_{nominal}$, is the desired thickness of the conducting layer unaffected by CMP or RIE. Top thickness, T_{top}, is the thickness variation due to CMP. Use THICKNESS_TOP table to define top thickness for a layer. Bottom thickness, T_{bottom}, is thickness variation due to RIE. Use THICKNESS_BOT table to define bottom thickness for a layer. See THICKNESS_TOP and THICKNESS_BOT Tables for syntax.

Figure 5-2. Thickness Variations

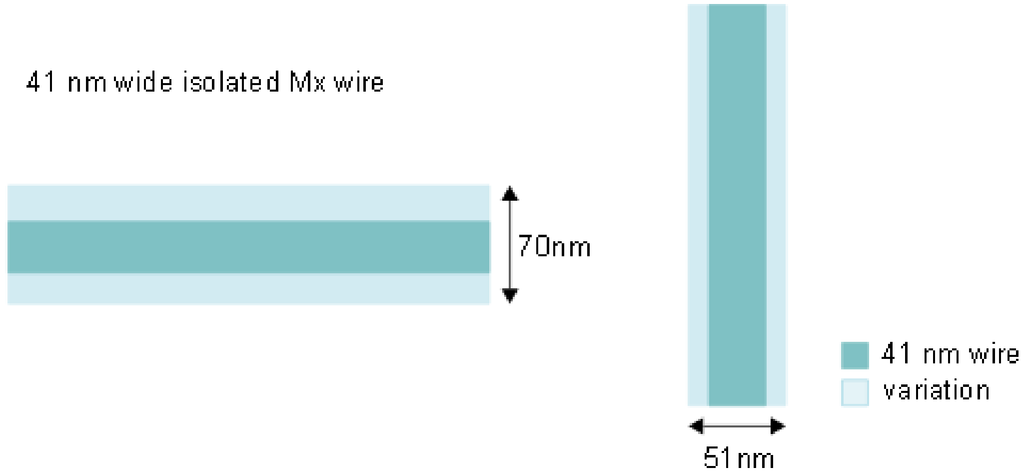


Retargeting

Retargeting is method used to apply various effects to conducting layers at different orientations. Width variation is supported in MIPT, where actual width is based on drawn width and spacing. Metal variation depends on the orientation of the wire. This requires an orientation dependent width variation feature in extraction.

Figure 5-3 illustrates the width variation for the horizontal and vertical orientations of a 41 nm wide isolated Mx wire. Two wires with the same width of 41nm become 70nm and 51nm wide during manufacturing, since they are at different orientations.

Figure 5-3. Orientation-based Width Variation



In order to support orientation-based biasing, two width tables are needed for each direction, x and y. One to describe the actual width in the preferred direction and another to describe the actual width in the non-preferred direction.

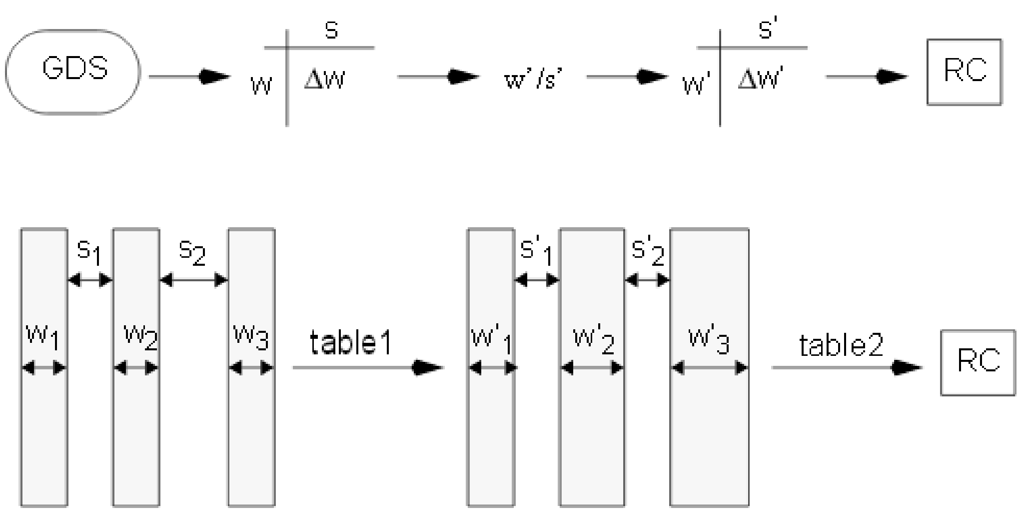
The bias_type parameter specified for the layer in your MIPT file designates whether the table is preferred or nonpreferred. This parameter may only be used with two-dimensional width or multi_bias property tables. For more information on multi_bias property tables, see MULTI_BIAS Table. For an example on how to define a multiple width table using table syntax, see "Multiple Width Table Example" on page 151.

Multiple Width Tables

The multiple width tables are used for advanced OPC modeling and calibration. The tables describe a final delta width variation. In this case, spacing refers to edge-based spacing.

The diagram in Figure 5-4 describes how the drawn width and drawn spacing are obtained from the GDS. The first table applies the width bias or delta w to the drawn width. Width bias is a function of drawn width and drawn spacing. During calibration the tool calculates the new adjusted values for w and s. The second table then applies a second bias delta w' to the adjusted width, which will result in the final width adjustment that is used for extraction. Delta w' is a function of the adjusted width and spacing.

Figure 5-4. Drawn Width and Spacing for Multiple Width Tables



The MULTI_BIAS Table must be specified together with a RHO_T Table and a thickness table type in your MIPT file in order to get accurate results. Thickness table type refers to thickness, thickness_top, and thickness_bot syntax. See THICKNESS Table and THICKNESS_TOP and THICKNESS_BOT Tables for more information. See "Multiple Width Table Example" on page 151 for an example on how to define a multiple width table using table syntax.

Gate Fringe Capacitance

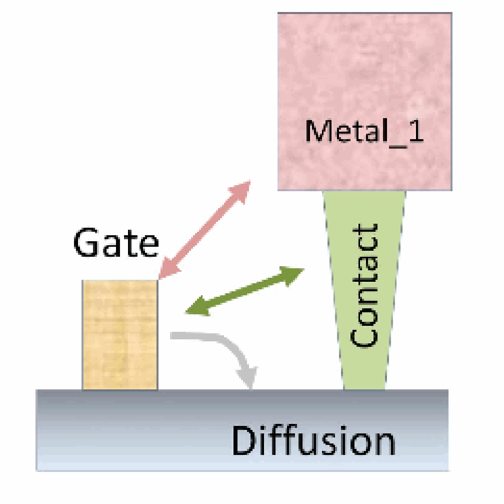
The capacitance fringe scaling table applies to gate layers only. The gate or capacitance fringe (CF tables) are provided by the foundry and are used to adjust the gate to source-drain (S/D) capacitance by adding or subtracting the capacitance values provided in the CF tables.

By default, the capacitance values from the CF tables are added to the source-drain capacitance. To subtract the CF table values from the gate to source-drain capacitance during gate to diffusion capacitance extraction, set the GATE_FRINGE Table parameter ignore_gate_to_diff to no.

The gate to source/drain capacitance consists of three components as shown in Figure 5-5:

- gate to metal1
- gate to contact
- gate to diffusion

Figure 5-5. Gate to S/D Capacitance of a planar MOS



Gate to diffusion capacitance depends on the gate-to-contact spacing and contact-to-contact spacing as shown in Figure 5-6. The larger the distance between a gate and contact, the larger the gate-to-diffusion capacitance. Gate-to-diffusion capacitance also increases with contact-to-contact spacing.

Figure 5-6. Gate to Diffusion Capacitance

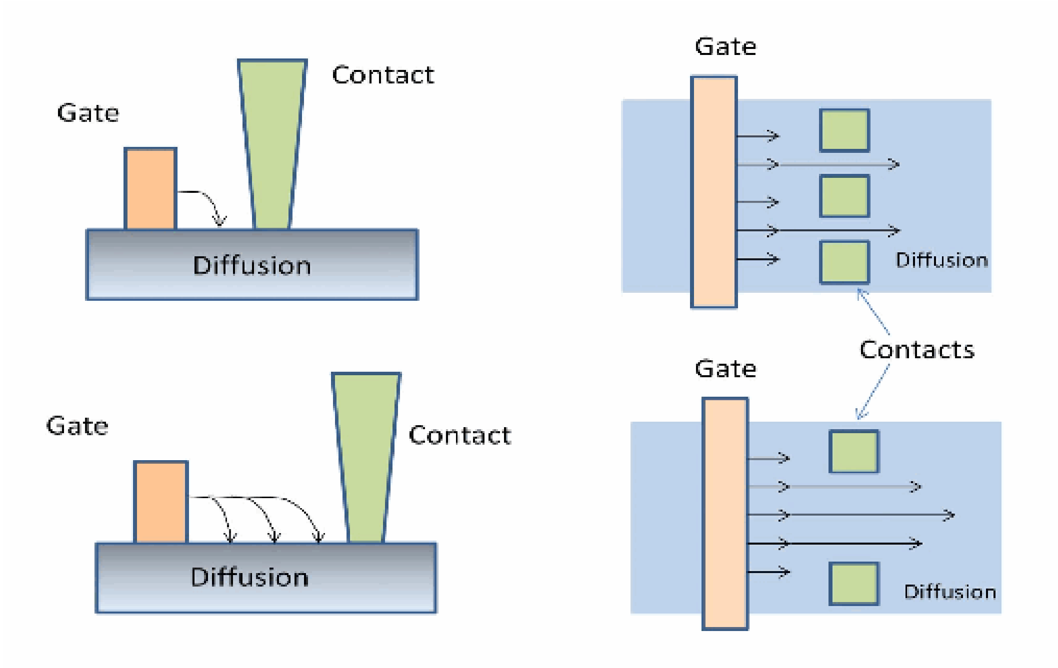


Table Syntax

Tables provide a unified way to input process variation information into xCalibrate.

Tables are typically specified in-line in the MIPT file, but can also be referenced using the use keyword. Providing the table data in-line avoids having to keep track of multiple files for a calibration. See Use Directive for more information on the use keyword.

Layout data contains shapes that are at desired or drawn dimensions. The actual width of each conductor varies from drawn dimension due to deterministic process variation. For example, the actual width of a metal path may be more narrow than the drawn width.

The on-chip variation effects can be modeled in Calibre xRC. This tool uses the variation data described by tables of discrete values in the MIPT file. Local density, width, and spacing data are used to make adjustments to the drawn shapes. The adjusted shapes are used when calculating parasitic effects. Some foundries also supply loading effects in their in-die information in order to model the changes in conductor thickness after reactive ion-etching.

Tables can be defined for any supported process variation parameter such as width, thickness, sheet resistance, resistivity, temperature coefficients tc1 and tc2, gate fringe, and contact bias.

You can specify tables to be resistive (R), capacitive (C), or both (RC). Each type of table has a default setting. The user must specify what type of table property information is stored. If a table applies only to resistance, such as RSH, then specifying it to be a C or RC table will generate an error message.

Most table data is specified as DRAWN dimensions (dim_type = drawn). Some data can be specified in ACTUAL dimensions, by setting dim_type = actual in the table.

Tables specified in-line are always associated with a layer. A layer may have zero or more tables associated with it. The order of the tables within a layer is not relevant. Tables are supported for layer types:

- Conductor
- Poly
- Diffusion
- Contact
- Via

For dielectric layers, inter- and intra-layer dielectric thickness are adjusted automatically during calibration based on the conductor thickness variation. So no table support is needed for dielectric layers.

Each table type is associated with a specific set of ordered variables. A one dimensional table has a single variable. When a table is used to describe a parameter, such as tc1, that singleton parameter is no longer required in the layer definition since it has a table representation.

A one dimensional table syntax is:

```
table = table_reference_name {
   property = table_property
   table_type = {R | C | RC}
   dim_type = {drawn | actual}
   value_type = {absolute | delta}
   equation = {<equation to evaluate, optional>}
   variable1 = {w1 w2  ... wn}
   value = {v(w1) v(w2) ... v(wn) }
}
```

Two dimensional table syntax has two variables:

```
table = table_reference_name {
   property = table_property
   table_type = {R | C | RC}
   dim_type = {drawn | actual}
   value_type = {absolute | delta}
   bias_type = {preferred | nonpreferred}
   enclosure = {max | min}
   layers = { up | down | both }
   equation = {<equation to evaluate, optional>}
   via_size = { x y }
   variable1 = {w1 w2  ... wn}
   variable2  = {s1 s2  ... sn}
   value = { v(w1,s1) v(w1,s2) ... v(w1,sn),
            ...,
           v(wn,s1) v(wn,s2) ... v(wn,sn)}
}
```

and so forth.

Table 5-1 describes the Table keyword.

Table 5-1. Table Keyword

Table Keyword	Description
table	Required keyword used to define a table construct.
table_reference_name	Required user specified name that identifies the structure.

Table 5-2 lists the Table parameters.

Table 5-2. Table Parameters

Table Parameters	Description
property	Required keyword whose value specifies the property type of the table. See Table 5-3 for a complete list of valid table property types.
table_type	Required keyword whose value specifies the type of table being described. Must be either R, C, or RC.
dim_type	Required keyword whose value specifies the type of dimensions being used. Must be either drawn or actual. Not required for rpv_vs_count table.
value_type	Required keyword whose value specifies the type of values being used. Must be either absolute or delta.
bias_type	Optional keyword whose value specifies whether the table describes the actual width in the preferred or non-preferred direction. Must be defined as either preferred or nonpreferred. Can only be used in two-dimensional width or multi_bias property tables.
enclosure	Optional keyword that specifies whether to use the min or max values from the up and down via_resistance or via_resistance2 table structure.
equation	Optional keyword whose value specifies an equation to evaluate in the table structure.
layers	Optional keyword that specifies whether the via resistance is dependent on the upper, lower, or both upper and lower layer's enclosure. Can only be used in the via_resistance or via_resistance2 table structure.
via_size	Optional parameter whose values specify the x,y dimensions of a square via enclosed in braces ({ }). Can only be used in the two-dimensional via_resistance2 property table.
variable	Required parameter that specifies the type of values being used. The values specified for a ***variable*** are enclosed in braces and must be listed in ascending order (smallest to largest). See Table 5-4 which lists the available variable keywords and their unit value.
value	A keyword whose value specifies the values applied to the associated variable in the table definition. Values supplied must be either absolute or delta to match the value_type setting. For rpv_vs_count, value is replaced with ratio.

Table Property

The *property* parameter defines the table's property. The table_type is required and must be either R, C, or RC. If both are present, the table is applied to both resistance and capacitance. Not every keyword is supported with every table type. For example, if the R keyword is specified with a capacitance table, then an error message is generated and calibration will stop.

Foundry modeling teams derive table data from silicon process technology test chips to support the table property content. For example, in conducting layer definitions there are four table property types that can be used for modeling resistance. Listed in order of precedence they are rho_t, rho, rsh_t, and rsh. For thickness dependent data they would define either a rho_t or a rsh_t table, otherwise they would define a rho or a rsh table.

Note

Tables of RESISTANCE property type are only allowed for via and contact layers.

Table 5-3 shows which options are available for each table property. It also identifies which properties can be in drawn or actual dimensions, or both, and what value type setting should be used.

Table 5-3. Available Table Property Types

Table Property	Table Type			Dimension Type		Value Type	
	R	C	RC	Drawn	Actual	Absolute	Delta
WIDTH	Y	Y	Y	Y	N	Y	N
MULTI_BIAS	Y	Y	Y	Y	N	N	Y
VWIDTH	N	Y	Y	Y	N	Y	N
VLENGTH	N	Y	Y	Y	N	Y	N
VRESISTANCE	Y	N	N	Y	N	Y	N
VIA_RESISTANCE	Y	N	N	Y	N	Y	N
VIA_RESISTANCE2	Y	N	N	Y	N	Y	N
THICKNESS	Y	Y	Y	Y	Y	Y	N
THICKNESS_TOP	Y	Y	Y	Y	Y	N	Y
THICKNESS_BOT	Y	Y	Y	Y	Y	N	Y
COUPLING	N	Y	N	N	Y	Y	N
GATE_FRINGE	N	Y	N	Y	N	Y	N
GATE_FRINGE4	N	Y	N	Y	N	Y	N
GATE_FRINGE_SCALE	N	Y	N	Y	N	Y	N
GATE_TOTAL	N	Y	N	N	Y	Y	N

Table 5-3. Available Table Property Types (cont.)

Table Property	Table Type			Dimension Type		Value Type	
	R	C	RC	Drawn	Actual	Absolute	Delta
GATE_INTRINSIC	N	Y	N	N	Y	Y	N
S2D_COUPLING	N	Y	N	N	Y	Y	N
SD_INTRINSIC	N	Y	N	N	Y	Y	N
SD_TOTAL	N	Y	N	N	Y	Y	N
RSH	Y	N	N	Y	Y	Y	N
RSH_T	Y	N	N	N	Y	Y	N
RHO	Y	N	N	Y	N	Y	N
RHO_T	Y	N	N	N	Y	Y	N
RESISTANCE	Y	N	N	Y	N	Y	N
INVERSE_RG_FACTOR_1	Y	N	N	Y	N	Y	N
INVERSE_RG_FACTOR_2	Y	N	N	Y	N	Y	N
TC1	Y	N	N	Y	N	Y	N
TC2	Y	N	N	Y	N	Y	N
CONTACT_WIDTH	N	Y	N	Y	N	Y	N
CONTACT_BIAS	N	Y	N	Y	N	Y	N
LI_GATE_FRINGE	N	Y	N	Y	N	Y	N
LI1_WIDTH	N	Y	N	Y	N	Y	N
LI2_WIDTH	N	Y	N	Y	N	Y	N
DIELECTRIC_CONSTANT	N	Y	N	Y	N	Y	N
EXTRA_WIDTH	Y	Y	Y	Y	Y	Y	Y
DAMAGE_THICKNESS	N	Y	N	Y	N	Y	Y
SIDEWALL_K	N	Y	N	Y	N	Y	N
TSV_CAPACITANCE	N	Y	N	Y	N	Y	N
TSV_RESISTANCE	Y	N	N	Y	N	Y	N
VERTICAL_RSH	Y	N	N	N	Y	Y	N
RPV_VS_COUNT	Y	N	N	N	N	Y	N

For information on the variables used for each table see Variable Keywords.

The following sections provide information on the table syntax for the available table property types:

CONTACT_BIAS Table

For contact_bias define variables width and spacing. Used for capacitance, BIAS values may be positive, negative or zero, and are associated with contact or via layers.

Table Syntax

```
table = my_contact_bias_table {
   property = contact_bias
   table_type = C
   dim_type = drawn
   value_type = absolute
   width = {w1 w2 ... wn}
   spacing = {s1 s2 ... sn}
   value = {v(w1,s1) v(w1,s2) ... v(w1,sn),
          ...,
          v(wn,s1) v(wn,s2) ... v(wn,sn)}
}
```

CONTACT_WIDTH Table

For contact_width define variables co_co_spacing and gate_co_spacing. Contact_width table entries for value are absolute values applied to the variation property, a floating point number in microns. Contact_width tables are associated with gate layers.

Table Syntax

```
table = my_contact_width_table {
   property = contact_width
   table_type = C
   dim_type = drawn
   value_type = absolute
   co_co_spacing = {c1 c2 ... cn}
   gate_co_spacing = {g1 g2 ... gn}
   value = {v(c1,g1) v(c1,g2) ... v(c1,gn),
          v(c2,g1) v(c2,g2) ... v(c2,gn),
          ...,
          v(cn,g1) v(cn,g2) ... v(cn,gn)}
}
```

COUPLING Table

The coupling table is used to specify the gate to source/drain capacitance correction values for a finFET device. This table should only be specified in seed layers associated with a multigate device.

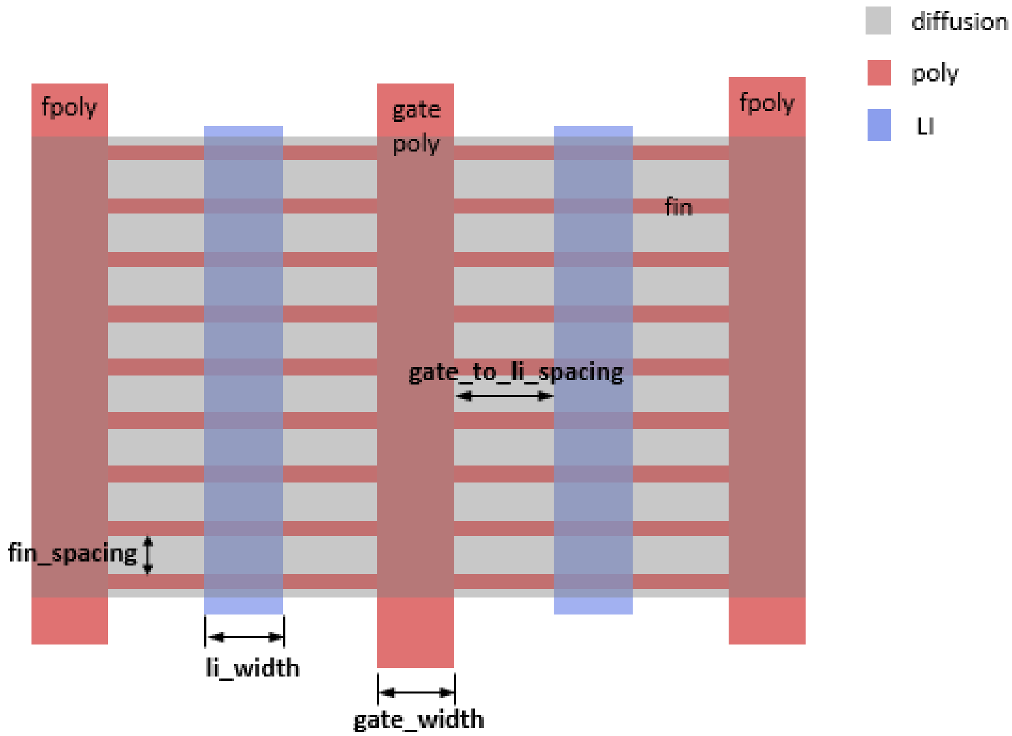
Table Syntax

```
table = my_coupling_table {
   property = coupling
   table_type = C
   dim_type = actual
   value_type = absolute
   shielding_layer = layer_name
   fin_count = {fc1 fc2 ... fcn}
   gate_width = {gwidth1 gwidth2 ... gwidthn}
   li_width = {li_width1 li_width2 ... li_widthn}
   fin_spacing = {fs1 fs2 ... fsn}
   gate_to_li_spacing = {g2li_spacing1 g2li_spacing2 ... g2li_spacingn}
   value = {v1 v2 ... vn)
}
```

Description

The coupling table uses variables shielding_layer, fin_count, gate_width, li_width, fin_spacing, and gate_to_li_spacing. The shielding_layer specifies the outer shielding layer. The values in the value list (v_1, v_2, ... v_n) are a function of fin_count, gate_width, li_width, fin_spacing and gate_to_li_spacing variables. In the example figure, Local Interconnect FinFET Example, fin_count is 9.

Figure 5-7. Local Interconnect FinFET Example



Use the COUPLING table together with S2D_COUPLING, SD_INTRINSIC, GATE_INTRINSIC, SD_TOTAL, and GATE_TOTAL tables to control different finFET device effects.

DAMAGE_THICKNESS Table

Use the damage_thickness table to specify the damaged thickness variation values for a planar dielectric layer versus the width and spacing of a coplanar conductor. Used for capacitance calculations of damaged sidewall k.

Table Syntax

```
table = my_damage_thickness_table {
   property = damage_thickness
   table_type = C
   dim_type = drawn
   value_type = delta | absolute
   sidewall_dielectric = coplanar_dielectric_layername
   width = {w1 w2 ... wn}
   spacing = {s1 s2 ... sn}
   value = {v(w1,s1) v(w1,s2) ... v(w1,sn),
            ...,
           v(wn,s1) v(wn,s2) ... v(wn,sn)}
}
```

Description

To apply damage_thickness for a dielectric layer, specify the damage_thickness table in the parent conductor layer and set the sidewall_dielectric variable to the coplanar dielectric layer name. The sidewall_dielectric variable is optional in the damage_thickness table.

See damage_thickness for additional information on damage thickness.

DIELECTRIC_CONSTANT Table

The dielectric_constant table defines the properties of width-spacing dependent conformal dielectric layer for gate geometries.

Table Syntax

```
table = my_dielectric_constant_table {
   property = dielectric_constant
   table_type = C
   dim_type = drawn
   value_type = absolute
   li_layer = M0
   gate_width = {w1 w2 ... wn}
   gate_to_li_spacing = {s1 s2 ... sn}
   value = {v(w1,s1) v(w1,s2) ... v(w1,sn),
             ...,
             v(wn,s1) v(wn,s2) ... v(wn,sn)}
}
```

EXTRA_WIDTH Table

An EXTRA_WIDTH table can be used to specify varying extra width values for side walls based on width and spacing for a conductor with a trapezoid shape.

Table Syntax

```
table = my_extra_width_table {
   property = extra_width
   table_type = R | C | RC
   dim_type = {drawn | actual}
   value_type = absolute | delta
   width = {w1 w2 ... wn}
   spacing = {s1 s2 ... sn}
   value = {v(w1,s1) v(w1,s2) ... v(w1,sn),
            ...,
            v(wn,s1) v(wn,s2) ... v(wn,sn)}
}
```

Description

EXTRA_WIDTH table must define variables width and spacing. If the dim_type is actual, then actual values are used for spacing. If the dim_type is drawn, then drawn values are used for spacing. The table entries for value can be absolute or delta values applied to the variation property. These are positive floating point numbers.

GATE_FRINGE Table

A gate_fringe table defines variables co_co_spacing and gate_co_spacing. The gate_fringe table entries for *value* are absolute values applied to the variation property, a positive floating point number in femtofarads. The gate_fringe table may consist of a single value if that is what corresponds to the spice device model.

Table Syntax

```
table = my_gate_fringe_table {
   property = gate_fringe
   table_type = C
   dim_type = drawn
   value_type = absolute
   ignore_gate_to_diff = yes | no
   co_co_spacing = {c1 c2 ... cn}
   gate_co_spacing = {g1 g2 ... gn}
   value = {v(c1,g1) v(c1,g2) ... v(c1,gn),
            v(c2,g1) v(c2,g2) ... v(c2,gn),
            ...,
            v(cn,g1) v(cn,g2) ... v(cn,gn)}
}
```

Description

Specify different gate_fringe tables for different MOS transistor types. Use gate_fringe tables to specify the fringe capacitance (CF) values.

The ignore_gate_to_diff variable setting determines whether the CF values are added or subtracted from the extracted gate to S/D capacitance value. If ignore_gate_to_diff is yes, the CF values are added; this is the default. If ignore_gate_to_diff is no, the CF values are subtracted.

GATE_FRINGE4 Table

A gate_fringe4 table defines variables co_co_spacing, gate_co_spacing, gate_width, and gate_length. The gate_fringe4 table entries for *value* are absolute values applied to the variation property, a positive floating point number in femtofarads. The gate_fringe4 table may consist of a single value if that is what corresponds to the spice device model.

Table Syntax

```
table = my_gate_fringe_table {
   property = gate_fringe4
   table_type = C
   dim_type = drawn
   value_type = absolute
   ignore_gate_to_diff = yes | no
   co_co_spacing = {c1 c2 ... cn}
   gate_co_spacing = {g1 g2 ... gn}
   gate_width = {w1 w2 ... wn}
   gate_length = {l1 l2 ... ln}
   value = {v(c1,g1,w1,l1) v(c1,g1,w1,l2) ... v(c1,g1,w1,ln),
            v(c1,g1,w2,l1) v(c1,g1,w2,l2) ... v(c2,g1,wn,ln),
            ...,
            v(cn,gn,w1,l1) v(cn,gn,w1,l2) ... v(cn,gn,wn,ln)}
}
```

Description

Specify different gate_fringe4 tables for different MOS transistor types. Use gate_fringe4 tables to specify the fringe capacitance (CF) values that require gate width and gate length detail.

The ignore_gate_to_diff variable setting determines whether the CF values are added or subtracted from the extracted gate to S/D capacitance value. If ignore_gate_to_diff is yes, the CF values are added; this is the default. If ignore_gate_to_diff is no, the CF values are subtracted.

GATE_FRINGE_SCALE Table

The gate_fringe_scale table specifies the scaling factor of the fringe capacitance (CF values) when two devices share the same diffusion area such as multi-finger transistors. This table can only be specified together with the gate_fringe table.

Table Syntax

```
table = my_gate_fringe_scale_table {
   property = gate_fringe_scale
   table_type = C
   dim_type = drawn
   value_type = absolute
   ignore_gate_to_diff = yes | no
   co_co_spacing = {c1 c2 ... cn}
   gate_co_spacing = {g1 g2 ... gn}
   value = {v(c1,g1) v(c1,g2) ... v(c1,gn),
            v(c2,g1) v(c2,g2) ... v(c2,gn),
            ...,
            v(cn,g1) v(cn,g2) ... v(cn,gn)}
}
```

Description

A gate_fringe_scale table defines variables co_co_spacing and gate_co_spacing. The gate_fringe_scale table entries for *value* are absolute values applied to the variation property, a positive floating point number in femtofarads. The gate_fringe_scale table may consist of a single value if that is what corresponds to the spice device model. Specify different gate_fringe_scale tables for different MOS transistor types. Use a gate_fringe_scale table together with a gate_fringe table to specify the scaling factor of the fringe capacitance (CF) values.

The ignore_gate_to_diff variable setting determines whether the CF values are added or subtracted from the extracted gate to S/D capacitance value. If ignore_gate_to_diff is yes, the CF values are added; this is the default. If ignore_gate_to_diff is no, the CF values are subtracted.

GATE_INTRINSIC Table

The gate_intrinsic table is used to specify the gate intrinsic capacitance correction values for a finFET device. This table should only be specified in seed layers associated with a multigate device.

Table Syntax

```
table = my_coupling_table {
   property = gate_intrinsic
   table_type = C
   dim_type = actual
   value_type = absolute
   shielding_layer = layer_name
   fin_count = {fc1 fc2 ... fcn}
   gate_width = {gwidth1 gwidth2 ... gwidthn}
   li_width = {li_width1 li_width2 ... li_widthn}
   fin_spacing = {fs1 fs2 ... fsn}
   gate_to_li_spacing = {g2li_spacing1 g2li_spacing2 ... g2li_spacingn}
   value = {v1 v2 ... vn)
}
```

Description

The gate_intrinsic table uses variables shielding_layer, fin_count, gate_width, li_width, fin_spacing, and gate_to_li_spacing. The shielding_layer specifies the outer shielding layer. The values in the value list (v_1, v_2, ... v_n) are a function of fin_count, gate_width, li_width, fin_spacing and gate_to_li_spacing variables. Figure 5-7 illustrates the spacing and width measurements.

Use the GATE_INTRINSIC table together with COUPLING, S2D_COUPLING, SD_TOTAL, SD_INTRINSIC, and GATE_TOTAL tables to control different finFET device effects.

GATE_TOTAL Table

The gate_total table is used to specify the gate capacitance correction values for a finFET device. This table should only be specified in seed layers associated with a multigate device.

Table Syntax

```
table = my_coupling_table {
   property = gate_total
   table_type = C
   dim_type = actual
   value_type = absolute
   shielding_layer = layer_name
   fin_count = {fc1 fc2 ... fcn}
   gate_width = {gwidth1 gwidth2 ... gwidthn}
   li_width = {li_width1 li_width2 ... li_widthn}
   fin_spacing = {fs1 fs2 ... fsn}
   gate_to_li_spacing = {g2li_spacing1 g2li_spacing2 ... g2li_spacingn}
   value = {v1 v2 ... vn)
}
```

Description

The gate_total table uses variables shielding_layer, fin_count, gate_width, li_width, fin_spacing, and gate_to_li_spacing. The shielding_layer specifies the outer shielding layer. The values in the value list (v_1, v_2, ... v_n) are a function of fin_count, gate_width, li_width, fin_spacing and gate_to_li_spacing variables. Figure 5-7 illustrates the spacing and width measurements.

Use the GATE_TOTAL table together with COUPLING, S2D_COUPLING, SD_INTRINSIC, GATE_INTRINSIC, and SD_TOTAL tables to control different finFET device effects.

INVERSE_RG_FACTOR_1 and INVERSE_RG_FACTOR_2 Tables

The inverse_rg_factor_1 and inverse_rg_factor_2 tables are used to define the delta resistance that is added to the gate terminal when vias are on a gate geometry. These tables can only be specified for seed layers and are typically specified together.

Table Syntax

inverse_rg_factor_1 table syntax:

```
table = my_inverse_rg_factor1_table {
   property = inverse_rg_factor_1
   table_type = R
   dim_type = drawn
   value_type = absolute
   fin_count = {fc1 fc2 ... fcn}
   ratio1 = {ratio1 ratio2 ... ration}
   value = {v1 v2 ... vn)
}
```

inverse_rg_factor_2 table syntax:

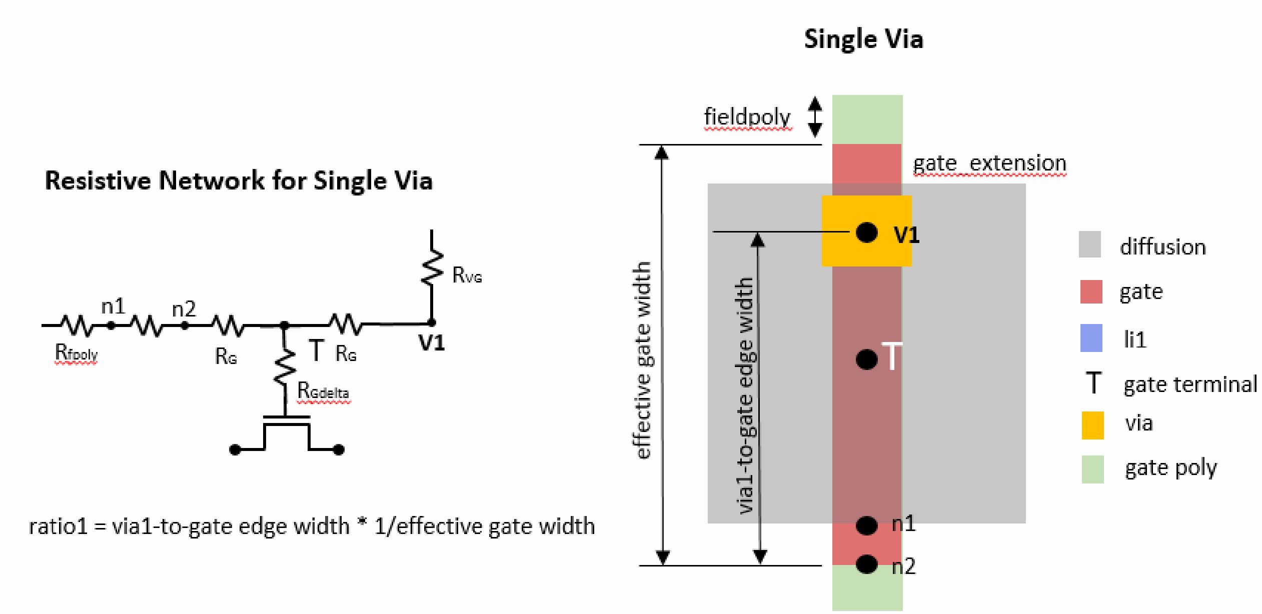
```
table = my_inverse_rg_factor2_table {
   property = inverse_rg_factor_2
   table_type = R
   dim_type = drawn
   value_type = absolute
   fin_count = {fc1 fc2 ... fcn}
   ratio1 = {ratio1 ratio2 ... ration}
   ratio2 = {ratio1 ratio2 ... ration}
   value ={v1 v2 ... vn}
}
```

Description

Use an inverse_rg_factor_1 and inverse_rg_factor_2 table to specify the delta gate resistance values for the gate terminal.

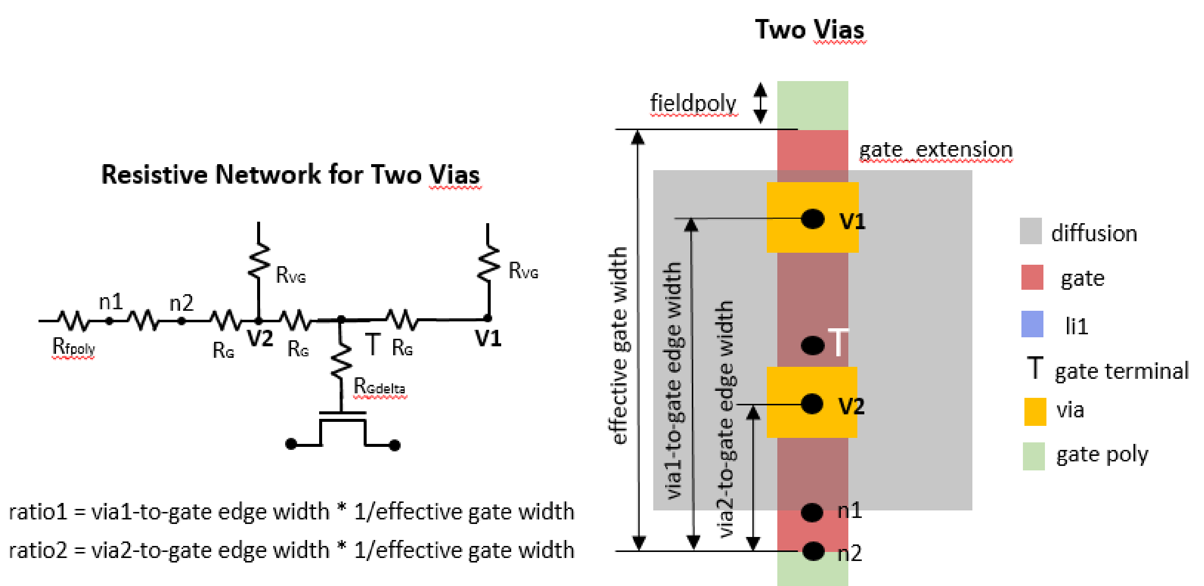
An inverse_rg_factor_1 table uses the fin_count and ratio1 variables. Use inverse_rg_factor_1 table to model a single via on the gate (Figure 5-8).

Figure 5-8. Single Via



An inverse_rg_factor_2 table uses the fin_count, ratio1, and ratio2 variables. Use inverse_rg_factor_2 table type to model two vias on the gate (Figure 5-9).

Figure 5-9. Two Vias



For the inverse_rg_factor_1 table, the fin_count defines the number of fins in the device. Ratio1 is the ratio of the distance from the via center to the bottom gate extension edge and the effective gate width. The values are the ratio of the model resistance to effective resistance.

For the inverse_rg_factor_2 table, the fin_count defines the number of fins in the device. Ratio1 and ratio2 are the ratio of the distance from each via center to the bottom gate extension edge and the effective gate width for each via. The values are the ratio of the model resistance to effective resistance.

LI_GATE_FRINGE Table

The li_gate_fringe defines the table for local interconnect capacitance.

Table Syntax

```
table = my_li_gate_fringe_table {
   property = li_gate_fringe
   table_type = C
   dim_type = {drawn | percentage}
   value_type = absolute
   li1 = {li1₁ li1₂}
   li2 = {li2₁ li2₂}
   value = {v(li11,li21) v(li11,li22)
            v(li12,li21) v(li12,li22)}

}
```

Description

If li1 and li2 variables represent percentage values, then the variable values should be within the range [0..1]. If li1 and li2 variables represent drawn values, then the variable values should be positive floating point values.

LI1_WIDTH and LI2_WIDTH Tables

The li1_width and li2_width define tables for local interconnect capacitance.

Table Syntax

li1_width table syntax:

```
table = my_li1_width_table {
   property = li1_width
   table_type = C
   dim_type = {drawn | percentage}
   value_type = absolute
   li1 = {li1₁ li1₂}
   li2 = {li2₁ li2₂}
   value = {v(li11,li21) v(li11,li22)
            v(li12,li21) v(li12,li22)}
}
```

li2_width table syntax:

```
table = my_li2_width_table {
   property = li2_width
   table_type = C
   dim_type = {drawn | percentage}
   value_type = absolute
   li1 = {li1_1 li1_2}
   li2 = {li2_1 li2_2}
   value = {v_(li11,li21) v_(li11,li22)
            v_(li12,li21) v_(li12,li22)}
}
```

Description

If li1 and li2 variables represent percentage values, then the variable values should be within the range [0..1]. If li1 and li2 variables represent drawn values, then the variable values should be positive floating point values.

MULTI_BIAS Table

Use the multi_bias table with a rho_t table and a thickness table to control etching accuracy.

Table Syntax

Two Level:

```
table = my_multi_bias_table {
   property = multi_bias
   table_type = {R | C | RC}
   dim_type = drawn
   value_type = absolute | delta
   bias_type = {preferred | nonpreferred}
   width_1 = {w_1 w_2 ... w_n}
   spacing_1 = {s_1 s_2 ... s_n}
   value_1 = {v_(w1,s1) v_(w1,s2) ... v_(w1,sn),
              ...,
             v_(wn,s1) v_(wn,s2) ... v_(wn,sn)}
   width_2 = {w_1 w_2 ... w_n}
   spacing_2 = {s_1 s_2 ... s_n}
   value_2 = {v_(w1,s1) v_(w1,s2) ... v_(w1,sn),
              ...,
             v_(wn,s1) v_(wn,s2) ... v_(wn,sn)}
}
```

Three Level:

```
table = my_multi_bias_table {
   property = multi_bias
   table_type = {R | C | RC}
   dim_type = drawn
   value_type = absolute | delta
   bias_type = {preferred | nonpreferred}
   width_1 = {w_1 w_2 ... w_n}
   spacing_1 = {s_1 s_2 ... s_n}
   value_1 = {v_(w1,s1) v_(w1,s2) ... v_(w1,sn),
              ...,
             v_(wn,s1) v_(wn,s2) ... v_(wn,sn)}
   width_2 = {w_1 w_2 ... w_n}
   spacing_2 = {s_1 s_2 ... s_n}
   value_2 = {v_(w1,s1) v_(w1,s2) ... v_(w1,sn),
              ...,
             v_(wn,s1) v_(wn,s2) ... v_(wn,sn)}
   width_3 = {w_1 w_2 ... w_n}
   spacing_3 = {s_1 s_2 ... s_n}
   value_3 = {v_(w1,s1) v_(w1,s2) ... v_(w1,sn),
              ...,
             v_(wn,s1) v_(wn,s2) ... v_(wn,sn)}
}
```

Four Level:

```
table = my_multi_bias_table {
   property = multi_bias
   table_type = {R | C | RC}
   dim_type = drawn
   value_type = absolute | delta
   bias_type = {preferred | nonpreferred}
   width_1 = {w_1 w_2 ... w_n}
   spacing_1 = {s_1 s_2 ... s_n}
   value_1 = {v_(w1,s1) v_(w1,s2) ... v_(w1,sn),
              ...,
             v_(wn,s1) v_(wn,s2) ... v_(wn,sn)}
   width_2 = {w_1 w_2 ... w_n}
   spacing_2 = {s_1 s_2 ... s_n}
   value_2 = {v_(w1,s1) v_(w1,s2) ... v_(w1,sn),
              ...,
             v_(wn,s1) v_(wn,s2) ... v_(wn,sn)}
   width_3 = {w_1 w_2 ... w_n}
   spacing_3 = {s_1 s_2 ... s_n}
   value_3 = {v_(w1,s1) v_(w1,s2) ... v_(w1,sn),
              ...,
             v_(wn,s1) v_(wn,s2) ... v_(wn,sn)}
   width_4 = {w_1 w_2 ... w_n}
   spacing_4 = {s_1 s_2 ... s_n}
   value_4 = {v_(w1,s1) v_(w1,s2) ... v_(w1,sn),
              ...,
             v_(wn,s1) v_(wn,s2) ... v_(wn,sn)}
}
```

Description

The multi_bias table contains two, three, or four sets of variables and values. Foundries can choose the number of sets to use based on how many different effects they are trying to model. Examples of effects that can be modeled with multi_bias tables include reactive ion etch (RIE), optical process correction (catOPC), and width bias. These effects are nested sequentially using this multi-bias approach.

The value parameter values are return values that are absolute or delta values. An absolute value_type means the returned values are absolute width. A delta value_type means the returned values are either delta or bias values. Bias values may be positive, negative, or zero. A positive delta is an enlargement of metal size. Positive bias makes the conductor larger and negative bias (or etch) makes the conductor smaller. A difference between ITF and MIPT formats is that ITF only uses delta value_type where as MIPT is nested and can use either absolute or delta value_type.

This table type must be used with a rho_t table and a thickness table. When specified, the multi_bias table takes precedence over other table types such as width.

RESISTANCE Table

Resistance table should only be specified for via or contact layers. It is area based to handle square vias and contacts. Resistance table entries for value are absolute values applied to the variation property. Resistance is a positive floating point number in ohms per contact.

Table Syntax

```
table = my_resistance_table {
   property = resistance
   table_type = R
   dim_type = drawn
   value_type = absolute
   area = {a1 a2 ... an}
   value = {r1 r2 r3 ... rn}
}
```

Description

If a TC1 table is specified, then a resistance table must also exist. If TC1 and TC2 tables are specified, there must also be a resistance specification for that layer. A resistance table can be used without TC1 and TC2.

RHO Table

For RHO table property define variables width and spacing or width and length. The RHO table entries for value are absolute values applied to the variation property, a positive floating point number in ohm-microns.

Table Syntax

RHO table syntax using width and spacing variables:

```
table = my_rho_table {
   property = rho
   table_type = R
   dim_type = drawn
   value_type = absolute
   width = {w1 w2 ... wn}
   spacing = {s1 s2 ... sn}
   value = {v(w1,s1) v(w1,s2) ... v(w1,sn),
            ...,
            v(wn,s1) v(wn,s2) ... v(wn,sn)}
}
```

RHO table syntax using width and length variables:

```
table = my_rho_table {
   property = rho
   table_type = R
   dim_type = drawn
   value_type = absolute
   width = {w1 w2 ... wn}
   length = {l1 l2 ... ln}
   value = {v(w1,l1) v(w1,l2) ... v(w1,ln),
            ...,
            v(wn,l1) v(wn,l2) ... v(wn,ln)}
}
```

Description

Resistance can be modeled by specifying the resistivity parameter or by defining an RHO table for the layer. When resistance (R) is modeled with an RHO table and a width table with its table_type keyword set to R or RC is specified for the layer, the width table is used for computing resistance.

Use width and length variables when defining gate layers for FinFET devices.

RHO_T Table

For rho_t define variables width and thickness. The rho_t table entries for value are absolute values applied to the variation property, a positive floating point number in ohm-microns. Use the rho_t table when the resistance data is thickness dependent.

Table Syntax

```
table = my_rho_t_table {
   property = rho_t
   table_type = R
   dim_type = actual
   value_type = absolute
   width = {w1 w2 ... wn}
   thickness = {t1 t2 ... tn}
   value = {v(w1,t1) v(w1,t2) ... v(w1,tn),
            ...,
            v(wn,t1) v(wn,t2) ... v(wn,tn)}
}
```

Description

Resistance can be modeled by specifying the resistivity parameter or by defining an rho_t table for the layer. When resistance (R) is modeled with an rho_t table and a width table with its table_type keyword set to R or RC is specified for the layer, the width table is used for computing resistance.

RPV_VS_COUNT Table

The resistance-per-via versus count (RPV_VS_COUNT) table returns a via resistance multiplication factor based on the number of vias in the array that is used when calculating via resistance.

Table Syntax

```
table = my_rpv_vs_count_table {
   property = rpv_vs_count
   table_type = R
   value_type = absolute
   via_size = { x y }
   max_spacing = maximum_distance_between_via_arrays
   count = {c1 c2 ... cn}
   ratio = {ratio1 ratio2 ... ration}
}
```

Description

Use a rpv_vs_count table, defined within a via layer definition, to specify the via array configurations and number of vias in each via cluster.

This table is one-dimensional area-based table used to define multiple via arrays. Different array configurations can have the same total via count. Rpv_vs_count entries are absolute values. A single via size (area) is allowed as defined by width and length in the via_size variable definition. Count is the total number of vias in each array. In the rpv_vs_count table, the max_spacing variable is used to specify the maximum spacing between via arrays.

RSH Table

An RSH table must define variables width and spacing or width and length. However, an actual dimension RSH table does not depend on spacing. Specify dummy values for spacing when defining an actual dimension RSH table. If the dim_type is drawn, then drawn values are used for spacing.

Table Syntax

RSH table syntax using width and spacing variables:

```
table = my_rsh_table {
   property = rsh
   table_type = R
   dim_type = {drawn | actual}
   rsh_type = {drawn | actual}
   value_type = absolute
   width = {w1 w2 ... wn}
   spacing = {s1 s2 ... sn}
   value = {v(w1,s1) v(w1,s2) ... v(w1,sn),
            ...,
            v(wn,s1) v(wn,s2) ... v(wn,sn)}
}
```

RSH table syntax using width and length variables:

```
table = my_rsh_table {
   property = rsh
   table_type = R
   dim_type = {drawn | actual}
   value_type = absolute
   width = {w1 w2 ... wn}
   length = {l1 l2 ... ln}
   value = {v(w1,l1) v(w1,l2) ... v(w1,ln),
            ...,
            v(wn,l1) v(wn,l2) ... v(wn,ln)}
}
```

Description

RSH table entries for value are absolute values applied to the variation property. These are positive floating point numbers in ohms per square.

Resistance can be modeled by specifying the r_sheet parameter or by defining an RSH table for the layer. When resistance (R) is modeled with an RSH table and a WIDTH table with its table_type keyword set to R or RC is specified for the layer, the WIDTH table is not used for computing resistance.

The optional rsh_type variable is used only in the width and spacing format of the RSH table. The rsh_type variable specifies whether to use actual or drawn width dimensions for resistance calculations during extraction. Use this parameter when you need to use drawn dimensions from the table and actual dimensions from the models to calculate the resistance values. The rsh_type parameter cannot be used in the width and length format of the RSH table.

Use the width and length format when defining gate layers for FinFET devices.

RSH_T Table

For rsh_t define variables width and thickness. RSH_T table entries for value are absolute values applied to the variation property, a positive floating point number in ohms per square. Use an rsh_t table when the resistance data is thickness dependent.

Table Syntax

```
table = my_rsh_t_table {
   property = rsh_t
   table_type = R
   dim_type = actual
   value_type = absolute
   width = {w1 w2 ... wn}
   thickness = {t1 t2 ... tn}
   value = {v(w1,t1) v(w1,t2) ... v(w1,tn),
            ...,
            v(wn,t1) v(wn,t2) ... v(wn,tn)}
}
```

Description

Resistance can be modeled by specifying the r_sheet parameter or by defining an rsh_t table for the layer. When resistance (R) is modeled with an rsh_t table and a width table with its table_type keyword set to R or RC is specified for the layer, the width table is not used for computing resistance.

S2D_COUPLING Table

The s2d_coupling table is used to specify the source to drain capacitance correction values for a finFET device. This table should only be specified in seed layers associated with a multigate device.

Table Syntax

```
table = my_coupling_table {
   property = s2d_coupling
   table_type = C
   dim_type = actual
   value_type = absolute
   shielding_layer = layer_name
   fin_count = {fc1 fc2 ... fcn}
   gate_width = {gwidth1 gwidth2 ... gwidthn}
   li_width = {li_width1 li_width2 ... li_widthn}
   fin_spacing = {fs1 fs2 ... fsn}
   gate_to_li_spacing = {g2li_spacing1 g2li_spacing2 ... g2li_spacingn}
   value = {v1 v2 ... vn)
}
```

Description

The s2d_coupling table uses variables shielding_layer, fin_count, gate_width, li_width, fin_spacing, and gate_to_li_spacing. The shielding_layer specifies the outer shielding layer. The values in the value list (v_1, v_2, ... v_n) are a function of fin_count, gate_width, li_width, fin_spacing and gate_to_li_spacing variables. Figure 5-7 illustrates the spacing and width measurements.

Use the S2D_COUPLING table together with COUPLING, SD_INTRINSIC, GATE_INTRINSIC, SD_TOTAL, and GATE_TOTAL tables to control different finFET device effects.

SD_INTRINSIC Table

The sd_intrinsic table is used to specify the source/drain intrinsic capacitance correction values for a finFET device. This table should only be specified in seed layers associated with a multigate device.

Table Syntax

```
table = my_coupling_table {
   property = sd_intrinsic
   table_type = C
   dim_type = actual
   value_type = absolute
   shielding_layer = layer_name
   fin_count = {fc1 fc2 ... fcn}
   gate_width = {gwidth1 gwidth2 ... gwidthn}
   li_width = {li_width1 li_width2 ... li_widthn}
   fin_spacing = {fs1 fs2 ... fsn}
   gate_to_li_spacing = {g2li_spacing1 g2li_spacing2 ... g2li_spacingn}
   value = {v1 v2 ... vn)
}
```

Description

The sd_intrinsic table uses variables shielding_layer, fin_count, gate_width, li_width, fin_spacing, and gate_to_li_spacing. The shielding_layer specifies the outer shielding layer. The values in the value list (v_1, v_2, ... v_n) are a function of fin_count, gate_width, li_width, fin_spacing and gate_to_li_spacing variables. Figure 5-7 illustrates the spacing and width measurements.

Use the SD_INTRINSIC table together with COUPLING, S2D_COUPLING, SD_TOTAL, GATE_INTRINSIC, and GATE_TOTAL tables to control different finFET device effects.

SD_TOTAL Table

The sd_total table is used to specify the total source/drain capacitance correction values for a finFET device. This table should only be specified in seed layers associated with a multigate device.

Table Syntax

```
table = my_coupling_table {
   property = sd_total
   table_type = C
   dim_type = actual
   value_type = absolute
   shielding_layer = layer_name
   fin_count = {fc1 fc2 ... fcn}
   gate_width = {gwidth1 gwidth2 ... gwidthn}
   li_width = {li_width1 li_width2 ... li_widthn}
   fin_spacing = {fs1 fs2 ... fsn}
   gate_to_li_spacing = {g2li_spacing1 g2li_spacing2 ... g2li_spacingn}
   value = {v1 v2 ... vn)
}
```

Description

The sd_total table uses variables shielding_layer, fin_count, gate_width, li_width, fin_spacing, and gate_to_li_spacing. The shielding_layer specifies the outer shielding layer. The values in the value list (v_1, v_2, ... v_n) are a function of fin_count, gate_width, li_width, fin_spacing and gate_to_li_spacing variables. Figure 5-7 illustrates the spacing and width measurements.

Use the SD_TOTAL table together with COUPLING, S2D_COUPLING, SD_INTRINSIC, GATE_INTRINSIC, and GATE_TOTAL tables to control different finFET device effects.

SIDEWALL_K Table

The sidewall_k table defines the side wall dielectric constant changes for various spacing.

Table Syntax

```
table = my_sidewall_k_table {
   property = sidewall_k
   table_type = C
   dim_type = drawn
   value_type = absolute
   sidewall_dielectric = coplanar_dielectric_layername
   corners = {DP_1 DP_2 ... DP_n}
   width = {w_1 w_2 ... w_n}
   spacing = {s_1 s_2 ... s_n}
   value_1 = {dc1_1 dc1_2 dc1_3 ... dc1_n}
   value_2 = {dc2_1 dc2_2 dc2_3 ... dc2_n}
...
   value_n = {dcn_1 dcn_2 dcn_3 ... dcn_n}
}
```

Description

Use this table type to appropriately adjust the dielectric constants and account for the mask offsets. Specify the corners (DP) that the table applies to with the corners variable keyword. Optionally use the sidewall_dielectric variable to specify the planar dielectric associated with the sidewall_k table in order to determine the base EPS during calibration. Specify the dielectric constants (dc) for various spacing for each corner with a corresponding value list.

TC1 and TC2 Tables

The TC1 and TC2 tables specify variation values for first-order temperature coefficient of resistance (TC1) and second-order temperature coefficient of resistance (TC2) in conducting and square via layers. Use the TC1 and TC2 table syntax with width and length variables for rectangular shaped vias.

Table Syntax

TC1 and TC2 table syntax using width variable:

```
table = my_tc1_table {
   property = tc1
   table_type = R
   dim_type = drawn
   value_type = absolute
   width = {w_1 w_2 w_3 ... w_n}
   value = {tc1_1 tc1_2 tc1_3 ... tc1_n}
}

table = my_tc2_table {
   property = tc2
   table_type = R
   dim_type = drawn
   value_type = absolute
   width = {w_1 w_2 w_3 ... w_n}
   value = {tc2_1 tc2_2 tc2_3 ... tc2_n}
}
```

TC1 and TC2 table syntax using width and length variables:

```
table = my_tc1_table {
   property = tc1
   table_type = R
   dim_type = drawn
   value_type = absolute
   width = {w_1 w_2 w_3 ... w_n}
   length = {l_1 l_2 l_3 ... l_n}
   value = {tc1_1(w1,l1) tc1_2(w1,l2) ... tc1_n(w_1,l_n),
            ...,
            tc1_1(wn,l1) tc1_2(wn,l2) ... tc1_n(wn,ln)}
}

table = my_tc2_table {
   property = tc2
   table_type = R
   dim_type = drawn
   value_type = absolute
   width = {w_1 w_2 w_3 ... w_n}
   length = {l_1 l_2 l_3 ... l_n}
   value = {tc2_1(w1,l1) tc2_2(w1,l2) ... tc2_n(w1,ln),
            ...,
            tc2_1(wn,l1) tc2_2(wn,l2) ... tc2_n(wn,ln)}

}
```

Description

TC1 and TC2 table entries for the value parameter are absolute values and applied to the variation property. The values are floating point numbers in microns.

For both conductor and via layers, you can specify a TC1 table without a TC2 table. But, you cannot specify a TC2 table without a TC1 table. Actual TC1 and TC2 can be used together with actual rsh_t and rho_t tables.

The syntax using width and length variables can only be used for via layers. When specifying TC1 tables for via layers, a resistance table must also be defined. If TC1 and TC2 tables are specified, there must also be a resistance specification for the via layer. The length variable is only required for rectangular vias.

For conductor layers, TC1 and TC2 may be specified without a resistance table. The syntax using width and length variables is not allowed.

THICKNESS Table

Use a thickness table to specify the total thickness due to all effects for a layer.

Table Syntax

thickness table syntax as a function of width and spacing:

```
table = my_thickness_spacing_table {
   property = thickness
   table_type = {R | C | RC}
   dim_type = {drawn | actual}
   value_type = absolute
   width = {w1 w2  ... wn}
   spacing = {s1 s2  ... sn}
   value = {v(w1,s1) v(w1,s2) ... v(w1,sn),
            ...,
            v(wn,s1) v(wn,s2) ... v(wn,sn)}
}
```

thickness table syntax as a function of width and density:

```
table = my_thickness_density_table {
   property = thickness
   table_type = {R | C | RC}
   dim_type = {drawn | actual}
   value_type = absolute
   width = {w1 w2  ... wn}
   density_type = {drawn | actual}
   density = {d1 d2  ... dn}
   value = {v(w1,d1) v(w1,d2) ... v(w1,dn),
            ...,
            v(wn,d1) v(wn,d2) ... v(wn,dn)}
}
```

Description

This includes the measurements for nominal, top, and bottom thickness as shown in Figure 5-2 in "Thickness Variation". Total thickness is represented by the following equation:

$$T_{total} = T_{nominal} + T_{top} + T_{bot}$$

Thickness tables can be varied as a function of width and spacing, or width and density. Translation from density to spacing uses the equation:

```
density = width/(width+spacing)
```

then

```
spacing = width*((1-density)/density)
```

Thickness table cannot exist together with thickness_top table for the same layer; however, they can exist together in the same mipt file as long as they are specified for different layers. When specifying thickness tables, use either width/spacing tables or width/density tables but not a combination of both.

The width and spacing values can be drawn or actual dimensions. The values are the absolute thickness value for a given width/spacing pair. Values are a floating point number in microns. For more information on total thickness, see “Thickness Variation” on page 103.

The optional density_type variable specifies whether to use drawn or actual density in the density calculations for the layer. Use the density_type variable in width/density tables, whether they are equation or table-based.

THICKNESS_TOP and THICKNESS_BOT Tables

Thickness_top and thickness_bot tables can exist together, be specified individually, or not at all. When specifying both thickness_top and thickness_bot tables, use either width/spacing tables or width/density tables but not a combination of both.

Table Syntax

Thickness_top and thickness_bot tables varied as a function of width and spacing:

```
table = my_thickness_top_spacing_table {
   property = thickness_top
   table_type = {R | C | RC}
   dim_type = {drawn | actual}
   value_type = delta
   width = {w1 w2 ... wn}
   spacing = {s1 s2 ... sn}
   value = {v(w1,s1) v(w1,s2) ... v(w1,sn),
            ...,
            v(wn,s1) v(wn,s2) ... v(wn,sn)}
}

table = my_thickness_bot_spacing_table {
   property = thickness_bot
   table_type = {R | C | RC}
   dim_type = {drawn | actual}
   value_type = delta
   width ={w1 w2 ... wn}
   spacing = {s1 s2 ... sn}
   value = {v(w1,s1) v(w1,s2) ... v(w1,sn),
            ...,
            v(wn,s1) v(wn,s2) ... v(wn,sn)}
}
```

Thickness_top and thickness_bot tables as a function of width and density:

```
table = my_thickness_top_density_table {
   property = thickness_top
   table_type = {R | C | RC}
   dim_type = {drawn | actual}
   value_type = delta
   width = {w_1 w_2 ... w_n}
   density = {d_1 d_2 ... d_n}
   value = {v_(w1,d1) v_(w1,d2) ... v_(w1,dn),
            ...,
            v_(wn,d1) v_(wn,d2) ... v_(wn,dn)}
}

table = my_thickness_bot_density_table {
   property = thickness_bot
   table_type = {R | C | RC}
   dim_type = {drawn | actual}
   value_type = delta
   width = {w_1 w_2 ... w_n}
   density = {d_1 d_2 ... d_n}
   value = {v_(w1,d1) v_(w1,d2) ... v_(w1,dn),
            ...,
            v_(wn,d1) v_(wn,d2) ... v_(wn,dn)}
}
```

Description

The width and spacing values can be drawn or actual dimensions. The values are the delta changes in thickness_bot or thickness_top that correspond to a given width/spacing pair. The values are positive when the top edge of the conductor moves up and the bottom of the conductor moves down, for example, when the overall thickness of the conductor increases. The values are negative when the top edge moves down and the bottom edge moves up, for example, when the overall thickness of the conductor decreases. Values are a floating point number in microns.

TSV_CAPACITANCE Table

The tsv_capacitance table specifies the capacitance values for parameterized coupling models (PCM) between TSVs in a TSV layer definition for either GDS or LEF/DEF design flows. TSV_capacitance table values can be specified for either a analog or digital flow. xCalibrate inserts the corresponding table information into the PEX rule files.

Table Syntax

Analog tsv_capacitance table syntax:

```
table = my_tsv_capacitance_table {
   property = tsv_capacitance
   table_type = C
   dim_type = drawn
   value_type = absolute
   spacing = {t1 t2 t3 ... tmax}
   value = {c1 c2 c3 ... cmax}
}
```

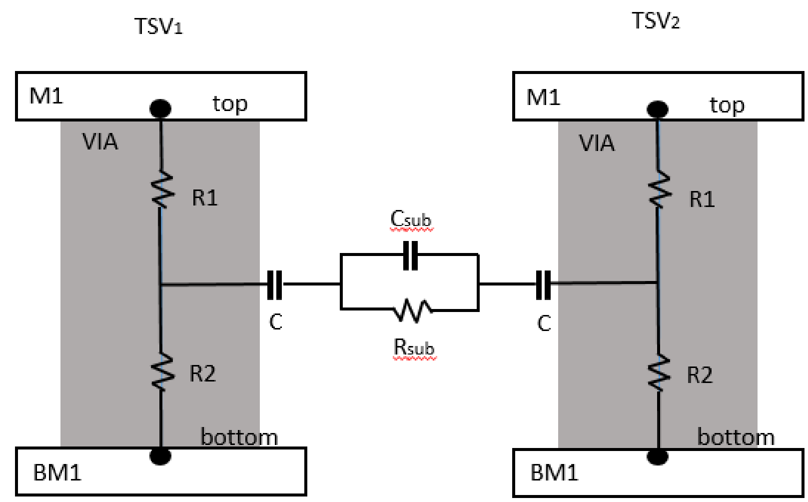
Digital tsv_capacitance table syntax:

```
table = my_tsv_capacitance_eff_table {
   property = tsv_capacitance
   table_type = C
   dim_type = drawn
   value_type = absolute
   frequency = {f1 f2 ... fn}
   spacing = {s1 s2 ... sn}
   value = {v(f1,s1) v(f1,s2) ... v(f1,sn),
            ...,
            v(fn,s1) v(fn,s2) ... v(fn,sn)}
}
```

Description

The analog flow requires a tsv_capacitance table and a tsv_resistance table. The tsv_capacitance table contains the coupling capacitance values as a function of distance (t_{dis}) between two TSVs (C_{sub} in Figure 5-10). BM1 is the from_layer and M1 is the to_layer. The distance is measured from the edge of one TSV to the other TSV.

Figure 5-10. tsv_capacitance and tsv_resistance for Analog Flow

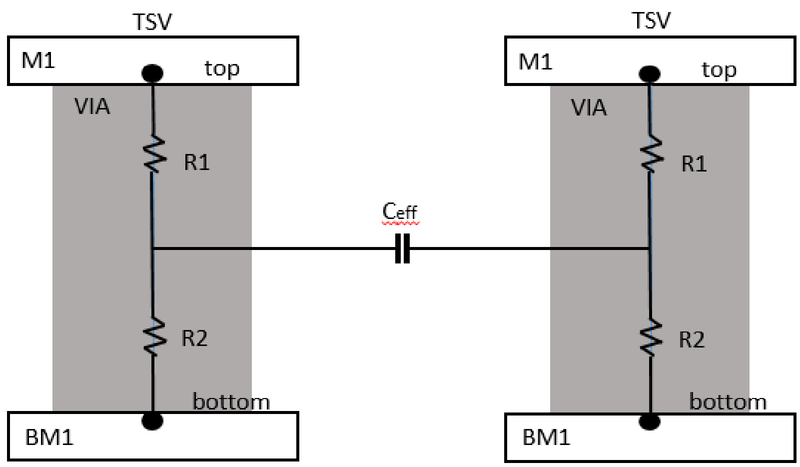


t_{dis}	t_1	t_2	t_3	t_4	t_5	. . .	t_{max}
C_{sub}	C_1	C_2	C_3	C_4	C_5	. . .	C_{max}
R_{dis}	R_1	R_2	R_3	R_4	R_5	. . .	R_{max}

The tsv_capacitance table must be specified together with the analog tsv_resistance table. For tsv_resistance table syntax, see TSV_RESISTANCE Table.

The digital flow requires the two-dimensional version of the tsv_capacitance table which describes the effective capacitance (C_{eff}) as a function of the frequency in hertz and distance (S) in microns between two TSVs (see Figure 5-11).

Figure 5-11. tsv_capacitance for Digital Flow



$TSV_{spacing}$ \ *freq*	f_1 (Hz)	f_2	f_3
S_1 (um)	C_{11}	C_{12}	C_{13}
S_2	C_{21}	C_{22}	C_{23}

The TSV layer definition can contain analog tsv tables, tsv_capacitance table and tsv_resistance table, and a digital tsv_capacitance table. Calibre xACT determines which couplings should be considered and uses the tsv_capacitance table to compute the parasitic coupling capacitance written to the extracted netlist. The TSV must be defined as a via in the LVS rules.

TSV_RESISTANCE Table

The tsv_resistance table specifies the resistance values for parameterized coupling models (PCM) between TSVs in a TSV layer definition for GDS design flows.

Table Syntax

Analog tsv_resistance table syntax:

```
table = my_tsv_resistance_table {
   property = tsv_resistance
   table_type = R
   dim_type = drawn
   value_type = absolute
   spacing = {t1 t2 t3 ... tmax}
   value = {R1 R2 R3 ... Rmax}
}
```

Description

The analog flow requires a tsv_resistance table and a tsv_capacitance table. The tsv_resistance table contains the resistance values as a function of distance (t_{dis}) between two TSVs (R_{sub} in Figure 5-10 on page 142). BM1 is the from_layer and M1 is the to_layer. The distance is measured in microns from the edge of one TSV to the other TSV.

The analog tsv_resistance table must be specified together with the analog tsv_capacitance table in the same TSV layer definition.

VERTICAL_RSH Table

Use the vertical_rsh table to specify vertical resistance from the bottom of a gate to the middle of a horizontal gate.

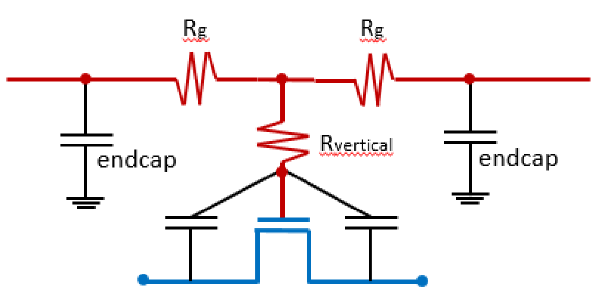
Table Syntax

```
table = my_vertical_rsh_table {
   property = vertical_rsh
   table_type = R
   dim_type = actual
   value_type = absolute
   width = {w1 w2 ... wn} #variable1
   length = {l1 l2 ... ln} #variable2
   value = { v(w1,l1) v(w1,l2) ... v(w1,ln),
               ...,
         v(wn,l1) v(wn,l2) ... v(wn,ln) }
}
```

Description

The vertical gate resistance model connects the gate pin to the horizontal gate resistors as shown in Figure 5-12. This creates a "T" resistor model with the gate pin at the bottom of the "T" and the poly endcaps at the two ends of the top of the "T".

Figure 5-12. Vertical Gate Resistance Model



VIA_RESISTANCE Table

Use a via_resistance table to specify resistance for via layers. A via_resistance table uses width, length, overlap_width, and overlap_length variable types. The width and length value_type are in absolute dimensions. Use this table type for square vias.

Table Syntax

```
table = my_via_resistance_table {
   property = via_resistance
   table_type = R
   dim_type = drawn | actual
   layers = up | down | both
   enclosure = min | max
   value_type = absolute
   width = {w1 w2}
   length = {l1 l2 ... ln}
   overlap_width = {w1 w2 ... wn}
   overlap_length = {l1 l2 ... ln}
   value ={v(ow1,ol1) v(ow1,ol2) ... v(ow1,oln),
            ...,
           v(own,ol1) v(own,ol2) ... v(own,oln)}
}
```

Description

The via_resistance table has only min and max values for width and length variables. The dim_type for overlap_width and overlap_length variables can be drawn(default) or actual, and should always be symmetric around the via center.

The enclosure setting determines how the enclosure polygon is derived based on whether it is set to min or max; The default is max. If layers is set to both, the tool uses the overlap between lower and upper layers; this is the default behavior for layers.

VIA_RESISTANCE2 Table

Use a via_resistance2 table, together with the optional via layer parameter via_r_select_up_down, to specify small and large resistance values for the same via size. The table can only be specified in via layers. A via_resistance2 table uses via_size, overlap_width, and overlap_length variable types. The value_type is absolute dimensions.

Table Syntax

```
table = my_via_resistance2_table {
   property = via_resistance2
   table_type = R
   dim_type = drawn
   layers = up | down | both
   enclosure = min | max
   value_type = absolute
   via_size = {x y}
   overlap_width = {w1 w2 ... wn}
   overlap_length = {l1 l2 ... ln}
   value ={v(ow1,ol1) v(ow1,ol2) ... v(ow1,oln),
            ...,
            v(own,ol1) v(own,ol2) ... v(own,oln)}
}
```

Description

The via_resistance2 table has only two values for the via_size variable. The dim_type for overlap_width and overlap_length variables is drawn. Up to 10 different via_resistance2 tables can be specified per via layer.

The enclosure setting determines how the enclosure polygon is derived based on whether it is set to min or max; The default is max. If layers is set to both, the tool uses the overlap between lower and upper layers; this is the default behavior for layers.

VRESISTANCE Table

Use a vresistance table to specify contact or via resistance. A vresistance table uses width and length variable types. The width and length values are drawn dimensions. Use this table type for vias and contacts that are not square. The vresistance table takes precedence over the specification of n1 and n2 parameters.

Table Syntax

```
table = my_vresistance_table {
   property = vresistance
   table_type = R
   dim_type = drawn
   value_type = absolute
   width = {w1 w2 ... wn}
   length = {l1 l2 ... ln}
   value ={v(w1,l1) v(w1,l2) ... v(w1,ln),
            ...,
            v(wn,l1) v(wn,l2) ... v(wn,ln)}
}
```

VWIDTH and VLENGTH Tables

Use vwidth and vlength tables together to specify square or rectangular vias. Vwidth and vlength tables use width and length variable types. The width and length values are drawn dimensions. Value entries are actual dimensions. For rectangular vias, use the longer dimension for vlength.

Table Syntax

vwidth table syntax:

```
table = my_vwidth_table {
   property = vwidth
   table_type = { C | RC}
   dim_type = drawn
   value_type = absolute
   width = {w1 w2 ... wn} #variable1
   length = {l1 l2 ... ln} #variable2
   value = { v(w1,l1) v(w1,l2) ... v(w1,ln),
                ...,
         v(wn,l1) v(wn,l2) ... v(wn,ln) }
}
```

vlength table syntax:

```
table = my_vlength_table {
   property = vlength
   table_type = {C | RC}
   dim_type = drawn
   value_type = absolute
   width = {w1 w2 ... wn} #variable1
   length = {l1 l2 ... ln} #variable2
   value = { v(w1,l1) v(w1,l2) ... v(w1,ln),
                ...,
         v(wn,l1) v(wn,l2) ... v(wn,ln) }
}
```

WIDTH Table

A width table uses width and spacing variable types. Width table entries for value are absolute values applied to the variation property, a positive floating point number in microns.

Table Syntax

```
table = my_width_table {
   property = width
   table_type = { R | C | RC}
   dim_type = drawn
   value_type = absolute
   bias_type = {preferred | nonpreferred}
   width = {w1 w2 ... wn}
   spacing = {s1 s2 ... sn}
   value = { v(w1,s1) v(w1,s2) ... v(w1,sn),
               ...,
             v(wn,s1) v(wn,s2) ... v(wn,sn) }
}
```

See “Width Table Example” on page 150 for an example.

Variable Keywords

The *variable* parameter is a table keyword.

Table 5-4 lists the available variable keywords and their unit value. Following the variable keyword is a list of values in brackets. The values in this list must be in increasing order from smallest to largest. These list values define the axis values for the table’s lookups. Values outside this range are set at the minimum or maximum value listed. Each table can have a maximum of two variable definitions. If the table bounds are exceeded, use values at the extent of the table.

Multi-dimensional Table Syntax has two or more variables and corresponding value array. The order of the values in the array is the same as in the two-dimensional table.

Multiple variable lines can be present in a table. The table entries for value are the absolute or delta values applied to the variation property.

If a bias has been set in the MIPT file, an error message should be generated if a table has settings that are out of range according to that bias.

The adjusted data in the value arrays must conform to the corresponding minimum values defined in the MIPT file. The adjusted widths must not be smaller than the minimum width for the layer. The adjusted spacing must not be smaller than the minimum spacing for the layer.

The required parameter ***value*** is the last entry of the table. The table entries for value are the absolute or delta values applied to the variation property. The end of line character of each line in the value array must be a comma (,). The value parameter can be specified as ***value***, ***value***$_1$, and ***value***$_2$ depending on the type of table. Tables with multiple values use ***value***$_1$ and ***value***$_2$.

Table 5-4. Variable Keywords

Variable Keyword	Value
area	Positive increasing floating point number in microns.
width	Positive increasing floating point number in microns.
length	Positive increasing floating point number in microns.
thickness	Positive increasing floating point number in microns.
spacing	Positive increasing floating point number in microns.
width1	Positive increasing floating point number in microns.
spacing1	Positive increasing floating point number in microns.
width2	Positive increasing floating point number in microns.
spacing2	Positive increasing floating point number in microns.
density	Positive increasing floating point number in microns less than or equal to 1.
co_co_spacing	Positive increasing floating point number in microns.
gate_co_spacing	Positive increasing floating point number in microns.
gate_width	Positive increasing floating point number in microns.
gate_length	Positive increasing floating point number in microns.
gate_to_li_spacing	Positive increasing floating point number in microns. Minimum spacing between a seed layer and its li layer.
li_layer	List of one or more space separated local interconnect layer names.
li1	If li1 layer exists, value must be 1. If li1 layer does not exist, value must be 0. Required for all LI table types.
li2	If li2 layer exists, value must be 1. If li2 layer does not exist, value must be 0. Optional for all LI table types.
corners	List of one or more space separated corner names
sidewall_dielectric	Dielectric layer name of the planar dielectric associated with the sidewall_k table or damage_thickness table.
frequency	List of one or more space separated frequency values in hertz used with the digital tsv_capacitance table.

Table Examples

Various examples are provided to demonstrate table structure usage.

Width Table Example

This is an example of a width table defined in an MIPT file:

```
conductor = metal4 {
   min_width = 0.1
   min_spacing = 0.1
   thickness = 0.2
   table = m4_width_table {
      property = width
      table_type = R
      dim_type = drawn
      value_type = absolute
      width = {0.1 0.2 0.25}
      spacing = {0.15 0.25 0.5}
      value = {0.2 0.3 0.5,
               0.2 0.3 0.5,
               0.2 0.3 0.5}
   }
}
```

Multiple Width Table Example

This is an example of a multiple width table defined in an MIPT file:

```
conductor = metal1 {
   thickness = 0.092
   min_width = 0.046
   min_spacing = 0.046

   table = rho_t_table {
      property = rho_t
      table_type = R
      dim_type = actual
      value_type = delta
      width = {0.065 0.08 0.14}
      thickness = {0.04 0.06 0.15 2.5}
      value = {0.036 0.032  0.03  0.028,
               0.035 0.0318 0.029 0.0278,
               0.034 0.0315 0.028 0.0276}
   }
   table = thickness_top_table {
      property = thickness_top
      table_type = RC
      dim_type = drawn
      value_type = delta
      width = {0.045 0.08 0.2 2.5}
      spacing = {0.045 0.08 0.2 2.5}
      value = {0 0.002 0.005 0.006,
               0.003 0.004 0.006 0.007,
              -0.001 0.001 0.03 0.004,
              -0.005 -0.0045 -0.002 0.0001}
   }
   table = multi_bias_table {
      property = multi_bias
      table_type = RC
      dim_type = drawn
      value_type = delta
      width1 = {0.045 0.08 0.14 0.2}
      spacing1 = {0.045 0.1 0.15 0.25}
      value1 = { -0.002 0.004 0.015 0.02,
                 -0.004 0.002 0.012 0.018,
                 -0.006 -0.002 0.01 0.016,
                 -0.01 -0.004 -0.002 -0.002}
      width2 = {0.046 0.085 0.2 1.0 3.2}
      spacing2 = {0.05 0.1 0.5 2.5}
      value2 = { 0.002 0.005 0.01 0.02,
                -0.001 0.001 0.008 0.018,
                -0.003 -0.001 0.005 0.017,
                 0.0005 0.0007 0.001 0.0015,
                 0.002 0.0022 0.0022 0.01}
   }
}
```

Thickness Table Example

This is an example of a thickness table used to specify the total thickness for a specific layer:

```
conductor = metal1 {
   thickness = 0.092
   min_width = 0.046
   min_spacing = 0.046

   table = metal1_rho_t_table {
      property = rho_t
      table_type = R
      dim_type = actual
      value_type = absolute
      width = {0.065 0.08 0.14}
      thickness = {0.04 0.06 0.15 2.5}
      value = {0.036 0.032  0.03  0.028,
               0.035 0.0318 0.029 0.0278,
               0.034 0.0315 0.028 0.0276}
   }

   table = metal1_thickness_table {
      property = thickness
      table_type = RC
      dim_type = drawn
      value_type = delta
      width = {0.045 0.08 0.2 2.5}
      spacing = {0.045 0.08 0.2 2.5}
      value = {0.092 0.094 0.097 0.098,
               0.095 0.096 0.098 0.099,
               0.093 0.093 0.122 0.096,
               0.097 0.0965 0.094 0.0921}
   }

   table = metal1_multi_bias_table {
      property = multi_bias
      table_type = RC
      dim_type = drawn
      value_type = delta
      width1 = {0.045 0.08 0.14 0.2}
      spacing1 = {0.045 0.1 0.15 0.25}
      value1 = { -0.002 0.004 0.015 0.02,
                -0.004 0.002 0.012 0.018,
                -0.006 -0.002 0.01 0.016,
                -0.01 -0.004 -0.002 -0.002}
      width2 = {0.046 0.085 0.2 1.0 3.2}
      spacing2 = {0.05 0.1 0.5 2.5}
      value2 = { 0.002 0.005 0.01 0.02,
               -0.001 0.001 0.008 0.018,
               -0.003 -0.001 0.005 0.017,
               0.0005 0.0007 0.001 0.0015,
               0.002 0.0022 0.0022 0.01}
   }
}
```

Multiple Via Type Example

This is an example of how to specify multi-via types such as square and rectangular vias using vwidth and vlength tables defined in an MIPT file.

Given the following via dimensions for Via1:

```
         Drawn dimension            Actual dimension
        [ width x length ]         [ width x length ]

Via1    0.04725 x 0.04725           0.08022 x 0.08022
Via1    0.04725 x 0.12285           0.05145 x 0.12705
```

In the MIPT file specify the rectangular via using vwidth and vlength tables:

```
table = vwidth_table {
   property = vwidth
   table_type = RC
   dim_type = drawn
   value_type = absolute
   width = {0.04725}
   length = {0.04725 0.12285}
   value = {0.08022 0.05145}
}

table = vlength_table {
   property = vlength
   table_type = RC
   dim_type = drawn
   value_type = absolute
   width = {0.04725}
   length = {0.04725 0.12285}
   value = {0.08022 0.12705}
}
```

Resistance Table For Square and Rectangular Contacts Example

This is an example of how to specify multiple-sized square and rectangular contacts using vresistance, vwidth, and vlength tables defined in an MIPT file. The vresistance, vwidth, and vlength tables can also be used in via layer definitions.

The following contact dimensions:

```
            Drawn dimension            Actual dimension
           [ width x length ]         [ width x length ]

contact    0.04725 x 0.04725           0.08022 x 0.08022
contact    0.12285 x 0.12285           0.12705 x 0.12705
```

can be used to define both square and rectangular contacts with the dimensions:

```
square contacts:  0.04725 x 0.04725
                  0.12285 x 0.12285

rectangular contacts:   0.04725 x 0.12285
                        0.12285 x 0.04725
```

This is an example of a contact layer defined in an MIPT file:

```
contact = pcont {
   measured_from = p_od
   measured_to = m1
   max_length = 0
   max_area = 0.016
   tc1 = 0
   tc2 = 0
   min_width = 0.03
   min_spacing = 0.03
   resistance = 363.023
   enclosure_down = 0
   enclosure_up = 0

   table = resistance_table {
      property = vresistance
      table_type = R
      dim_type = drawn
      value_type = absolute
      width = {0.04725 0.12285}
      length = {0.04725 0.12285}
      value = {200 190,
               190 100}
   }

   table = vwidth_table {
      property = vwidth
      table_type = RC
      dim_type = drawn
      value_type = absolute
      width = {0.04725 0.12285}
      length = {0.04725 0.12285}
      value = {0.08022 0.05145
               0.12705 0.12705}
   }

   table = vlength_table {
      property = vlength
      table_type = RC
      dim_type = drawn
      value_type = absolute
      width = {0.04725 0.12285}
      length = {0.04725 0.12285}
      value = {0.08022 0.12705
               0.05145 0.12705}
   }
}
```

Inner Fringe Capacitance (CFI) Example

Use the ignore_gate_to_diff gate_fringe table parameter to control whether the capacitance values provided in the CF tables are added to or subtracted from the gate to source-drain capacitance. The CF values come from the SPICE model parameters. Specify different gate_fringe tables for different MOS transistor types. The tables can consist of a single value if that is what corresponds to the SPICE device model.

In the MIPT file, to subtract CF table values from the gate to source-drain capacitance during gate to diffusion capacitance extraction, set ignore_gate_to_diff table parameter to no. For example:

```
seed = my_gate {
   thickness = 0.031
   min_width = 0.018
   min_spacing = 0.060
   r_sheet = 55
   layer_bias = 0.0063
   tc1 = 2.2e-4
   tc2 = 0.0
   devices = { my_device }
   ignore_gate_to_diff = no
   top_thickness = 0.014
   top_spacing = 0.020
   top_enclosure = 0.015
   table = my_gate_fringe_table {
      property = gate_fringe
      table_type = C
      dim_type = drawn
      value_type = absolute
      co_co_spacing = {0.01}
      gate_co_spacing = {0.01}
      value = {0.08022 0.05145}
   } # end table
} # end seed layer definition
```

Chapter 6
MIPT Keyword Dictionary

This chapter contains descriptions for the MIPT keywords and parameters used to create a MIPT file. These appear in alphabetical order.

airgap

Parameter for layer(s): dielectric, conductor. device_li, diffusion. li, pcaux, poly, resistor, seed, src_drn

An optional conformal dielectric parameter that specifies the airgap parameters for dielectric and conductor layers. The mingap value is the largest spacing value.

Syntax

airgap = {**rectangle** | **triangle**} **min_gap (*value*) spacing (*s1* ... *sn*) width (*w1* ... *wn*) thickness (*t1* ... *tn*) dist_from_surface (*b1* ... *bn*)**

Parameters

- **rectangle** | **triangle**

  A required parameter that specifies the shape of the air gap.

- **min_gap (*value*)**

  A required keyword value set that specifies the threshold value for the spacing between conductors. If the spacing between conductors is less than the min_gap value, then an air gap (dielectric void) occurs. Conductors further apart than min_gap do not have air gaps. The ***value*** must be enclosed in parentheses.

- **spacing (*s1* ... *sn*)**

  A required keyword value set of floating point values that must be greater than or equal to zero and enclosed in parentheses. Used to define the spacing between conductors as shown in Figure 6-1 and Figure 6-2, or the spacing between thin conformal layers with no air gaps as shown in Figure 6-3 and Figure 6-4. Values must be specified in ascending order.

- **width (*w1* ... *wn*)**

  A required keyword value set of floating point values that must be greater than or equal to zero and enclosed in parentheses. Used to define the width of the air gap at its base.

- **thickness (*t1* ... *tn*)**

  A required keyword value set of floating point values that must be greater than or equal to zero and enclosed in parentheses. Used to define the height of the rectangle or triangle.

- **dist_from_surface (*b1* ... *bn*)**

  A required keyword value set of floating point values that must be enclosed in parentheses. Used to define the distance to the base of the air gap from the bottom surface of the dielectric in which the air gap is defined.

Description

The spacing between conductors is shown in Figure 6-1 and Figure 6-2.

Figure 6-1. Rectangular Airgap Parameters

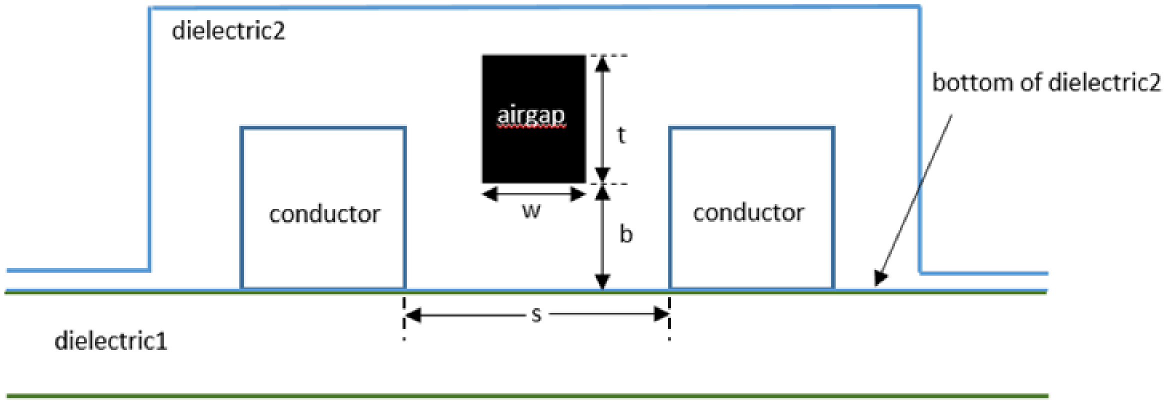
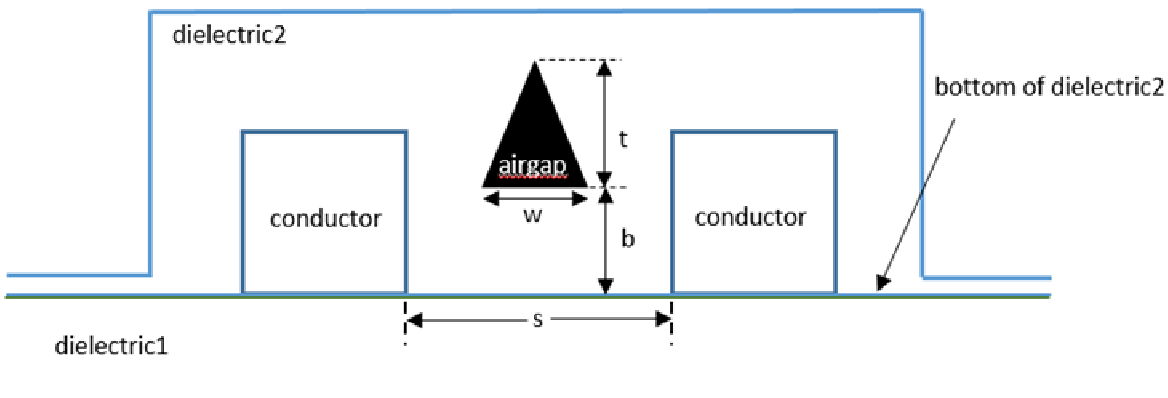


Figure 6-2. Triangular Airgap Parameters



The air gap measurements with thin conformal layers with no air gaps as is shown in Figure 6-3 and Figure 6-4.

Figure 6-3. Airgap With Thin Conformal Dielectric Layer

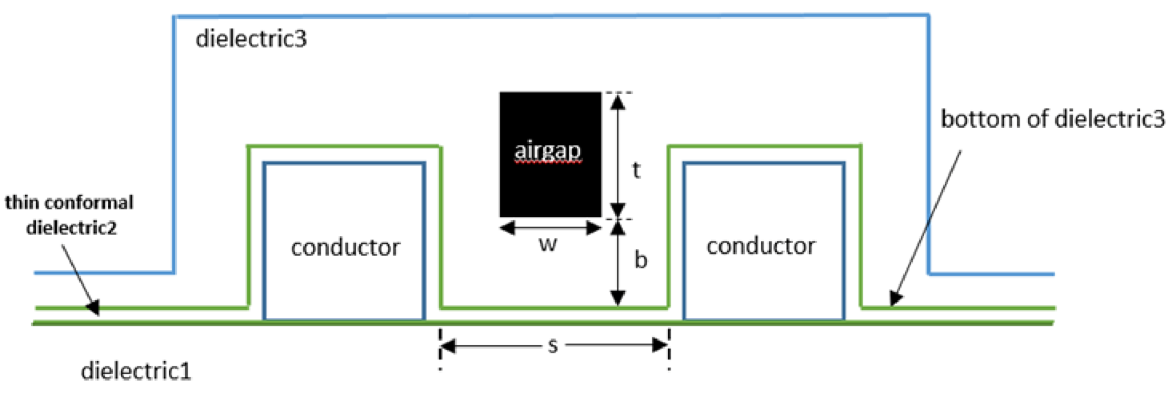
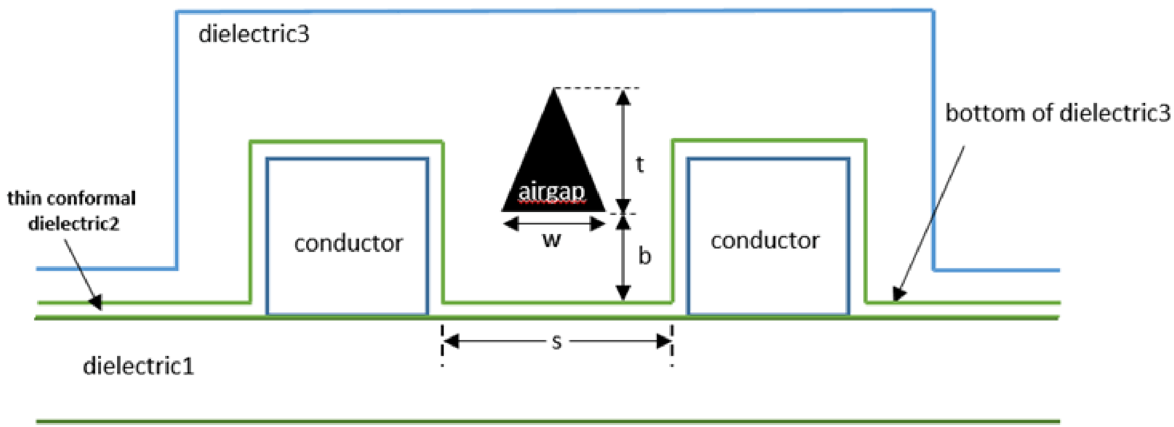


Figure 6-4. Triangular Airgap With Thin Conformal Dielectric Layer



The distance from surface parameter, b, is a negative value when the air gap falls below the bottom of the dielectric in which the air gap is defined as shown in Figure 6-5 and Figure 6-6.

Figure 6-5. Airgap With Negative Base Measurement

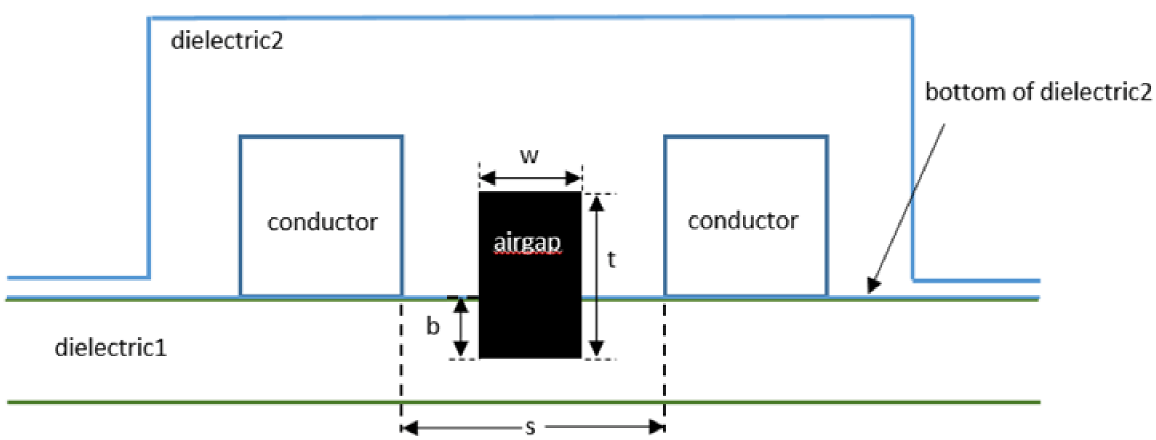
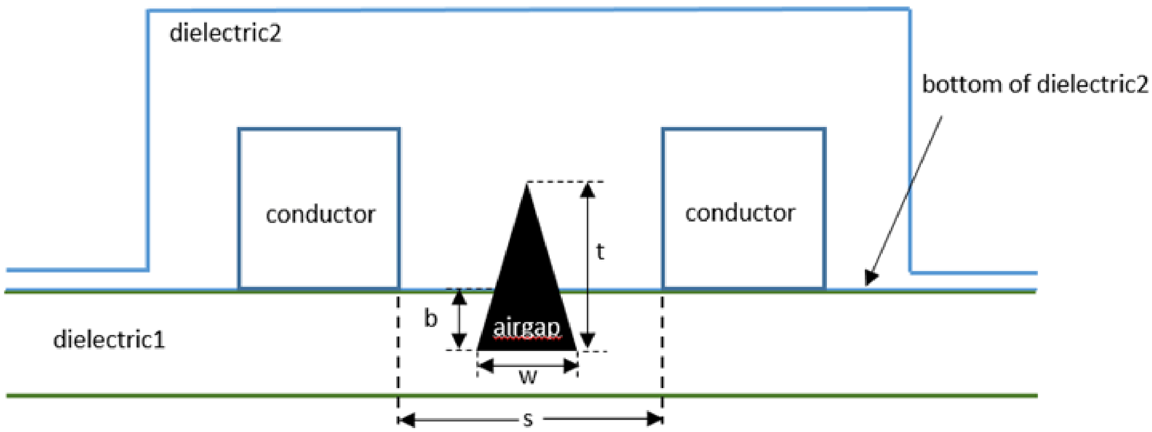


Figure 6-6. Triangular Airgap With Negative Base Measurement



Examples

Example 1

In the following example, air gaps form at spacings below min_gap (0.3), but no air gaps exist for spacings larger than min_gap (0.3):

```
airgap = rectangle min_gap (0.3) spacing (0.15 0.2) width (0.075 0.08)
thickness (0.05 0.06) dist_from_surface (0.04 0.05)
```

Example 2

In this example, the air gaps fall below the bottom of the dielectric:

```
airgap = rectangle min_gap (0.5) spacing (0.38 0.4) width (0.22 0.27)
thickness (0.62 0.63) dist_from_surface ( -0.15 -0.28)
```

area

Parameter for layer(s): base_via, contact, via

A required parameter used by contact and via layers that specifies the area of contact in distance_units squared. This parameter is required if min_width is not specified. The shape of the contact or via is assumed to be square, with dimensions of sqrt(area) * sqrt(area).

Syntax

area = *value*

Parameters

- ***value***

  A value that specifies the area of contact or via in distance_units squared.
  area = min_width * min_width.

Examples

```
area = 0.25
```

author

Type: Global Parameters

An optional global parameter used to specify the author of the MIPT file and other related author information. The specified string is added to the beginning of the calibrated rule file during calibration. This parameter can only be specified once. If it is a single string without spaces, the quotation marks are not required. If the string is enclosed in quotes, spaces are allowed.

Syntax

author = *author_info*

Parameters

- *author_info*

  A string that specifies the author of the MIPT file and other related author information.

Examples

Example 1

```
author = Stack_Master
```

Example 2

```
author = “Stack Master”
```

background_dielectric

Type: Global Parameters

An optional global parameter whose real number or integer value specifies the dielectric constant (permittivity). It can only be specified once.

Syntax

background_dielectric = *value*

Parameters

- *value*

  A real number or integer value that specifies the dielectric constant (permittivity). It can only be specified once. The default is 1.0.

Examples

```
background_dielectric = -0.5
```

base

Type: Base layer

A layer definition keyword used to define a base reference plane layer.

Syntax

base = ***layer_name*** **{**
  thickness = ***layer_thickness***
  ztop = *top_z-coordinate*
  zbottom = *bottom_z-coordinate*
  bulk_min_width = *value*
  bulk_resistance = *value*
  hidden = <u>false</u> | true
  }

Parameters

- ***layer_name***

  A required unique user-specified name.

- **thickness**

  A required layer parameter that specifies the thickness of the base layer in the z-direction.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- bulk_min_width

  An optional layer parameter that specifies the minimum width of the bulk layer in microns.

- bulk_resistance

  An optional layer parameter that specifies the sheet resistance of the bulk layer in ohms.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Description

Part of the layer definition syntax, this layer definition keyword is used to optionally define a base reference plane layer. The base keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line.

It is possible to have a single base plane or multiple base layers. A single base plane is used to specify a reference from which all dimensions are defined. Multiple base layers are used in a technology stack where each base layer definition represents a different possible ground plane height. For more information on base layers, see Base.

Examples

```
base = my_base_layer {
   thickness = 1
   ztop = 0
}
```

base_via

Type: Base_via layer

Part of the layer definition syntax, this optional layer definition keyword is used to define via properties for the contact to base layer (substrate tap). Use this layer type to model the tap resistance of a via to well or substrate layers. Layers of this type can be specified multiple times in the MIPT file.

Syntax

base_via = ***layer_name* {**
measured_from = ***layername*** # base layer_name
measured_to = ***layername*** # upper diffusion layer_name
area = ***area_of_via***
min_width = ***minimum_allowed_metal_width***
min_spacing = ***minimum_allowed_metal_spacing***
resistance = ***resistance_per_via***
enclosure_down = ***enclosure_with_respect_to_lower_conductor***
enclosure_up = ***enclosure_with_respect_to_upper_conductor***
max_area = *maximum_via_area*
max_length = *maximum_distance_between_vias*
min_length = *length_of_shape*
tc1 = *resistance_temperature1_coefficient*
tc2 = *resistance_temperature2_coefficient*
extra_width = *extra-width_of_side_walls*
extrapolation = yes | no
ignore_caps = yes | no
ronly_layers = '{'*space_delimited_list_of_r-only_layernames*'}'
hidden = false | true
trim = no | yes | actual | merge | bridge
interpolation = gperarea | invgperarea
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- **measured_from**

  A required parameter that specifies the base layer_name.

- **measured_to**

  A required parameter that specifies the upper diffusion layer_name.

- **area**

  A required parameter if min_width is not specified. Specifies the area of via in distance_units squared. This parameter cannot be specified with min_width parameter.

- **min_width**

  A required parameter that specifies the minimum allowed metal width if area parameter is not specified. This parameter cannot be specified with area parameter. The shape is assumed to be square, with dimensions of min_width * min_width.

- **min_spacing**

  A required parameter that specifies the minimum allowed metal spacing.

- **resistance**

  A required parameter that specifies the resistance per via or contact. The value is specified in Ohms.

- **enclosure_down**

  A required parameter that specifies enclosure with respect to a lower conductor. This value is typically 0, and defaults to 0 if not specified.

- **enclosure_up**

  A required parameter that specifies enclosure with respect to an upper conductor. This value is typically 0, and defaults to 0 if not specified.

- max_area

  An optional parameter that specifies the area threshold for large area via connections.

- max_length

  An optional parameter that specifies the maximum distance between distributed vias.

- min_length

  An optional parameter that specifies the length of the shape. This parameter may be used together with min_width parameter, but cannot be used with area parameter. The shape is assumed to be rectangular, with dimensions of min_width * min_length.

- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.

- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.

- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- extrapolation

  An optional parameter that specifies whether or not to extrapolate the variable via_resistance table parameters for the layer.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

- trim

  An optional parameter that controls how resistance is calculated for the via layer. The default is yes, if this parameter is not specified.

- interpolation

  An optional parameter that controls how the area-based tables interpolate resistance values for fixed or variable sized vias. The default is gperarea.

Examples

This example specifies a base_via layer called *n_welltap* with a resistance of 1e-06 ohms between base layer *mybase* and diffusion layer *diff1*:

```
base_via = n_welltap {
   measured_from = mybase
   measured_to = diff1
   resistance = 1e-06
   enclosure_up = 0
   enclosure_down = 0
   interpolation = invgperarea
   min_width = 0.01
   min_spacing = 0.01
}
```

bias_type

table structure parameter

An optional table structure parameter used to specify whether the actual width values described in the table are for the preferred or non-preferred direction. This parameter may only be used in two-dimensional width or multi_bias property tables.

Syntax

bias_type = preferred | nonpreferred

Parameters

- preferred

  A keyword used to specify that the actual width parameters described in this table are for the preferred direction.

- nonpreferred

  A keyword used to specify that the actual width parameters described in this table are for the non-preferred direction.

Description

An optional table structure parameter that specifies whether the actual width values described in the table are for the preferred or non-preferred direction.

Both a preferred and a nonpreferred table are required for a particular property in the layer definition. If one is missing, then calibration stops with the following error:

```
ERROR: Missing preferred/nonpreferred table for layer layername
```

Once you have specified the preferred and nonpreferred tables for a given property, additional tables for that property type are not permitted. For information on orientation-based width variation see Retargeting.

Examples

Example 1

```
bias_type = preferred
```

Example 2

The following is an example of a two dimensional WIDTH table specified with bias_type:

```
poly = my_poly2 {
       thickness = 0.25
       table = my_table5 {
         property = width
         table_type = RC
         dim_type = drawn
         value_type = absolute
         bias_type = preferred
         width1 = {0.027 0.045 0.068 0.125 0.2 0.5}
         spacing1 = {0.072 0.081 0.09 0.135 0.18 0.3 0.4 0.6 1.2}
         value1 = {0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 ,
                   0.048 0.048 0.048 0.048 0.048 0.048 0.048 0.048 0.048}
         width2 = {0.027 0.045 0.068 0.125 0.2 0.5}
         spacing2 = {0.072 0.081 0.09 0.135 0.18 0.3 0.4 0.6 1.2}
         value2 = {0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 ,
                   0.048 0.048 0.048 0.048 0.048 0.048 0.048 0.048 0.048}
      }
       table = my_table6 {
         property = width
         table_type = RC
         dim_type = drawn
         value_type = absolute
         bias_type = nonpreferred
         width1 = {0.027 0.045 0.068 0.125 0.2 0.5}
         spacing1 = {0.072 0.081 0.09 0.135 0.18 0.3 0.4 0.6 1.2}
         value1 = {0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 ,
                   0.048 0.048 0.048 0.048 0.048 0.048 0.048 0.048 0.048}
         width2 = {0.027 0.045 0.068 0.125 0.2 0.5}
         spacing2 = {0.072 0.081 0.09 0.135 0.18 0.3 0.4 0.6 1.2}
         value2 = {0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 ,
                   0.048 0.048 0.048 0.048 0.048 0.048 0.048 0.048 0.048}
      }
}
```

bot_enclosure

Parameter for layer(s): TSV

An optional TSV layer parameter that specifies the bottom enclosure.

Syntax

bot_enclosure = *value*

Parameters

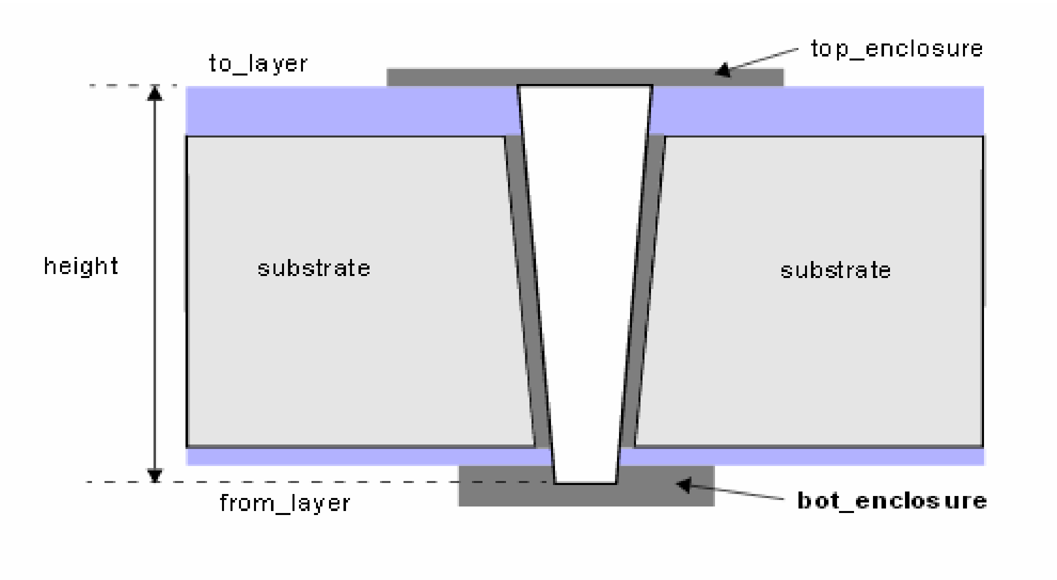
- *value*

  A floating point value that specifies the bottom enclosure covered by the metal "from_layer". The default is 0 if you do not specify this parameter.

Description

An optional parameter that specifies the bottom enclosure covered by the metal "from_layer" for the TSV. This parameter is used only in TSV layer definitions. The bot_enclosure is shown in Figure 6-7.

Figure 6-7. bot_enclosure for TSV Layer



For more information on the TSV layers, see TSV.

Examples

```
bot_enclosure = 0.34
```

botthk

Parameter for layer(s): dielectric

An optional dielectric layer parameter that specifies the conformal coating thickness on the bottom of a conductor.

Syntax

botthk = *value*

Parameters

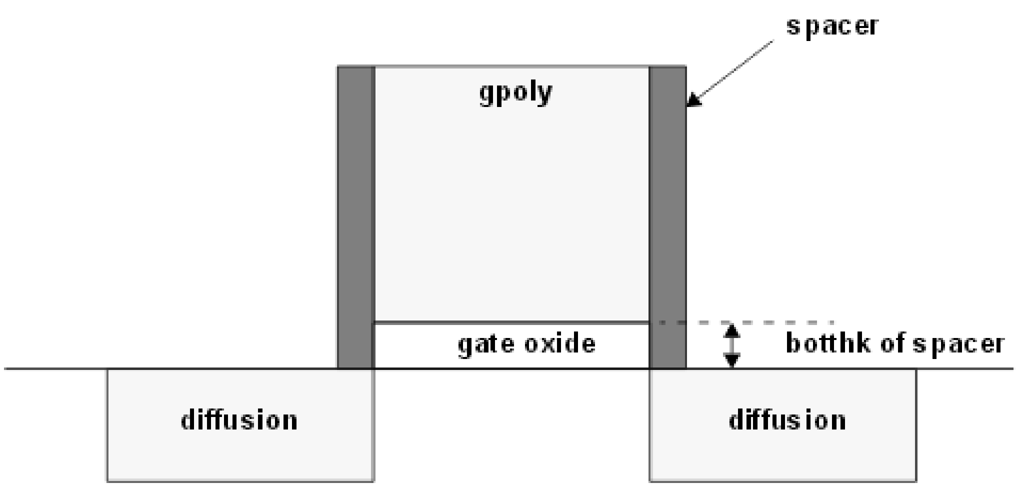
- *value*

  A positive or negative floating point value that specifies the conformal coating thickness on the bottom of a conductor.

Description

Use this dielectric parameter when diel_type is set to conformal, trench, or spacer. Allows the bottom for the conformal to be moved down by the amount specified for botthk. The botthk measurement is shown in Figure 6-8.

Figure 6-8. botthk Parameter for a Spacer Dielectric



See Dielectric for more information on the dielectric layer type.

Examples

```
botthk = 0.02
```

bottom_thickness

Parameter for layer(s): contact

An optional parameter used by diffusion contact layers that specifies the lower section of the diffusion contact.

Syntax

bottom_thickness = *value*

Parameters

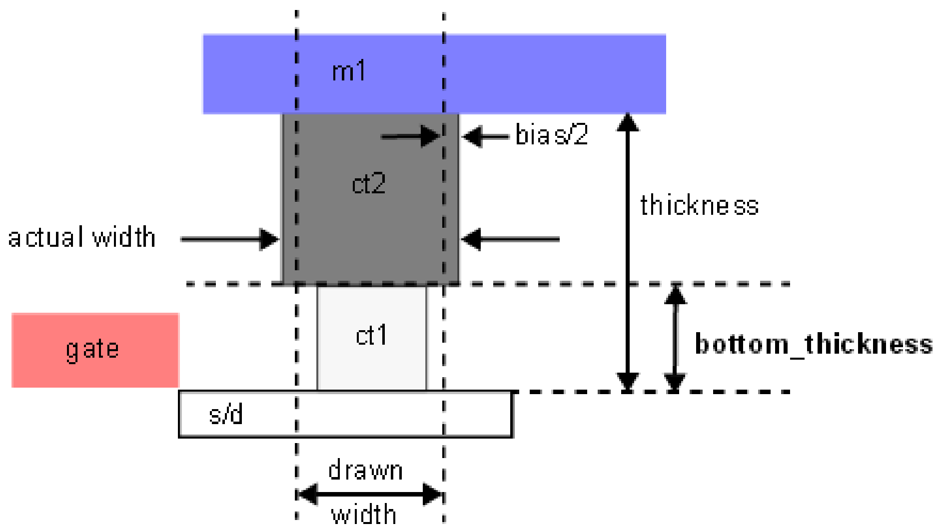
- *value*

  A positive floating point value that specifies the lower section of the diffusion contact.

Description

The optional bottom_thickness parameter is used to represent the lower section of the diffusion contact as seen in Figure 6-9.

Figure 6-9. bottom_thickness Parameter for Contact Layer



See Contact for more information on the contact layer type.

Examples

```
bottom_thickness = 0.15
```

bulk_min_width

Parameter for layer(s): base, substrate, tap, well

An optional layer parameter that specifies the minimum width of the bulk layer. It can be specified for base, substrate, tap, and well layer definitions.

Syntax

bulk_min_width = *value*

Parameters

- *value*

  A non-negative floating point value used to specify the minimum width in microns.

Examples

Given the following layer definitions:

```
substrate = nsub {
   zbottom = -50
   ztop = -0.7
   resistivity = 101800
   eps = 11.9
   bulk_resistance = 100
   bulk_min_width = 0.5
}

well = nwell {
   zbottom = -0.7
   thickness = 0.7
   resistivity = 100
   eps = 11.9
   bulk_resistance = 10
   bulk_min_width = 0.5
}

tap = ntap {
   zbottom = 0
   ztop = 0.01
   resistivity = 0.1
   bulk_resistance = 2
   bulk_min_width = 0.1
}
```

calibration adds the following SVRF statements to the calibrated rule file:

```
PEX RESISTANCE PARAMETERS XCALIBRATE_nsub BULKRESISTANCE 100 \
   BULKMINWIDTH 0.5
PEX RESISTANCE PARAMETERS XCALIBRATE_nwell BULKRESISTANCE 10 \
   BULKMINWIDTH 0.5
PEX RESISTANCE PARAMETERS XCALIBRATE_ntap BULKRESISTANCE 2 \
   BULKMINWIDTH 0.1
```

bulk_resistance

Parameter for layer(s): base, substrate, tap, well

An optional layer parameter that specifies the sheet resistance of the bulk layer. It is used to calculate resistance between the bulk pin of a transistor and the nearest substrate or well connection. This parameter can be specified for base, substrate, tap, and well layer definitions.

Syntax

bulk_resistance = *value*

Parameters

- *value*

  A non-negative floating point value used to specify the sheet resistance in ohms.

Examples

Given the following layer definitions:

```
base = base_b {
   zbottom = -1
   thickness = 1
   bulk_resistance = 15
}
```

calibration adds the following SVRF statement to the calibrated rule file:

```
PEX RESISTANCE PARAMETERS XCALIBRATE_base_b BULKRESISTANCE 15
```

calibration_type

Type: Global Parameters

A required global parameter declared at the beginning of your MIPT file that specifies the type of calibration used.

Syntax

calibration_type = {**RONLY** | **FIELDSOLVER** | **RULEBASED** | **ALL**}

Parameters

- **RONLY**

  A keyword which specifies that the MIPT file is only intended for RONLY calibrations. When doing an RONLY calibration, only the technology header and resistance rules are calibrated. It may not be suitable for other calibrations due to insufficient data or accuracy of data.

- **FIELDSOLVER**

  A keyword which specifies that the MIPT file is only intended for FIELDSOLVER calibrations. When doing a FIELDSOLVER calibration, only the technology header and resistance rules are calibrated. It may not be suitable for other calibrations due to insufficient data or accuracy of data.

- **RULEBASED**

  A keyword which specifies that the MIPT file is only intended for RULEBASED calibrations. This type produces the same output as is currently expected for traditional calibrations. It may not be suitable for other calibrations due to insufficient data or accuracy of data.

- **ALL**

  A keyword which specifies that the MIPT file is intended for ALL calibrations. It may be used for other calibrations as specified on the command line.

Description

This required global parameter specifies the type of calibration used or possible uses of the MIPT file. It can only be specified once. This parameter accepts a list of values or a single value to indicate the possible uses of a specific MIPT file. Possible calibration types are RONLY, FIELDSOLVER, RULEBASED and ALL. Multiple values must be separated by a space and enclosed in braces ({ }). For more information on global parameters, see Global Parameters.

Examples

Example 1

```
calibration_type = RONLY
```

Example 2

```
calibration_type = {RONLY RULEBASED}
```

cap_unit

Type: Global Parameters

An optional global parameter that specifies the units for capacitance values.

Syntax

cap_unit = *unit*

Parameters

- *unit*

  A value that specifies the units for capacitance values.

  Acceptable values are:

  af — attofarads

  ff — femtofarads (default)

  pf — picofarads

  nf — nanofarads

  uf — microfarads

  mf — millifarads

  f — farads

  The default is femtofarads.

Examples

```
cap_unit = pf
```

capacitive_only_etch

Parameter for layer(s): conductor, device_li, pad, poly

An optional parameter, used for conductor, device_li, pad, and poly layer definitions, that specifies the layer bias for capacitance only.

Syntax

capacitive_only_etch = *value*

Parameters

- *value*

  A floating point value that specifies the layer bias for capacitance only.

Description

This parameter can be specified for Conductor, Device_li, Pad, and Poly layer definitions. It is equivalent to the capacitive_only_etch found in the standardized Interconnect Technology Format (ITF).

Layer_bias applies to both resistance and capacitance. When capacitive_only_etch is specified, the layer_bias applies to capacitance only and the value of layer_bias is determined using the following equation:

layer_bias = -2 * capacitive_only_etch

Figure 6-52 in “layer_bias” shows the layer_bias measurement.

Examples

```
device_li = CA {
   thickness = 0.059
   min_width = 0.026
   min_spacing = 0.051
   capacitive_only_etch = -0.0057
}
```

channel_er

Parameter for layer(s): multigate

An optional multigate layer parameter used to specify the difference in permittivity of the gate oxide and its lateral side.

Syntax

channel_er = *value*

Parameters

- *value*

  A floating point value that specifies the difference in permittivity of the gate oxide and its lateral side.

Description

An optional multigate layer parameter that specifies the difference in permittivity of the gate oxide and its lateral side. This parameter is used only in multigate layer definitions. For more information on the multigate layer definition, see Multigate.

Examples

```
channel_er = 8.0
```

comment

Type: Global Parameters

An optional global parameter used to specify a comment string that is placed in the calibrated rule file.

Syntax

comment = *comment_info*

Parameters

- *comment_info*

  A string that describes comments and other information for the manufacturing technology (process) being used.

Description

An optional global parameter used to create comments and other information for the manufacturing technology (process) being used. Comments specified with the comment parameter are placed in the calibrated rule file directly below the command line comment separated with a blank line. They are located towards the top of the un-encrypted section of the calibrated rule deck.

The comment parameter can only be specified once. If it is a single string without spaces, the quotation marks are not required. If the string is enclosed in quotes, spaces are allowed. If the description requires multiple lines, then it should be enclosed in braces ({ }).

For more information on global parameters, see Global Parameters.

Examples

Example 1

```
comment = MIPT_2_0_ekit_file
```

Example 2

```
comment = “MIPT 2.0 ekit file”
```

Example 3

```
comment = {
            MIPT 2.0
            ekit
            file
}
```

conductor

Type: Conductor layer

A layer definition keyword used to define a conductor layer.

Syntax

conductor = ***layer_name* {**
thickness = ***z_direction_metal_thickness***
min_width = ***minimum_allowed_metal_width***
min_spacing = ***minimum_allowed_metal_spacing***
resistivity = ***resistivity_value***
r_sheet = ***sheet_resistance***
metal_fill = '{'*fill_ratio fill_spacing fill_width* [floating | grounded]'}'
thickness_type = <u>absolute</u> | relative
ztop = *top_ z-coordinate*
zbottom = *bottom_z-coordinate*
measured_from = *layername*
airgap = {rectangle | triangle} min_gap (*v*) spacing (*s1* ... *sn*) width (*w1* ... *wn*)
thickness (*t1* ... *tn*) dist_from_surface (*b1* ... *bn*)
coplanar_min_spacing = *value*
tc1 = *resistance_temperature1_coefficient*
tc2 = *resistance_temperature2_coefficient*
max_rlength = *length_for_resistance_fracturing*
max_width = *maximum_allowed _metal_width*
max_spacing = *maximum_allowed_metal_spacing*
maxwidth_for_minspacing = *value*
devices = '{'$device_1$ $device_2$ ... $device_n$'}' # a space delimited list of device names
layer_bias = *bias_actual_width_relative_to_drawn_width*
capacitive_only_etch = *layer_bias_for_capacitance_only*
resistive_only_etch = *layer_bias_for_resistance_only*
cond_type = nonplanar
trap_style = top | middle | bottom # where trapezoid measurements are taken from
extra_width = *extra-width_of_side_walls* # for a conductor with a trapezoid shape
swstep = *count*
extension = *amount_layer_extends_past_diffusion_or_contact_layers*
ignore_caps = yes | <u>no</u>
ronly_layers = '{'*space_delimited_list_of_r-only_layer_names*'}'
density_window = '{' '{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'...'{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}''}'
widths = '{' w_1 w_2 w_3 ... w_n '}'
spacings = '{' s_1 s_2 s_3 ... s_n '}'
hidden = <u>false</u> | true
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- **thickness**

  A required layer parameter that specifies the thickness of the metal layer in the z-direction.

- **min_width**

  A required layer parameter that specifies the minimum allowed drawn width of metal objects on this layer.

- **min_spacing**

  A required layer parameter that specifies the minimum allowed drawn spacing between conductors on this layer.

- **resistivity**

  A required layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- **r_sheet**

  A required layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- metal_fill

  An optional conducting layer parameter. It is a set of values enclosed in braces, used to define a list of virtual fill parameters.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- measured_from

  An optional conducting layer parameter that measures zbottom or ztop relative to the specified layer.

- airgap

  An optional parameter that specifies the airgap parameters for the conductor layer.

- coplanar_min_spacing

  An optional conducting layer parameter that specifies the minimum spacing value between coplanar layers.
- tc1

  An optional conducting layer parameter that specifies the resistance temperature1 coefficient.
- tc2

  An optional conducting layer parameter that specifies the resistance temperature2 coefficient.
- max_rlength

  An optional conducting layer parameter that specifies the maximum length of a wire. This parameter is used for resistance fracturing, which determines how many pieces a resistor should be broken into for representation in a distributed network.
- max_width

  An optional parameter that specifies the maximum allowed metal width.
- max_spacing

  An optional parameter that specifies the maximum allowed metal spacing.
- maxwidth_for_minspacing

  An optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
- devices

  An optional layer parameter that specifies a space delimited list of device names enclosed in braces. This parameter can only be specified in the conductor layer definition that is associated with a device. Specifying the devices parameter for conductor metal layers that are not part of a device is not allowed.
- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width relative to drawn width.
- capacitive_only_etch

  An optional parameter that specifies the layer bias for capacitance only.
- resistive_only_etch

  An optional parameter that specifies the layer bias for resistance only.
- cond_type

  An optional parameter that specifies the conductor layer is non-planar. Nonplanar is the only permitted value.

- trap_style

  An optional conducting layer parameter that defines where trapezoid measurements are taken from. Permitted values are top, middle, or bottom.

- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- swstep

  An optional conducting layer parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.

- extension

  An optional conducting layer parameter that specifies the amount this layer extends past diffusion or contact layers.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- density_window

  An optional conducting layer parameter specified as {{w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$} ... {w_n h_n $factor_n$ $stepx_n$ $stepy_n$}}. The braces are required for grouping.

- widths

  An optional conducting layer parameter that specifies a space delimited list of floating point width values enclosed in braces ({ }).

- spacings

  An optional conducting layer parameter that specifies a space delimited list of floating point spacing values enclosed in braces ({ }).

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Description

Part of the layer definition syntax, this layer definition keyword is used to define the electrical and physical characteristics of conductor layers. The conductor keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line.

Conductor layers are conducting layers that are also referred to as metal layers. You are required to specify at least one conducting layer in your MIPT file. Layers of this type can be specified multiple times. For more information on conductor layers, see Conductor.

Examples

```
conductor = M1 {
   ztop = 0.0
   measured_from = poly_diel7
   thickness = 0.09
   min_width = 0.045
   min_spacing = 0.045
   resistivity = 0.0366
   tc1 = use my_Mx_layers_tc1
   tc2 = use my_Mx_layers_tc2
}
```

cond_type

Parameter for layer(s): conductor

An optional conductor layer parameter used to specify that the conductor layer is non-planar. Use only in MIPT files for Calibre xRC rule decks.

Syntax

cond_type = nonplanar

Parameters

- nonplanar

  A keyword used to specify that the conductor layer is non-planar.

Description

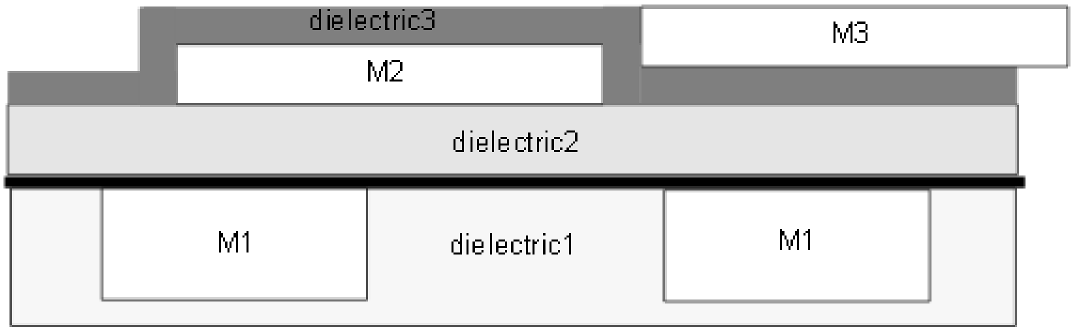
An optional conductor layer parameter used to specify that the conductor layer is non-planar. Use this parameter to define conductors sitting on top of non-planar or conformal dielectrics. The conductor layer is treated as a normal planar conductor, if this parameter is not specified.

This parameter is used only in conductor layer definitions. For more information on the conductor layer definition, see Conductor.

Examples

If a conductor, M3, is defined after a conformal dielectric, dielectric3, it is placed coplanar to the conductor the conformal is associated with, M2, as shown in Figure 6-10.

Figure 6-10. Coplanar M2 and M3



To properly place the conductor M3 over the conformal dielectric, dielectric3, as shown in Figure 6-11, specify the cond_type parameter set to non-planar in the definition of conductor M3. For example:

```
conductor = M3 {
   thickness = 0.24
   min_width = 0.13
   min_spacing = 0.12
   r_sheet = 0.38
   cond_type = nonplanar
}
```

Figure 6-11 shows an example of a conductor over conformal dielectric.

Figure 6-11. Conductor Over Nonplanar Dielectric

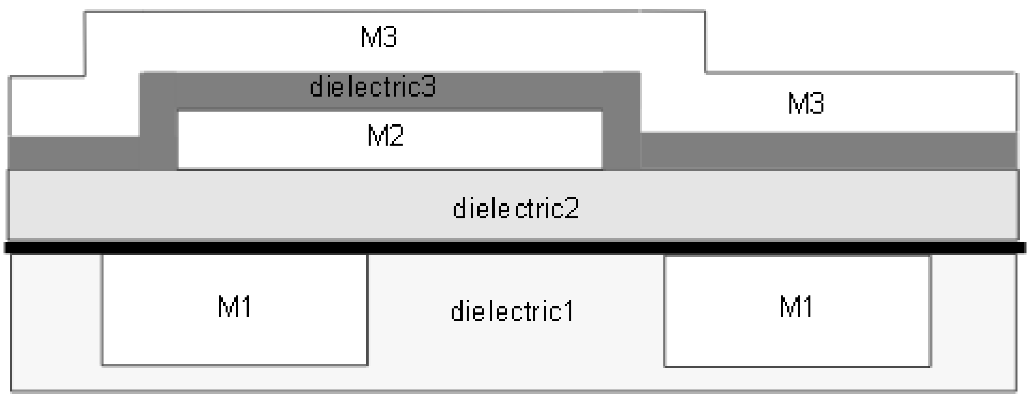
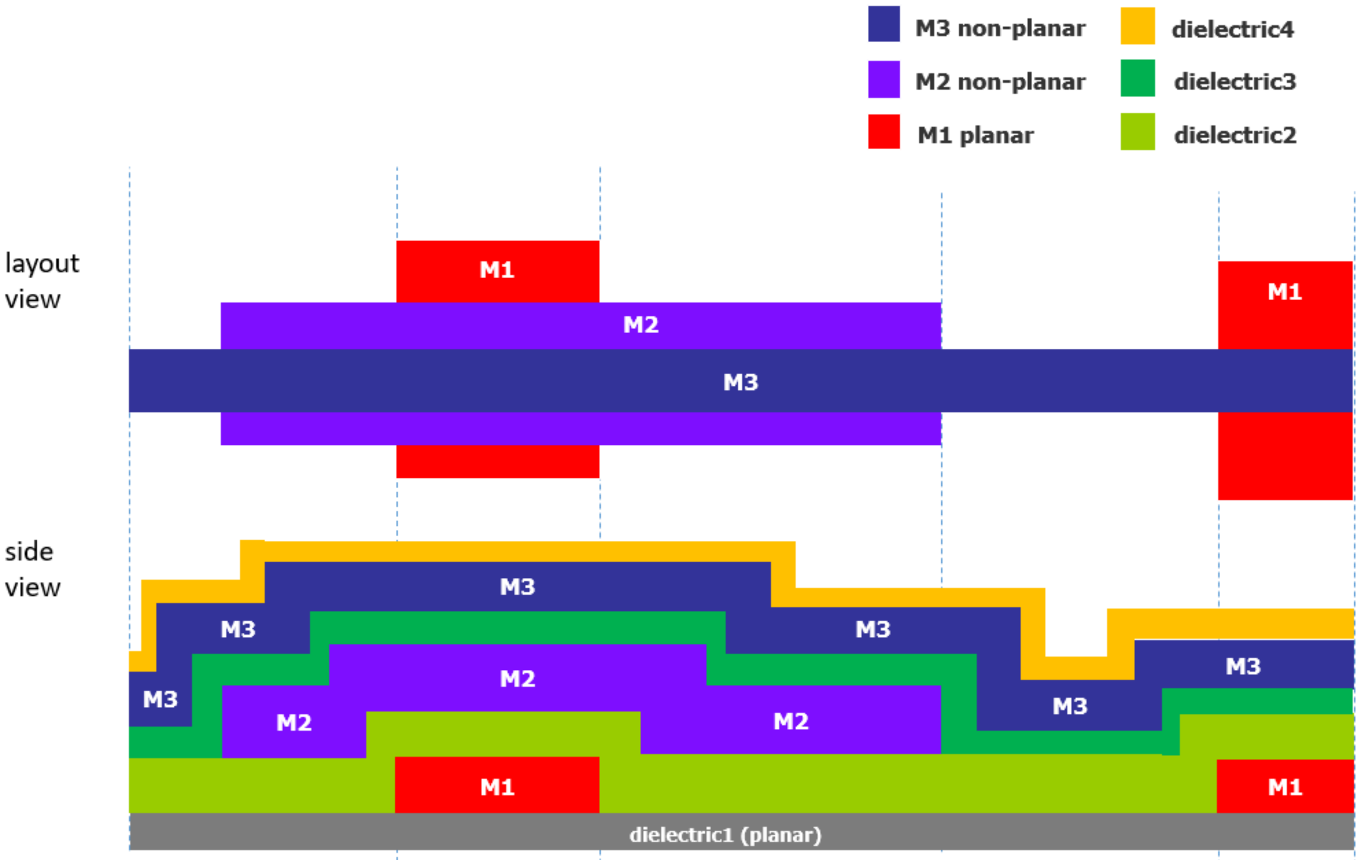


Figure 6-12 shows an example where more than one conductor is defined as non-planar. In this case, the conductor layer definitions for M2 and M3 both specify cond_type as nonplanar. For example:

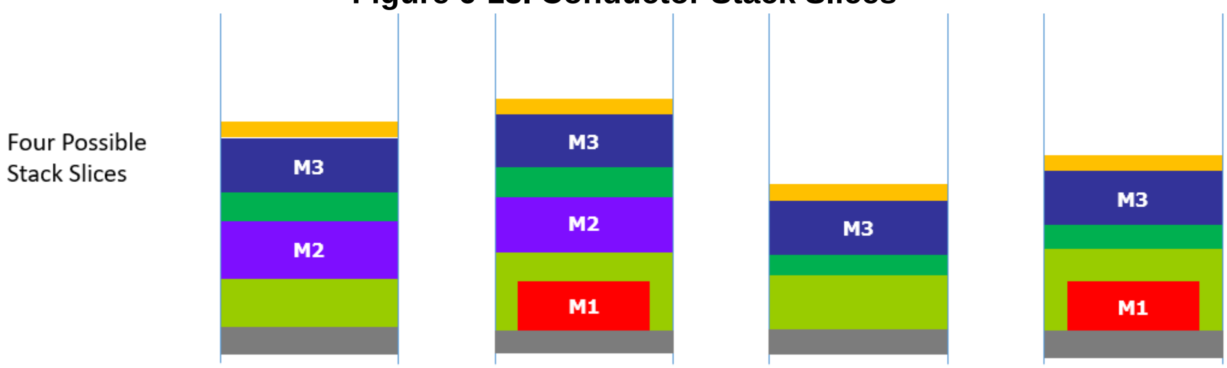
```
...
dielectric = dielectric1 {
   eps = 4.1
   thickness = 0.2
   diel_type = planar
}
conductor = M1 {
   thickness = 0.24
   min_width = 0.13
   min_spacing = 0.12
   r_sheet = 0.38
}
dielectric = dielectric2 {
   eps = 5.8
   thickness = 0.1
   diel_type = conformal
   swthk = 0.1
   topthk = 0.1
   ref_layer = M1
}
conductor = M2 {
   thickness = 0.24
   min_width = 0.13
   min_spacing = 0.12
   r_sheet = 0.38
   cond_type = nonplanar
}
dielectric = dielectric3 {
   eps = 5.8
   thickness = 0.1
   diel_type = conformal
   swthk = 0.1
   topthk = 0.1
   ref_layer = M2
}
conductor = M3 {
   thickness = 0.24
   min_width = 0.13
   min_spacing = 0.12
   r_sheet = 0.38
   cond_type = nonplanar
}
dielectric = dielectric4 {
   eps = 5.8
   thickness = 0.1
   diel_type = conformal
   swthk = 0.1
   topthk = 0.1
   ref_layer = M3
}
...
```

Figure 6-12. Multiple Nonplanar Layers



During calibration, a profile is generated for each possible conductor layer stack. The stack is sliced at different elevations for each conductor layer combination. This is referred to as stack slicing. In this example, there are four possible layer combinations as shown in Figure 6-13. A profile is generated for each of these stacks.

Figure 6-13. Conductor Stack Slices



contact

Type: Contact layer

A layer definition keyword used to define a contact layer.

Syntax

contact = *layer_name* {
measured_from = ***lower_conductor_layername***
measured_to = ***upper_conductor_layername***
area = ***area_of_contact***
min_width = ***minimum_allowed_metal_width***
min_spacing = ***minimum_allowed_metal_spacing***
resistance = ***resistance_per_via***
enclosure_down = ***enclosure_with_respect_to_lower_conductor***
enclosure_up = ***enclosure_with_respect_to_upper_conductor***
min_length = *length_of_shape*
bottom_thickness = *lower_section_value*
tc1 = *resistance_temperature1_coefficient*
tc2 = *resistance_temperature2_coefficient*
gate_to_cont_spacing_min = *minimum_allowed_gate_to_contact_spacing*
extra_width = *measurement_of_side_walls*
extrapolation = yes | no
ignore_caps = yes | no
layer_bias = *bias_actual_relative_to_drawn_contact_width&length*
ronly_layers = '{'*space_delimited_list_of_r-only_layernames*'}'
src_drn_contact_layers = '{'*space_delimited_list_of_srcdrn_contact_layernames*'}'
hidden = false | true
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- **measured_from**

  A required parameter that specifies the lower conductor layer_name.

- **measured_to**

  A required parameter that specifies the upper conductor layer_name.

- **area**

  A required parameter if min_width is not specified. Specifies the area of the contact in distance_units squared.

- **min_width**

  A required parameter that specifies the minimum allowed metal width if the area parameter is not specified. This parameter cannot be specified with the area parameter. The shape is assumed to be square, with dimensions of min_width * min_width.

- **min_spacing**

  A required parameter that specifies the minimum allowed metal spacing.

- **resistance**

  A required parameter that specifies the resistance per via or contact. The value is specified in Ohms.

- **enclosure_down**

  A required parameter that specifies enclosure with respect to a lower conductor. This value is typically 0, and defaults to 0 if not specified.

- **enclosure_up**

  A required parameter that specifies enclosure with respect to an upper conductor. This value is typically 0, and defaults to 0 if not specified.

- min_length

  An optional parameter that specifies the length of the shape. This parameter may be used together with min_width parameter, but cannot be used with area parameter. The shape is assumed to be rectangular, with dimensions of min_width * min_length.

- bottom_thickness

  An optional parameter that specifies the lower section of a diffusion contact. The value must be a positive floating point number.

- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.

- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.

- gate_to_cont_spacing_min

  An optional parameter that specifies the minimum allowed gate to contact spacing.

- extra_width

  An optional parameter that specifies the extra width of side walls for a conductor with a trapezoid shape.

- extrapolation

  An optional parameter that specifies whether or not to extrapolate the variable via_resistance table parameters for the layer.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width and length relative to drawn width and length for the contact.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- src_drn_contact_layers

  An optional contact layer parameter that specifies the layer or layers that make up the source/drain contact region of devices. Enclose the list in braces ({ }).

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Description

Part of the layer definition syntax, the contact layer definition keyword is used to define the electrical and physical characteristics of contact layers. The contact keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line.

Layers of this type may be specified multiple times. For more information on contact, see Contact.

Examples

```
contact = my_contact {
  resistance = 1.1
  min_width = 0.13
  min_spacing = 0.15
  enclosure_up = 0.00
  enclosure_down = 0.00
  measured_from = polyX
  measured_to = metal1
}
```

contact_width2

Parameter for layer(s): pcaux, seed

An optional pcaux and seed layer parameter for all contact types except diffusion contact. This parameter specifies the ct2 bias width for a seed layer contact.

Syntax

contact_width2 = *value*

Parameters

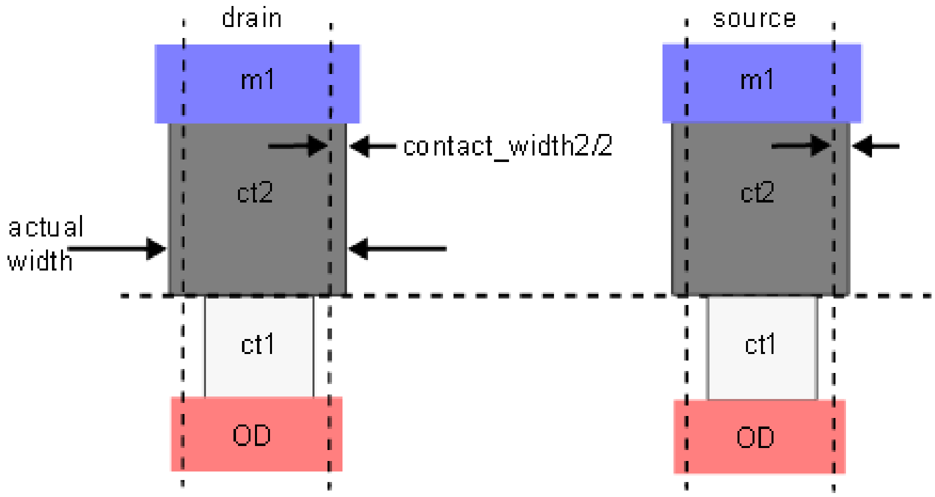
- *value*

  A floating point value used to specify the ct2 bias.

Description

The optional contact_width2 parameter is used to represent the upper section of the diffusion contact as seen in Figure 6-14.

Figure 6-14. contact_width2 Parameter for Seed Layer



Use a contact_width table to define ct1. For more information on contact_width tables, see CONTACT_WIDTH Table. See Seed for more information on the seed layer type.

Examples

```
contact_width2 = 0.24
```

coplanar_min_spacing

Type: conductor, device_li, diffusion, ground, li, pcaux, poly, resistor

An optional parameter for conducting layer definitions conductor, device_li, diffusion, ground, li, pcaux, poly, and resistor that specifies the minimum spacing used between coplanar layers.

Syntax

coplanar_min_spacing = *value*

Parameters

- *value*

  A positive floating point value that represents the minimum spacing used between coplanar layers.

Examples

```
coplanar_min_spacing = 0.04
```

copy

Parameter for layer(s): ground

An optional ground layer parameter that specifies a space delimited list of layer names to be mapped to ground layers under the gate.

Syntax

copy = ‘{’*layer_name*$_1$ *layer_name*$_2$ ... *layer_name*$_n$‘}’

Parameters

- ‘{’*layer_name*$_1$ *layer_name*$_2$ ... *layer_name*$_n$‘}’

  A space delimited list of layer names enclosed in braces ({ }).

Description

In most cases, the ground layer is a copy of the gate layer and associates the ground layer with the gate layer in the layout patterns in the sdcont profile. For example, if there are two ground layers N and P associated with two N and P diffusion layers, the gate layers associated with each diffusion layer go in the copy list for the associated ground layers.

Examples

```
ground = tpdiff_device_gnd {
   zbottom = -0.01
   thickness = 0.1
   resistivity = 11
   copy = {pgate pgate_srn}
}
```

corner

Type: Global Parameters

An optional global parameter that specifies the corner variation to be applied during extraction.

Syntax

corner = typical | cbest | rcbest | cworst | rcworst

Parameters

- typical | cbest | rcbest | cworst | rcworst

  A value that specifies the corner variation to be applied during extraction. The default is typical.

Description

Process corners represent the extremes of variations of fabrication parameters within which a circuit must function correctly. A circuit that has been etched onto the wafer and runs on devices fabricated at these process corners may run slower or faster than specified and at lower or higher temperatures and voltages. Use the corner parameter to specify the corner variation you want applied during calibration.

The process corner variation choices are typical, cbest, rcbest, cworst, and rcworst. The default is the process name if neither this parameter or the "-corner" command line option is specified. The corner parameter can only be specified once.

Calibration generates the PEX Corner statement and updates the reference corners in the PEX Corner Custom statement with the specified corner variation in the calibrated rule file.

For more information on global parameters, see Global Parameters. For information on SVRF statements, see the *Standard Verification Rule Format (SVRF) Manual*.

Examples

Example 1

Specifying the following global parameter in your .mipt file:

```
corner = cbest
```

generates SVRF statements similar to the following in your rules.C and rules.R files:

```
PEX CORNER CBEST

PEX CORNER CUSTOM dp_corner1 dp_corner2 REFERENCE cbest cbest
VARIATION XCALIBRATE__M1
mask_offset_x 1 delta 0.007 -0.007
mask_offset_y 1 delta 0.007 -0.007
```

Example 2

Specifying the following global parameter in your .mipt file:

```
corner = typical
```

generates SVRF statements similar to the following in your rules.C and rules.R files:

```
PEX CORNER TYPICAL

PEX CORNER CUSTOM dp_corner1 dp_corner2 REFERENCE typical typical
VARIATION XCALIBRATE__M1
mask_offset_x 1 delta 0.007 -0.007
mask_offset_y 1 delta 0.007 -0.007
```

count

Parameter for Table(s): RPV_VS_COUNT Table

An optional parameter whose values specify the number of vias in the array. Can only be used in the one-dimensional rpv_vs_count property table.

Syntax

count = { c_1 c_2 ... c_n}

Parameters

- c_1 c_2 ... c_n

  A list of one or more values that represent the number of vias in an array enclosed in required braces ({ }).

Examples

```
count = { 12 16 32 }
```

damage_eps

Parameter for layer(s): dielectric

An optional parameter that specifies the damaged relative permittivity (dielectric constant). This changes the dielectric constant in a planar dielectric where the damaged portion touches the conductor laterally or vertically. This parameter can only be specified if diel_type is set to planar and must be specified with damage_thickness parameter.

Syntax

damage_eps = ***value***

Parameters

- ***value***

  A floating point value that specifies the damaged relative permittivity (dielectric constant).

Examples

```
dielectric = diel1 {
   diel_type = planar
   thickness = 0.1
   eps = 2.5
   damage_thickness = 0.001
   damage_eps = 3.7
}

dielectric = diel2 {
   diel_type = conformal
   thickness = 1
   eps = 2.5
   swthk = 0.075
   topthk = 0.00
   bottthk = 0.00
   ref_layer = m1
}
```

damage_thickness

Parameter for layer(s): dielectric

An optional dielectric layer parameter that specifies the damaged thickness of the dielectric layer in the z-direction where it touches the conductor laterally or vertically. This parameter can only be specified if diel_type is set to planar and must be specified with damage_eps.

Syntax

damage_thickness **=** ***value***

Parameters

- ***value***

  A floating point value that specifies the damaged thickness of the dielectric layer in the z-direction.

Examples

```
dielectric = diel1 {
   diel_type = planar
   thickness = 0.1
   eps = 2.5
   damage_thickness = 0.001
   damage_eps = 3
}

dielectric = diel2 {
   diel_type = conformal
   thickness = 1
   eps = 2.5
   swthk = 0.075
   topthk = 0.00
   bottthk = 0.00
   ref_layer = m1
}
```

date

Type: Global Parameters

An optional global parameter used to specify the date the MIPT file containing the description of the process was created. The specified string is added to the beginning of the calibrated rule file during calibration. It can only be specified once. If *date* is a single string without spaces, the quotation marks are not required. If the string is enclosed in quotes, spaces are allowed.

Syntax

date = *date*

Parameters

- *date*

  A string that specifies the date the MIPT file containing the description of the process was created.

Examples

Example 1

```
date = August_9_2010
```

Example 2

```
date = 8/9/10
```

Example 3

```
date = "August 9, 2010"
```

density_window

Parameter for layer(s): conductor, device_li, diffusion, ground, li, pad, pcaux, poly, resistor, seed, src_drn

An optional conducting layer parameter that takes a list of values that are applied to any conductor that has a thickness table.

Syntax

density_window = '{' '{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'... '{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}' '}'

Parameters

- {*w h factor stepx stepy*}

  One or more space separated lists of values applied to any conductor that has a thickness table. These lists must be enclosed in braces ({ }).

  w — window size.

  h — optional window height. The default is the value specified for window size, *w*.

  factor — optional value between 0 and 1 that specifies the weight applied to the density window. The default is 1.

  stepx stepy — optional value pair that specifies the grid increment for which the fill density windows are tiled. The default is half the window size, *w*.

Description

If thickness variation is based on width and density, one or more density windows may be specified. The list of values must be enclosed in braces ({ }). There is default density window size is the cell size. This parameter may optionally be specified by conducting layer definitions, conductor, poly, diffusion, and li.

During calibration, a PEX Density Window statement is generated and written to the calibrated rule files for each conductor that specifies the density_window parameter.

For more information on the conductor keyword, see Conductor. For more information on the poly keyword, see Poly. For more information on the diffusion keyword, see Diffusion. For more information on the li keyword, see li.

Examples

```
density_window = {{10 10 0.5 5 5} {30 20 0.1 10 10} {50 50 0.4 25 25}}
```

depletion_width

Parameter for layer(s): TSV

A required TSV layer parameter that specifies the depletion region width of a Through-Silicon-Via (TSV).

Syntax

depletion_width = *value*

Parameters

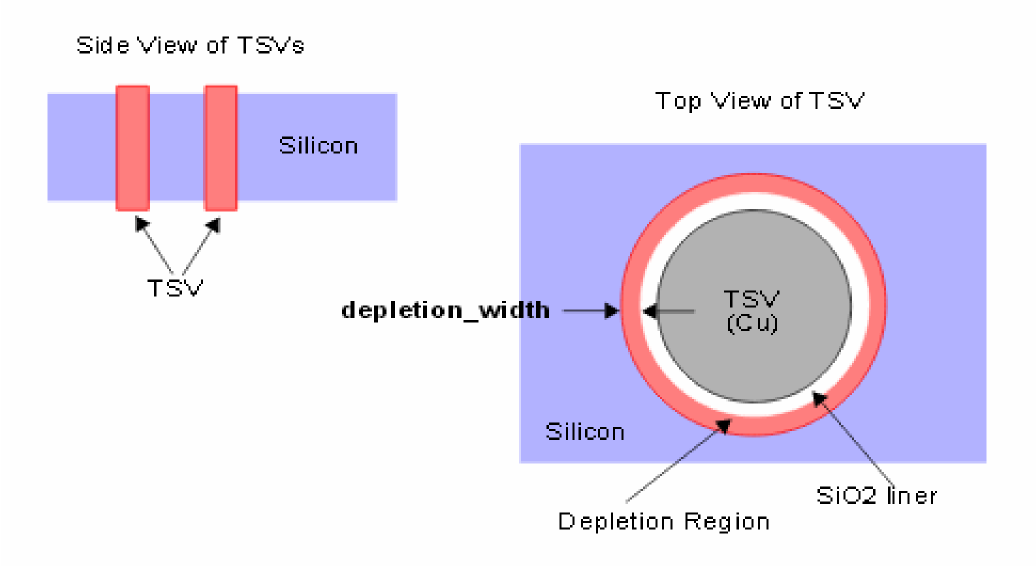
- *value*

  A floating point value that specifies the depletion region width.

Description

The depletion_width measurement is shown in Figure 6-15.

Figure 6-15. depletion_width for TSV Layers



For more information on TSV layers, see TSV.

Examples

```
depletion_width = 10.4
```

derived

Type: Derived layer

The derived layer keyword is an optional keyword used to define the electrical characteristics of a conducting layers. The derived keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. This layer type cannot alter any physical parameters of a conducting layer and can be specified more than once.

Syntax

derived = ***layer_name* {**
  derived_type = {**src_drn** | **src_drn_contact** | **seed** | **ronly** | **ronly_contact** | **ronly_via**}
  resistance = *resistance_per_via*
  resistivity = *resistivity_value*
  r_sheet = *sheet_resistance*
  tc1 = *resistance_temperature1_coefficient*
  tc2 = *resistance_temperature2_coefficient*
  max_rlength = *length_for_resistance_fracturing*
  maxwidth_for_minspacing = *value*
  ignore_caps = yes | no
  hidden = false | true
  }

Parameters

- ***layer_name***

  A required unique user-specified name.

- **derived_type**

  A required parameter that defines the derived layer type. Derived types are only valid for use with specific layers and the syntax checker will verify these are used appropriately. Accepted values are src_drn, src_drn_contact, seed, ronly, ronly_contact, or ronly_via.

- **resistance**

  A required parameter that specifies the resistance per via or contact. The value is specified in Ohms. This parameter cannot be specified with resistivity or r_sheet.

- **resistivity**

  A required layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- **r_sheet**

  A required layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.

- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.

- max_rlength

  An optional conducting layer parameter that specifies the maximum length allowed for resistance fracturing.

- maxwidth_for_minspacing

  An optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for the derived layer. May only be specified if derived_type is ronly or ronly_contact. The default is no, if this parameter is not specified.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Examples

```
derived = my_src_drn { # Table usage; preferred syntax
   derived_type = src_drn
   resistivity = 0.0355
   tc1 = my_local_tc1 use my_Mx_layers_tc1
   tc2 = my_local_tc2 use my_Mx_layers_tc2
}

diffusion = my_diff {
   # CONDUCTOR PARAMETERS
   src_drn_layers = {my_src_drn}
}
```

derived_type

Parameter for layer(s): derived

A required derived layer parameter that describes the specified derived layer type. Derived types are only valid for use with specific layers.

Syntax

derived_type = {**src_drn** | **src_drn_contact** | **seed** | **ronly** | **ronly_contact** | **ronly_via**}

Parameters

- **src_drn**

  Specifies that the type of derived layer is src_drn. Valid for diffusion layers only.

- **src_drn_contact**

  Specifies that the type of derived layer is src_drn_contact. Valid for contact layers only.

- **seed**

  Specifies that the type of derived layer is seed. Valid for poly layers only.

- **ronly**

  Specifies that the type of derived layer is ronly. Valid for conductor and seed layers only.

- **ronly_contact**

  Specifies that the type of derived layer is ronly_contact. Valid for contact layers only.

- **ronly_via**

  Specifies that the type of derived layer is ronly_via. Valid for via layers only.

Examples

```
derived_type = src_drn
```

device_li

Type: Device_li layer

The required device_li layer definition keyword is used to define the device region of the li layer. The device_li keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. Layers of this type represent a physical layer in the stack and can be specified multiple times. This layer definition is required when you are not performing RONLY extraction.

Syntax

device_li = *layer_name* {
thickness = ***z-direction_ metal_ thickness***
min_width = *minimum_allowed_metal_width*
min_spacing = ***minimum_allowed_metal_spacing***
resistivity = ***resistivity_value***
r_sheet = ***sheet_resistance***
metal_fill = '{'*fill_ratio fill_spacing fill_width* [floating | grounded] '}'
ztop = *top_ z-coordinate*
zbottom = *bottom_z-coordinate*
measured_from = *layername*
airgap = {rectangle | triangle} min_gap (*v*) spacing (*s1* ... *sn*) width (*w1* ... *wn*)
thickness (*t1* ... *tn*) dist_from_surface (*b1* ... *bn*)
thickness_type = absolute | relative
coplanar_min_spacing = *value*
tc1 = *resistance_temperature1_coefficient*
tc2 = *resistance_temperature2_coefficient*
max_rlength = *length_for_resistance_fracturing*
max_width = *maximum_allowed _metal_width*
max_spacing = *maximum_allowed_metal_spacing*
maxwidth_for_minspacing = *value*
devices = '{'$device_1$ $device_2$ ... $device_n$'}' # a space delimited list of device names
layer_bias = *bias_actual_width_relative_to_drawn_width*
capacitive_only_etch = *layer_bias_for_capacitance_only*
resistive_only_etch = *layer_bias_for_resistance_only*
trap_style = top | middle | bottom # where trapezoid measurements are taken from
extra_width = *extra-width_of_side_walls* # for a conductor with a trapezoid shape
swstep = *count*
extension = *amount_layer_extends_past_diffusion_or_contact_layers*
ignore_caps = yes | no
ronly_layers = '{'*space_delimited_list_of_r-only_layer_names*'}'
density_window = '{' '{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'...'{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}''}'
hidden = false | true
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- **thickness**

  A required layer parameter that specifies the thickness of the metal layer in the z-direction.

- **min_width**

  A required layer parameter that specifies the minimum allowed drawn width of metal objects on this layer.

- **min_spacing**

  A required parameter that specifies the minimum allowed metal spacing.

- **resistivity**

  A required layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- **r_sheet**

  A required layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- metal_fill

  An optional conducting layer parameter. It is a set of values enclosed in braces, used to define a list of virtual fill parameters.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- measured_from

  An optional conducting layer parameter that measures zbottom or ztop relative to the specified layer.

- airgap

  An optional parameter that specifies the airgap parameters for the device_li layer.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.

- coplanar_min_spacing

  An optional conducting layer parameter that specifies the minimum spacing value between coplanar layers.
- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.
- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.
- max_rlength

  An optional conducting layer parameter that specifies the maximum length allowed for resistance fracturing.
- max_width

  An optional parameter that specifies the maximum allowed metal width.
- max_spacing

  An optional parameter that specifies the maximum allowed metal spacing.
- maxwidth_for_minspacing

  An optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
- devices

  An optional layer parameter that specifies a space delimited list of device names enclosed in braces. This parameter is used to specify the devices the layer definition applies to.
- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width relative to drawn width.
- capacitive_only_etch

  An optional parameter that specifies the layer bias for capacitance only.
- resistive_only_etch

  An optional parameter that specifies the layer bias for resistance only.
- trap_style

  An optional conducting layer parameter that defines where trapezoid measurements are taken from. Permitted values are top, middle, or bottom.
- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- swstep

  An optional conducting layer parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.

- extension

  An optional conducting layer parameter that specifies the amount this layer extends past diffusion or contact layers.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- density_window

  An optional conducting layer parameter specified as {{w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$} ... {w_n h_n $factor_n$ $stepx_n$ $stepy_n$}}. The braces are required for grouping.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Examples

```
device_li = m0_od2 {
   ztop = 0.0
   measured_from = poly_diel7
   thickness = 0.09
   min_width = 0.045
   min_spacing = 0.045
   resistivity = 0.0366
   tc1 = use my_Mx_layers_tc1
   tc2 = use my_Mx_layers_tc2
}
```

device_li_layers

Parameter for layer(s): li

An optional li layer parameter that specifies a space delimited list of device_li layer names identifying the device local interconnect layers associated with it.

Syntax

device_li_layers = {$li\_layer\_name_1$ $li\_layer\_name_2$ ... $li\_layer\_name_n$}

Parameters

- {$li\_layer\_name_1$ $li\_layer\_name_2$ ... $li\_layer\_name_n$}

  A space delimited list of device_li layer names enclosed in braces ({ }).

Examples

```
li = M0_STI1 {
   min_width = 0.02600
   min_spacing = 0.05800
   resistivity = 0.67
   device_li_layers = {M0_OD1}
}

device_li = M0_OD1 {
   min_spacing = 0.05800
   resistivity = 0.672
   layer_bias = -0.0070
   devices = { ngate_mac ngate_15_mac
               pgate_mac pgate_15_mac}
}
```

devices

Parameter for layer(s): conductor, device_li, diffusion, li, pcaux, poly, resistor, seed, src_drn

An optional layer parameter that specifies the devices the layer definition applies to.

Syntax

devices = {$device_1$ $device_2$ ... $device_n$}

Parameters

- {$device_1$ $device_2$ ... $device_n$}

  A space delimited list of device names enclosed in braces ({ }).

Description

The devices parameter is used to specify which devices the layer definition applies to so that xCalibrate can generate the corresponding device models in the calibrated rule files. Without the devices parameter, xCalibrate pairs the first poly layer encountered in the MIPT file with the first diffusion layer as it parses the file, the second poly layer with the second diffusion layer, and so forth.

The device names can be any name you choose; they do not have to match the LVS device definition.

The devices parameter can only be specified in the conductor layer that defines metal layers associated with a device. For example, typically, there is only one conductor layer defined as metal1 that is shared by default across all devices, and the devices parameter is not required. However, if you are using more than one conductor layer to define metal1, then you must include the devices parameter in each metal1 conductor layer definition to specify the associated device. Specifying the devices parameter for metal layers that are not part of a device is not allowed.

Examples

Example 1

```
poly = GATE_FINAL{
   thickness = 0.1
   min_width = 0.045
   min_spacing = 0.12
   extension = 0.14
   devices = {NMOS PMOS}
}
```

```
diffusion = diff {
   thickness = 0.16
   min_width = 0.045
   min_spacing = 0.12
   devices = {NMOS PMOS}
   src_drn_layers = {NDIFSI_FINAL PDIFSI_FINAL}
   extension = 0.03
}
```

Example 2

Given two MOS devices, MOS_LV and MOS_HV in the stack whose gates are physically different.

```
seed = poly_LV {
   ...
   devices = {MOS_LV}
   ...
}

seed = poly_HV {
   ...
   devices = {MOS_HV}
   ...
}

diff = diff1{
   ...
   Src_drn_layers = {tndiff tpdiff}
   devices = {MOS_LV MOS_HV}
   ...
}
```

The devices parameter controls which layers belong to the same device. Without the devices statement, the models generated by xCalibrate may not be what you expect. In this example, the devices keyword is used to define two sdcont profiles; one made of poly_LV with tndiff and tpdiff, and one made of poly_HV with tndiff and tpdiff for which xCalibrate generates corresponding device models in the rule files.

Example 3

Use the devices parameter to control which metal1 layer is used by a device. The following two conductor layer definitions describe two different metal1 layers, M1A and M1B.

```
seed = npoly1 {
   ...
   devices = {NMOS1}
   ...
}

seed = npoly2 {
   ...
   devices = {NMOS2}
   ...
}

seed = ppoly1 {
   ...
   devices = {PMOS1}
   ...
}

seed = ppoly2 {
   ...
   devices = {PMOS2}
   ...
}
. . .
conductor = M1A {
   ztop = 0.0
   measured_from = poly_diel7
   thickness = 0.09
   min_width = 0.045
   min_spacing = 0.045
   resistivity = 0.0366
   extension = 0.06
   devices = {NMOS1 PMOS1}
}

conductor = M1B {
   ztop = 0.0
   measured_from = poly_diel7
   thickness = 0.12
   min_width = 0.055
   min_spacing = 0.055
   resistivity = 0.033
   extension = 0.06
   devices = {NMOS2 PMOS2}
}
. . .
```

Devices NMOS1 and PMOS1 use metal1 layer M1A, and devices NMOS2 and PMOS2 use metal1 layer M1B. When defining multiple metal1 layers you must also divide the seed layers into multiple corresponding seed layers for the devices as shown in this example.

dielectric

Type: Dielectric layer

Part of the layer definition syntax, the dielectric layer definition keyword is used to optionally define a insulating layers. The dielectric keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. Layers of this type represent a physical layer in the stack and can be specified multiple times.

Syntax

dielectric = *layer_name* {
    diel_type = planar | conformal | rsd_conformal | trench | mask | spacer | spacer_mask
    thickness = *layer_thickness*
    eps = *value*
    ref_layer = *reference_layername* # required for conformal, trench, and spacer diel_types
    swslope = *value*
    swstep = *count*
    swthk = *value* # required for conformal, trench, and spacer diel_types
    topthk = *value* # required for conformal, trench, and spacer diel_types
    botthk = *value* # required for conformal, trench, and spacer diel_types
    ztop = *top_z-coordinate*
    zbottom = *bottom_z-coordinate*
    measured_from = *layername*
    airgap = {rectangle | triangle} min_gap (*v*) spacing (*s1* ... *sn*) width (*w1* ... *wn*)
            thickness (*t1* ... *tn*) dist_from_surface (*b1* ... *bn*)
    thickness_type = <u>absolute</u> | relative
    damage_thickness = *value*
    damage_eps = *value*
    hidden = <u>false</u> | true
    }

Parameters

- ***layer_name***

  A required unique user-specified name.

- **diel_type**

  A required parameter that specifies the type of dielectric being described. Legal values are planar, conformal, rsd_conformal, dielectric, mask, spacer, or spacer_mask.

- **thickness**

  A required layer thickness parameter that specifies the thickness of the dielectric layer in the z-direction. The thickness may be a zero value if diel_type is conformal or rsd_conformal. For a trench dielectric where diel_type is conformal or rsd_conformal, the thickness must be 0.

- **eps**

  A required value that specifies the relative permittivity (dielectric constant).

- ref_layer

  An optional parameter when diel_type is planar. A required parameter when diel_type is set to conformal, rsd_conformal, trench, or spacer.

- swslope

  An optional dielectric layer parameter used to specify the angle of the side wall in degrees. May only be used when diel_type is set to conformal or trench.

- swstep

  An optional dielectric layer parameter that specifies the number of rectangles used to model trapezoid shapes. May only be used when diel_type is set to conformal. Used only by the Calibre xACT 3D field solver.

- swthk

  An optional parameter when diel_type is planar. A required parameter when diel_type is set to conformal, rsd_conformal, trench, or spacer.

- topthk

  An optional parameter when diel_type is planar. A required parameter when diel_type is set to conformal, rsd_conformal, trench, or spacer.

- botthk

  An optional parameter when diel_type is planar. A required parameter when diel_type is set to conformal, rsd_conformal, trench, or spacer.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- measured_from

  An optional parameter whose value specifies the layer whose top surface is used as the base from which the layer's bottom is measured. Measures zbottom or ztop relative to the given layer. This parameter is required if thickness_type parameter is specified.

- airgap

  An optional conformal dielectric parameter that specifies the airgap parameters for the dielectric layer.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.

- damage_thickness

  An optional parameter that specifies the damaged thickness a planar dielectric layer in the z-direction where it touches the conductor laterally or vertically. This parameter can only be specified if diel_type is set to planar and must be specified with damage_eps.

- damage_eps

  An optional parameter that specifies the damaged relative permittivity (dielectric constant). This parameter can only be specified if diel_type is set to planar and must be specified with damage_thickness parameter.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Examples

Example 1

The following is an example of a planar dielectric layer and conformal dielectric layer:

```
dielectric = diel1 {
   diel_type = planar
   thickness = 0.1
   eps = 2.5
}

dielectric = diel2 {
   diel_type = conformal
   thickness = 1
   eps = 2.5
   swthk = 0.075
   topthk = 0.2
   botthk = 0.1
   ref_layer = m1
}
```

Example 2

Placed seed layers do not inherit the conformal dielectric layer from the seed layer's parent poly layer. If you want conformal dielectric around the placed seed layer, then you need to define a conformal dielectric layer that references the seed layer and has the same parameters as the seed layers parent poly layer. For example:

```
poly = fpoly {
   ...
   seed_layers = {ngate}
   ...
}

dielectric = spacer {
   thickness = 0
   eps = 4
   diel_type = conformal
   ref_layer = fpoly
   topthk = 0.1
   swthk = 0.1
}

// placed seed layer
seed = ngate {
   ...
}

dielectric = spacer_seed {
   thickness = 0
   eps = 4
   diel_type = conformal
   ref_layer = ngate    // placed seed layer name
   topthk = 0.1
   swthk = 0.1
}
```

diel_over_gate_bottom

Parameter for layer(s): seed

An optional seed layer parameter that specifies the distance (in microns) from the gate poly to the bottom of a dielectric air gap. This parameter must be specified together with diel_over_gate_width and diel_over_gate_thickness seed layer parameters.

Syntax

diel_over_gate_bottom = *value*

Parameters

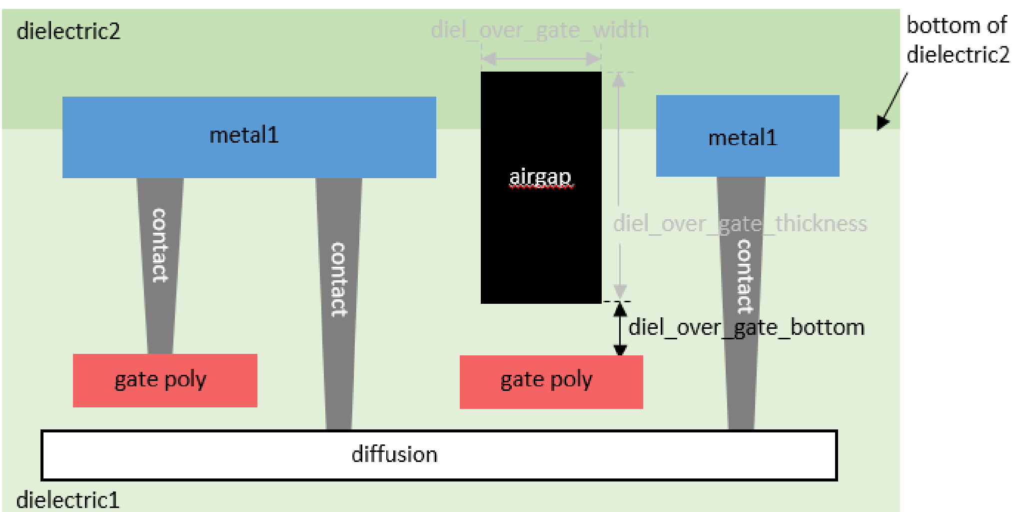
- *value*

  A floating point value in microns that specifies the distance from the gate poly to the bottom of a dielectric air gap.

Description

Use the diel_over_gate_bottom parameter together with the diel_over_gate_width and diel_over_gate_thickness parameters in the seed layer definition to describe an air gap over the gate as shown in Figure 6-16.

Figure 6-16. diel_over_gate_bottom



Examples

```
seed = pc_mod {
    r_sheet = 8.8
    devices = {mos}
    tc1 = 0.003
    tc2 = 0.00
    diel_over_gate_width = 0.3
    diel_over_gate_bottom = 0.2
    diel_over_gate_thickness = 0.5
}
```

diel_over_gate_eps

Parameter for layer(s): seed

An optional seed layer parameter that specifies the dielectric gap permittivity where the gap is centered above the gate poly and runs the length of the gate. This parameter can be optionally specified together with diel_over_gate_bottom, diel_over_gate_width, and diel_over_gate_thickness seed layer parameters.

Syntax

diel_over_gate_eps = *value*

Parameters

- *value*

  A floating point value that specifies the dielectric gap permittivity for a dielectric air gap. The default is 1 if this parameter is not specified.

Examples

```
seed = pc_mod {
    r_sheet = 8.8
    devices = {mos}
    tc1 = 0.003
    tc2 = 0.00
    diel_over_gate_width = 0.3
    diel_over_gate_bottom = 0.2
    diel_over_gate_thickness = 0.5
    diel_over_gate_extrawidth = 0.009
    diel_over_gate_eps = 1.5
}
```

diel_over_gate_extrawidth

Parameter for layer(s): seed

An optional seed layer parameter that specifies the extra width of the top side of a trapezoid-shaped air gap. This parameter can be optionally specified together with diel_over_gate_bottom, diel_over_gate_width, and diel_over_gate_thickness seed layer parameters.

Syntax

diel_over_gate_extrawidth = *value*

Parameters

- *value*

  A floating point value (in microns) that specifies the extra width of the top side of a trapezoid-shaped gap. If this parameter is not specified, then the default is 0 for a rectangular shaped gap.

Examples

```
seed = pc_mod {
    r_sheet = 8.8
    devices = {mos}
    tc1 = 0.003
    tc2 = 0.00
    diel_over_gate_width = 0.3
    diel_over_gate_bottom = 0.2
    diel_over_gate_thickness = 0.5
    diel_over_gate_extrawidth = 0.009
    diel_over_gate_eps = 1.5
}
```

diel_over_gate_thickness

Parameter for layer(s): seed

An optional seed layer parameter that specifies the thickness (in microns) of a dielectric air gap. This parameter must be specified together with diel_over_gate_width and diel_over_gate_bottom seed layer parameters.

Syntax

diel_over_gate_thickness = *value*

Parameters

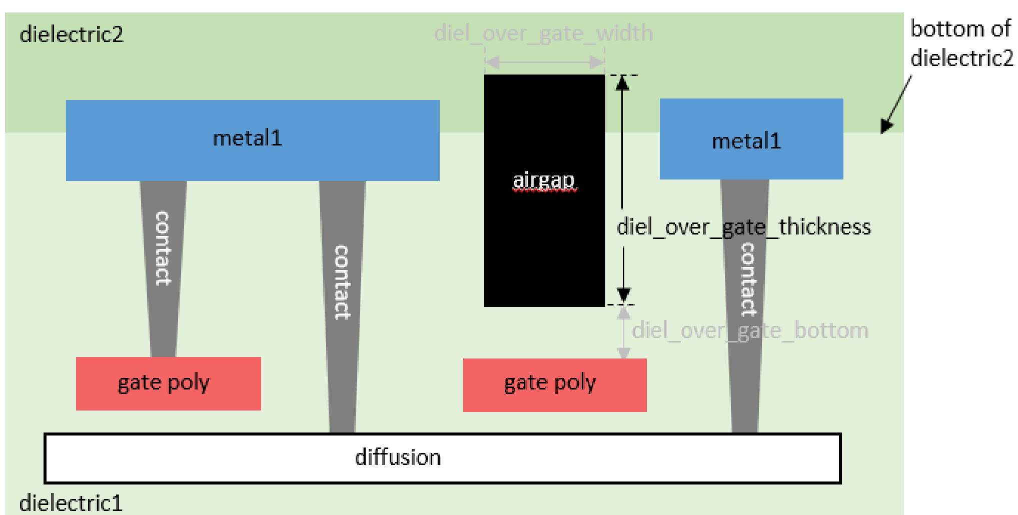
- *value*

  A floating point value in microns that specifies the thickness of a dielectric air gap.

Description

Use the diel_over_gate_thickness parameter together with the diel_over_gate_width and diel_over_gate_bottom parameters in the seed layer definition to describe an air gap over the gate as shown in Figure 6-17.

Figure 6-17. diel_over_gate_thickness



Examples

```
seed = pc_mod {
    r_sheet = 8.8
    devices = {mos}
    tc1 = 0.003
    tc2 = 0.00
    diel_over_gate_width = 0.3
    diel_over_gate_bottom = 0.2
    diel_over_gate_thickness = 0.5
}
```

diel_over_gate_width

Parameter for layer(s): seed

An optional seed layer parameter that describes the width (in microns) of a dielectric air gap. This parameter must be specified together with diel_over_gate_bottom and diel_over_gate_thickness seed layer parameters.

Syntax

diel_over_gate_width = *value*

Parameters

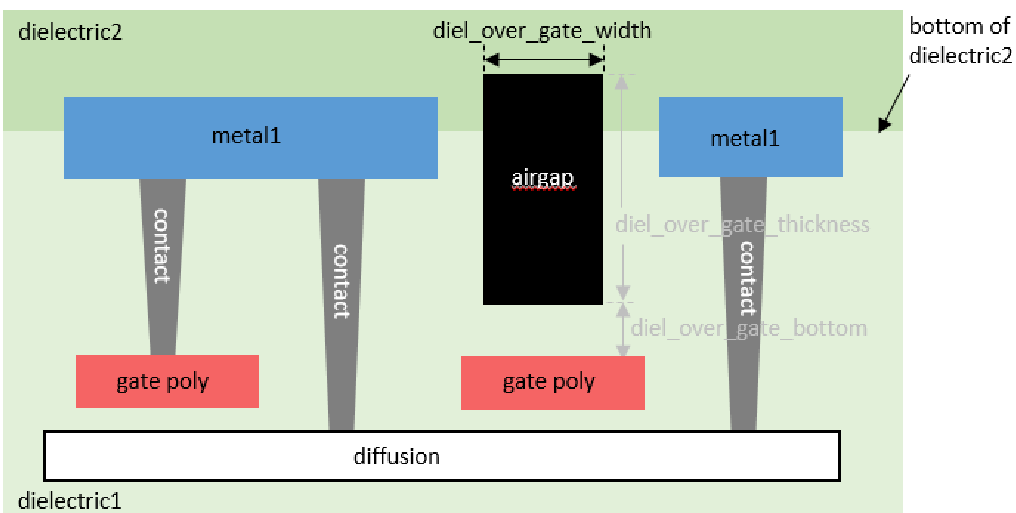
- *value*

  A floating point value in microns that specifies the width of a dielectric air gap.

Description

Use the diel_over_gate_width parameter together with the diel_over_gate_bottom and diel_over_gate_thickness parameters in the seed layer definition to describe an air gap over the gate as shown in Figure 6-18.

Figure 6-18. diel_over_gate_width



Examples

```
seed = pc_mod {
    r_sheet = 8.8
    devices = {mos}
    tc1 = 0.003
    tc2 = 0.00
    diel_over_gate_width = 0.3
    diel_over_gate_bottom = 0.2
    diel_over_gate_thickness = 0.5
}
```

diel_type

Parameter for layer(s): dielectric

A required dielectric layer parameter that describes the type of dielectric layer being defined.

Syntax

diel_type = planar | conformal | rsd_conformal | trench | mask | spacer | spacer_mask

Parameters

- **planar**

  Specifies a planar dielectric.

- **conformal**

  Specifies a conformal dielectric.

- **rsd_conformal**

  Specifies a rsd conformal dielectric.

- **trench**

  Specifies a trench dielectric.

- **mask**

  Specifies a local dielectric for extraction with field solvers. Use only with Calibre xACT 3D.

- **spacer**

  Specifies a spacer dielectric.

- **spacer_mask**

  Specifies a painted dielectric over a conductor. Used for slotted metal when performing 3D extraction. Use only with Calibre xACT 3D.

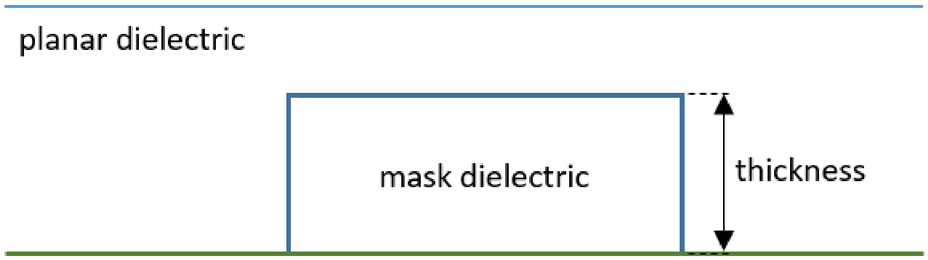
Description

Legal values for diel_type are planar, conformal, rsd_conformal, trench, mask, spacer, or spacer_mask. Specifying more than twenty conformals per layer generates an error message and stops calibration. A layer cannot have a diel_type equal to both conformal and trench.

Set diel_type to mask when defining the local dielectric layer, typically around a device. The mask subtype does not change the top of the current dielectric stack for relative layer definitions. It should have the required thickness and eps parameters. If zbottom is not specified, the mask subtype sits on top of the last regular dielectric, and does not change the top of the stack. It is coplanar with the planar dielectrics. It may be trapezoid in shape (have an extra_width parameter), and may have its own conformal or trench dielectrics. This dielectric type is only used by Calibre xACT 3D. Figure 6-19 shows an example of a mask dielectric.

Set diel_type to spacer_mask to model holes preventing fracturing and off angle problems during extraction. This dielectric type is only used by Calibre xACT 3D.

Figure 6-19. Mask Dielectric



Set diel_type to conformal when defining a dielectric layer over a diffusion. If diel_type is conformal, the sidewall thickness (swthk), top thickness (topthk), and bottom thickness (botthk) of the conformal dielectric as well as a reference layer (ref_layer) for the conformal layer must be specified. Figure 6-20, Figure 6-21, and Figure 6-22 show examples of conformal dielectrics.

Figure 6-20. Conformal Dielectric

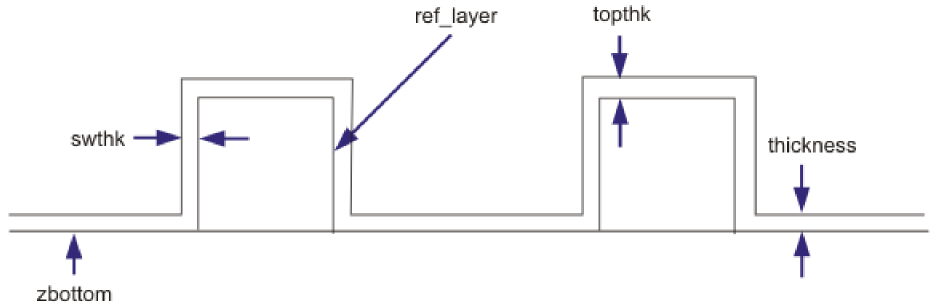


Figure 6-21. Conformal Dielectric With Negative Top Thickness

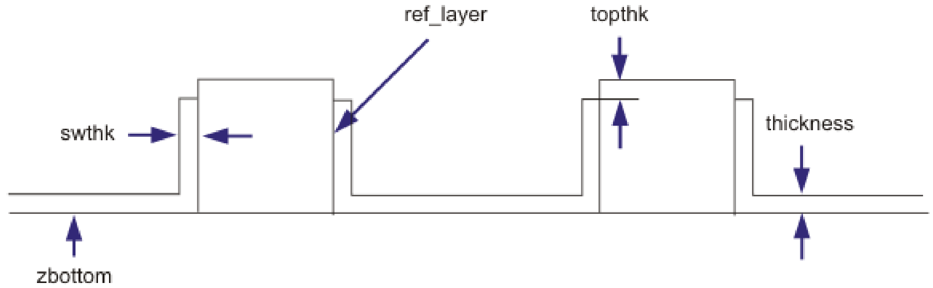
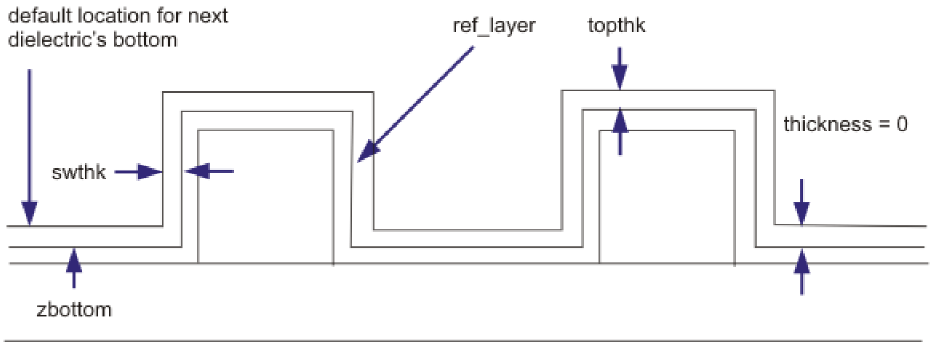


Figure 6-22. Double Conformal Dielectrics



Set diel_type to rsd_conformal when defining a dielectric layer over an rsd layer. If diel_type is rsd_conformal, the sidewall thickness (swthk), top thickness (topthk), and bottom thickness (botthk) of the rsd_conformal dielectric as well as a reference layer (ref_layer) for the rsd_conformal layer must be specified.

Figure 6-23 shows an example of an rsd_conformal dielectric.

Figure 6-23. RSD_conformal Dielectric

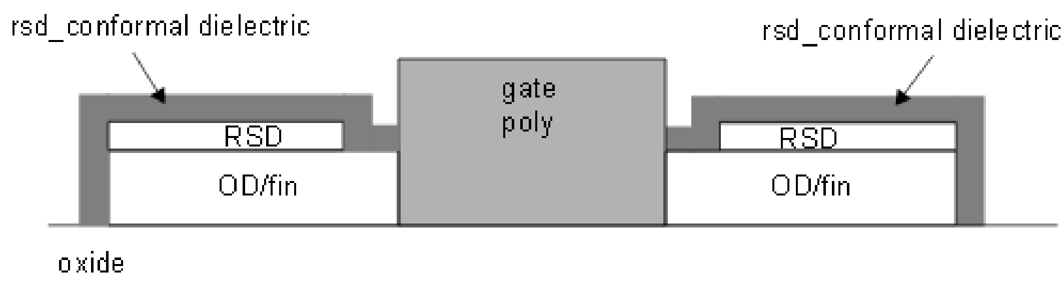


Figure 6-24 and Figure 6-25 are examples of trench dielectrics.

Figure 6-24. Trench Dielectric With Positive Top Thickness

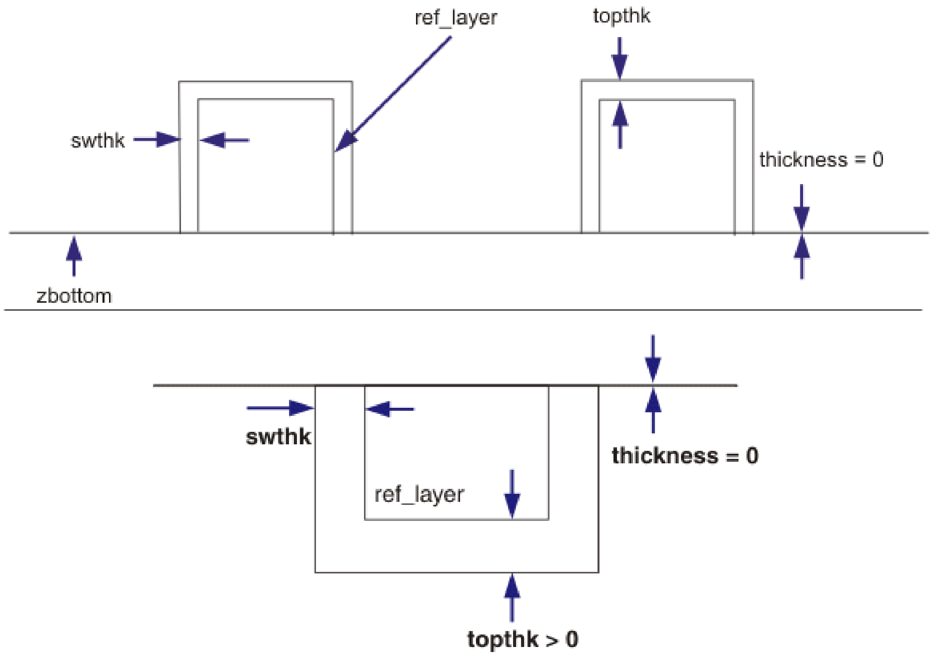


Figure 6-25. Trench Dielectric With Negative Top Thickness

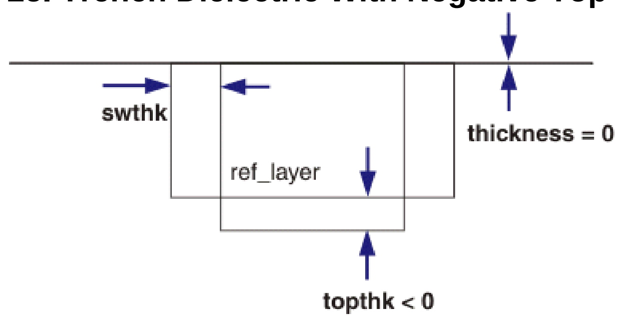
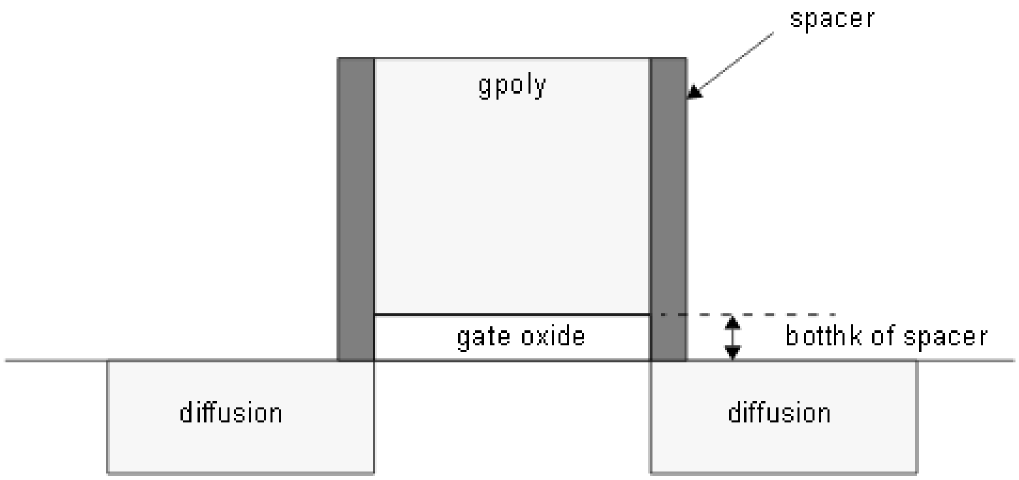


Figure 6-26 shows an example of a spacer dielectric. Spacer dielectrics are conformal dielectrics used to self align contacts near the gate or other similar operations.

Figure 6-26. Spacer Dielectric



For more information on the dielectric layers, see Dielectric.

Examples

```
dielectric = diel1 {
   diel_type = planar
   thickness = 0.1
   eps = 2.5
}

dielectric = diel2 {
   diel_type = conformal
   thickness = 1
   eps = 2.5
   swthk = 0.075
   topthk = 0.00
   bottthk = 0.00
   ref_layer = m1
}
```

diffusion

Type: Diffusion layer

Part of the layer definition syntax, the diffusion layer definition keyword is used to define the electrical and physical characteristics of conducting layers. Layers of this type can be specified multiple times. This layer definition is required when you are not performing RONLY extraction.

Syntax

diffusion = *layer_name* {
thickness = ***z-direction_metal_ thickness***
min_width = ***minimum_allowed_metal_width***
min_spacing = ***minimum_allowed_metal_spacing***
resistivity = ***resistivity_value***
r_sheet = ***sheet_resistance***
metal_fill = '{'*fill_ratio fill_spacing fill_width* [floating | grounded] '}'
ztop = *top_ z-coordinate*
zbottom = *bottom_z-coordinate*
measured_from = *layername*
airgap = {rectangle | triangle} min_gap (*v*) spacing (*s1* ... *sn*) width (*w1* ... *wn*)
thickness (*t1* ... *tn*) dist_from_surface (*b1* ... *bn*)
thickness_type = absolute | relative
coplanar_min_spacing = *value*
tc1 = *resistance_temperature1_coefficient*
tc2 = *resistance_temperature2_coefficient*
max_rlength = *length_for_resistance_fracturing*
max_width = *maximum_allowed _metal_width*
max_spacing = *maximum_allowed_metal_spacing*
maxwidth_for_minspacing = *value*
devices = '{'$device_1$ $device_2$ ... $device_n$'}' # a space delimited list of device names
layer_bias = *bias_actual_width_relative_to_drawn_width*
trap_style = top | middle | bottom # where trapezoid measurements are taken from
extra_width = *extra-width_of_side_walls* # for a conductor with a trapezoid shape
swstep = *count*
extension = *amount_layer_extends_past_diffusion_or_contact_layers*
ignore_caps = yes | no
ronly_layers = '{'*space_delimited_list_of_r-only_layer_names*'}'
density_window ='{' '{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'...'{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}' '}'
src_drn_layers = '{'*space_delimited_list_of_src_drn_layer_names* '}'
rsd_enclosure = *raised_source/drain_edge_bias_value*
rsd_thickness = *raised_source/drain_height_value*
rsd_spacing = *raised_source/drain_to_gate_spacing_value*
rsd_swslope = {*Sx Sy*}
widths = '{' w_1 w_2 w_3 ... w_n '}'
spacings = '{' s_1 s_2 s_3 ... s_n '}'

hidden = <u>false</u> | true
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- **thickness**

  A required layer parameter that specifies the thickness of the diffusion metal layer in the z-direction.

- **min_width**

  A required layer parameter that specifies the minimum allowed drawn width of metal objects on this layer.

- **min_spacing**

  A required parameter that specifies the minimum allowed metal spacing.

- **resistivity**

  A required layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- **r_sheet**

  A required layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- metal_fill

  An optional conducting layer parameter. It is a set of values enclosed in braces, used to define a list of virtual fill parameters.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- measured_from

  An optional conducting layer parameter that measures zbottom or ztop relative to the specified layer.

- airgap

  An optional parameter that specifies the airgap parameters for the diffusion layer.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.
- coplanar_min_spacing

  An optional conducting layer parameter that specifies the minimum spacing value between coplanar layers.
- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.
- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.
- max_rlength

  An optional conducting layer parameter that specifies the maximum length allowed for resistance fracturing.
- max_width

  An optional parameter that specifies the maximum allowed metal width.
- max_spacing

  An optional parameter that specifies the maximum allowed metal spacing.
- maxwidth_for_minspacing

  An optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
- devices

  An optional layer parameter that specifies a space delimited list of device names enclosed in braces. This parameter is used to specify the devices the layer definition applies to.
- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width relative to drawn width.
- trap_style

  An optional conducting layer parameter that defines where trapezoid measurements are taken from. Permitted values are top, middle, or bottom.
- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- swstep

  An optional conducting layer parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.

- extension

  An optional conducting layer parameter that specifies the amount this layer extends past diffusion or contact layers.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- density_window

  An optional conducting layer parameter specified as {{w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$} ... {w_n h_n $factor_n$ $stepx_n$ $stepy_n$}}. The braces are required for grouping.

- src_drn_layers

  An optional layer parameter that specifies a space delimited list of src_drn layer names enclosed in braces ({ }).

- rsd_enclosure

  An optional layer parameter that specifies the raised source/drain edge bias.

- rsd_thickness

  An optional conducting layer parameter that specifies the raised source/drain height.

- rsd_spacing

  An optional conducting layer parameter that specifies the raised source/drain to gate spacing.

- rsd_swslope

  An optional conducting layer parameter that specifies the sidewall slope of the associated raised source/drain layer. Parameters rsd_thickness and rsd_spacing must be specified with this parameter.

- widths

  An optional conducting layer parameter that specifies a space delimited list of floating point width values enclosed in braces ({ }).

- spacings

  An optional conducting layer parameter that specifies a space delimited list of floating point spacing values enclosed in braces ({ }).

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Description

The diffusion keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line.

A diffusion layer must be named as the **measured_to** or **measured_from** layer for at least one contact layer in the MIPT file. A warning is generated during calibration for diffusion layers that do not have a corresponding contact layer.

Examples

```
diffusion = diff1 {
   thickness = 0.09
   min_width = 0.045
   min_spacing = 0.045
   resistivity = 0.0366
   src_drn_layers = {tndiff tpdiff}
}
```

dim_type

table structure parameter

A required table structure parameter that must be either drawn or actual. Note that not every keyword is supported with every table type.

Syntax

dim_type = **drawn** | **actual**

Parameters

- **drawn**

  Specifies that the table should treat dimension values as drawn.

- **actual**

  Specifies that the table should treat dimension values as actual.

Description

A required table structure parameter that specifies the types of values specified in the table. "Table Property" on page 113 shows which dimension type settings should be used for each table property. For more information on tables, see "Table Syntax" on page 109.

Examples

```
dim_type = drawn
```

distance_unit

Type: Global Parameters

An optional global parameter that specifies the units for distance values. It can only be specified once.

Syntax

distance_unit = *unit*

Parameters

- *unit*

  A value that specifies the units for distance values.

  Acceptable values are:

  a — angstrom

  nm — nanometer

  um — micrometer or micron

  The default is micrometers (1e-06).

Examples

```
distance_unit = nm
```

enclosure

table structure parameter

An optional table structure parameter used to specify whether to use the min or max values from the up and down via_resistance or via_resistance2 table structure. This parameter may only be used in two-dimensional via_resistance or via_resistance2 property tables.

Syntax

enclosure = max | min

Parameters

- max

  A keyword that specifies to use the maximum values. This is default.

- min

  A keyword that specifies to use the minimum values.

Examples

```
table = table_up {
   property = via_resistance2
   table_type = R
   layers = up
   enclosure = min
   dim_type = drawn
   value_type = absolute
   via_size = { 0.022 0.022 }
   overlap_width = { 0.02175 }
   overlap_length = { 0.02175 }
   value = { 30.2 }
}
table = table_down {
   property = via_resistance2
   table_type = R
   layers = down
   enclosure = min
   dim_type = drawn
   value_type = absolute
   via_size = { 0.022 0.022 }
   overlap_width = { 0.02175 }
   overlap_length = { 0.02175 }
   value = { 40 }}
```

enclosure_down

Parameter for layer(s): required by contact, ubump; optional for base_via, via

A layer parameter used by via, base_via, contact, and ubump layers that specifies the enclosure with respect to a lower conductor.

Syntax

enclosure_down = ***value***

Parameters

- ***value***

  A floating point value used to specify the enclosure with respect to a lower conductor. The default is 0 if this parameter is not specified.

Examples

```
enclosure_down = 0.01
```

enclosure_up

Parameter for layer(s): required by contact, ubump; optional for base_via, via.

A layer parameter used by via, base_via, contact, or ubump layers that specifies the enclosure with respect to an upper conductor.

Syntax

enclosure_up = ***value***

Parameters

- ***value***

  A floating point value used to specify the enclosure with respect to an upper conductor. The default is 0 if this parameter is not specified.

Examples

```
enclosure_up = 0.01
```

env

Type: Global Parameters

An optional global parameter used to specify one or more environment variables used during calibration.

Syntax

env = '{'
"*EVname1,value1*"
"*EVname2,value2*"
...
"*EVnameN,valueN*"
'}'

Parameters

- "*EVname,value*"

  One or more comma separated pairs of strings enclosed in quotation marks (""). Each pair of strings must appear on its own line.

  EVname — environment variable name.

  value — a real number or string

Description

Use this parameter to specify any environment variables you want to set prior to the start of the MIPT calibration. Each quoted pair must appear on its own line and must be enclosed in braces ({ }) as shown in the syntax.

Examples

In this example, specifying the following env parameter in the *example.mipt* file:

```
env = {
"XCAL_INTRINSIC_IGNORE_ALL,on"
"XCAL_ENABLE_LOADING_EFFECT,1"
}
```

is the same as setting the environment variables in a c-shell window:

```
setenv XCAL_INTRINSIC_IGNORE_ALL on
setenv XCAL_ENABLE_LOADING_EFFECT 1
```

prior to running calibration:

```
xcalibrate -exec example.mipt
```

eps

Parameter for layer(s): dielectric

A required parameter for dielectric layer definitions that specifies the relative permittivity (dielectric constant).

Syntax

eps = ***value***

Parameters

- ***value***

  A floating point value that specifies the relative permittivity (dielectric constant).

Examples

```
dielectric = diel1 {
   diel_type = planar
   thickness = 0.1
   eps = 2.5
}

dielectric = diel2 {
   diel_type = conformal
   thickness = 1
   eps = 2.5
   swthk = 0.075
   topthk = 0.00
   bottthk = 0.00
   ref_layer = m1
}
```

equation

table structure parameter

An optional table structure parameter used to specify an in-die equation.

Syntax

equation = ‘{’*equation_to_evaluate*’}’

Parameters

- ‘{’*equation_to_evaluate*‘}’

  An equation enclosed in braces ({ }) used to specify the thickness variation (in-die) equation.

Description

Specifies an in-die equation for the layer. When specified, this equation is copied to the PEX Tables section of the generated rules file. The equation definition follows the rules in the *Standard Verification Rule Format (SVRF) Manual*.

Examples

```
table = my_thickness_table {
   property = thickness
   table_type = R
   dim_type = drawn
   value_type = delta
   equation = {
      property T
      T = density () + 2
   }
} # end table
```

extension

Parameter for layer(s): conductor, device_li, diffusion, li, pcaux, poly, resistor, seed, src_drn

An optional conducting layer parameter that specifies the amount the conductor layer extends past the diffusion or contact layers. For diffusion layer, this specifies the amount diffusion extends past the contact layer.

Syntax

extension = *value*

Parameters

- *value*

  A floating point value used to specify the amount the conductor layer extends past the diffusion or contact layers.

Description

Extensions should be defined for layers that conform only to the MOS transistor. If the extension parameter is not specified for diffusion, src_drn, poly, or seed layers and conductor layers of front-end-of-line (FEOL) or middle-of-line (MOL), then the following warning is generated during calibration:

```
WARNING: Layer <layer_name> has no extension.
```

If a poly layer does not have an extension parameter specified but the seed layers do, then no warning message for the poly layer is generated.

Examples

```
extension = 0.05
```

extra_width

Parameter for layer(s): base_via, conductor, contact, device_li, diffusion, ground, li, poly, resistor, seed, src_drn, TSV, via

An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

Syntax

extra_width = *value*

Parameters

- *value*

  A floating point value used to specify the extra-width of side walls for a conductor with a trapezoid shape.

Description

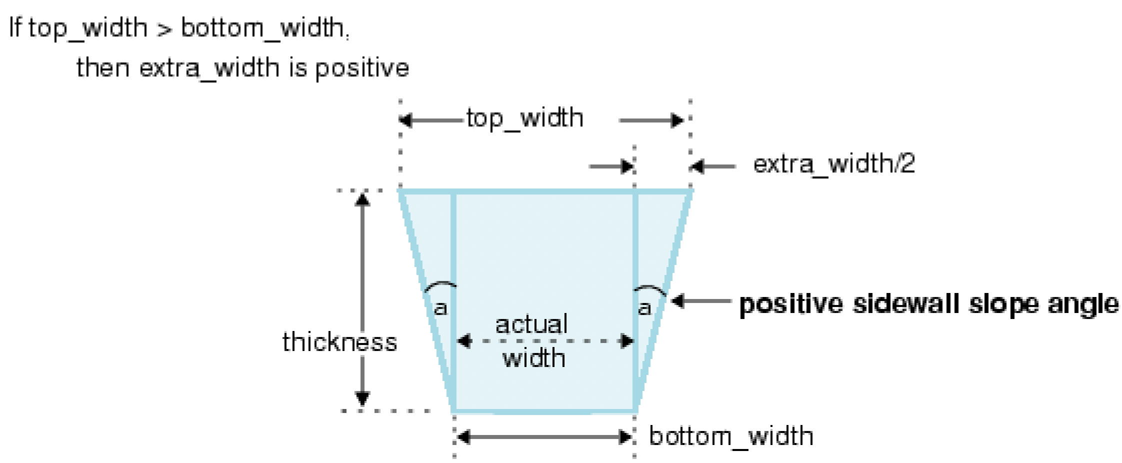
The extra-width of side walls for a conductor with a trapezoid shape is shown in Figure 6-27. To simplify this diagram, drawn width and actual width are shown as equal.

The value of extra_width is determined using one of the following two equations:

- extra_width = top_width - bottom_width
- extra_width = 2 * thickness * tan(a)

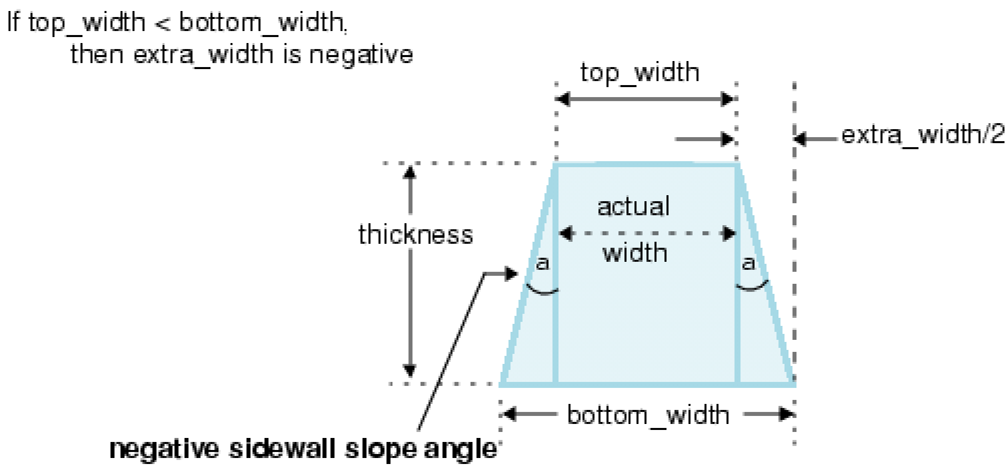
If top_width is greater than bottom_width, then the extra_width value is positive.

Figure 6-27. Extra_width for Positive Trapezoid Style



If top_width is less than bottom_width, then extra width value is negative.

Figure 6-28. Extra_width for Negative Trapezoid Style



For more information on the conductor keyword, see Conductor. For more information on the poly keyword, see Poly. For more information on the diffusion keyword, see Diffusion. For more information on the li keyword, see li.

Examples

```
extra_width = 0.05
```

extrapolation

Parameter for layer(s): base_via, contact, via

Optional parameter that specifies whether or not to extrapolate the variable via_resistance table boundary values.

Syntax

extrapolation = yes | no

Parameters

- yes

  Specifies to extrapolate the variable via_resistance table boundary values for the layer. This is the default if this parameter is not specified.

- no

  Specifies to limit the upper and lower via_resistance table boundary values. The variable boundaries are set by the overlap_width and overlap_length table syntax. Via dimensions are fixed.

Examples

To turn off extrapolation of resistance table values for *my_contact*, set the extrapolation parameter to *no* in the layer definition:

```
contact = my_contact {
  resistance = 1.1
  min_width = 0.13
  min_spacing = 0.15
  enclosure_up = 0.00
  enclosure_down = 0.00
  measured_from = polyX
  measured_to = metal1
  extrapolation = no
}
```

fin_bias

Parameter for layer(s): multigate

An optional multigate layer parameter used to specify the biasing effect on the fin layer.

Syntax

fin_bias = *value*

Parameters

- *value*

  A floating point value that specifies the biasing effect for fin width and fin spacing. The default is 0.

Description

The fin_bias parameter represents the biasing effect on the fin layers where:

actual_fin_width = drawn_fin_width + fin_bias

actual_fin_spacing = drawn_fin_spacing - fin_bias

When fin_bias is not specified, the values for fin_width and fin_spacing are considered actual values. For more information on the multigate layer definition, see Multigate.

Examples

```
multi_gate = fin_layer {
   fin_spacing = 0.023
   fin_width = 0.01
   fin_length = 0.010
   fin_thickness = 0.039
   fin_bias = 0.02
...
}
```

fin_count

Parameter for Table(s): INVERSE_RG_FACTOR_1 and INVERSE_RG_FACTOR_2 Tables

A table parameter whose values specify the number of fins in the device. Can only be used in the inverse_rg_factor_1 and inverse_rg_factor_2 tables.

Syntax

fin_count = {fc_1 fc_2 ... fc_n}

Parameters

- fc_1 fc_2 ... fc_n

  A list of one or more integer values that specifies the number of fins in the device enclosed in required braces ({ })

Examples

```
fin_count = { 2 3 }
```

fin_length

Parameter for layer(s): multigate

A required multigate layer parameter used to specify the length of the fin.

Syntax

fin_length = *value*

Parameters

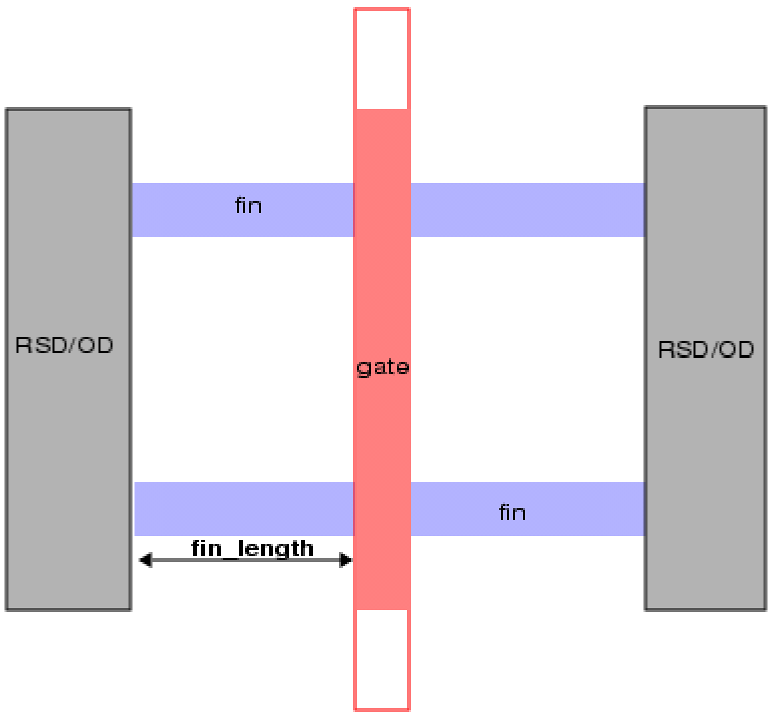
- *value*

  A floating point value that specifies the length of the fin.

Description

A required multigate layer parameter that specifies the length of the fin. This parameter is used only in multigate layer definitions. The fin_length is shown in Figure 6-29.

Figure 6-29. fin_length of the multigate layer



For more information on the multigate layer definition, see Multigate.

Examples

```
fin_length = 0.032
```

fin_spacing

Parameter for layer(s): multigate

A required multigate layer parameter used to specify the spacing between fins.

Syntax

fin_spacing = *value*

Parameters

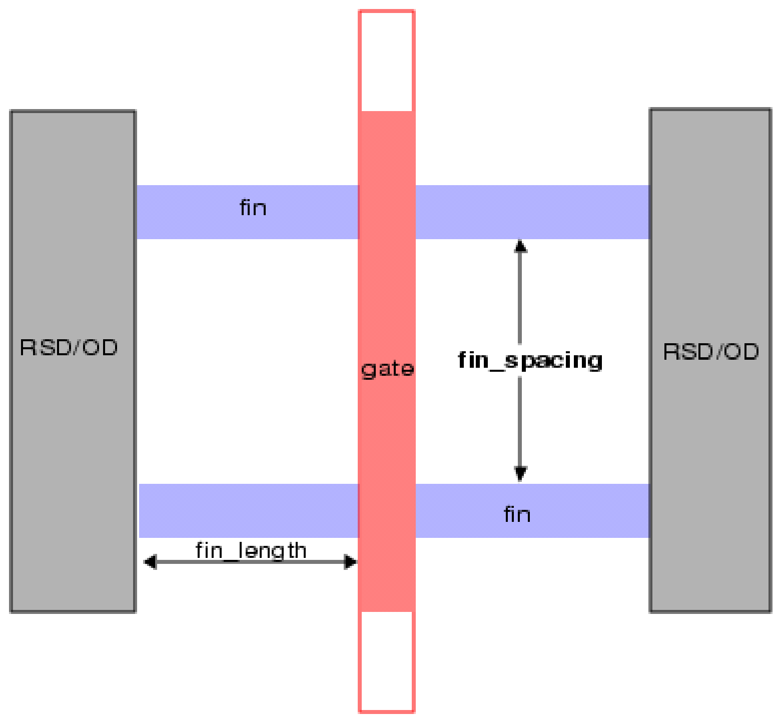
- *value*

  A floating point value that specifies the spacing between fins.

Description

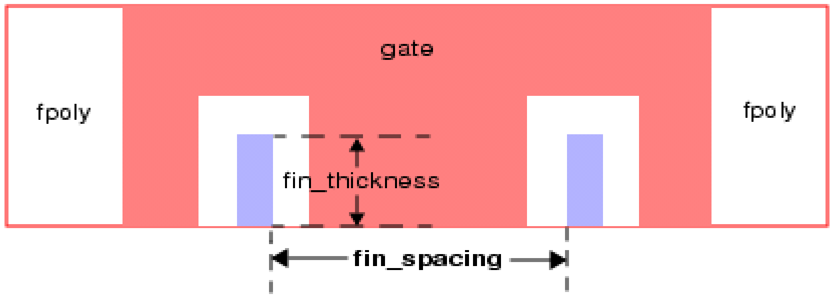
A required multigate layer parameter that specifies the spacing between fins. This parameter is used only in multigate layer definitions. The top-view of the fin_spacing is shown in Figure 6-30.

Figure 6-30. fin_spacing of the multigate layer top-view



The side view of the fin_spacing is shown in Figure 6-31.

Figure 6-31. fin_spacing of the multigate layer side-view



For more information on the multigate layer definition, see Multigate.

Examples

```
fin_spacing = 0.016
```

fin_thickness

Parameter for layer(s): multigate

A required multigate layer parameter used to specify the thickness of the fin.

Syntax

fin_thickness = *value*

Parameters

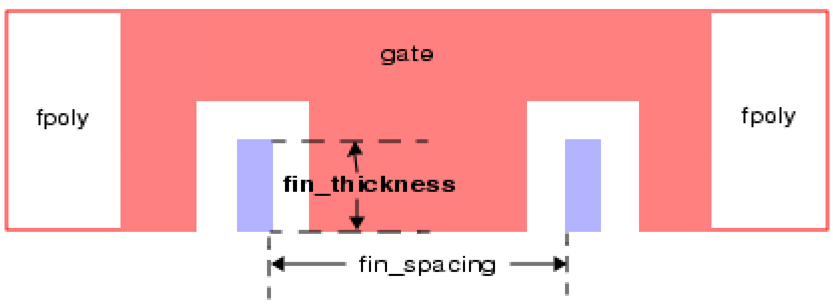
- *value*

  A floating point value that specifies the thickness of the fin.

Description

A required multigate layer parameter that specifies the thickness of the fin. This parameter is used only in multigate layer definitions. The fin_thickness is shown in Figure 6-32.

Figure 6-32. fin_thickness of the multigate layer



For more information on the multigate layer definition, see Multigate.

Examples

```
fin_thickness = 0.032
```

fin_width

Parameter for layer(s): multigate

A required multigate layer parameter used to specify the width of the fin.

Syntax

fin_width = *value*

Parameters

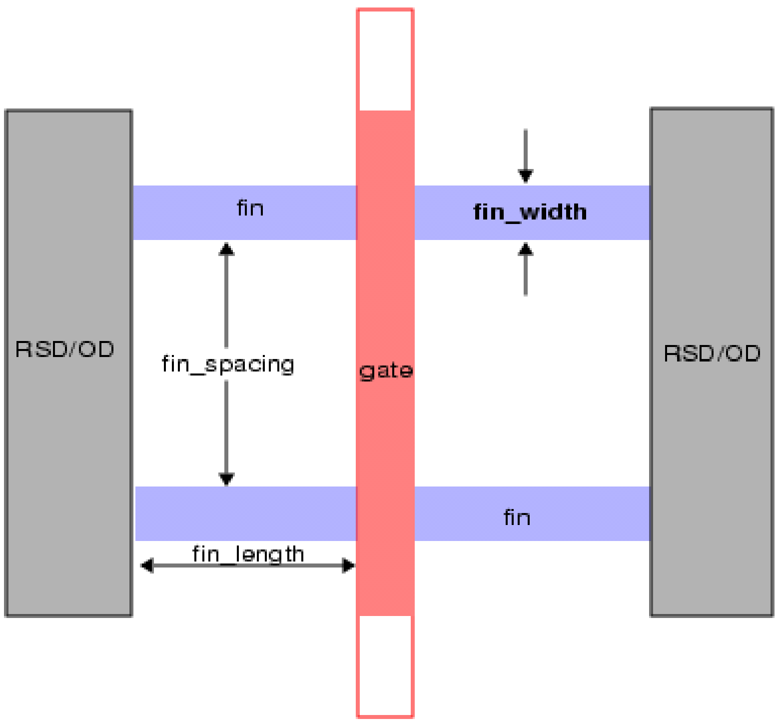
- *value*

  A floating point value that specifies the width of the fin.

Description

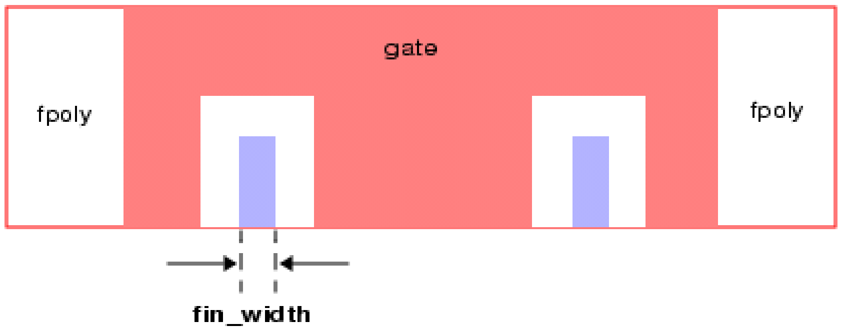
A required multigate layer parameter that specifies the width of the fin. This parameter is used only in multigate layer definitions. The top-view of the fin_width is shown in Figure 6-33.

Figure 6-33. fin_width of the multigate layer top-view



The side view of the fin_width is shown in Figure 6-34.

Figure 6-34. fin_width of the multigate layer side-view



For more information on the multigate layer definition, see Multigate.

Examples

```
fin_width = 0.016
```

gate_diffusion_layer_pair

Parameter for layer(s): multigate

A required list of one or more layer pairs used to specify which gate and diffusion layers can be paired together to describe different device types with the same device parameters.

Syntax

gate_diffusion_layer_pair = '{' (*layer1 layer2*) [**(*layer1 layer2*)**...] **'}'**

Parameters

- **(*layer1 layer2*)**

  A pair of gate and diffusion layers enclosed in parentheses used to describe a device.

Description

A required list of one or more layer pairs used to specify which gate and diffusion layers can be paired together to describe different device types with the same device parameters. The list of pairs must be specified in brackets; pairs must be specified in parentheses. This parameter is used only in multigate layer definitions. For more information on the multigate layer definition, see Multigate.

Examples

```
gate_diffusion_layer_pair = {(PGATE PDIFF) (NGATE NDIFF)}
```

gate_extension

Parameter for layer(s): multigate, seed

An optional multigate and seed layer parameter that specifies the distance the gate layer extends outside the diffusion area.

Syntax

gate_extension = *value*

Parameters

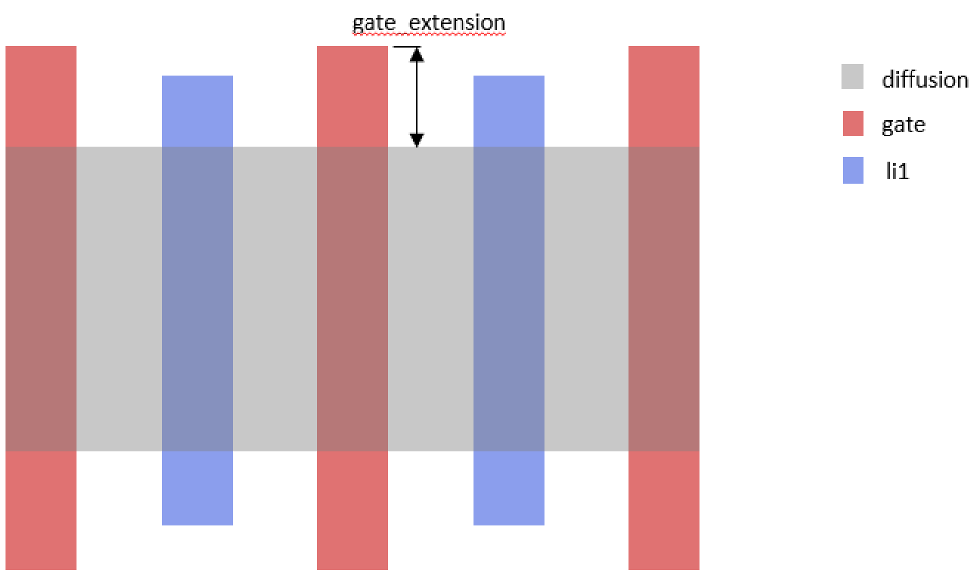
- *value*

  A floating point value in microns that specifies the distance the gate layer extends outside the diffusion area.

Description

Use this parameter to control gate resistance for more accurate modeling. The gate_extension parameter specifies the amount of the device poly layer that extends past the diffusion region for resistance calculation as shown in Figure 6-35.

Figure 6-35. gate_extension for FinFET



A warning is generated during calibration if this parameter is not specified in the multigate or seed layer definition.

Examples

gate_extension = 0.01

gate_oxide_er

Parameter for layer(s): multigate

A required multigate layer parameter used to specify the permittivity of the gate oxide.

Syntax

gate_oxide_er = *value*

Parameters

- *value*

  A floating point value that specifies the permittivity of the gate oxide.

Description

A required multigate layer parameter that specifies the permittivity of the gate oxide. This parameter is used only in multigate layer definitions. For more information on the multigate layer definition, see Multigate.

Examples

```
gate_oxide_er = 6.0
```

gate_oxide_side_t

Parameter for layer(s): multigate

An optional multigate layer parameter used to specify the side thickness of the gate oxide.

Syntax

gate_oxide_side_t = *value*

Parameters

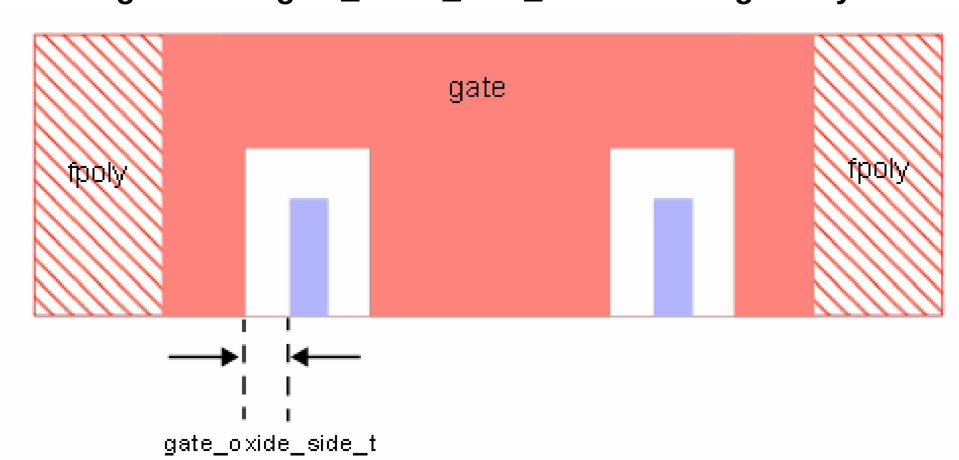
- *value*

  A floating point value that specifies the side thickness of the gate oxide.

Description

An optional multigate layer parameter that specifies the side thickness of the gate oxide. This parameter is used only in multigate layer definitions. The gate_oxide_side_t is shown in Figure 6-36.

Figure 6-36. gate_oxide_side_t of the multigate layer



For more information on the multigate layer definition, see Multigate.

Examples

```
gate_oxide_side_t = 6.0
```

gate_oxide_top_t

Parameter for layer(s): multigate

A required multigate layer parameter used to specify the top thickness of the gate oxide.

Syntax

gate_oxide_top_t = *value*

Parameters

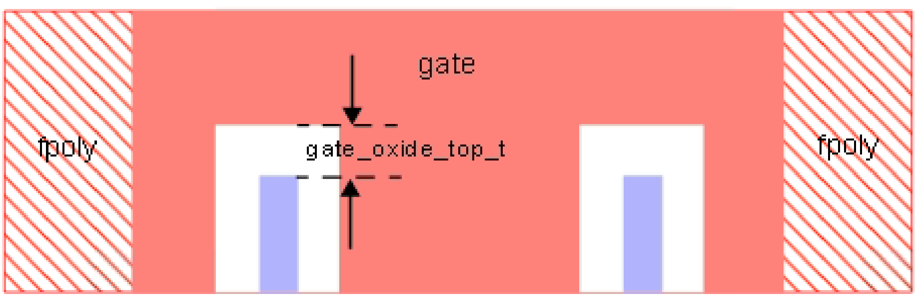
- *value*

  A floating point value that specifies the top thickness of the gate oxide.

Description

A required multigate layer parameter that specifies the top thickness of the gate oxide. This parameter is used only in multigate layer definitions. The gate_oxide_top_t is shown in Figure 6-37.

Figure 6-37. gate_oxide_top_t of the multigate layer



For more information on the multigate layer definition, see Multigate.

Examples

```
gate_oxide_top_t = 6.0
```

gate_poly_side_t

Parameter for layer(s): multigate

An optional multigate layer parameter used to specify the gate poly thickness between the side of the gate poly and the side of the gate oxide.

Syntax

gate_poly_side_t = *value*

Parameters

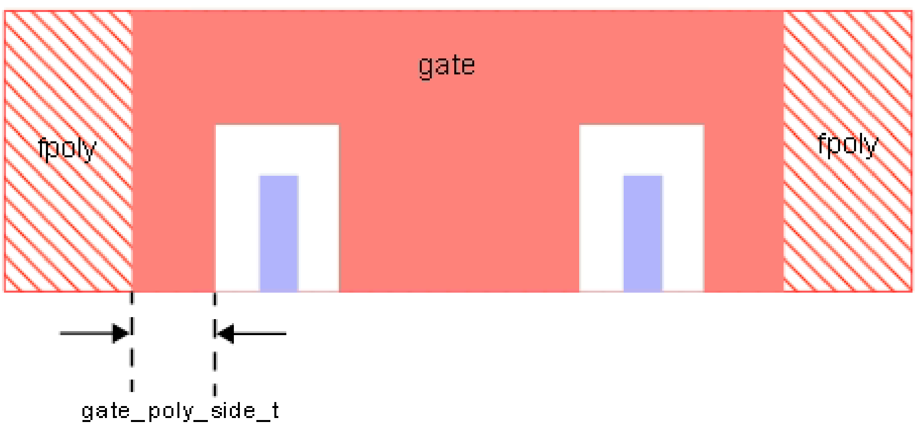
- *value*

  A floating point value that specifies the gate poly thickness between the side of the gate poly and the side of the gate oxide.

Description

An optional multigate layer parameter that specifies the gate poly thickness between the side of the gate poly and the side of the gate oxide. This parameter is used only in multigate layer definitions. The gate_poly_side_t is shown in Figure 6-38.

Figure 6-38. gate_poly_side_t of the multigate layer



For more information on the multigate layer definition, see Multigate.

Examples

```
gate_poly_side_t = 6.0
```

gate_poly_top_t

Parameter for layer(s): multigate

A required multigate layer parameter used to specify the gate poly thickness between the top of the gate poly and the top of the gate oxide.

Syntax

gate_poly_top_t = *value*

Parameters

- *value*

  A floating point value that specifies the gate poly thickness between the top of the gate poly and the top of the gate oxide.

Description

A required multigate layer parameter that specifies the gate poly thickness between the top of the gate poly and the top of the gate oxide. This parameter is used only in multigate layer definitions. The gate_poly_top_t is shown in Figure 6-39.

Figure 6-39. gate_poly_top_t of the multigate layer

For more information on the multigate layer definition, see Multigate.

Examples

```
gate_poly_top_t = 6.0
```

gate_to_cont_min_spacing

Parameter for layer(s): pcaux, seed

An optional pcaux and seed layer parameter that specifies a diffusion contact within a gate or seed layer type. This parameter specifies the minimum spacing between the gate poly layer and the diffusion contact layer (the contact from diffusion to metal1) in a MOS transistor structure. If a contact layer is specified with gate_to_cont_spacing_min, and a pcaux or seed layer is specified with gate_to_cont_min_spacing, the setting for gate_to_cont_spacing_min takes precedence.

Syntax

gate_to_cont_min_spacing = *value*

Parameters

- *value*

  A floating point value used to specify the minimum spacing between the gate poly layer and the diffusion contact layer.

Examples

```
gate_to_cont_min_spacing = 0.05
```

gate_to_cont_spacing_min

Parameter for layer(s): contact

An optional contact layer parameter used only for diffusion contacts. This parameter specifies the minimum allowed spacing between the gate and diffusion contact in a MOS transistor structure. If a contact layer is specified with gate_to_cont_spacing_min, and a pcaux or seed layer is specified with gate_to_cont_min_spacing, the setting for gate_to_cont_spacing_min takes precedence.

Syntax

gate_to_cont_spacing_min = *value*

Parameters

- *value*

  A floating point value used to specify the minimum allowed gate to contact spacing.

Examples

```
gate_to_cont_spacing_min = 0.05
```

gate_to_LI1_min_spacing

Parameter for layer(s): pcaux, seed

An optional pcaux and seed layer parameter for all contact types except diffusion contact. This parameter specifies the minimum spacing between the gate poly layer and the li layer touching the diffusion layer.

Syntax

gate_to_LI1_min_spacing = *value*

Parameters

- *value*

  A floating point value used to specify the minimum spacing between the gate poly layer and the li layer touching the diffusion layer.

Examples

```
gate_to_LI1_min_spacing = 0.05
```

gate_to_LI2_min_spacing

Parameter for layer(s): pcaux, seed

An optional pcaux and seed layer parameter for all contact types except diffusion contact. This parameter specifies the minimum spacing between the gate poly layer and the second li layer.

Syntax

gate_to_LI2_min_spacing = *value*

Parameters

- *value*

  An optional floating point value used to specify the minimum spacing between the gate poly layer and the second li layer.

Examples

```
gate_to_LI2_min_spacing = 0.05
```

gate_to_via_min_spacing

Parameter for layer(s): pcaux, seed

An optional pcaux and seed layer parameter for all contact types except diffusion contact. This parameter specifies the minimum spacing between the gate poly layer and the diffusion via layer.

Syntax

gate_to_via_min_spacing = *value*

Parameters

- *value*

  A floating point value used to specify the minimum spacing between the gate poly layer and the diffusion via layer.

Examples

```
gate_to_via_min_spacing = 0.05
```

gate_to_via_spacing_min

Parameter for layer(s): via

An optional via layer parameter used to specify the minimum allowed gate to via spacing.

Syntax

gate_to_via_spacing_min = *value*

Parameters

- *value*

  A floating point value used to specify the minimum allowed gate to via spacing.

Examples

```
gate_to_via_spacing_min = 0.05
```

ground

Type: Ground layer

Part of the layer definition syntax, the ground layer definition keyword is used to define the electrical and physical characteristics of conducting ground layers. The ground keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. Layers of this type can be specified multiple times.

Syntax

ground = *layer_name* {
  thickness =*z-direction_ metal_ thickness*
  min_width = *minimum_allowed_metal_width*
  min_spacing = ***minimum_allowed_metal_spacing***
  resistivity = ***resistivity_value***
  r_sheet = ***sheet_resistance***
  thickness_type = absolute | relative
  ztop = *top_ z-coordinate*
  zbottom = *bottom_z-coordinate*
  measured_from = *layername*
  coplanar_min_spacing = *value*
  tc1 = *resistance_temperature1_coefficient*
  tc2 = *resistance_temperature2_coefficient*
  max_rlength = *length_for_resistance_fracturing*
  max_width = *maximum_allowed _metal_width*
  max_spacing = *maximum_allowed_metal_spacing*
  layer_bias = *bias_actual_width_relative_to_drawn_width*
  trap_style = top | middle | bottom # where trapezoid measurements are taken from
  extra_width = *extra-width_of_side_walls* # for a conductor with a trapezoid shape
  swstep = *count*
  ignore_caps = yes | no
  virtual = yes | no
  ronly_layers = '{'*space_delimited_list_of_r-only_layer_names*'}'
  density_window = '{''{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'...'{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}''}'
  copy = '{'*space_delimited_list_of_layer_names*'}'
  hidden = false | true
  }

Parameters

- ***layer_name***

  A required unique user-specified name.

- **thickness**

  A required layer parameter that specifies the thickness of the metal layer in the z-direction.

- **min_width**

  A required layer parameter that specifies the minimum allowed drawn width of metal objects on this layer.

- **min_spacing**

  A required layer parameter that specifies the minimum allowed drawn spacing between conductors on this layer.

- **resistivity**

  A required layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- **r_sheet**

  A required layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- measured_from

  An optional conducting layer parameter that measures zbottom or ztop relative to the specified layer.

- coplanar_min_spacing

  An optional conducting layer parameter that specifies the minimum spacing value between coplanar layers.

- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.

- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.

- max_rlength

  An optional conducting layer parameter that specifies the maximum length of a wire. This parameter is used for resistance fracturing, which determines how many pieces a resistor should be broken into for representation in a distributed network.

- max_width

  An optional parameter that specifies the maximum allowed metal width.

- max_spacing

  An optional parameter that specifies the maximum allowed metal spacing.

- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width relative to drawn width.

- trap_style

  An optional conducting layer parameter that defines where trapezoid measurements are taken from. Permitted values are top, middle, or bottom.

- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- swstep

  An optional conducting layer parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.

- extension

  An optional conducting layer parameter that specifies the amount this layer extends past diffusion or contact layers.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- virtual

  An optional ground layer parameter that specifies whether or not to treat the ground layer as virtual. The default is no, if this parameter is not specified.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- density_window

  An optional conducting layer parameter specified as {{w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$} ... {w_n h_n $factor_n$ $stepx_n$ $stepy_n$}}. The braces are required for grouping.

- copy

  An optional conducting layer parameter that specifies a space delimited list of layer names enclosed in braces ({ }) to be mapped to ground layers under the gate.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Examples

Example 1

The following is an example of a ground layer definition:

```
ground = my_ground {
   zbottom = -0.01
   thickness = 0.1
   resistivity = 11
}
```

Example 2

The following is an example of a virtual ground layer definition:

```
ground = my_virtual_ground {
   zbottom = -0.01
   thickness = 0.1
   resistivity = 11
   virtual = yes
}
```

half_node_scale_factor

Type: Global Parameters

An optional global parameter that specifies the amount by which to scale the layout. It can only be specified once. This parameter is required for Calibre xACT and Calibre xACT 3D flows.

Syntax

half_node_scale_factor = *scale*

Parameters

- *scale*

  A positive floating point value used to specify the amount by which to scale the layout.

Description

Specifies the amount by which to scale the layout. Scaling affects a polygon's width, length, and spacing properties. Calibration does not scale the layout, but generates the PEX Magnify SVRF statement with the specified scale value. The default is to not produce the PEX Magnify statement when this parameter has not been specified.

For Calibre xACT and Calibre xACT 3D, your MIPT file must contain this parameter in order to ensure that scaling is properly handled.

Examples

Specifying the following global parameter in your MIPT file

```
half_node_scale_factor = 0.75
```

generates the following SVRF statement in your rules.C and rules.R files:

```
PEX MAGNIFY 0.75
```

height

Parameter for layer(s): TSV

A required TSV layer parameter used to specify the height of the through silicon via.

Syntax

height = *value*

Parameters

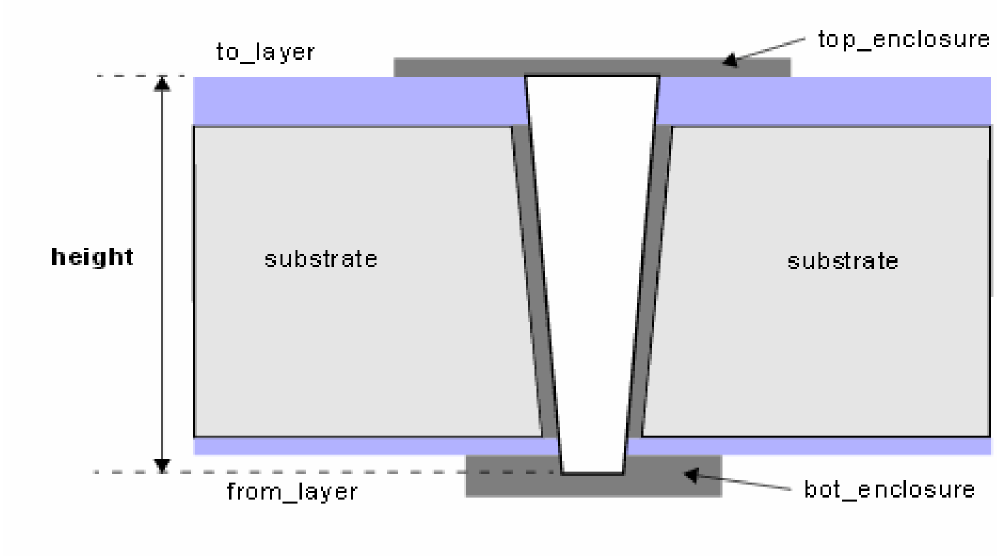
- *value*

  A floating point value that specifies the height of the through silicon via.

Description

A required TSV parameter that specifies the height of the through silicon via. This parameter is used only in TSV layer definitions. The height is shown in Figure 6-40.

Figure 6-40. height of TSV Layer



For more information on TSV layer type, see TSV.

Examples

```
height = 10.4
```

hidden

Parameter for layer(s): all Layer Definitions

An optional layer parameter that controls whether or not the layer is encrypted in the calibrated rule files. Can be used in all layer type definitions.

Syntax

hidden = false | true

Parameters

- false

  Specifies not to encrypt the layer in the calibrated rule files. This is the default.

- true

  Specifies to encrypt the layer in the calibrated rule files.

Examples

To encrypt the my_diel_mipt layer information in the calibrated rule files, specify hidden=true in the layer definition:

```
dielectric = my_diel_mipt {
   diel_type = mask
   zbottom = 0.071
   thickness = 11.0
   eps = 1.69
   hidden = true
}
```

hollow_radius

Parameter for layer(s): TSV

A required TSV layer parameter used to specify the radius of the hollow area for through silicon via.

Syntax

hollow_radius = *value*

Parameters

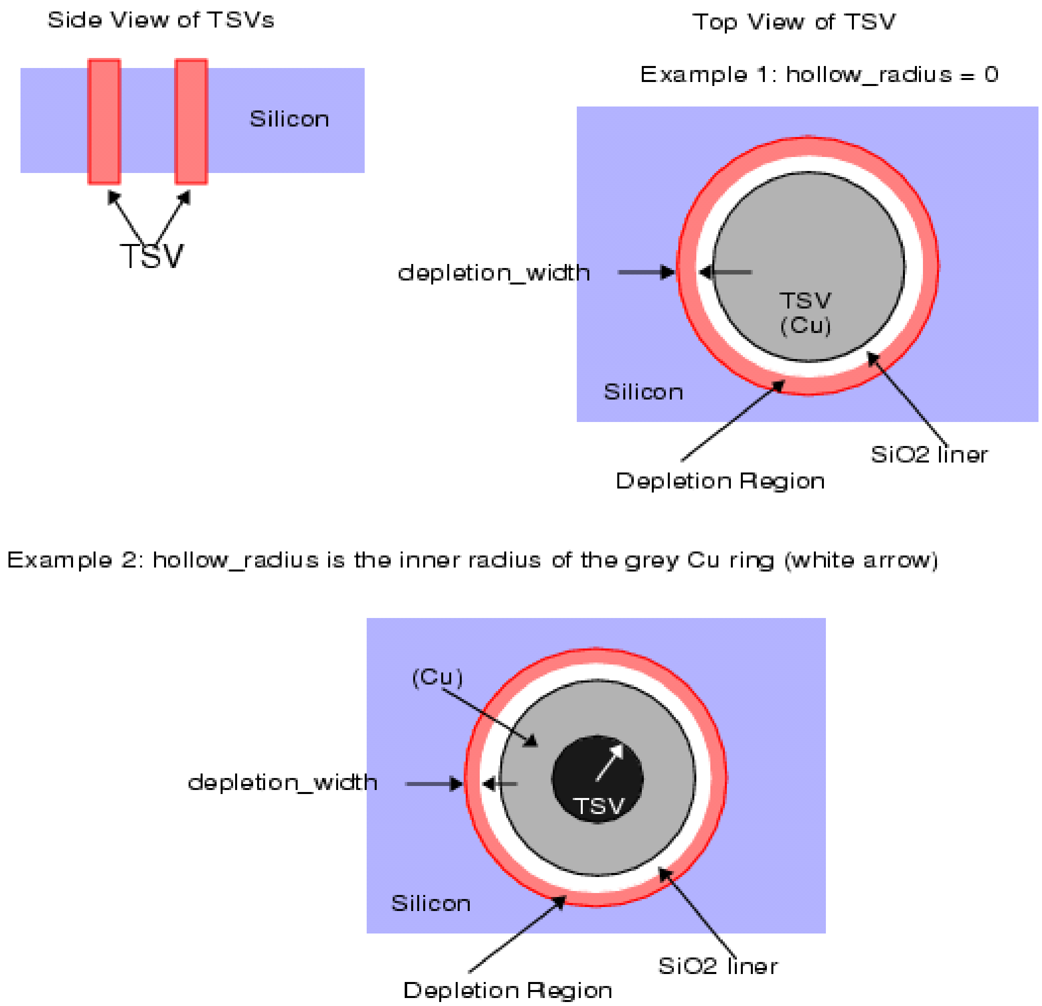
- *value*

  A floating point value used to specify the radius of the hollow area.

Description

A required TSV layer parameter that specifies the radius of the hollow area for through silicon via. The hollow_radius measurement is shown in Figure 6-41.

Figure 6-41. hollow_radius for TSV Layers



For more information on TSV layer type, see TSV.

Examples

Example 1

In Figure 6-41, Example 1 is an example of a solid TSV. In this case hollow radius is 0.

```
hollow_radius = 0
```

Example 2

In Figure 6-41, Example 2 is an example of a hollow TSV. In this case the hollow radius is a non-zero value, for example:

```
hollow_radius = 0.87
```

ignore_caps

Parameter for layer(s): base_via, conductor, contact, derived, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn, via

An optional parameter that controls whether or not all capacitance for the layer is ignored during extraction. This parameter may be specified for all conducting layers and via layers. For derived layers, this parameter can only be specified if the derived_type parameter is set to ronly or ronly_contact.

Syntax

ignore_caps = yes | no

Parameters

- yes

  Specifies to ignore the capacitance for the layer.

- no

  Specifies to include the capacitance for the layer. This is the default.

Examples

```
ignore_caps = yes
```

ignore_diff_intrinsic

Parameter for layer(s): seed

An optional seed layer parameter that controls whether or not capacitance from diffusion to ground is ignored by calibration when generating the calibrated rule files.

Syntax

ignore_diff_intrinsic = yes | no

Parameters

- yes

  Specifies to ignore the capacitance from diffusion to ground. This is the default.

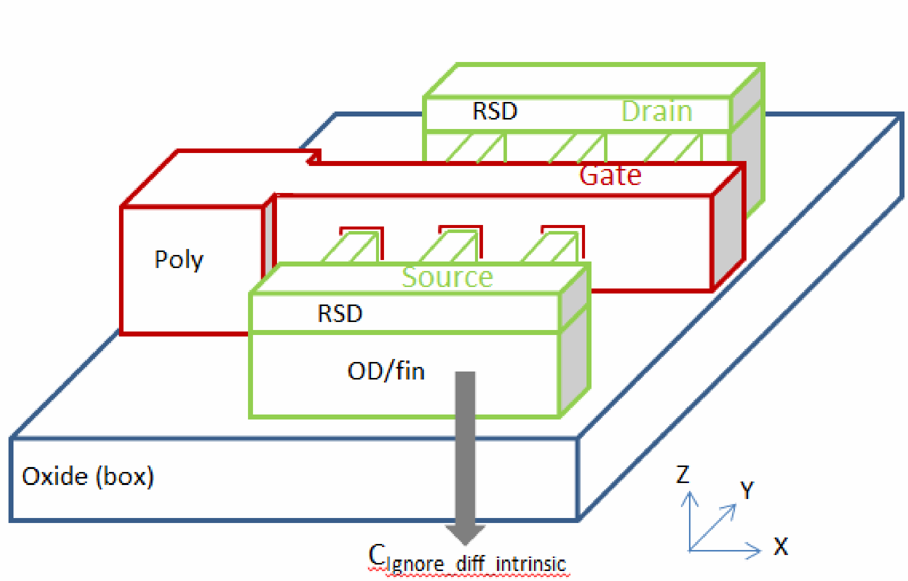
- no

  Specifies to include the capacitance from diffusion to ground. A warning is generated during calibration when this parameter is set to no.

Description

The optional ignore_diff_intrinsic parameter controls whether or not capacitance from diffusion to ground is ignored. For example, in Figure 6-42 the ignored capacitance would be from OD/fin to ground.

Figure 6-42. Ignore_diff_intrinsic Parameter for Seed Layer



See Seed for more information on the seed layer type.

Examples

```
ignore_diff_intrinsic = no
```

ignore_diff_to_diff_under_poly

Parameter for layer(s): pcaux, seed

An optional pcaux and seed layer parameter that controls whether or not diffusion to diffusion under poly capacitance is ignored by calibration when generating the calibrated rule files.

Syntax

ignore_diff_to_diff_under_poly = yes | no

Parameters

- yes

  Specifies to ignore the diffusion to diffusion under poly capacitance. This is the default.

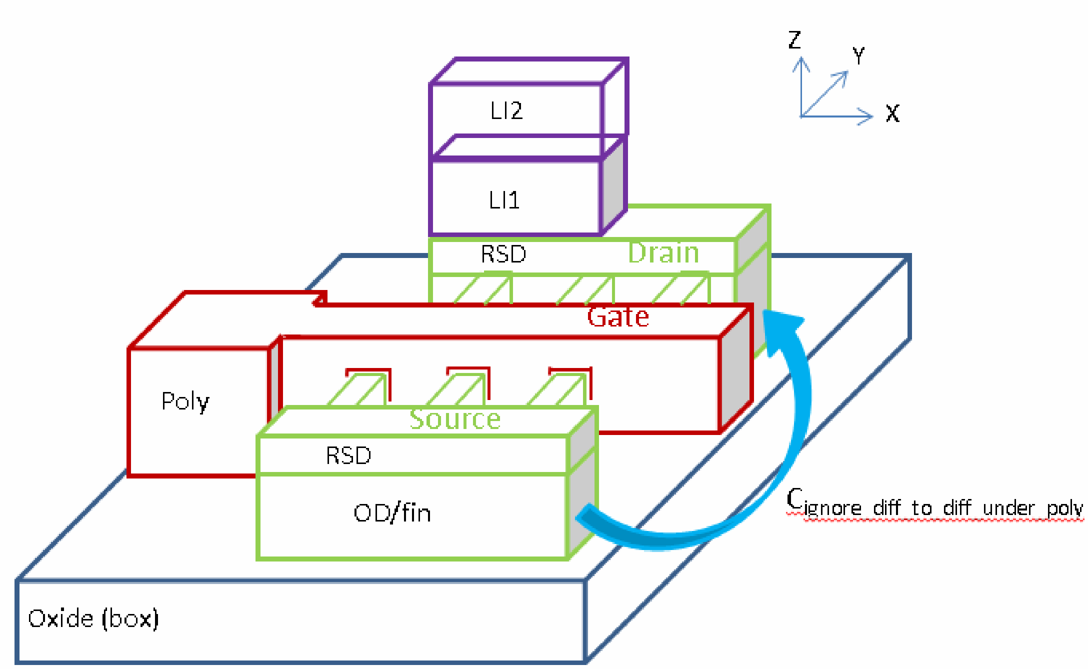
- no

  Specifies to include the diffusion to diffusion under poly capacitance. A warning is generated during calibration when this parameter is set to no.

Description

Figure 6-43 shows the capacitance controlled by the ignore_diff_to_diff_under_poly parameter.

Figure 6-43. Capacitance for Ignore_diff_to_diff_under_poly Parameter



Examples

```
seed = NAGATE3 {
   thickness = 0.04300
   min_width = 0.02700
   min_spacing = 0.07200
   resistivity = 1.6984
   ignore_endcap = yes
   ignore_diff_to_diff_under_poly = no
   ignore_gate_to_diff = yes
}
```

ignore_endcap

Parameter for layer(s): pcaux, seed

An optional pcaux and seed layer parameter that controls whether or not poly extension to source and drain coupling is ignored by calibration when generating the calibrated rule files.

Syntax

ignore_endcap = yes | no

Parameters

- yes

  Specifies to ignore the poly extension to source and drain coupling.

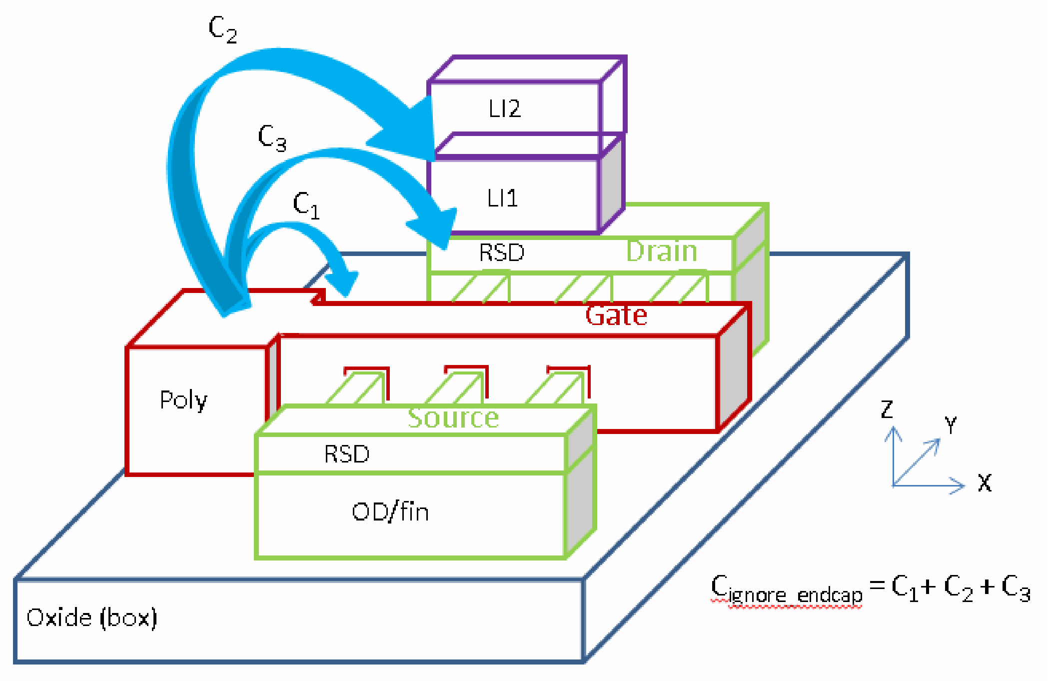
- no

  Specifies to include the poly extension to source and drain coupling. This is the default.

Description

Figure 6-44 shows the capacitance controlled by the ignore_endcap parameter.

Figure 6-44. Capacitance affected by Ignore_endcap Parameter



Examples

```
seed = NAGATE3 {
   thickness = 0.04300
   min_width = 0.02700
   min_spacing = 0.07200
   resistivity = 1.6984
   ignore_endcap = yes
   ignore_diff_to_diff_under_poly = no
   ignore_gate_to_diff = yes
}
```

ignore_gate_intrinsic

Parameter for layer(s): seed

An optional seed layer parameter that controls whether or not capacitance from the gate poly layer to ground is ignored by calibration when generating the calibrated rule files.

Syntax

ignore_gate_intrinsic = <u>yes</u> | no

Parameters

- <u>yes</u>

  Specifies to ignore the capacitance from the gate poly layer to ground. This is the default.

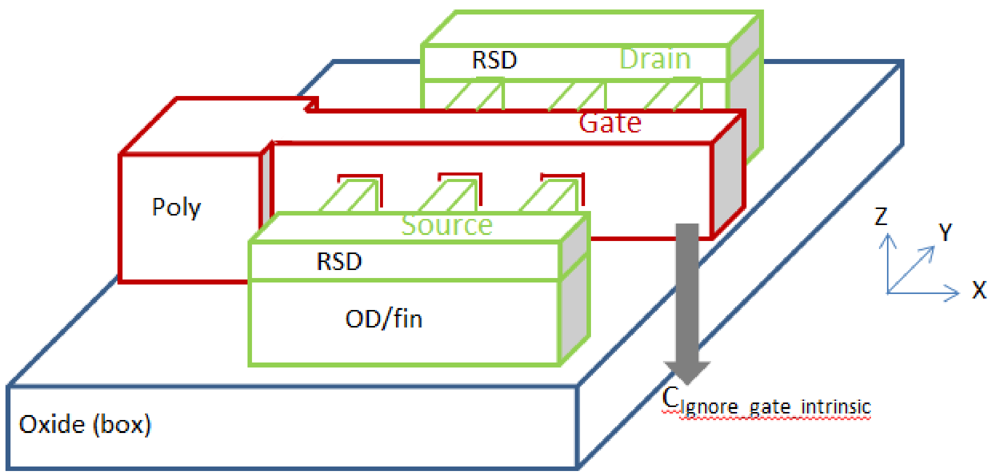
- no

  Specifies to include the capacitance from the gate poly layer to ground.

Description

The optional ignore_gate_intrinsic parameter controls whether or not capacitance from the gate poly layer to ground is ignored. For example, in Figure 6-45 the ignored capacitance would be from gate poly to ground.

Figure 6-45. Ignore_gate_intrinsic Parameter for Seed Layer



See Seed for more information on the seed layer type.

Examples

```
ignore_gate_intrinsic = no
```

ignore_gate_to_diff

Parameter for layer(s): pcaux, seed

An optional pcaux and seed layer parameter that controls whether or not the coupling capacitance between gates to diffusion is ignored by calibration when generating the calibrated rule files.

Syntax

ignore_gate_to_diff = yes | no

Parameters

- yes

  Specifies to ignore the coupling capacitance between gates to diffusion. This is the default.

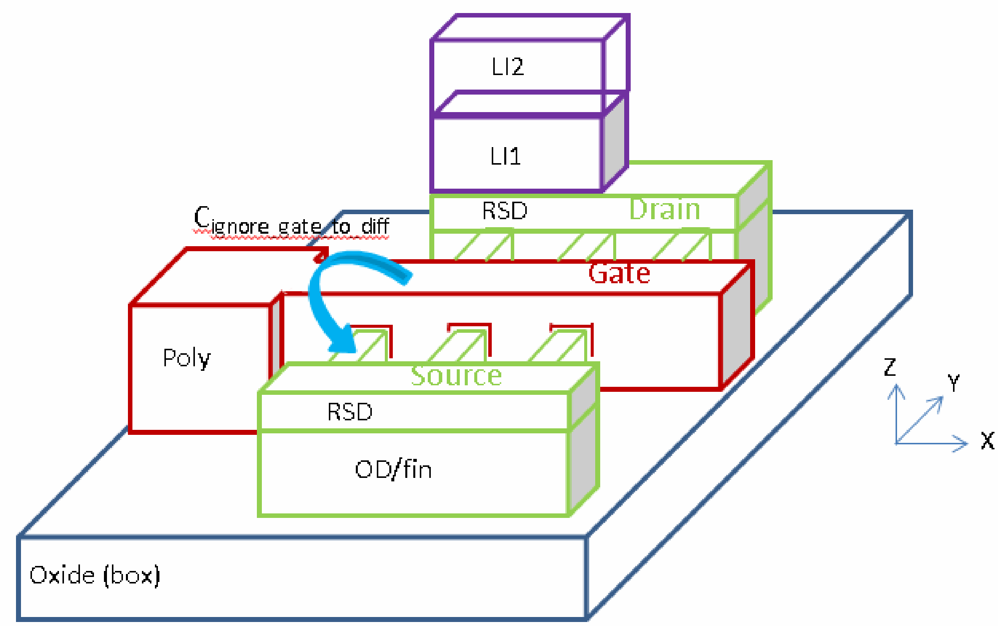
- no

  Specifies to include the coupling capacitance between gates to diffusion. A warning is generated during calibration when this parameter is set to no.

Description

Figure 6-46 shows the capacitance controlled by the ignore_gate_to_diff parameter.

Figure 6-46. Capacitance for Ignore_gate_to_diff Parameter



Examples

```
seed = NAGATE3 {
   thickness = 0.04300
   min_width = 0.02700
   min_spacing = 0.07200
   resistivity = 1.6984
   ignore_endcap = yes
   ignore_diff_to_diff_under_poly = no
   ignore_gate_to_diff = yes
}
```

ignore_gateext_to_diff

Parameter for layer(s): pcaux, seed

An optional pcaux and seed layer parameter that controls whether or not the coupling capacitance between the gate extension to diffusion and rsd is ignored by calibration when generating the calibrated rule files.

Syntax

ignore_gateext_to_diff = yes | no

Parameters

- yes

  Specifies to ignore the coupling capacitance between gate extension to diffusion and rsd.

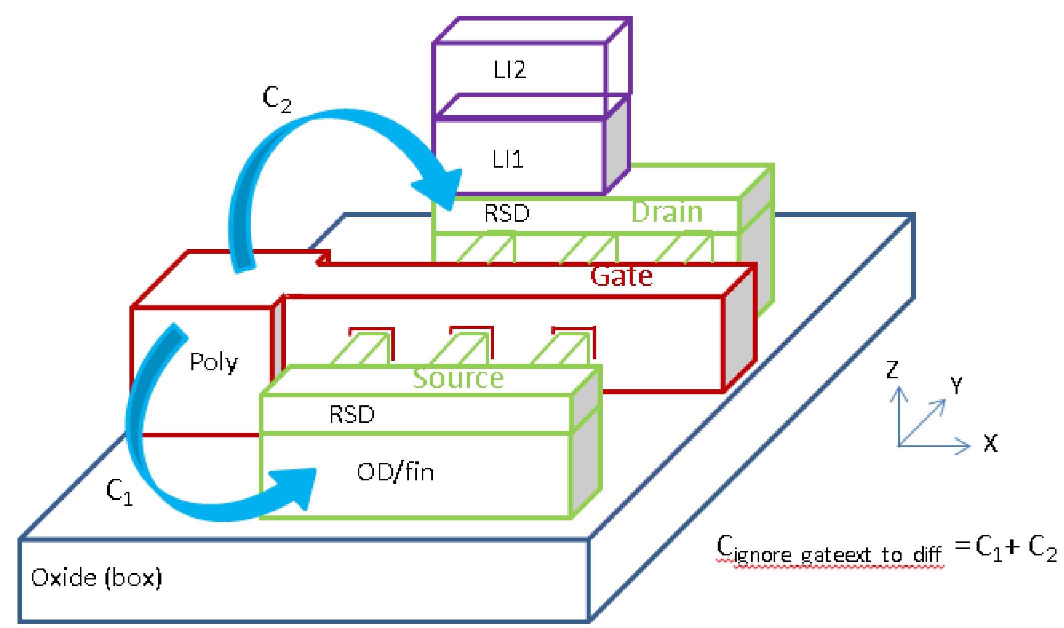
- no

  Specifies to include the coupling capacitance between gate extension to diffusion and rsd. This is the default.

Description

Figure 6-47 shows the capacitance controlled by the ignore_gateext_to_diff parameter.

Figure 6-47. Capacitance for Ignore_gateext_to_diff Parameter



This parameter is ignored if it is set to yes with ignore_endcap parameter set to yes, and issues the following warning:

```
WARNING: "Ignore the ignore_gateext_to_diff parameter. SVRF is generated
based on ignore_endcap parameter setting".
```

Examples

```
seed = NAGATE3 {
   thickness = 0.04300
   min_width = 0.02700
   min_spacing = 0.07200
   resistivity = 1.6984
   ignore_gate_to_diff = no
   ignore_gateext_to_diff = yes
}
```

ignore_gateext_to_diff_only

Parameter for layer(s): pcaux, seed

An optional pcaux and seed layer parameter that controls whether or not the coupling capacitance between the gate extension to diffusion is ignored by calibration when generating the calibrated rule files.

Syntax

ignore_gateext_to_diff_only = yes | <u>no</u>

Parameters

- yes

  Specifies to ignore the coupling capacitance between gate extension to diffusion. A warning is generated during calibration when this parameter is set to yes.

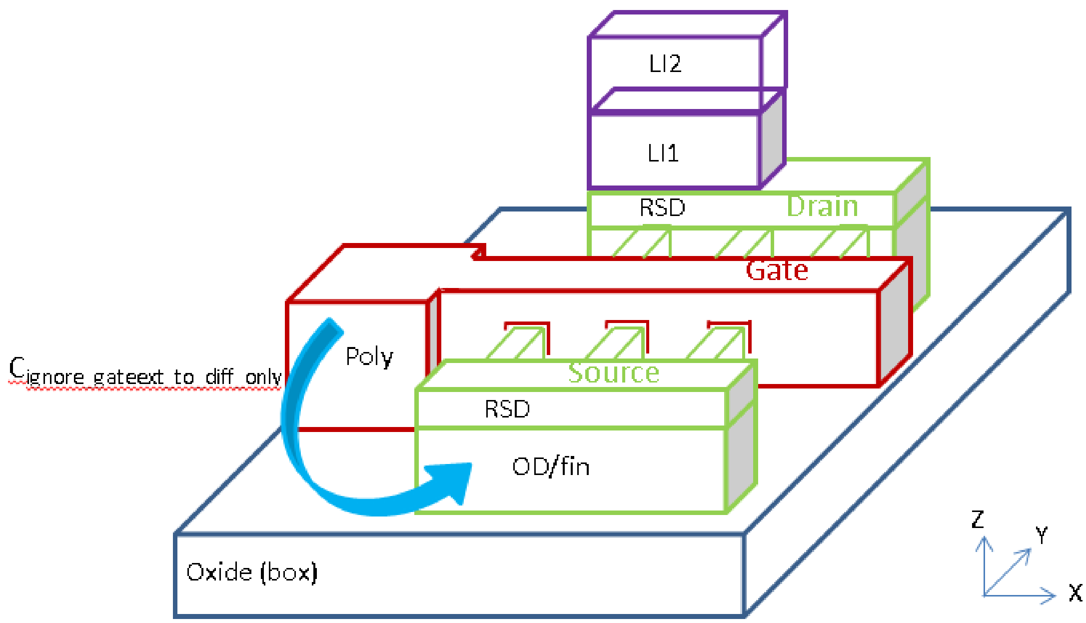
- <u>no</u>

  Specifies to include the coupling capacitance between gate extension to diffusion. This is the default.

Description

Figure 6-48 shows the capacitance controlled by the ignore_gateext_to_diff_only parameter. Note that capacitance from the gate extension to diffusion is ignored regardless of how the ignore_endcap parameter is specified.

Figure 6-48. Capacitance for Ignore_gateext_to_diff_only Parameter



Examples

```
seed = NAGATE3 {
   thickness = 0.04300
   min_width = 0.02700
   min_spacing = 0.07200
   resistivity = 1.6984
   ignore_gate_to_diff = no
   ignore_gateext_to_diff_only = yes
}
```

ignore_li1

Parameter for layer(s): seed

An optional seed layer parameter that controls whether or not the coupling capacitance between gate to li1 and poly to li1 in FinFET structures is ignored by calibration when generating the calibrated rule files.

Syntax

ignore_li1 = yes | no

Parameters

- yes

  Specifies to ignore the coupling capacitance between gate to li1 and poly to li1.

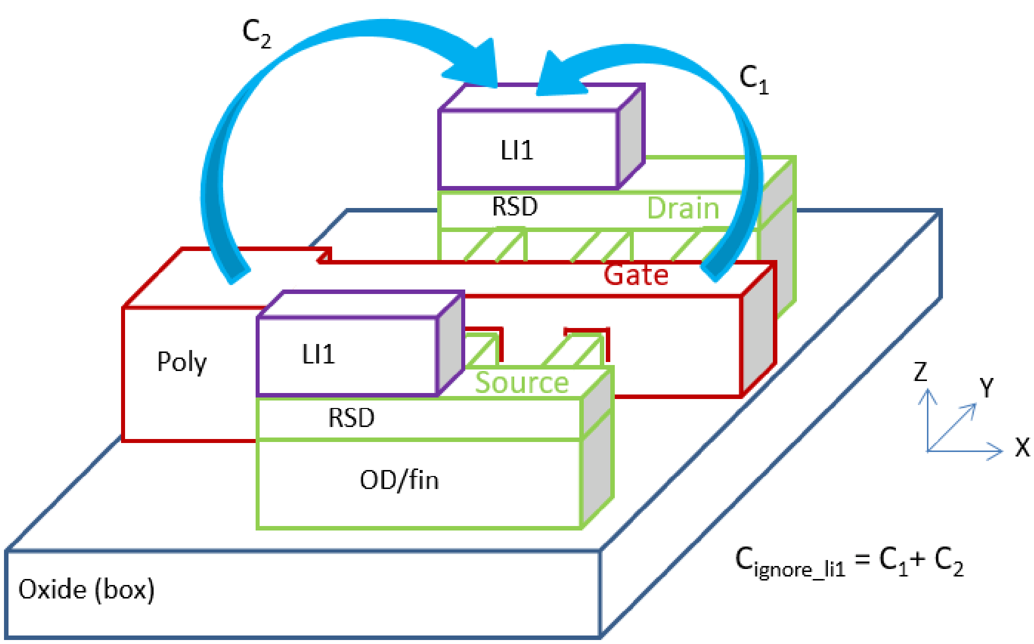
- no

  Specifies to include the coupling capacitance between gate to li1 and poly to li1 This is the default.

Description

The optional ignore_li1 parameter controls whether or not the coupling capacitance between gate to li1 and between poly to li1 is ignored. Figure 6-49 shows the capacitance controlled by the ignore_li1 parameter.

Figure 6-49. Capacitance Controlled by ignore_li1 Parameter



Examples

```
seed = NAGATE3 {
     thickness = 0.04300
     min_width = 0.02700
     min_spacing = 0.07200
     resistivity = 1.6984
     ignore_li1 = yes
     ignore_endcap = yes
     ignore_diff_to_diff_under_poly = no
     ignore_gate_to_diff = yes
}
```

ignore_rsd

Parameter for layer(s): seed

An optional seed layer parameter that controls whether or not the poly to raised source/drain coupling is ignored by calibration.

Syntax

ignore_rsd = yes | no

Parameters

- yes

  Specifies to ignore the capacitance between the poly and raised source/drain layer.

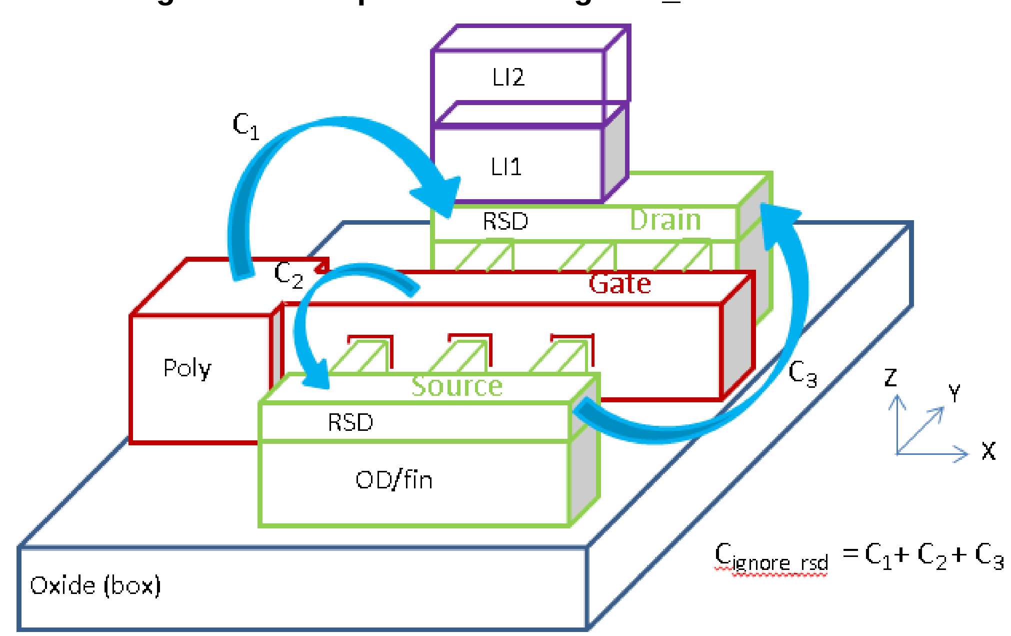
- no

  Specifies to include the capacitance between the poly and raised source/drain layer. This is the default.

Description

The optional ignore_rsd parameter controls whether or not capacitance from the poly to raised source/drain coupling is ignored. For example, in Figure 6-50 the ignored capacitance would be from rsd to poly.

Figure 6-50. Capacitance for Ignore_rsd Parameter



Examples

```
seed = NAGATE3 {
   thickness = 0.04300
   min_width = 0.02700
   min_spacing = 0.07200
   resistivity = 1.6984
   ignore_gate_to_diff = no
   ignore_rsd = yes
}
```

ignore_rsd_intrinsic

Parameter for layer(s): seed

An optional seed layer parameter that controls whether or not the capacitance between the raised source/drain layer and ground is ignored by calibration.

Syntax

ignore_rsd_intrinsic = yes | no

Parameters

- yes

  Specifies to ignore the capacitance between the raised source/drain layer and ground.

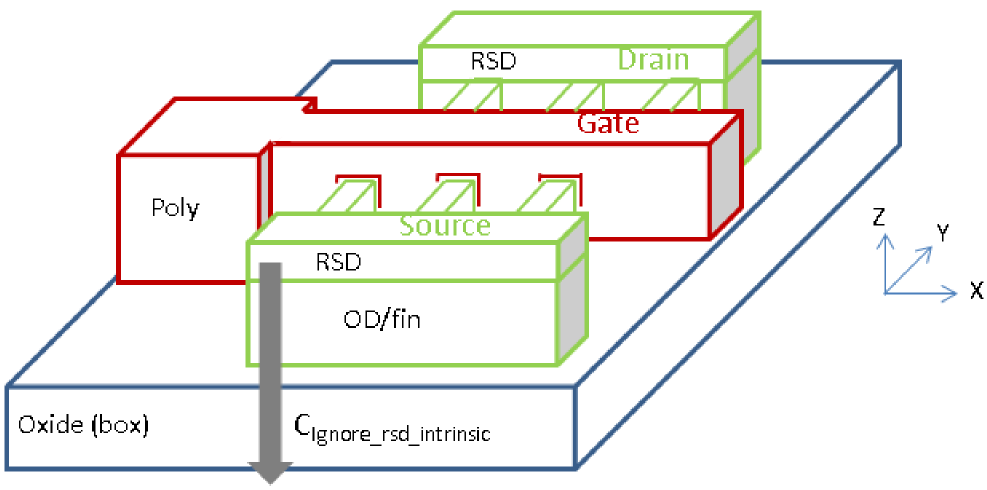
- no

  Specifies to include the capacitance between the raised source/drain layer and ground. This is the default.

Description

The optional ignore_rsd_intrinsic parameter controls whether or not capacitance from the rsd layer to ground is ignored. For example, in Figure 6-51 the ignored capacitance would be from rsd to ground.

Figure 6-51. Capacitance for Ignore_rsd_intrinsic Parameter



Examples

```
seed = NAGATE3 {
   thickness = 0.04300
   min_width = 0.02700
   min_spacing = 0.07200
   resistivity = 1.6984
   ignore_gate_to_diff = no
   ignore_rsd_intrinsic = yes
}
```

inductance_unit

Type: Global Parameters

An optional global parameter that specifies the units for inductance values to use when calculating the self inductance of vias.

Syntax

inductance_unit = *unit*

Parameters

- *unit*

  A value that specifies the unit of inductance to use for calculating the self inductance of vias.

  Acceptable values are:

  ah — attohenry

  fh — femtohenry

  ph — picohenry

  nh — nanohenry

  uh — microhenry

  mh — millihenry

  h — henry

  The default is picohenry (ph).

Description

Calibration does not generate inductance calculations but sets the units used during inductance extraction. The default is to not produce the Unit Inductance SVRF statement. For more information on global parameters, see Global Parameters.

Examples

```
inductance_unit = ph
```

insulator

Parameter for layer(s): TSV

An optional TSV layer parameter used to specify one or more insulators for the TSVs in the process. The comma separated value pairs must be enclosed in braces ({ }). Insulators must be described using either a conformal dielectric layer or with the insulator parameter.

Syntax

insulator = {{$thickness_1$, ER_1} {$thickness_2$, ER_2 }... {$thickness_n$, ER_n}}

Parameters

- *thickness*

  A floating point value used to specify the thickness of the insulator.

- *ER*

  A floating point value used to specify the dielectric constant for the insulator.

Examples

Example 1

The following example is a TSV description using the insulator parameter for a TSV with one insulator type where thickness of the insulator is 0.25 and the dielectric constant is 4.8:

```
tsv = my_tsv {
   measured_from = base_M1
   measured_to = BM
   radius = 3.2
   hollow_radius = 0
   height = 48.5
   top_enclosure = 4
   bot_enclosure = 4
   resistivity = 0.02
   depletion_width = 0
   insulator = {{0.25, 4.8}}
}
```

Example 2

This example defines a TSV insulator using a conformal dielectric layer. For example:

```
tsv = tsv {
   measured_from = BM1
   measured_to = M1
   radius = 6.0
   hollow_radius = 0
   height = 50.50
   top_enclosure = 0
   bot_enclosure = 0
   resistivity = 0.1
   depletion_width = 0.5
}
dielectric = tsv_sidewall {
   diel_type = conformal
   thickness = 0.0
   eps = 3.2
   ref_layer = tsv
   topthk = 0.0
   swthk = 0.3
}
```

interpolation

Parameter for layer(s): base_via, via

An optional parameter used by via layers that specifies how the area-based tables interpolate resistance values for fixed or variable sized vias.

Syntax

interpolation = gperarea | invgperarea

Parameters

- gperarea

  Specifies the standard interpolation as G = 1/R for fixed via sizes. This setting increases the accuracy of the interpolated values. This is the default.

- invgperarea

  Specifies the standard interpolation as G = R for rectangular vias with variable length. This setting is used when working with high resistance conductors.

Examples

Example 1

To use gperarea for BEOL layer specify the following parameter in your via layer definition:

```
interpolation = gperarea
```

Example 2

To use invgperarea for TAP (device) vias specify the following parameter in your via layer definition:

```
interpolation = invgperarea
```

layer_bias

Parameter for layer(s): conductor, contact, device_li, diffusion, ground, li, poly, resistor, seed, src_drn, via

An optional conducting layer parameter that specifies the bias on the edge of metal objects with respect to the drawn width of the layer, or the drawn width and length if the layer is a contact or via layer.

Syntax

layer_bias = *value*

Parameters

- *value*

  A positive or negative floating point value that specifies the bias on the edge of metal objects with respect to the drawn width of the layer, or the drawn width and length of the contact or via.

Description

Figure 6-52 illustrates the bias on the edge of metal objects with respect to the drawn width of the layer.

Figure 6-52. Definition of Layer Bias

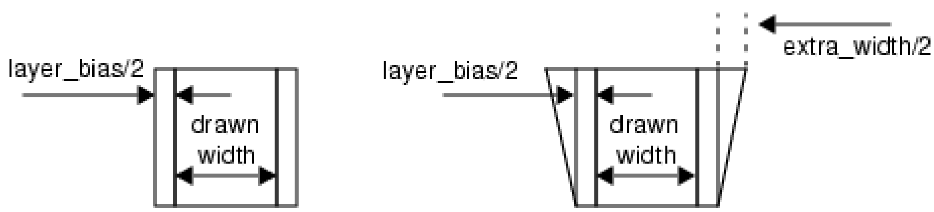
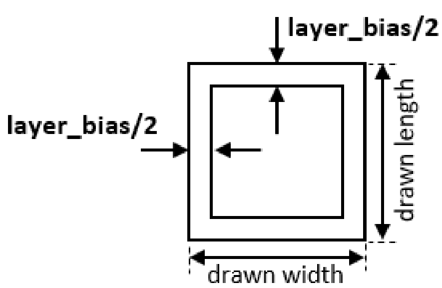


Figure 6-53 illustrates the bias on the edge of metal objects with respect to the drawn width and length of a contact or via layer.

Figure 6-53. Layer Bias for a Contact or Via



If there is no in-die table or opc bias table, then actual width uses the layer_bias setting as follows:

actual_width = drawn_width + layer_bias

If there is an in-die table or opc bias table and layer_bias is specified, then the table setting is used and the layer_bias setting is ignored.

The process variation of the width is described through variation data from the table. For more information on process variation, see Process Variation. For more information on the conducting layers, see Layer Definitions.

Examples

```
layer_bias = 0.50
```

layer_name

Layer Definitions syntax

Part of the layer definition syntax, this layer definition parameter specifies a required unique user-specified name.

Syntax

***layer_type* = *layer_name* {**
list_of_parameters

}

Parameters

- ***layer_type***

  A required parameter that specifies the layer type keyword. Valid layer type keywords include Base, Dielectric, Conductor, Resistor, Ground, Derived, Poly, Diffusion, li, Device_li, Contact, Via, Src_drn, Seed, and TSV. When you define a layer, the layer type keyword must come first followed by an equal sign then the layer name. For more information on layer types, see Layer Definitions.

- ***layer_name***

  A required unique user-specified name.

- ***list_of_parameters***

  This is the list of required and optional parameters which varies depending on the layer type. For more information on what the list of parameters should be, refer to the list supplied for the layer type you wish to specify.

Examples

In the following example the ***layer_name*** parameter is *my_base_layer*:

```
base = my_base_layer {
   thickness = 1
   ztop = 0.
}
```

layers

table structure parameter

An optional table structure parameter used to specify whether the via resistance is dependent on the upper, lower, or both upper and lower layer's enclosure defined with a via_resistance or via_resistance2 table structure. This parameter may only be used in two-dimensional via_resistance or via_resistance2 property tables.

Syntax

layers = up | down | both

Parameters

- up

  A keyword that specifies that the via resistance is dependent on the upper layer's enclosure.

- down

  A keyword that specifies that the via resistance is dependent on the lower layer's enclosure

- both

  A keywords that specifies that the via resistance is dependent on both upper and lower layer's enclosure. This is the default.

Examples

```
table = table_up {
   property = via_resistance2
   table_type = R
   layers = up
   enclosure = min
   dim_type = drawn
   value_type = absolute
   via_size = { 0.022 0.022 }
   overlap_width = { 0.02175 }
   overlap_length = { 0.02175 }
   value = { 30.2 }
}
table = table_down {
   property = via_resistance2
   table_type = R
   layers = down
   enclosure = min
   dim_type = drawn
   value_type = absolute
   via_size = { 0.022 0.022 }
   overlap_width = { 0.02175 }
   overlap_length = { 0.02175 }
   value = { 40 }}
```

li

Type: li layer

Part of the layer definition syntax, the li layer definition keyword is used to define the electrical and physical characteristics of local interconnect layers. The li keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. Layers of this type can be specified multiple times. This layer definition is required when you are not performing RONLY extraction.

Syntax

li = ***layer_name*** **{**
  thickness = ***z-direction_ metal_ thickness***
  min_width = ***minimum_allowed_metal_width***
  min_spacing = ***minimum_allowed_metal_spacing***
  resistivity = ***resistivity_value***
  r_sheet = ***sheet_resistance***
  metal_fill = '{'*fill_ratio fill_spacing fill_width* [floating | grounded] '}'
  ztop = *top_ z-coordinate*
  zbottom = *bottom_z-coordinate*
  measured_from = *layername*
  airgap = {rectangle | triangle} min_gap (*v*) spacing (*s1* ... *sn*) width (*w1* ... *wn*)
    thickness (*t1* ... *tn*) dist_from_surface (*b1* ... *bn*)
  thickness_type = <u>absolute</u> | relative
  coplanar_min_spacing = *value*
  tc1 = *resistance_temperature1_coefficient*
  tc2 = *resistance_temperature2_coefficient*
  max_rlength = *length_for_resistance_fracturing*
  max_width = *maximum_allowed _metal_width*
  max_spacing = *maximum_allowed_metal_spacing*
  maxwidth_for_minspacing = *value*
  devices = '{'$device_1$ $device_2$ ... $device_n$'}' # a space delimited list of device names
  layer_bias = *bias_actual_width_relative_to_drawn_width*
  trap_style = top | middle | bottom # where trapezoid measurements are taken from
  extra_width = *extra-width_of_side_walls* # for a conductor with a trapezoid shape
  swstep = *count*
  extension = *amount_layer_extends_past_diffusion_or_contact_layers*
  ignore_caps = yes | <u>no</u>
  device_li_layers = *space_delimited_list_of_device_li_layer_names*
  ronly_layers = '{'*space_delimited_list_of_r-only_layer_names*'}'
  density_window = '{' '{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'...'{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}''}'
  widths = '{' w_1 w_2 w_3 ... w_n '}'
  spacings = '{' s_1 s_2 s_3 ... s_n '}'
  hidden = <u>false</u> | true
  }

Parameters

- ***layer_name***

  A required unique user-specified name.

- **thickness**

  A required layer parameter that specifies the thickness of the metal layer in the z-direction.

- **min_width**

  A required layer parameter that specifies the minimum allowed drawn width of metal objects on this layer.

- **min_spacing**

  A required parameter that specifies the minimum allowed metal spacing.

- **resistivity**

  A required layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- **r_sheet**

  A required layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- metal_fill

  An optional conducting layer parameter. It is a set of values enclosed in braces, used to define a list of virtual fill parameters.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- measured_from

  An optional conducting layer parameter that measures zbottom or ztop relative to the specified layer.

- airgap

  An optional parameter that specifies the airgap parameters for the li layer.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.

- coplanar_min_spacing

  An optional conducting layer parameter that specifies the minimum spacing value between coplanar layers.

- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.

- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.

- max_rlength

  An optional conducting layer parameter that specifies the maximum length allowed for resistance fracturing.

- max_width

  An optional parameter that specifies the maximum allowed metal width.

- max_spacing

  An optional parameter that specifies the maximum allowed metal spacing.

- maxwidth_for_minspacing

  An optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.

- devices

  An optional layer parameter that specifies a space delimited list of device names enclosed in braces. This parameter is used to specify the devices the layer definition applies to.

- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width relative to drawn width.

- trap_style

  An optional conducting layer parameter that defines where trapezoid measurements are taken from. Permitted values are top, middle, or bottom.

- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- swstep

  An optional conducting layer parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.

- extension

  An optional conducting layer parameter that specifies the amount this layer extends past diffusion or contact layers.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- device_li_layers

  An optional li layer parameter that specifies a space delimited list of device_li layer names enclosed in braces ({ }).

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- density_window

  An optional conducting layer parameter specified as $\{\{w_1\ h_1\ factor_1\ stepx_1\ stepy_1\}\ ...\ \{w_n\ h_n\ factor_n\ stepx_n\ stepy_n\}\}$. The braces are required for grouping.

- widths

  An optional conducting layer parameter that specifies a space delimited list of floating point width values enclosed in braces ({ }).

- spacings

  An optional conducting layer parameter that specifies a space delimited list of floating point spacing values enclosed in braces ({ }).

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Examples

```
li = li_1 {
   ztop = 0.0
   measured_from = poly_diel7
   thickness = 0.09
   min_width = 0.045
   min_spacing = 0.045
   resistivity = 0.0366
   tc1 = use my_Mx_layers_tc1
   tc2 = use my_Mx_layers_tc2
}
```

li_device_model

Type: Global Parameters

An optional global parameter that defines the type of local interconnect model generated by calibration.

Syntax

li_device_model = default | single_li_layer | dual_li_layer_model1 | dual_li_layer_model2 | fdsoi_single_li | fdsoi_dual_li | planar_single_li | planar_dual_li

Parameters

- default

  Specifies the CMOS device model. This is the default if this global parameter is not specified.

- single_li_layer

  Specifies a single layer LI model for an SOI process.

- dual_li_layer_model1

  Specifies a two layer LI model which includes gate to diffusion coupling.

- dual_li_layer_model2

  Specifies a two layer LI model which excludes gate to diffusion coupling.

- fdsoi_single_li

  Specifies a single layer LI model for a fully-depleted (FD) SOI process.

- fdsoi_dual_li

  Specifies a two layer LI model for a fully-depleted (FD) SOI process.

- planar_single_li

  Specifies a single layer LI model for a planar process.

- planar_dual_li

  Specifies a two layer LI model for a planar process.

Description

Legal values for li_device_model are default, single_li_layer, dual_li_layer_model1, dual_li_layer_model2, fdsoi_single_li, fdsoi_dual_li, planar_single_li, or planar_dual_li. The number of LI layers defined in the MIPT file must match the number of layers allowed for the specified li_device_model parameter. If the number of layers for the parameter do not match the number of LI layers in the MIPT file, then an error message is generated.

If this parameter is not specified, then the CMOS device model is used by default. If this parameter is specified, but there are no LI layers in the MIPT file, no errors or warnings are generated and the effect of the parameter is ignored. This parameter can only be specified once. For more information on global parameters, see Global Parameters.

Set device_li_model to fdsoi_single_li, fdsoi_dual_li, planar_single_li, or planar_dual_li to activate xACT only calibration mode in fdsoi or planar non-FinFET processes. and generate the PEX Qualification Mode XACT statement in the calibrated rules files.

Examples

Example 1

Include the following global parameter definition in your MIPT file to specify two LI layers which include gate to diffusion coupling:

```
li_device_model = dual_li_layers_model1
```

Example 2

If more than one definition appears in the MIPT file:

```
li_device_model = dual_li_layers_model1
li_device_model = dual_li_layers_model2
```

The following error message is issued by the checker:

```
Error XCAL_2_002: MIPT Parser Error: “The parameter li_device_model is
defined more than once: li_device_model = dual_li_layers_model2” near line
5 in file tech_45nm_2_0.mipt.
```

li_layers

Parameter for layer(s): device_li

An optional device_li layer parameter that specifies a space delimited list of li layer names used to associate the device_li layer with other li layers. Use this parameter to maintain proper physical connectivity between coplanar layers after applying bias.

Syntax

li_layers = {$li\_layer\_name_1$ $li\_layer\_name_2$ ... $li\_layer\_name_n$}

Parameters

- {$li\_layer\_name_1$ $li\_layer\_name_2$ ... $li\_layer\_name_n$}

  A space delimited list of li layer names enclosed in braces ({ }).

Examples

```
device_li = M0_OD1 {
   thickness = 0.09
   min_width = 0.045
   min_spacing = 0.045
   resistivity = 0.672
   layer_bias = -0.0070
   measured_from = ILD_nm
   zbottom = 0
   li_layers = {M0_ST1}
}

li = M0_ST1 {
   min_width = 0.02600
   min_spacing = 0.05800
   resistivity = 0.67
   measured_from = FOX
   zbottom = 0
}
```

li1_extension

Parameter for layer(s): multigate, seed

An optional multigate and seed layer parameter that specifies the distance the li1 layer extends outside the diffusion area.

Syntax

li1_extension = *value*

Parameters

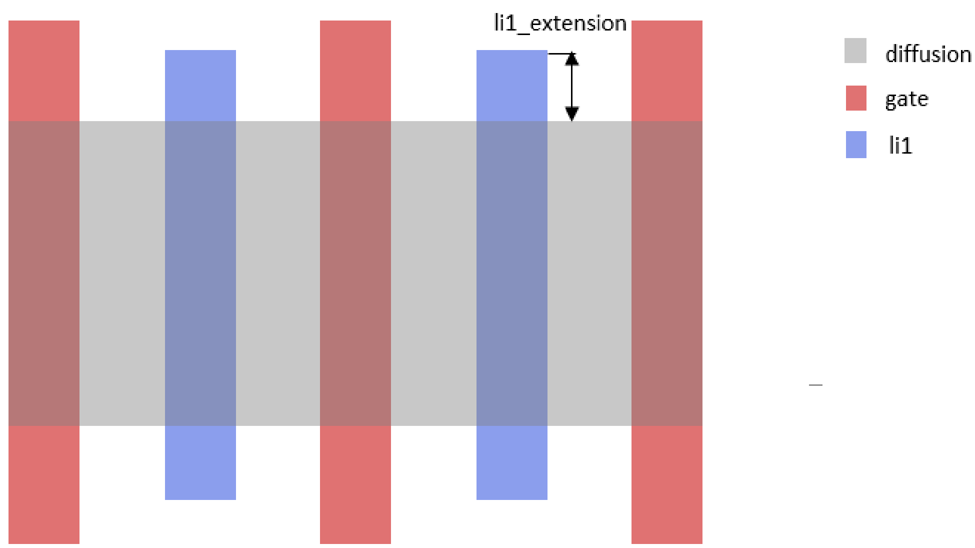
- *value*

  A floating point value in microns that specifies the distance the li1 layer extends past the diffusion area.

Description

Use this parameter to control li1 resistance for more accurate modeling of a FinFET device. The li1_extension parameter specifies the amount of the li1 layer that extends past the diffusion region as shown in Figure 6-54.

Figure 6-54. li1_extension for FinFET



A warning is generated during calibration if this parameter is not specified in the multigate or seed layer definition.

Examples

li1_extension = 0.01

max_area

Parameter for layer(s): base_via, via

An optional parameter used by via and base_via layers that specifies the area threshold in square microns for large area via connections.

Syntax

max_area = *value*

Parameters

- *value*

  A value used to specify the area threshold for large area via connections in square microns.

Description

Use this parameter in your via layer definitions to control how vias are modeled. During calibration, this parameter sets the MAXAREA parameter in the PEX Resistance Parameters statement for the via layer. See Via layer type for a complete list of via layer definition parameters. See Base_via layer type for a complete list of base_via layer definition parameters.

Examples

```
max_area = 150
```

max_length

Parameter for layer(s): base_via, via

An optional parameter used by via and base_via layers that specifies the maximum distance in microns between distributed vias.

Syntax

max_length = *value*

Parameters

- *value*

  A value used to specify the maximum distance between distributed vias in microns.

Description

Use this parameter in your via layer definitions to control the distribution of reduced vias. The distance between the distributed vias will not exceed this limit. During calibration, this parameter sets the MAXLENGTH parameter in the PEX Resistance Parameters statement for the via layer. See Via layer type for a complete list of via layer definition parameters. See Base_via layer type for a complete list of base_via layer definition parameters.

Examples

```
max_length = 50
```

max_rlength

Parameter for layer(s): conductor, derived, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn

An optional conducting layer parameter that specifies the maximum length of a wire to use when calculating how many pieces a resistor should be broken into for representation in a distributed network.

Syntax

max_rlength = *value*

Parameters

- *value*

  A value used to specify the maximum length of a wire to use when calculating how many pieces a resistor should be broken into.

Description

If this parameter is unset, it defaults to a value of 0 and the parasitic resistors are limited to 100 microns (um) in length. Two-dimensional fracturing is not performed (no width fracturing). Leaving this parameter unset is suitable for interconnect layers such as polysilicon and metal.

If this parameter is set to a value, then a two-dimensional grid is created, with a maximum length of each resistor being set to the specified value. This is appropriate for substrate layers such as nwell and psub.

For more information on the conducting layers, see Layer Definitions.

Examples

```
max_rlength = 100
```

max_spacing

Parameter for layer(s): conductor, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn

Parameter for Table(s): RPV_VS_COUNT Table

When specified in a layer definition, max_spacing is an optional conducting layer parameter that specifies the maximum allowed metal spacing. When specified in the rpv_vs_count table, max_spacing specifies the maximum spacing between via arrays.

Syntax

max_spacing = *value*

Parameters

- *value*

  A floating point value used to specify the maximum allowed metal spacing or the maximum spacing between via arrays in a rpv_vs_count table specification.

Examples

Example 1

In a layer definition:

```
conductor = poly {
   thickness = 0.1
   min_width = 0.046
   min_spacing = 0.048
   max_width = 0.2
   max_spacing = 5
   r_sheet = 9
}
```

Example 2

Defined within an rpv_vs_count table:

```
via = via1 {
   ...
   table = my_res_per_via {
      property = rpv_vs_count
      table_type = R
      value_type = absolute
      max_spacing = 3
      via_size = {0.026 0.026}
      count = { 4 6 8}
      ratio = { 0.2 0.3 0.4}
   }
   ...
}
```

max_width

Parameter for layer(s): conductor, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn

An optional conducting layer parameter that specifies the maximum allowed metal width.

Syntax

max_width = *value*

Parameters

- *value*

  A floating point value used to specify the maximum allowed metal width.

Examples

```
max_width = 0.50
```

maxwidth_for_minspacing

Parameter for layer(s): conductor, derived, device_li, diffusion, li, pcaux, poly, seed, src_drn

An optional conducting layer parameter that specifies a width range value used to determine the min_spacing value for calibration. This is a floating point value in microns.

Usage

maxwidth_for_minspacing = *value*

Arguments

- *value*

  A floating point value in microns that specifies a width range used to determine the min_spacing value for calibration. The default is ten times the value specified for min_width.

Examples

```
conductor = M1 {
   thickness = 0.063
   hidden = {thickness}
   min_width = 0.034
   maxwidth_for_minspacing = 0.0483
   min_spacing = 0.02
   tc1 = 1.565e-03
   tc2 = -3.159e-07
   extra_width = 0.012
   swstep = 3
   widths = {0.029 0.0483 0.067667 0.145 }
   spacings = {0.009 0.0149698 0.0248995 0.041415 0.068887 0.114581
               0.190584 0.317001 0.52727 1.74}
}
```

measured_from

Parameter for layer(s): optional for conductor, device_li, dielectric, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn; required by base_via, contact, TSV, ubump, via

A layer parameter that specifies the layer whose top surface is used as the base from which the layer's bottom is measured.

Syntax

measured_from = *layername*

Parameters

- *layername*

  A string value used to specify the name of the layer whose top surface is used as the base from which the layer's bottom is measured.

Description

For a conformal dielectric layer, the top is from the planar part. For a trench dielectric layer, the top is from the bottom of the reference layer. The measured_from parameter can only reference layers lower in the stack. For more information on dielectric layer definition, see Dielectric.

For contact and via layers, this is a required parameter that specifies the lower conductor layer name. For base_via layers measured_from must be the base layer name. For more information on contact layer definition, see Contact. For more information on via layer definition, see Via. For more information on base_via layer definition, see Base_via.

For TSV layers, this is a required parameter that specifies the back side metal layer name that the TSV connects to. For more information on TSV layer definition, see TSV.

For ubump layers the parameter specifies the bottom most layer that the uBump connects to. The parameters measured_from and measured_to are interchangeable for ubump layers.

Examples

```
measured_from = poly1
```

measured_to

Parameter for layer(s): base_via, contact, TSV, ubump, via

A required layer parameter that must be specified for base_via, contact, TSV, via, and ubump layer definitions.

Syntax

measured_to = *layername*

Parameters

- *layername*

  A string value used to specify the upper conductor layer name for contact and via layer type, the diffusion layer name for base_layer type, the front metal layer for TSV layer type, or topmost layer for ubump layer type.

Description

For contact and via layers the measured_to parameter specifies the layer whose top surface is used as the base from which the layer's bottom is measured. For base_via layers the parameter specifies the diffusion layer that the base layer connects to. For TSV layers the parameter specifies the front metal layer that the TSV connects to. For ubump layers the parameter specifies the topmost layer that the ubump connects to. The parameters measured_from and measured_to are interchangeable for ubump layers.

Examples

```
measured_to = metal1
```

metal_fill

Parameter for layer(s): conductor, device_li, diffusion, li, pcaux, poly, resistor, seed, src_drn

An optional conducting layer parameter used to define a list of virtual fill parameters. The virtual fill parameters are specified as a set of values enclosed in braces ({ }).

Syntax

metal_fill = '{'*fill_ratio fill_spacing fill_width* [floating | grounded]'}'

Parameters

- *fill_ratio*

  A positive floating point number in the range [0..1], used as minimum density value by virtual fill model.

- *fill_spacing*

  A positive floating point number.

- *fill_width*

  A positive floating point number.

- [floating | grounded]

  A literal value used to specify the type of fill as either floating or grounded.

Examples

```
metal_fill = {0.9 0.1 0.5 grounded}
```

mid_width

Parameter for layer(s): ubump

A required parameter that specifies the middle metal width of the ubump layer.

Syntax

mid_width =*value*

Parameters

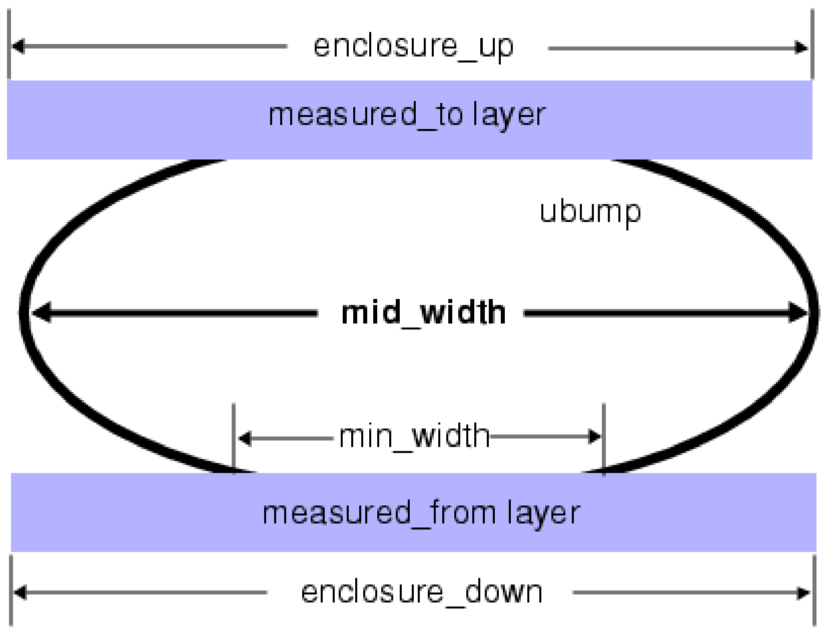
- ***value***

  A required floating point value used to specify the middle metal width of the ubump.

Description

In Figure 6-55, mid_width is the widest diameter of the ubump.

Figure 6-55. mid_width of ubump



Examples

```
mid_width = 1.7
```

min_actual_contact_length

Parameter for layer(s): poly, seed

An optional poly and seed layer parameter used to change the minimum actual length of a corresponding diffusion contact per device.

Syntax

min_actual_contact_length = *value*

Parameters

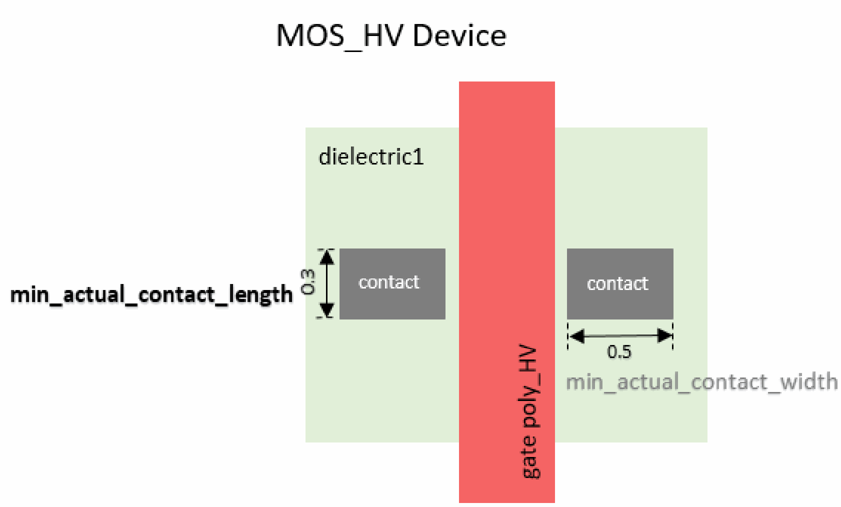
- *value*

  A floating point value used to specify the minimum actual length. The default is the value specified for min_actual_contact_width.

Description

The optional min_actual_contact_length parameter is used to modify the contact min_length for a specific device. Figure 6-56 shows the min_actual_contact_length measurement.

Figure 6-56. min_actual_contact_length Parameter



Examples

The min_actual_contact_length parameter specified in a device seed layer overrides the contact min_length defined in the diffusion contact layer. In the following example, the seed layer (poly_HV device) has a different contact width and length from that specified in the diffusion contact layer diff_cont:

```
seed = poly_LV {
   ...
   devices = {MOS_LV}
}

seed = poly_HV {
   min_actual_contact_width = 0.5
   min_actual_contact_length = 0.3
   ...
   devices = {MOS_HV}
}

diffusion = diff1 {
   src_drn_layers = {tndiff tpdiff}
   devices = {MOS_HV MOS_LV}
   ...
}

contact = diff_cont {
   min_width = 0.4
   min_length = 0.4
   ...
}
```

min_actual_contact_width

Parameter for layer(s): poly, seed

An optional poly and seed layer parameter used to change the minimum actual width of a corresponding diffusion contact per device.

Syntax

min_actual_contact_width = *value*

Parameters

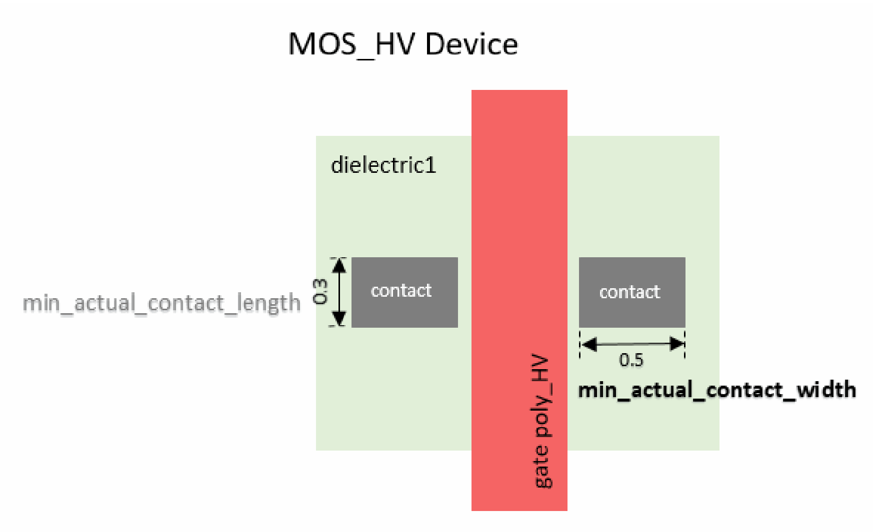
- *value*

  A floating point value used to specify the minimum actual width. The default is the value specified for min_width in the diffusion contact layer.

Description

The optional min_actual_contact_width parameter is used to modify the contact min_width for a specific device. Figure 6-57 shows the min_actual_contact_width measurement.

Figure 6-57. min_actual_contact_width Parameter



Examples

The min_actual_contact_width parameter specified in a device seed layer overrides the contact min_width defined in the diffusion contact layer. In the following example, the seed layer (poly_HV device) has a different contact width and length from that specified in the diffusion contact layer diff_cont:

```
seed = poly_LV {
   ...
   devices = {MOS_LV}
}

seed = poly_HV {
   min_actual_contact_width = 0.5
   min_actual_contact_length = 0.3
   ...
   devices = {MOS_HV}
}

diffusion = diff1 {
   src_drn_layers = {tndiff tpdiff}
   devices = {MOS_HV MOS_LV}
   ...
}

contact = diff_cont {
   min_width = 0.4
   min_length = 0.4
   ...
}
```

min_length

Parameter for layer(s): base_via, contact, via

An optional parameter used by via, base_via, or contact layers that specifies the length of the shape. This parameter can be used together with the min_width parameter, but cannot be used with the area parameter. The shape is assumed to be rectangular, with dimensions of min_width * min_length.

Syntax

min_length = *value*

Parameters

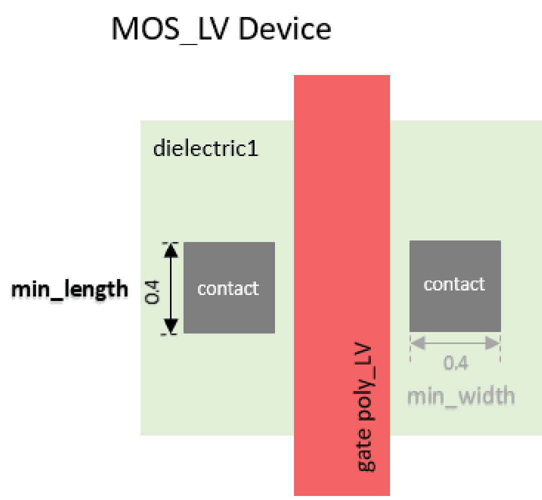
- *value*

  A floating point value used to specify the length of the shape in distance_units.

Description

Figure 6-58 illustrates a min_length measurement.

Figure 6-58. min_length Parameter



Examples

```
min_length = 0.15
```

min_spacing

Parameter for layer(s): base_via, conductor, contact, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn, via

A required parameter for conducting layer definitions conductor, device_li, diffusion, ground, li, pcaux, poly, and resistor that specifies the minimum drawn spacing of metal objects on this layer. This parameter is optional for seed and src_drn layer definitions. For contact, via, and base_via layer definitions, this parameter is required to specify the spacing between vias (or contacts) and conductors on their respective layers.

Syntax

min_spacing = *value*

Parameters

- ***value***

  A required floating point value used to specify the minimum drawn spacing of metal objects on this layer.

Examples

```
min_spacing = 0.125
```

min_width

Parameter for layer(s): base_via, conductor, contact, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn, ubump, via

A required parameter for conducting layer definitions base_via, conductor, contact, device_li, diffusion, ground, li, pcaux, poly, resistor, via, and ubump that specifies the minimum drawn width of metal objects on this layer. This parameter is optional for seed and src_drn layer definitions.

Syntax

min_width = ***value***

Parameters

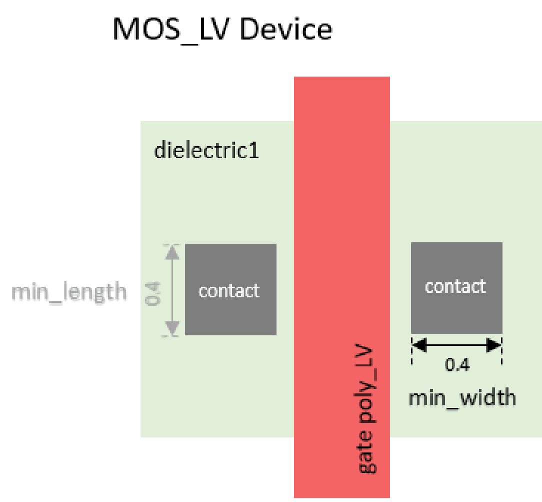
- ***value***

  A required floating point value used to specify the minimum drawn width of metal objects on this layer.

Description

Figure 6-59 illustrates a min_width measurement.

Figure 6-59. min_width Parameter



Examples

```
min_width = 0.17
```

mipt_version

Type: Global Parameters

A required global parameter that specifies the version of the MIPT syntax used in the MIPT file. It can only be specified once.

Syntax

mipt_version = ***version_number***

Parameters

- ***version_number***

  A value used to specify the version number of the MIPT syntax used in the file. The value must be 2.0.

Examples

```
mipt_version = 2.0
```

multigate

Type: Multigate layer

Part of the layer definition syntax, this optional layer definition keyword is used to define the electric and physical characteristics of a multigate device. The multigate keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. Layers of this type can be specified multiple times in the MIPT file.

Syntax

multigate = *layer_name* {
fin_spacing = *space_between_fins*
fin_width = *width_of_fin*
fin_length = *length_of_fin*
fin_thickness = *thickness_of_fin*
fin_bias = *bias_value*
gate_oxide_er = *gate_oxide_permittivity*
gate_oxide_top_t = *gate_oxide_top_thickness*
gate_oxide_side_t = *gate_oxide_side_thickness*
gate_poly_top_t = *poly_top_thickness*
gate_poly_side_t = *poly_side_thickness*
channel_er = *permittivity_diff_value*
gate_diffusion_layer_pair = '{' ***list_of_one_or_more_layer_pairs***'}'
gate_extension = *value*
li1_extension = *value*
rsd_enclosure = *raised_source/drain_edge_bias*
rsd_thickness = *raised_source/drain_height*
rsd_spacing = *raised_source/drain_to_gate_spacing*
trench_contact_extension_length = *value*
rsd_swslope = {*Sx Sy*}
hidden = false | true
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- **fin_spacing**

  A required layer parameter that specifies the space between fins.

- **fin_width**

  A required layer parameter that specifies the width of the fin.

- **fin_length**

  A required layer parameter that specifies the length of the fin.

- **fin_thickness**

  A required layer parameter that specifies the thickness of the fin.
- fin_bias

  An optional layer parameter that specifies the biasing effect on the fin.
- **gate_oxide_er**

  A required layer parameter that specifies the permittivity of the gate oxide.
- **gate_oxide_top_t**

  A required layer parameter that specifies the top thickness of the gate oxide.
- gate_oxide_side_t

  An optional layer parameter that specifies the side thickness of the gate oxide.
- **gate_poly_top_t**

  A required layer parameter that specifies the gate poly thickness between the top of the gate poly and the top of the gate oxide.
- gate_poly_side_t

  An optional layer parameter that specifies the gate poly thickness between the side of the gate poly and the side of the gate oxide.
- channel_er

  An optional layer parameter that specifies the difference in permittivity of the gate oxide and its lateral side.
- **gate_diffusion_layer_pair**

  A required list of one or more layer pairs used to specify which gate and diffusion layers can be paired together to describe different device types with the same device parameters. The list of pairs must be specified in brackets; pairs must be specified in parentheses.
- gate_extension

  An optional parameter that specifies the distance the gate layer extends outside the diffusion area.
- li1_extension

  An optional parameter that specifies the distance the li1 layer extends outside the diffusion area.
- rsd_enclosure

  An optional layer parameter that specifies the raised source/drain edge bias.
- rsd_thickness

  An optional conducting layer parameter that specifies the raised source/drain height.

- rsd_spacing

  An optional conducting layer parameter that specifies the raised source/drain to gate spacing.

- rsd_swslope

  An optional conducting layer parameter that specifies the sidewall slope of the associated raised source/drain layer. Parameters rsd_thickness and rsd_spacing must be specified with this parameter.

- trench_contact_extension_length

  An optional layer parameter that specifies the nominal extension of the trench contact beyond the diffusion.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Examples

```
multigate = fin1 {
   fin_spacing = 0.016
   fin_width = 0.016
   fin_length = 0.032
   fin_thickness = 0.010
   gate_oxide_top_t = 0.006
   gate_oxide_er = 4.0
   gate_poly_top_t = 0.008
   channel_er = 6.0
   gate_diffusion_layer_pair = {(PGATE PDIFF) (NGATE NDIFF)}
}
```

multigate parameter

Parameter for layer(s): pcaux, seed

An optional seed layer parameter used to control which default model for FinFETs is used during calibration.

Syntax

multigate = ignore | trim

Parameters

- ignore

  A keyword that specifies to replace diffusion with a dummy volume under raised source/drain.

- trim

  A keyword that specifies to keep diffusion under the raised source/drain (rsd) and not extend to gate.

Examples

```
multigate = trim
```

pad

Type: Pad layer

Part of the layer definition syntax, this optional layer definition keyword is used to define a conductor layer used to describe a CBUMP layer that has no routing, typically a square or round shape. The pad keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. Via table types VRESISTANCE, VWIDTH, and VLENGTH can be specified in this layer type.

Syntax

pad = ***layer_name*** **{**
  thickness = ***z_direction_metal_thickness***
  min_width =***minimum_allowed_metal_width***
  min_spacing = ***minimum_allowed_metal_spacing***
  resistivity = ***resistivity_value***
  r_sheet = ***sheet_resistance***
  metal_fill = '{'*fill_ratio fill_spacing fill_width* [floating | grounded]'}'
  thickness_type = <u>absolute</u> | relative
  ztop = *top_ z-coordinate*
  zbottom = *bottom_z-coordinate*
  measured_from = *layername*
  coplanar_min_spacing = *value*
  tc1 = *resistance_temperature1_coefficient*
  tc2 = *resistance_temperature2_coefficient*
  max_rlength = *length_for_resistance_fracturing*
  max_width = *maximum_allowed _metal_width*
  max_spacing = *maximum_allowed_metal_spacing*
  layer_bias = *bias_actual_width_relative_to_drawn_width*
  capacitive_only_etch = *layer_bias_for_capacitance_only*
  resistive_only_etch = *layer_bias_for_resistance_only*
  trap_style = top | middle | bottom # where trapezoid measurements are taken from
  extra_width = *extra-width_of_side_walls* #for a conductor with a trapezoid shape
  extension = *amount_layer_extends_past_diffusion_or_contact_layers*
  ronly_layers = '{'*space_delimited_list_of_r-only_layer_names*'}'
  density_window = '{''{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'...'{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}''}'
  hidden = <u>false</u> | true
  }

Parameters

- ***layer_name***

  A required unique user-specified name.

- **thickness**

  A required layer parameter that specifies the thickness of the metal layer in the z-direction.

- **min_width**

  A required layer parameter that specifies the minimum allowed drawn width of metal objects on this layer.

- **min_spacing**

  A required layer parameter that specifies the minimum allowed drawn spacing between conductors on this layer.

- **resistivity**

  A required layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- **r_sheet**

  A required layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- metal_fill

  An optional conducting layer parameter. It is a set of values enclosed in braces, used to define a list of virtual fill parameters.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- measured_from

  An optional conducting layer parameter that measures zbottom or ztop relative to the specified layer.

- coplanar_min_spacing

  An optional conducting layer parameter that specifies the minimum spacing value between coplanar layers.

- tc1

  An optional conducting layer parameter that specifies the resistance temperature1 coefficient.

- tc2

  An optional conducting layer parameter that specifies the resistance temperature2 coefficient.

- max_rlength

  An optional conducting layer parameter that specifies the maximum length of a wire. This parameter is used for resistance fracturing, which determines how many pieces a resistor should be broken into for representation in a distributed network.

- max_width

  An optional parameter that specifies the maximum allowed metal width.

- max_spacing

  An optional parameter that specifies the maximum allowed metal spacing.

- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width relative to drawn width.

- capacitive_only_etch

  An optional parameter that specifies the layer bias for capacitance only.

- resistive_only_etch

  An optional parameter that specifies the layer bias for resistance only.

- trap_style

  An optional conducting layer parameter that defines where trapezoid measurements are taken from. Permitted values are top, middle, or bottom.

- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- extension

  An optional conducting layer parameter that specifies the amount this layer extends past diffusion or contact layers.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- density_window

  An optional conducting layer parameter specified as {{w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$} ... {w_n h_n $factor_n$ $stepx_n$ $stepy_n$}}. The braces are required for grouping.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Examples

```
pad = pad1_layer {
   thickness = 0.1
   min_width = 0.1
   min_spacing = 0.25
   resistivity = 8.3
}
```

parallel_to_gate

Parameter for layer(s): via

An optional parameter used by via layers that specifies how the via width and length are measured.

Syntax

parallel_to_gate = no | yes

Parameters

- no

  Specifies the width is the shorter edge. This is the default.

- yes

  Specifies the length is the edge parallel to the gate and the width is the edge perpendicular to the gate.

Description

Use this parameter in your via layer definitions to describe how the via width and length are measured. For rectangular vias, the width is typically taken to be the shorter value from the width and length values. For smaller geometries certain via lengths may be shorter than the width, so specify the parallel_to_gate parameter set to yes to override the default behavior. See Via layer type for a complete list of via layer definition parameters.

Examples

```
parallel_to_gate = yes
```

pcaux

Type: PCaux layer

Part of the layer definition syntax, this optional layer definition keyword is used to define a floating (non-gate) poly layer abutting the diffusion in between or at the end of a device. The pcaux keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line.

Syntax

pcaux = *layer_name* {
thickness = *z-direction_metal_ thickness*
min_width = *minimum_allowed_metal_width*
min_spacing = *minimum_allowed_metal_spacing*
resistivity = *resistivity_value*
r_sheet = *sheet_resistance*
metal_fill = '{' *fill_ratio fill_spacing fill_width* [floating | grounded] '}'
ztop = *top_z-coordinate*
zbottom = *bottom_z-coordinate*
measured_from = *layername*
airgap = {rectangle | triangle} min_gap (*v*) spacing (*s1* ... *sn*) width (*w1* ... *wn*)
thickness (*t1* ... *tn*) dist_from_surface (*b1* ... *bn*)
thickness_type = <u>absolute</u> | relative
coplanar_min_spacing = *value*
tc1 = *resistance_temperature1_coefficient*
tc2 = *resistance_temperature2_coefficient*
max_rlength = *length_for_resistance_fracturing*
max_width = *maximum_allowed _metal_width*
max_spacing = *maximum_allowed_metal_spacing*
maxwidth_for_minspacing = *value*
multigate = ignore | trim
devices = '{'$device_1$ $device_2$ ... $device_n$'}' # a space delimited list of device names
layer_bias = *bias_actual_width_relative_to_drawn_width*
trap_style = top | middle | bottom # where trapezoid measurements are taken from
extra_width = *extra-width_of_side_walls* # for a conductor with a trapezoid shape
swstep = *count*
extension = *amount_layer_extends_past_diffusion_or_contact_layers*
ronly_layers = '{'*space_delimited_list_of_r-only_layer_names*'}'
density_window = '{''{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'...'{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}''}'
seed_layers = '{'*space_delimited_list_of_seed_layer_names*'}'
gate_to_cont_min_spacing = *minimum_spacing_between_gate/poly_and_diffusion_contact*
gate_to_via_min_spacing = *minimum_spacing_between_gate/poly_and_diffusion_via*
gate_to_LI1_min_spacing = *minimum_spacing_between_gate/poly_and_li_layer*
gate_to_LI2_min_spacing = *minimum_spacing_between_gate/poly_and_second_li_layer*
ignore_caps = yes | <u>no</u>
ignore_diff_to_diff_under_poly = <u>yes</u> | no

ignore_endcap = yes | <u>no</u>
ignore_gate_to_diff = <u>yes</u> | no
ignore_gateext_to_diff = yes | <u>no</u>
ignore_gateext_to_diff_only = yes | <u>no</u>
rsd_enclosure = *raised_source/drain_edge_bias*
rsd_thickness = *raised_source/drain_height*
rsd_spacing = *raised_source/drain_to_gate_spacing*
rsd_swslope = {*Sx Sy*}
hidden = <u>false</u> | true
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- **thickness**

  A required layer thickness parameter.

- **min_width**

  A required layer parameter that specifies the minimum allowed drawn width of metal objects on this layer.

- **min_spacing**

  A required layer parameter that specifies the minimum allowed metal spacing.

- resistivity

  An optional layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- r_sheet

  An optional layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- metal_fill

  An optional conducting layer parameter. It is a set of values enclosed in braces, used to define a list of virtual fill parameters.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.
- measured_from

  An optional conducting layer parameter that measures zbottom or ztop relative to the specified layer.
- airgap

  An optional parameter that specifies the airgap parameters for the pcaux layer.
- coplanar_min_spacing

  An optional conducting layer parameter that specifies the minimum spacing value between coplanar layers.
- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.
- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.
- max_rlength

  An optional conducting layer parameter that specifies the maximum length allowed for resistance fracturing.
- max_width

  An optional parameter that specifies the maximum allowed metal width.
- max_spacing

  An optional parameter that specifies the maximum allowed metal spacing.
- maxwidth_for_minspacing

  An optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
- multigate

  An optional parameter that specifies which default multigate model for FinFETs should be used during calibration.
- devices

  An optional layer parameter that specifies a space delimited list of device names enclosed in braces. This parameter is used to specify the devices the layer definition applies to.
- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width relative to drawn width.

- trap_style

  An optional conducting layer parameter that defines where trapezoid measurements are taken from. Permitted values are top, middle, or bottom.

- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- swstep

  An optional conducting layer parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.

- extension

  An optional conducting layer parameter that specifies the amount this layer extends past diffusion or contact layers.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- density_window

  An optional conducting layer parameter specified as {{w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$} ... {w_n h_n $factor_n$ $stepx_n$ $stepy_n$}}. The braces are required for grouping.

- seed_layers

  An optional layer parameter that specifies one or more gate layers referred to as seed layers, enclosed in braces ({ }). This parameter can only be specified for a poly or seed layer type.

- gate_to_cont_min_spacing

  An optional layer parameter that specifies the minimum spacing between the gate poly layer and the diffusion contact layer.

- gate_to_via_min_spacing

  An optional layer parameter that specifies the minimum spacing between the gate poly layer and the diffusion via layer.

- gate_to_LI1_min_spacing

  An optional layer parameter that specifies the minimum spacing between the gate poly layer and the li layer touching the diffusion layer.

- gate_to_LI2_min_spacing

  An optional layer parameter that specifies the minimum spacing between the gate poly layer and the second li layer touching the diffusion layer.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- ignore_diff_to_diff_under_poly

  An optional parameter set to either yes or no, that controls whether or not diffusion to diffusion under poly capacitance is ignored by calibration. The default is yes, if this parameter is not specified.

- ignore_endcap

  An optional parameter set to either yes or no, that controls whether or not poly extension to source and drain coupling is ignored by calibration. The default is no, if this parameter is not specified.

- ignore_gate_to_diff

  An optional parameter set to either yes or no, that controls whether or not the coupling capacitance between gates to diffusion is ignored by calibration. The default is yes, if this parameter is not specified.

- ignore_gateext_to_diff

  An optional parameter set to either yes or no, that controls whether or not the coupling capacitance between the gate extension to diffusion and rsd is ignored by calibration. The default is no, if this parameter is not specified.

- ignore_gateext_to_diff_only

  An optional parameter set to either yes or no, that controls whether or not the coupling capacitance between the gate extension to diffusion is ignored by calibration. The default is no, if this parameter is not specified.

- rsd_enclosure

  An optional layer parameter that specifies the raised source/drain edge bias.

- rsd_thickness

  An optional layer parameter that specifies the raised source/drain height.

- rsd_spacing

  An optional layer parameter that specifies the raised source/drain to gate spacing.

- rsd_swslope

  An optional conducting layer parameter that specifies the sidewall slope of the associated raised source/drain layer. Parameters rsd_thickness and rsd_spacing must be specified with this parameter.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Description

A pcaux (cfpoly) layer is an interconnect field poly between two devices whose thickness is the same as field poly and that is treated as a floating net. The width bias, high K gate oxide is different from field poly. For more information on pcaux layers, see PCaux.

Examples

Use a pcaux layer to specify a dummy interconnect field poly layer. For example:

```
pcaux = my_cpode_gate {
   measured_from = ppoly
   zbottom = 0
   thickness = 0.005
   rsd_spacing = 0.01
   min_width = 0.05
   min_spacing = 0.075
   devices - {pmos1 pmos2}
   gate_to_cont_min_spacing - 0.012
   resistivity = 7.8
   tc1 =0
   tc2 =0
}
```

plate_loading

Type: Global Parameters

An optional global parameter that specifies whether the models and rules apply loading effects to very wide wires or large plates. Setting plate_loading to yes turns on modeling of bottom plane thickness variation, also known as loading. By default, plate_loading is set to no. This option applies to all conductor layers that have thickness_bot tables. The plate_loading parameter can only be specified once.

Syntax

plate_loading = yes | <u>no</u>

Parameters

- yes

  Specifies to apply loading effects to plates. This setting turns on modeling of bottom plane thickness variation, also known as loading.

- <u>no</u>

  Specifies that loading effects are not applied to plates. This is the default behavior.

Examples

```
plate_loading = yes
```

poly

Type: Poly layer

Part of the layer definition syntax, the poly layer definition keyword is used to define the electrical and physical characteristics of conducting layers. Layers of this type can be specified multiple times. This layer definition is required when you are not performing RONLY extraction.

Syntax

poly = ***layer_name*** **{**
thickness = ***z-direction_metal_ thickness***
min_width = ***minimum_allowed_metal_width***
min_spacing = ***minimum_allowed_metal_spacing***
resistivity = ***resistivity_value***
r_sheet = ***sheet_resistance***
metal_fill = '{' *fill_ratio fill_spacing fill_width* [floating | grounded] '}'
ztop = *top_ z-coordinate*
zbottom = *bottom_z-coordinate*
measured_from = *layername*
airgap = {rectangle | triangle} min_gap (*v*) spacing (*s1* ... *sn*) width (*w1* ... *wn*)
thickness (*t1* ... *tn*) dist_from_surface (*b1* ... *bn*)
thickness_type = <u>absolute</u> | relative
coplanar_min_spacing = *value*
tc1 = *resistance_temperature1_coefficient*
tc2 = *resistance_temperature2_coefficient*
max_rlength = *length_for_resistance_fracturing*
max_width = *maximum_allowed _metal_width*
max_spacing = *maximum_allowed_metal_spacing*
maxwidth_for_minspacing = *value*
min_actual_contact_length = *value*
min_actual_contact_width = *value*
devices = '{'$device_1$ $device_2$ ... $device_n$'}' # a space delimited list of device names
layer_bias = *bias_actual_width_relative_to_drawn_width*
capacitive_only_etch = *layer_bias_for_capacitance_only*
resistive_only_etch = *layer_bias_for_resistance_only*
trap_style = top | middle | bottom # where trapezoid measurements are taken from
extra_width = *extra-width_of_side_walls* # for a conductor with a trapezoid shape
swstep = *count*
extension = *amount_layer_extends_past_diffusion_or_contact_layers*
ignore_caps = yes | <u>no</u>
ronly_layers = '{'*space_delimited_list_of_r-only_layer_names*'}'
density_window ='{' '{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'...'{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}''}'
seed_layers = '{'*space_delimited_list_of_seed_layer_names*'}'
widths = '{' w_1 w_2 w_3 ... w_n '}'
spacings = '{' s_1 s_2 s_3 ... s_n '}'

hidden = false | true
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- **thickness**

  A required layer parameter that specifies the thickness of the metal layer in the z-direction.

- **min_width**

  A required layer parameter that specifies the minimum allowed drawn width of metal objects on this layer.

- **min_spacing**

  A required parameter that specifies the minimum allowed metal spacing.

- **resistivity**

  A required layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- **r_sheet**

  A required layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- metal_fill

  An optional conducting layer parameter. It is a set of values enclosed in braces, used to define a list of virtual fill parameters.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- measured_from

  An optional conducting layer parameter that measures zbottom or ztop relative to the specified layer.

- airgap

  An optional parameter that specifies the airgap parameters for the poly layer.
- coplanar_min_spacing

  An optional conducting layer parameter that specifies the minimum spacing value between coplanar layers.
- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.
- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.
- max_rlength

  An optional conducting layer parameter that specifies the maximum length allowed for resistance fracturing.
- max_width

  An optional parameter that specifies the maximum allowed metal width.
- max_spacing

  An optional parameter that specifies the maximum allowed metal spacing.
- maxwidth_for_minspacing

  An optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
- min_actual_contact_length

  An optional poly and seed layer parameter that specifies to override the default minimum actual length of a diffusion contact for a specific device.
- min_actual_contact_width

  An optional poly and seed layer parameter that specifies to override the default minimum actual width of a diffusion contact for a specific device.
- devices

  An optional layer parameter that specifies a space delimited list of device names enclosed in braces. This parameter is used to specify the devices the layer definition applies to.
- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width relative to drawn width.
- capacitive_only_etch

  An optional parameter that specifies the layer bias for capacitance only.
- resistive_only_etch

  An optional parameter that specifies the layer bias for resistance only.

- trap_style

  An optional conducting layer parameter that defines where trapezoid measurements are taken from. Permitted values are top, middle, or bottom.

- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- swstep

  An optional conducting layer parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.

- extension

  An optional conducting layer parameter that specifies the amount this layer extends past diffusion or contact layers.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- density_window

  An optional conducting layer parameter specified as $\{\{w_1\ h_1\ factor_1\ stepx_1\ stepy_1\} \ldots \{w_n\ h_n\ factor_n\ stepx_n\ stepy_n\}\}$. The braces are required for grouping.

- seed_layers

  An optional layer parameter that specifies one or more gate layers referred to as seed layers, enclosed in braces ({ }). This parameter may only be specified for a poly or seed layer type.

- widths

  An optional conducting layer parameter that specifies a space delimited list of floating point width values enclosed in braces ({ }).

- spacings

  An optional conducting layer parameter that specifies a space delimited list of floating point spacing values enclosed in braces ({ }).

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Description

The poly keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line.

A poly layer must be named as the **measured_to** or **measured_from** layer for at least one contact layer in the MIPT file. A warning is generated during calibration for poly layers that do not have a corresponding contact layer.

Examples

Example 1

This example specifies a poly layer named ipoly:

```
poly = ipoly {
   thickness = 0.005
   min_width = 0.05
   min_spacing = 0.075
   r_sheet = 7.8
}
```

Example 2

In this example a new poly layer, my_poly, is created from an existing poly layer Mx using the Use Directive. The “with” modifier overrides existing properties and adds new properties to the my_poly layer definition.

```
poly = my_poly use Mx with {
   ztop = 0.0
   measured_from = poly_diel7
   thickness = 0.09
   min_width = 0.045
   min_spacing = 0.045
   resistivity = 0.0366
}
```

process

Type: Global Parameters

A required global parameter used to create filenames and directories of results. It can only be specified once. It must be a single string without spaces or special characters.

Syntax

process = *name*

Parameters

- ***name***

  A single string without spaces or special characters used to specify the name of the manufacturing technology (process) being used.

Examples

```
process = my_process_1
```

process_foundry

Type: Global Parameters

An optional global parameter used to specify the process foundry name. This parameter can only be specified once. The name must be a single string without spaces or special characters.

Syntax

process_foundry = ***name***

Parameters

- ***name***

  A single string without spaces or special characters used to specify the name of the foundry that produces the manufacturing technology (process) being used.

Examples

```
process_foundry = FastChip
```

property

table structure parameter

A required table structure parameter that specifies the property type of the table.

Syntax

property = ***table_property***

Parameters

- ***table_property***

  A required value that specifies the property type of the table.

Description

Specifies the property type of the table for the layer. Each table property has a specific set of ordered variables associated with them. For a complete list of valid table_property types and their associated settings refer to “Table Property” on page 113. For more information on table property, see “Table Property” on page 113.

Examples

This is an example of a conductor layer with a width table defined:

```
conductor = metal4 {
   min_width = 0.1
   min_spacing = 0.1
   thickness = 0.2
   table = m4_width_table {
      property = WIDTH
      table_type = R
      dim_type = drawn
      value_type = absolute
      width = {0.1 0.2 0.25}
      spacing = {0.15 0.25 0.5}
      value = {0.2 0.3 0.5,
               0.2 0.3 0.5,
               0.2 0.3 0.5}
   }
}
```

Note the table parameter property is set to WIDTH.

radius

Parameter for layer(s): TSV

A required TSV layer parameter used to specify the radius of the through silicon via.

Syntax

radius = ***value***

Parameters

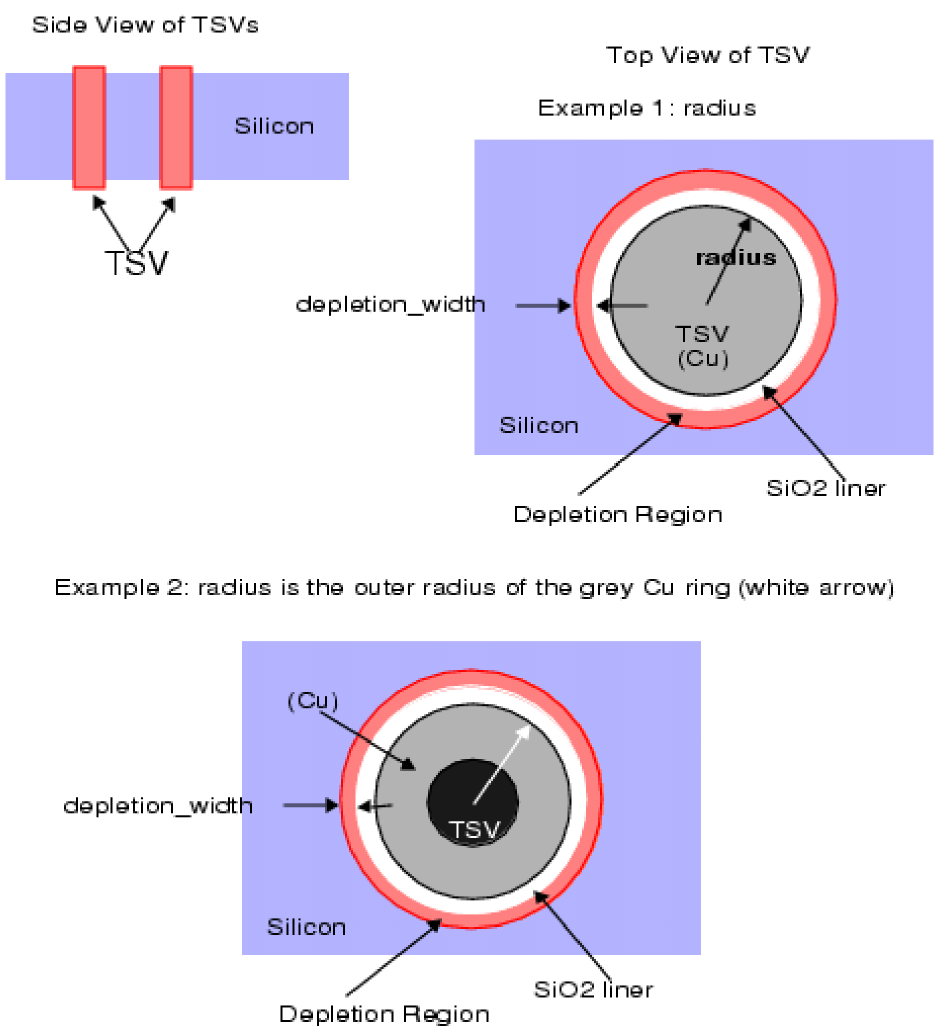
- ***value***

  A floating point value used to specify the radius of the TSV.

Description

The radius measurement for the TSV is shown in Figure 6-60.

Figure 6-60. radius for TSV Layers



For more information on TSV layer definition, see TSV.

Examples

Example 1

In Figure 6-60, Example 1 is an example of a solid TSV. In this case radius is 9.

```
radius = 9
```

Example 2

In Figure 6-60, Example 2 is an example of a hollow TSV. In this case the radius is 9.

```
radius = 9
```

ratio

Parameter for Table(s): RPV_VS_COUNT Table

A table parameter whose values specify a via resistance multiplication factor. The multiplication factor is applied when calculating resistance between via arrays. Can only be used in the one-dimensional rpv_vs_count property table.

Syntax

ratio = {$ratio_1$ $ratio_2$ ... $ratio_n$}

Parameters

- $ratio_1$ $ratio_2$ ... $ratio_n$

  A list of one or more floating point values that represent the via resistance multiplication factor enclosed in required braces ({ })

Examples

```
ratio = { 0.2 0.3 0.4 }
```

ratio1

Parameter for Table(s): INVERSE_RG_FACTOR_1 and INVERSE_RG_FACTOR_2 Tables

A table parameter whose values specify the ratio of the distance from the via center to the bottom gate extension edge and the effective gate width. Can only be used in the inverse_rg_factor_1 and inverse_rg_factor_2 tables.

Syntax

ratio1 = {$ratio_1$ $ratio_2$ ... $ratio_n$}

Parameters

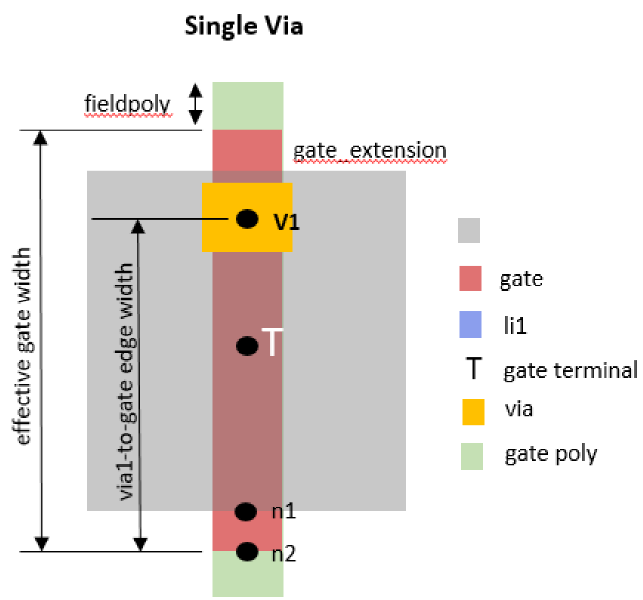
- $ratio_1$ $ratio_2$ ... $ratio_n$

  A list of one or more floating point values that represent the ratio of the distance from a via center to the bottom gate extension edge and the effective gate width enclosed in required braces ({ })

Description

Ratio1 is the ratio of the distance from the V1 center to the bottom of the gate extension edge and the effective gate width as shown in Figure 6-61.

Figure 6-61. Ratio1 for Single Via




```
ratio1 = (via1-to-gate edge width) * 1/(effective gate width)
```

Examples

```
ratio1 = { 0.0 0.5 1.0 }
```

ratio2

Parameter for Table(s): INVERSE_RG_FACTOR_1 and INVERSE_RG_FACTOR_2 Tables

A table parameter whose values specify the ratio of the distance from the via center to the bottom gate extension edge and the effective gate width. Use only when there are two vias on the gate. Ratio2 can only be used in the inverse_rg_factor_2 table.

Syntax

ratio2 = {*ratio*$_1$ *ratio*$_2$ ... *ratio*$_n$}

Parameters

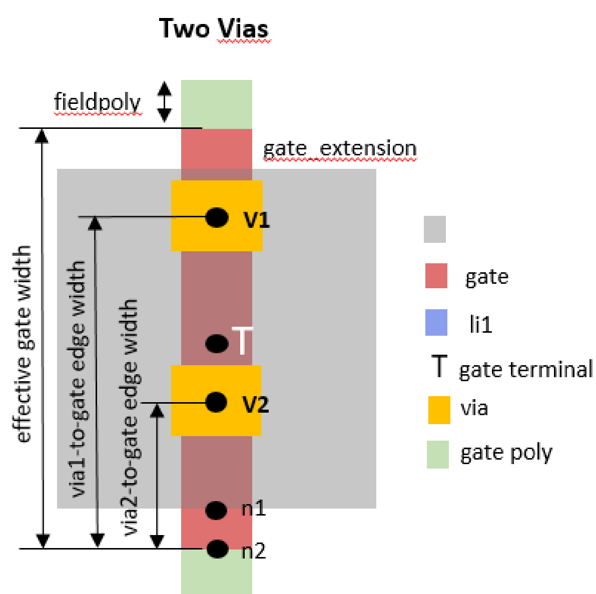
- *ratio*$_1$ *ratio*$_2$ ... *ratio*$_n$

  A list of one or more floating point values that represent the ratio of the distance from a via center to the bottom gate extension edge and the effective gate width enclosed in required braces ({ })

Description

Ratio2 is the ratio of the distance from the V2 center to the bottom of the gate extension edge and the effective gate width as shown in Figure 6-62.

Figure 6-62. Ratio2 For Two Vias




```
ratio2 = (via2-to-gate edge width) * 1/(effective gate width)
```

Examples

```
ratio1 = { 0.0 0.5 1.0 }
ratio2 = { 0.25 0.5 0.75 }
```

ref_layer

Parameter for layer(s): dielectric

A dielectric layer parameter that specifies the metal or dielectric layer around which the conformal coating is contoured.

Syntax

ref_layer = *reference_layername*

Parameters

- *reference_layername*

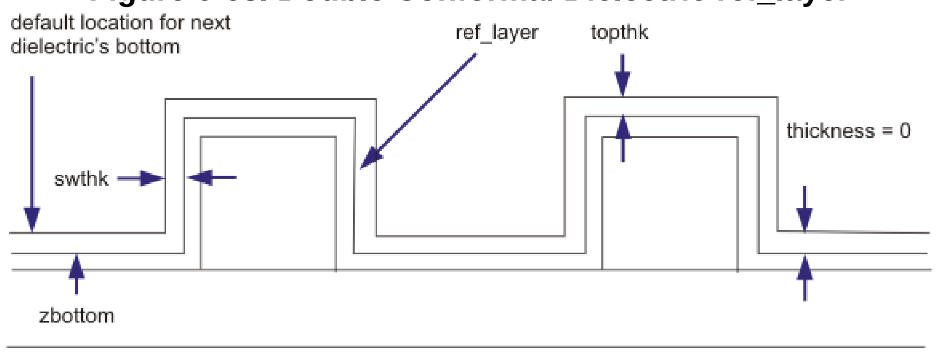
  A string value used to specify the metal or dielectric layer name.

Description

This is an optional parameter when diel_type is planar; a required parameter when diel_type is set to conformal, trench, or spacer.

The ref_layer parameter specifies the metal layer or dielectric layer around which the conformal coating (swthk and topthk) is contoured when diel_type is set to conformal. The reference layer is a metal layer in the case of a single conformal dielectric. The reference layer is a dielectric layer for the upper conformal layer in the case of double (and above) conformal dielectrics (see Figure 6-63).

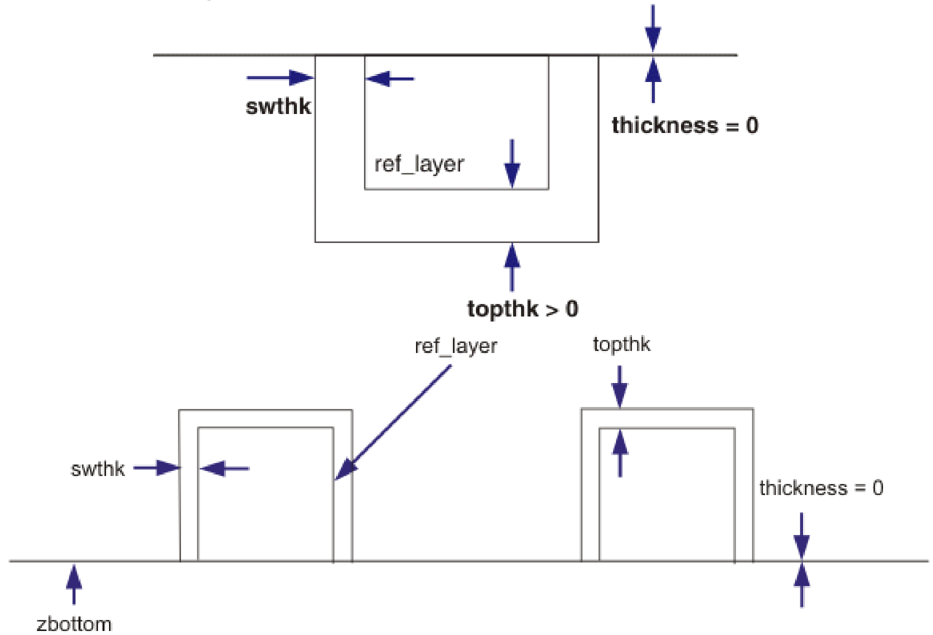
Figure 6-63. Double Conformal Dielectric ref_layer



The default reference layer is the previous conductor or conformal dielectric layer. Therefore, the reference layer may be omitted for planar dielectrics if the layers are defined in the correct order. Specifying the ref_layer allows out-of-order layer definitions. Once the layers have been placed, the previous layer must be a conformal dielectric or a conductor.

Figure 6-64 shows the reference layer when diel_type is set to trench.

Figure 6-64. Trench Dielectric ref_layer



For more information on dielectric layer definition, see Dielectric.

Examples

```
ref_layer = poly1
```

resistance

Parameter for layer(s): base_via, contact, derived, via

A required layer parameter used by base_via, contact, derived, and via layer definitions that specifies the resistance for the layer in ohms. When specified for derived layers, the derived_type must be src_drn_contact, ronly_contact, or ronly_via. This parameter generates n1/n2 tables during calibration. This parameter cannot be used with resistivity or r_sheet parameters in the layer specification.

Syntax

resistance = *value*

Parameters

- *value*

  A floating point value used to specify the resistance in units ohms for this specific contact, via, or derived layer.

Examples

```
resistance = 0.5
```

resistive_only_etch

Parameter for layer(s): conductor, device_li, pad, poly

An optional parameter, used for conductor, device_li, pad, and poly layer definitions, that specifies the layer bias for resistance only.

Syntax

resistive_only_etch = *value*

Parameters

- *value*

  A floating point value that specifies the layer bias for resistance only.

Description

This parameter can be specified for Conductor, Device_li, Pad, and Poly layer definitions. It is equivalent to the resistive_only_etch found in the standardized Interconnect Technology Format (ITF).

Layer_bias applies to both resistance and capacitance. When resistive_only_etch is specified, the layer_bias applies to resistance only and the value of layer_bias is determined using the following equation:

layer_bias = -2 * resistive_only_etch

Figure 6-52 in “layer_bias” shows the layer_bias measurement.

Examples

```
device_li = CA {
   thickness = 0.059
   min_width = 0.026
   min_spacing = 0.051
   resistive_only_etch = 0.0075
}
```

resistivity

Parameter for layer(s): required by conductor, derived, device_li, diffusion, ground, li, pcaux, poly, resistor, substrate, tap, TSV, well; optional for seed, src_drn.

A required parameter for layer definitions conductor, derived, device_li, diffusion, ground, li, poly, resistor, substrate, tap, TSV, and well that specifies the nominal metal resistance for this layer. This parameter is optional for pcaux, seed, and src_drn layer definitions.

Syntax

resistivity = *resistivity_value*

Parameters

- ***resistivity***

  A required floating point value used to specify the nominal metal resistance for this layer. The value can be specified as resistivity (rho). The units for resistivity are ohms * distance_units.

Description

This parameter specifies the nominal metal resistance for the layer. The value may also be specified as a table for resistivity (rho). The nominal value you specify for resistivity may be overridden by specifying the table. This parameter cannot be used if resistance or r_sheet have been defined for this layer. Specifying both resistivity and r_sheet generates an error during parsing. When specified in a TSV layer definition, the value specifies the resistivity of the TSV material.

Examples

```
resistivity = 0.347269
```

resistor

Type: Resistor layer

Part of the layer definition syntax, the resistor layer definition keyword is used to define the electrical and physical characteristics of a resistor layer. The resistor keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. Layers of this type can be specified multiple times.

Syntax

resistor = *layer_name* {
thickness = ***z-direction_metal_ thickness***
min_width = ***minimum_allowed_metal_width***
min_spacing = ***minimum_allowed_metal_spacing***
resistivity = ***resistivity_value***
r_sheet = ***sheet_resistance***
metal_fill = '{'*fill_ratio fill_spacing fill_width* [floating | grounded] '}'
thickness_type = absolute | relative
ztop = *top_ z-coordinate*
zbottom = *bottom_z-coordinate*
measured_from = *layername*
airgap = {rectangle | triangle} min_gap (*v*) spacing (*s1* ... *sn*) width (*w1* ... *wn*)
thickness (*t1* ... *tn*) dist_from_surface (*b1* ... *bn*)
coplanar_min_spacing = *value*
tc1 = *resistance_temperature1_coefficient*
tc2 = *resistance_temperature2_coefficient*
max_rlength = *length_for_resistance_fracturing*
max_width = *maximum_allowed_metal_width*
max_spacing = *maximum_allowed_metal_spacing*
devices = '{'$device_1$ $device_2$ ... $device_n$'}' # a space delimited list of device names
layer_bias = *bias_actual_width_relative_to_drawn_width*
trap_style = top | middle | bottom # where trapezoid measurements are taken from
extra_width = *extra-width_of_side_walls* # for a conductor with a trapezoid shape
swstep = *count*
extension = *amount_layer_extends_past_diffusion_or_contact_layers*
ignore_caps = yes | no
ronly_layers = '{'*space_delimited_list_of_r-only_layer_names* '}'
density_window = '{' '{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'...'{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}''}'
hidden = false | true
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- **thickness**

  A required layer thickness parameter that specifies the thickness of the metal layer in the z-direction.

- **min_width**

  A required layer parameter that specifies the minimum allowed drawn width of metal objects on this layer.

- **min_spacing**

  A required layer parameter that specifies the minimum allowed drawn spacing between resistors on this layer.

- **resistivity**

  A required layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- **r_sheet**

  A required layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- metal_fill

  An optional conducting layer parameter. It is a set of values enclosed in braces, used to define a list of virtual fill parameters.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- measured_from

  An optional conducting layer parameter that measures zbottom or ztop relative to the specified layer.

- airgap

  An optional parameter that specifies the airgap parameters for the resistor layer.

- coplanar_min_spacing

  An optional conducting layer parameter that specifies the minimum spacing value between coplanar layers.

- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.

- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.

- max_rlength

  An optional conducting layer parameter that specifies the maximum length of a wire. This parameter is used for resistance fracturing, which determines how many pieces a resistor should be broken into for representation in a distributed network.

- max_width

  An optional parameter that specifies the maximum allowed metal width.

- max_spacing

  An optional parameter that specifies the maximum allowed metal spacing.

- devices

  An optional layer parameter that specifies a space delimited list of device names enclosed in braces. This parameter is used to specify the devices the layer definition applies to.

- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width relative to drawn width.

- trap_style

  An optional conducting layer parameter that defines where trapezoid measurements are taken from. Permitted values are top, middle, or bottom.

- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- swstep

  An optional conducting layer parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.

- extension

  An optional conducting layer parameter that specifies the amount this layer extends past diffusion or contact layers.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- density_window

  An optional conducting layer parameter specified as {{w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$} ... {w_n h_n $factor_n$ $stepx_n$ $stepy_n$}}. The braces are required for grouping.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Examples

```
resistor = R1_RMOL {
   ztop = 0.0
   measured_from = poly_diel7
   thickness = 0.09
   min_width = 0.045
   min_spacing = 0.045
   resistivity = 0.0366
   tc1 = use my_Mx_layers_tc1
   tc2 = use my_Mx_layers_tc2
}
```

ronly_layers

Parameter for layer(s): base_via, conductor, contact, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn, via

An optional parameter for conducting layer definitions base_via, conductor, contact, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn, and via that specifies a space delimited list of r-only derived layer names that to define electrical variations for the layer. An r-only layer is a derived layer with derived_type equal to ronly, ronly_contact, or ronly_via.

Syntax

ronly_layers = '{'*layer*$_1$ *layer*$_2$ ... *layer*$_n$'}'

Parameters

- '{'*layer*$_1$ *layer*$_2$ ... *layer*$_n$'}'

  A space delimited list of r-only layer names enclosed in braces ({ }).

Examples

Example 1

This example specifies a derived ronly layers M1, M2, M3:

```
ronly_layers = {M1 M2 M3}
```

Example 2

For conductor and li layers where physical parameters remain the same, but electrical properties such as resistivity, tc1, tc2, and so forth may vary, you can specify a derived layer with a derived_type ronly with the varied properties. For example:

```
li = M1 {
   thickness = 0.052
   min_width = 0.021
   min_spacing = 0.06
   resistivity = 0.70
   tc1 = 1.5e-03
   ronly_layers = {M1_light}
}

derived = M1_light {
   derived_type = ronly
   resistivity = 0.24
   tc1 = 2.6e-03
}
```

Example 3

Electrical properties for contact and via layers may be varied using derived layers with a derived_type ronly_contact or ronly_via respectively. For example:

```
via = via12 {
   measured_from = ndiff
   measured_to = M1_norm
   min_width = 0.06
   min_spacing = 0.06
   ronly_layers = {via12_special}
   resistance = 14.2
}

derived = via12_special {
   derived_type = ronly_via
   resistance = 19.7
}
```

rsd_enclosure

Parameter for layer(s): diffusion, pcaux, seed, src_drn, multigate

An optional parameter used in diffusion, pcaux, seed, src_drn, multigate layer definitions that specifies the raised source/drain edge bias for the device.

Syntax

rsd_enclosure = *value*

Parameters

- *value*

  A floating point value used to specify the raised source/drain edge bias.

Examples

```
rsd_enclosure = 0.044
```

rsd_spacing

Parameter for layer(s): diffusion, pcaux, seed, src_drn, multigate

An optional parameter used in diffusion, pcaux, seed, src_drn, multigate layer definitions that specifies the raised source/drain to gate spacing for the device.

Syntax

rsd_spacing = *value*

Parameters

- *value*

  A floating point value used to specify the raised source/drain to gate spacing.

Examples

```
rsd_spacing = 0.044
```

rsd_swslope

Parameter for layer(s): diffusion, pcaux, seed, src_drn, multigate

An optional parameter used in diffusion, pcaux, seed, src_drn, and multigate layer definitions that specifies the angles of the sidewall in degrees for the associated raised source/drain (RSD) layer. Parameters rsd_thickness and rsd_spacing must be specified with rsd_swslope.

Syntax

Syntax 1:

rsd_swslope = {*Sx Sy*}

Syntax 2:

rsd_swslope = *Sy*

Parameters

- {*Sx Sy*} | *Sy*

  Sx Sy are integer values used to specify the outer angle of the sidewall slope in degrees. The default for both *Sx* and *Sy* is 90 degrees. The braces are only required if both *Sx* and *Sy* are specified. Do not use braces if only one value is specified. A single value specifies *Sy* and *Sx* defaults to 90 degrees.

Description

Rsd_swslope is an alternative way to define extra_width for trapezoids, allowing you to define different X and Y trapezoid shapes. *Sx* is the slope X direction representing the left and right side walls. *Sy* is the slope Y direction representing the top and bottom sidewalls.

The slope values are measured in degrees. A slope of 90 is vertical, meaning the shape is not a trapezoid and has 0 extra width. If the slope is less than 90, then the bottom of the trapezoid is smaller than the top (Figure 6-65) and has a positive extra width. If the slope is greater than 90, then the top of the trapezoid is smaller than the bottom (Figure 6-66) and has a negative extra width.

Figure 6-65. rsd_swslope less than 90 degrees

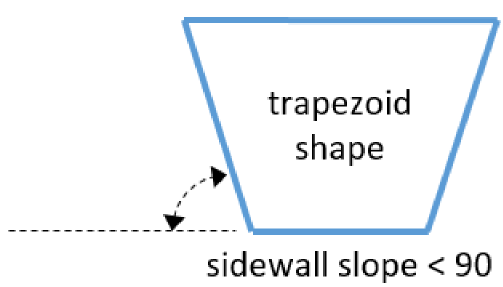
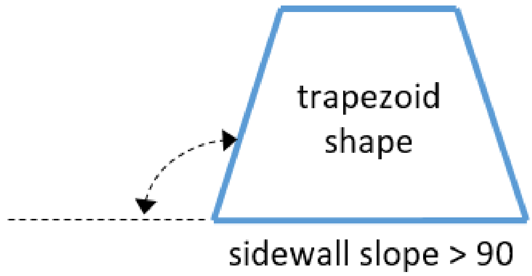


Figure 6-66. rsd_swslope greater than 90 degrees



Examples

Example 1

Including the following rsd_swslope parameter in your diffusion or src_drn layer definition produces a positive extra width:

```
rsd_swslope = {85 85}
```

Example 2

To only specify *Sy*, use the following syntax:

```
rsd_swslope = 45
```

This is equivalent to the following syntax:

```
rsd_swslope = {90 45}
```

rsd_thickness

Parameter for layer(s): diffusion, pcaux, seed, src_drn, multigate

An optional parameter used in diffusion, pcaux, seed, src_drn, and multigate layer definitions that specifies the raised source/drain height for the device.

Syntax

rsd_thickness = *value*

Parameters

- *value*

  A floating point value used to specify the raised source/drain height.

Examples

```
rsd_thickness = 0.034
```

r_sheet

Parameter for layer(s): conductor, derived, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn

A required parameter for conductor, derived, device_li, diffusion, ground, li, poly, and resistor layers that specifies the nominal metal resistance for this layer. This parameter is optional for pcaux, seed, and src_drn layer definitions.

Syntax

r_sheet = ***sheet_resistance***

Parameters

- ***sheet_resistance***

  A required floating point value used to specify the nominal metal resistance for this layer. The units for sheet resistance is ohms.

Description

A parameter that specifies the nominal metal resistance for this layer. The value may also be specified as a table for sheet resistance (rsh). The nominal value you specify for r_sheet may be overridden by specifying the table. This parameter cannot be used if resistance or r_sheet have been defined for this layer. Specifying both resistivity and r_sheet generates an error during parsing.

Examples

```
r_sheet = 0.09029
```

rsh_type

Type: Global Parameters

An optional global parameter that specifies how to interpret the indices for the sheet resistance (rsh) process variation table. By default, the indices refer to drawn values; this is the most common setting. If the parameter is set to actual, it affects the values used in the calculation of sheet resistance. It can only be specified once.

Syntax

rsh_type = { drawn | actual }

Parameters

- drawn

  Specifies that the table should treat indices in the process variation table as drawn values. This is the default.

- actual

  Specifies that the table should treat indices in the process variation table as actual values.

Examples

```
rsh_type = actual
```

seed

Type: Seed layer

Part of the layer definition syntax, this optional layer definition keyword is used to define the area of the poly over the diffusion layer. The seed keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line.

Syntax

seed = *layer_name* {
thickness = *z-direction_metal_ thickness*
min_width = *minimum_allowed_metal_width*
min_spacing = *minimum_allowed_metal_spacing*
resistivity = *resistivity_value*
r_sheet = *sheet_resistance*
metal_fill = '{' *fill_ratio fill_spacing fill_width* [floating | grounded] '}'
ztop = *top_ z-coordinate*
zbottom = *bottom_z-coordinate*
measured_from = *layername*
airgap = {rectangle | triangle} min_gap (*v*) spacing (*s1* ... *sn*) width (*w1* ... *wn*)
thickness (*t1* ... *tn*) dist_from_surface (*b1* ... *bn*)
thickness_type = absolute | relative
tc1 = *resistance_temperature1_coefficient*
tc2 = *resistance_temperature2_coefficient*
max_rlength = *length_for_resistance_fracturing*
max_width = *maximum_allowed _metal_width*
max_spacing = *maximum_allowed_metal_spacing*
maxwidth_for_minspacing = *value*
min_actual_contact_length = *value*
min_actual_contact_width = *value*
multigate = ignore | trim
devices = '{'$device_1$ $device_2$ ... $device_n$'}' # a space delimited list of device names
layer_bias = *bias_actual_width_relative_to_drawn_width*
trap_style = top | middle | bottom # where trapezoid measurements are taken from
extra_width = *extra-width_of_side_walls* # for a conductor with a trapezoid shape
swstep = *count*
extension = *amount_layer_extends_past_diffusion_or_contact_layers*
gate_extension = *value*
li1_extension = *value*
endcap_spacing = *value*
ronly_layers = '{'*space_delimited_list_of_r-only_layer_names*'}'
density_window = '{' '{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'...'{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}''}'
seed_layers = '{'*space_delimited_list_of_seed_layer_names*'}'
diel_over_gate_bottom = *distance_from_gate_poly_to_ag_bottom*
diel_over_gate_eps = *value*

diel_over_gate_extrawidth = *extra_width*
diel_over_gate_thickness = *thickness_of_diel_ag*
diel_over_gate_width = *width_of_diel_ag*
gate_to_cont_min_spacing = *minimum_spacing_between_gate/poly_and_diffusion_contact*
gate_to_via_min_spacing = *minimum_spacing_between_gate/poly_and_diffusion_via*
gate_to_LI1_min_spacing = *minimum_spacing_between_gate/poly_and_li_layer*
gate_to_LI2_min_spacing = *minimum_spacing_between_gate/poly_and_second_li_layer*
ignore_caps = yes | no
ignore_diff_intrinsic = yes | no
ignore_diff_to_diff_under_poly = yes | no
ignore_endcap = yes | no
ignore_gate_intrinsic = yes | no
ignore_gate_to_diff = yes | no
ignore_gateext_to_diff = yes | no
ignore_gateext_to_diff_only = yes | no
ignore_li1 = yes | no
ignore_rsd = yes | no
ignore_rsd_intrinsic = yes |no
rsd_enclosure = *raised_source/drain_edge_bias*
rsd_thickness = *raised_source/drain_height*
rsd_spacing = *raised_source/drain_to_gate_spacing*
rsd_swslope = {*Sx Sy*}
hidden = false | true
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- thickness

  An optional seed layer thickness parameter.

- min_width

  An optional seed layer parameter that specifies the minimum allowed drawn width of metal objects on this layer.

- min_spacing

  An optional seed layer parameter that specifies the minimum allowed metal spacing.

- resistivity

  An optional seed layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- r_sheet

  An optional seed layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- metal_fill

  An optional conducting layer parameter. It is a set of values enclosed in braces, used to define a list of virtual fill parameters.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- measured_from

  An optional conducting layer parameter that measures zbottom or ztop relative to the specified layer.

- airgap

  An optional parameter that specifies the airgap parameters for the seed layer.

- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.

- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.

- max_rlength

  An optional conducting layer parameter that specifies the maximum length allowed for resistance fracturing.

- max_width

  An optional parameter that specifies the maximum allowed metal width.

- max_spacing

  An optional parameter that specifies the maximum allowed metal spacing.

- maxwidth_for_minspacing

  An optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.

- min_actual_contact_length

  An optional poly and seed layer parameter that specifies to override the default minimum actual length of a diffusion contact for a specific device.

- min_actual_contact_width

  An optional poly and seed layer parameter that specifies to override the default minimum actual width of a diffusion contact for a specific device.

- multigate

  An optional parameter that specifies which default multigate model for FinFETs should be used during calibration.

- devices

  An optional layer parameter that specifies a space delimited list of device names enclosed in braces. This parameter is used to specify the devices the layer definition applies to.

- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width relative to drawn width.

- trap_style

  An optional conducting layer parameter that defines where trapezoid measurements are taken from. Permitted values are top, middle, or bottom.

- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- swstep

  An optional conducting layer parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.

- extension

  An optional conducting layer parameter that specifies the amount this layer extends past diffusion or contact layers.

- gate_extension

  An optional parameter that specifies the distance the gate layer extends outside the diffusion area.

- li1_extension

  An optional parameter that specifies the distance the li1 layer extends outside the diffusion area.

- endcap_spacing

  An optional parameter that specifies the distance between the layer geometry and the poly-endcap geometries on both ends.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- density_window

  An optional conducting layer parameter specified as {{w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$} ... {w_n h_n $factor_n$ $stepx_n$ $stepy_n$}}. The braces are required for grouping.

- seed_layers

  An optional layer parameter that specifies one or more gate layers referred to as seed layers, enclosed in braces ({ }). This parameter can only be specified for a poly or seed layer type.

- diel_over_gate_bottom

  An optional seed layer parameter that describes the distance (in microns) from the gate poly to the bottom of a dielectric air gap. This parameter must be specified together with diel_over_gate_width and diel_over_gate_thickness seed layer parameters.

- diel_over_gate_eps

  An optional seed layer parameter that specifies the dielectric gap permittivity where the gap is centered above the gate poly and runs the length of the gate. This parameter can be optionally specified together with diel_over_gate_bottom, diel_over_gate_width, and diel_over_gate_thickness seed layer parameters.

- diel_over_gate_extrawidth

  An optional seed layer parameter that specifies the extra width of the top side of a trapezoid-shaped gap. This parameter can be optionally specified together with diel_over_gate_bottom, diel_over_gate_width, and diel_over_gate_thickness seed layer parameters.

- diel_over_gate_thickness

  An optional seed layer parameter that describes the thickness (in microns) of a dielectric air gap. This parameter must be specified together with diel_over_gate_width and diel_over_gate_bottom seed layer parameters.

- diel_over_gate_width

  An optional seed layer parameter that describes the width (in microns) of a dielectric air gap. This parameter must be specified together with diel_over_gate_bottom and diel_over_gate_thickness seed layer parameters.

- gate_to_cont_min_spacing

  An optional layer parameter that specifies the minimum spacing between the gate poly layer and the diffusion contact layer.

- gate_to_via_min_spacing

  An optional layer parameter that specifies the minimum spacing between the gate poly layer and the diffusion via layer.

- gate_to_LI1_min_spacing

  An optional layer parameter that specifies the minimum spacing between the gate poly layer and the li layer touching the diffusion layer.

- gate_to_LI2_min_spacing

  An optional layer parameter that specifies the minimum spacing between the gate poly layer and the second li layer touching the diffusion layer.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- ignore_diff_intrinsic

  An optional parameter set to either yes or no, that controls whether or not capacitance from diffusion to ground is ignored by calibration. The default is yes, if this parameter is not specified.

- ignore_diff_to_diff_under_poly

  An optional parameter set to either yes or no, that controls whether or not diffusion to diffusion under poly capacitance is ignored by calibration. The default is yes, if this parameter is not specified.

- ignore_endcap

  An optional parameter set to either yes or no, that controls whether or not poly extension to source and drain coupling is ignored by calibration. The default is no, if this parameter is not specified.

- ignore_gate_intrinsic

  An optional parameter set to either yes or no, that controls whether or not capacitance from the gate poly layer to ground is ignored by calibration. The default is yes, if this parameter is not specified.

- ignore_gate_to_diff

  An optional parameter set to either yes or no, that controls whether or not the coupling capacitance between gates to diffusion is ignored by calibration. The default is yes, if this parameter is not specified.

- ignore_gateext_to_diff

  An optional parameter set to either yes or no, that controls whether or not the coupling capacitance between the gate extension to diffusion and rsd is ignored by calibration. The default is no, if this parameter is not specified.

- ignore_gateext_to_diff_only

  An optional parameter set to either yes or no, that controls whether or not the coupling capacitance between the gate extension to diffusion is ignored by calibration. The default is no, if this parameter is not specified.

- ignore_li1

  An optional parameter that controls whether or not the coupling capacitance between poly and li1 and gate and li1 is ignored by calibration. The default is no, if this parameter is not specified.

- ignore_rsd

  An optional parameter set to either yes or no, that controls whether or not the poly to raised source/drain coupling is ignored by calibration. The default is no, if this parameter is not specified.

- ignore_rsd_intrinsic

  An optional parameter set to either yes or no, that controls whether or not the capacitance between the raised source/drain layer and ground is ignored by calibration. The default is no, if this parameter is not specified.

- rsd_enclosure

  An optional layer parameter that specifies the raised source/drain edge bias.

- rsd_thickness

  An optional layer parameter that specifies the raised source/drain height.

- rsd_spacing

  An optional layer parameter that specifies the raised source/drain to gate spacing.

- rsd_swslope

  An optional conducting layer parameter that specifies the sidewall slope of the associated raised source/drain layer. Parameters rsd_thickness and rsd_spacing must be specified with this parameter.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Description

The seed layer must be defined with an associated poly layer by specifying the seed_layers parameter in the poly layer definition. Seed layers use the same parameters as poly layer type, however all seed layer parameters are optional, not required. The seed layer inherits parameter settings from its associated poly layer. Any parameters specified in the seed layer definition override the parameter setting specified in the parent poly layer. For more information on seed layers, see Seed.

Examples

Example 1

When the gate of a poly layer has different properties than the interconnect poly, use a seed layer to specify differences in physical parameters such as thickness. For example:

```
poly = PO {
   thickness = 0.005
   min_width = 0.05
   min_spacing = 0.075
   r_sheet = 7.8
   seed_layers = {PO_gate}
}

seed = PO_gate {
   thickness = 0.052
   r_sheet = 12
}
```

Example 2

In this example, the poly layer fp_poly has a physical location in the layer stack and specifies the poly layer over the field profile. The seed layer ppoly also has a physical location in the layer stack and is used to specify the device region of the poly layer. For example:

```
poly = fp_poly {
   thickness = 0.005
   min_width = 0.05
   min_spacing = 0.075
   resistivity = 3.8
   seed_layers = {ppoly}
}

seed = ppoly {
   thickness = 0.10
   min_width = 0.045
   min_spacing = 0.045
   resistivity = 0.0366
   seed_layers {PMOS1, PMOS2}
}

derived = NMOS1 {
   derived_type = seed
   table = NMOS1_width_RC {}
}

derived = NMOS2 {
   derived_type = seed
   table = NMOS2_width_RC {}
}
```

Example 3

MOS devices, with a dielectric centered above the gate, can be defined by a seed layer that uses the “diel_over_gate” parameters. The diel_over_gate_bottom, diel_over_gate_thickness, and diel_over_gate_width parameters must be specified together. The diel_over_gate_extrawidth and diel_over_gate_eps parameters are optional.

The LVS layer, defined by the dielectric gap over the seed, must be mapped to this calibrated seed layer using the PEX Map statement.

For example, if the following seed layer is defined in MIPT:

```
seed = pc_AG {
    r_sheet = 8.8
    devices = {low_cap_mos}
    tc1 = 0.003
    tc2 = 0.00
    diel_over_gate_width = 0.3
    diel_over_gate_bottom = 0.2
    diel_over_gate_thickness = 0.5
    diel_over_gate_extrawidth = 0.009
}

seed = pc_nonAG {
    r_sheet = 8.8
    devices = {mos}
    tc1 = 0.003
    tc2 = 0.00
}

poly = pc {
    thickness = 0.12
    min_width = 0.1
    min_spacing = 0.15
    resistivity = 1.07
    layer_bias= -0.04
    tc1 = 0.003
    tc2 = 0.00
    seed_layers = {pc_AG, pc_nonAG}
}
```

then, the layer definitions such as the following should be in the LVS rule file:

```
gate = PC_DEV NOT INTERACT AG_DEV
gate_lowcfet = PC_DEV AND AG_DEV
```

where PC_DEV is the gate layer name and AG_DEV is the air gap layer in LVS. And the following PEX Map statements should be included in your LVS or PEX rule file:

```
PEX MAP PC_AG    gate_lowcfet
PEX MAP PC_nonAG gate
```

to extract a dielectric air gap over gate poly.

seed_layers

Parameter for layer(s): poly, seed

An optional parameter used in poly and seed layer definitions that specifies one or more gate layers also referred to as seed layers. Use seed layers to describe different physical parameters such as thickness, width, and spacing for a poly layer.

Syntax

seed_layers = ‘{’*layer*$_1$ *layer*$_2$ ... *layer*$_n$‘}’

Parameters

- {*layer*$_1$ *layer*$_2$ ... *layer*$_n$}

  A space delimited list of one or more seed layer names enclosed in braces ({ }).

Examples

Example 1

You are allowed to specify more than one seed layer for a poly layer. For example:

```
poly = mpoly {
   ...
   seed_layers = {ngate pgate}
   ...
}
```

Example 2

When the gate of a poly layer has different properties than the interconnect poly, use a seed layer to specify differences in physical parameters such as resistivity. For example:

```
poly = mypoly {
   thickness = 0.055
   min_width = 0.05
   min_spacing = 0.075
   resistivity = 4.8
   seed_layers = {gate_poly}
}

seed = gate_poly {
   thickness = 0.09
   resistivity = 12
}
```

spacings

Parameter for layer(s): conductor, poly, diffusion, li

An optional conducting layer parameter that specifies a list of spacing values used to override modeled spacings for the layer. These floating point values must be enclosed in braces ({ }).

Syntax

spacings = '{' s_1 s_2 s_3 ... s_n '}'

Parameters

- '{' s_1 s_2 s_3 ... s_n '}'

  A space delimited list of one or more floating point spacing values enclosed in braces ({ }).

Examples

```
conductor = M1 {
   thickness = 0.063
   hidden = {thickness}
   min_width = 0.034
   min_spacing = 0.02
   tc1 = 1.565e-03
   tc2 = -3.159e-07
   extra_width = 0.012
   swstep = 3
   widths = {0.029 0.0483 0.067667 0.145 }
   spacings = {0.009 0.0149698 0.0248995 0.041415 0.068887 0.114581
               0.190584 0.317001 0.52727 1.74}
}
```

src_drn

Type: Src_drn layer

Part of the layer definition syntax, this optional layer definition keyword is used to define the diffusion layer area used to form a gate. The src_drn keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line.

Syntax

src_drn = *layer_name* {
thickness = *z-direction_metal_ thickness*
min_width = *minimum_allowed_metal_width*
min_spacing = *minimum_allowed_metal_spacing*
resistivity = *resistivity_value*
r_sheet = *sheet_resistance*
metal_fill = '{'*fill_ratio fill_spacing fill_width* [floating | grounded] '}'
ztop = *top_ z-coordinate*
zbottom = *bottom_z-coordinate*
measured_from = *layername*
airgap = {rectangle | triangle} min_gap (*v*) spacing (*s1* ... *sn*) width (*w1* ... *wn*)
thickness (*t1* ... *tn*) dist_from_surface (*b1* ... *bn*)
thickness_type = <u>absolute</u> | relative
tc1 = *resistance_temperature1_coefficient*
tc2 = *resistance_temperature2_coefficient*
max_rlength = *length_for_resistance_fracturing*
max_width = *maximum_allowed _metal_width*
max_spacing = *maximum_allowed_metal_spacing*
maxwidth_for_minspacing = *value*
devices = '{'$device_1$ $device_2$ ... $device_n$'}' # a space delimited list of device names
layer_bias = *bias_actual_width_relative_to_drawn_width*
trap_style = top | middle | bottom # where trapezoid measurements are taken from
extra_width = *extra-width_of_side_walls* # for a conductor with a trapezoid shape
swstep = *count*
extension = *amount_layer_extends_past_diffusion_or_contact_layers*
ignore_caps = yes | <u>no</u>
ronly_layers = '{'*space_delimited_list_of_r-only_layer_names*'}'
density_window = '{' '{'w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$'}'...'{'w_n h_n $factor_n$ $stepx_n$ $stepy_n$'}''}'
src_drn_layers = '{'*space_delimited_list_of_src_drn_layer_names*'}'
rsd_enclosure = *raised_source/drain_edge_bias_value*
rsd_thickness = *raised_source/drain_height_value*
rsd_spacing = *raised_source/drain_to_gate_spacing_value*
rsd_swslope = {*Sx Sy*}
hidden = <u>false</u> | true
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- thickness

  An optional layer thickness parameter that specifies the thickness of the src_drn layer.

- min_width

  An optional layer parameter that specifies the minimum allowed drawn width of metal objects on this layer.

- min_spacing

  An optional parameter that specifies the minimum allowed metal spacing.

- resistivity

  An optional layer parameter that specifies the nominal metal resistance for this layer. This parameter cannot be specified with r_sheet.

- r_sheet

  An optional layer parameter that specifies the sheet resistance for the layer. This parameter cannot be specified with resistivity.

- metal_fill

  An optional conducting layer parameter. It is a set of values enclosed in braces, used to define a list of virtual fill parameters.

- ztop

  The top z-coordinate that is typically zero. This parameter is optional if zbottom is specified; otherwise, this parameter is required.

- zbottom

  The bottom z-coordinate that is typically a negative value. This parameter is optional if ztop is specified; otherwise, this parameter is required.

- measured_from

  An optional conducting layer parameter that measures zbottom or ztop relative to the specified layer.

- airgap

  An optional parameter that specifies the airgap parameters for the src_drn layer.

- thickness_type

  An optional parameter set to either relative or absolute that defines how thickness is measured. This parameter is permitted only if measured_from is set. The default is absolute, if this parameter is not specified.

- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.
- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.
- max_rlength

  An optional conducting layer parameter that specifies the maximum length allowed for resistance fracturing.
- max_width

  An optional parameter that specifies the maximum allowed metal width.
- max_spacing

  An optional parameter that specifies the maximum allowed metal spacing.
- maxwidth_for_minspacing

  An optional conducting layer parameter that specifies a width range value in microns used to determine the min_spacing value for calibration.
- devices

  An optional layer parameter that specifies a space delimited list of device names enclosed in braces. This parameter is used to specify the devices the layer definition applies to.
- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width relative to drawn width.
- trap_style

  An optional conducting layer parameter that defines where trapezoid measurements are taken from. Permitted values are top, middle, or bottom.
- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.
- swstep

  An optional conducting layer parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.
- extension

  An optional conducting layer parameter that specifies the amount this layer extends past diffusion or contact layers.
- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- density_window

  An optional conducting layer parameter specified as {{w_1 h_1 $factor_1$ $stepx_1$ $stepy_1$} ... {w_n h_n $factor_n$ $stepx_n$ $stepy_n$}}. The braces are required for grouping.

- src_drn_layers

  Optional conducting layer parameter that specifies a space delimited list of src_drn layer names enclosed in braces ({ }).

- rsd_enclosure

  Optional conducting layer parameter that specifies the raised source/drain edge bias.

- rsd_thickness

  Optional conducting layer parameter that specifies the raised source/drain height.

- rsd_spacing

  Optional conducting layer parameter that specifies the raised source/drain to gate spacing.

- rsd_swslope

  An optional conducting layer parameter that specifies the sidewall slope of the associated raised source/drain layer. Parameters rsd_thickness and rsd_spacing must be specified with this parameter.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Description

The src_drn layer type represents a physical location in the layer stack. A src_drn layer must be defined with an associated diffusion layer by specifying the src_drn_layers parameter in the diffusion layer definition. Src_drn uses the same parameters as diffusion layer type, except all src_drn parameters are optional, not required. The src_drn layer inherits the parameter settings from its associated diffusion layer. Any parameters specified in the src_drn layer definition override the parameter setting specified in the parent diffusion layer. For more information on src_drn, see Src_drn.

A placed src_drn layer also requires an associated contact layer. The association is specified by setting the measured_from parameter in the contact layer definition to the src_drn layer name. If the src_drn layer is specified using a derived layer, where the derived_type parameter is set to src_drn, the derived layer cannot be used to specify the associated contact since derived layers do not have physical properties. In this case, the contact of the derived src_drn layer will be the same as (inherit) the contact of the associated diffusion layer for the measured_from parameter setting.

Examples

Example 1

In the following src_drn layer definition:

```
src_drn = diff1 {
   thickness = 0.09
   min_width = 0.045
   min_spacing = 0.045
   resistivity = 0.0366
   src_drn_layers = {tndiff tpdiff}
}
```

The values specified for thickness, min_width, min_spacing, resistivity, and src_drn_layers for the src_drn layer override the values specified in the parent diffusion layer.

Example 2

To properly define a placed src_drn layer you need an associated diffusion layer and contact layer. For example:

```
diffusion = diffA {
   ...
   src_drn_layers = {nsd}
   ...
}

src_drn = nsd {
   ...
}

contact = diffContact {
   ...
   measured_from = nsd
   measured_to = M1
   ...
}
```

src_drn_contact_layers

Parameter for layer(s): contact

An optional parameter used in contact layer definitions that specifies the layer or layers that make up the source/drain contact region of devices. Layer names must be enclosed in braces ({ }).

Syntax

src_drn_contact_layers = {*layer*$_1$ *layer*$_2$ ... *layer*$_n$}

Parameters

- {*layer*$_1$ *layer*$_2$ ... *layer*$_n$}

  A space delimited list of src_drn contact layer names enclosed in braces ({ }).

Examples

```
src_drn_contact_layers = {ndiff2 pdiff2}
```

src_drn_layers

Parameter for layer(s): diffusion, src_drn

An optional parameter used in diffusion and src_drn layer definitions that specifies a space delimited list of one or more src_drn layer names.

Syntax

src_drn_layers = {$layer_1$ $layer_2$ ... $layer_n$}

Parameters

- {$layer_1$ $layer_2$ ... $layer_n$}

  A space delimited list of src_drn layer names enclosed in braces ({ }).

Description

Use this parameter to associated a diffusion layer to src_drn layers that specify variations in diffusion thickness, spacing, and width. The src_drn layers describe the different physical parameters such as thickness, width, and spacing for a diffusion layer.

Examples

Example 1

You are allowed to specify more than one src_drn layer for a diffusion layer. For example:

```
diffusion = my_diff {
   ...
   src_drn_layers = {ndiff pdiff}
   ...
}
```

Example 2

When there are differences in diffusion thickness, spacing, or width, use a src_drn layer to specify the differences in physical parameters. The src_drn layers that specify these differences are defined in the diffusion layer with the src_drn_layers parameter. For example:

```
diffusion = diff1 {
   thickness = 0.09
   min_width = 0.045
   min_spacing = 0.045
   resistivity = 0.0366
   src_drn_layers = {pdiff ndiff}
}

src_drn = pdiff {
   thickness = 0.3
   r_sheet = 0.2
}
```

```
src_drn = ndiff {
   thickness = 0.32
   r_sheet = 0.17
}
```

substrate

Type: Substrate layer

Part of the layer definition syntax, this optional layer definition keyword is used to define the substrate layer. The substrate keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line.

Syntax

substrate = *layer_name* {
ztop = *top_z-coordinate*
zbottom = *bottom_z-coordinate*
thickness = *z-direction_metal_thickness*
resistivity = *resistivity_value*
eps = *value*
bulk_min_width = *value*
bulk_resistance = *value*
hidden = false | true
}

Parameters

- **ztop**

  A required top z-coordinate of the substrate layer.

- **zbottom**

  A required bottom z-coordinate of the substrate layer.

- **thickness**

  A required layer parameter that specifies the thickness of the substrate layer.

- **resistivity**

  A required parameter that specifies the substrate layer resistance as rho, typically greater than or equal to 0.

- **eps**

  A required value that specifies the relative permittivity (dielectric constant), typically greater than or equal to 0.

- bulk_min_width

  An optional layer parameter that specifies the minimum width of the bulk layer in microns.

- bulk_resistance

  An optional layer parameter that specifies the sheet resistance of the bulk layer in ohms.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Description

The substrate layer type can be specified once in the MIPT file. The ztop and zbottom parameters are required to establish the substrate. Substrate models need information about substrate layers. Substrate layers require resistivity, eps, and thickness. They cannot contain tables. They should not be coplanar with each other or have gaps. For more information on substrate layer structures, see Substrate Layer Structures.

Examples

```
base = base_layer {
   ztop = 0 # required to define the base plane
            # all other z-dimensions refer to this
   thickness = 1
}

substrate = substrate_layer {
   ztop = -0.18
   zbottom = -0.08
   resistivity = 0.11
   eps = 2.3
}

dielectric = STI {
   zbottom = 0 # optional
   diel_type = planar
   thickness = 0.27 # hSTI
   eps = 4
}
```

svrf_verbatim

Type: Global Parameters

An optional global parameter that specifies a text string or text block that allows the insertion of SVRF statements into the calibrated rule file.

Syntax

svrf_verbatim = *text*

Parameters

- *text*

  A value that specifies a text string or text block that allows the insertion of SVRF statements into the calibrated rule file. If the string contains spaces, then it must be enclosed in quotes (“ ”).

Description

This parameter should be used with caution as some SVRF statements cannot be declared multiple times when parsed by the SVRF parser. There is no checking for compliance with SVRF.

This parameter can be specified multiple times. The text specified by this parameter is written verbatim toward the top of the un-encrypted portion of the calibrated rule deck. It does not affect calibration. You can use this parameter with no value to create an empty line in the header of the calibrated rule file. If the string is enclosed in quotes, spaces are allowed. Multiple lines must be enclosed in braces ({ }). You can encrypt one or more of the lines enclosed in braces by using the #ENCRYPT and #DECRYPT directives.

For more information on global parameters, see Global Parameters. For more information on the encryption directives, see Encryption. For information on SVRF statements, see the *Standard Verification Rule Format (SVRF) Manual*.

Examples

Example 1

```
svrf_verbatim =
svrf_verbatim = "CAPACITANCE ORDER poly m1 m2"
svrf_verbatim = {
                     CAPACITANCE ORDER poly m1 m2
                     N4_PEX = M4 NOT GATE
}
```

Example 2

Given the following layer information:

Table 6-1. Layer Information

Layer Name	Sheet Resistance
psub	109.1
nwell	402.3
pwell	347.5
nplus	633.1
pplus	352.9

To enter the sheet resistance of each bulk layer in your MIPT file, use the svrf_verbatim keyword as follows:

```
svrf_verbatim = ”PEX RESISTANCE PARAMETERS psub BULKRESISTANCE 109.1”
svrf_verbatim = ”PEX RESISTANCE PARAMETERS nwell BULKRESISTANCE 402.3”
svrf_verbatim = ”PEX RESISTANCE PARAMETERS pwell BULKRESISTANCE 347.5”
svrf_verbatim = ”PEX RESISTANCE PARAMETERS nplus BULKRESISTANCE 633.1”
svrf_verbatim = ”PEX RESISTANCE PARAMETERS pplus BULKRESISTANCE 352.9”
```

Example 3

To encrypt certain SVRF statements in the calibrated rule files use the #ENCRYPT and #DECRYPT directives:

```
svrf_verbatim = {
   PEX REDUCE ANALOG YES
   #ENCRYPT
      CAPACITANCE ORDER poly m1 m2
      N4_PEX = M4 NOT GATE
   #DECRYPT
}
```

This example produces a rules.C file that has the following in the header section:

```
...
PEX REDUCE ANALOG YES
#DECRYPT%9"~6B$JA"P&@I1804%J<'_.H"#O9I&)M\:JU\<K4Y\
L%?1!#WIG$!:S7W6A%2T)>[;?UCA!"@3^UE:P%1QMT4:*!H;J;1Q!!"[X<'@?U.`WA<L7,O4"
R02A!!"R*E"CSA\+0^Y:"RO@VLX4!!!"E^W["MS@>7R,=2Q$9L>X5@N3\/
>>%CEC7)4!AXFL:Q!#,L#I!]@%B%5E2.(%Q,^$DQ!!")9OAR4C?L!@9&WA#J']E[1!!"8>KM$
5`!DR6L!&$_X65H#Q!!"\':3I,$\D]_J"]S#"N!!%!!!"
#ENCRYPT
...
```

swslope

Parameter for layer(s): dielectric

An optional conformal or trench dielectric layer parameter used to specify the angle of the side wall in degrees.

Syntax

swslope = *value*

Parameters

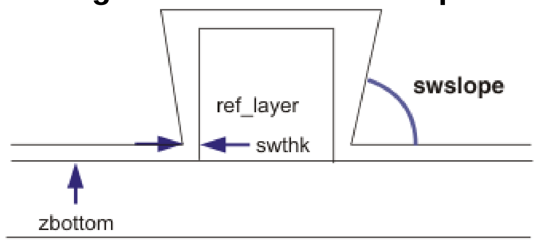
- *value*

  An integer valued used to specify the angle of the side wall in degrees. It should be a positive value less than 180. The default is 90 degrees.

Description

The angle of the side wall, swslope, is shown in Figure 6-67. If this parameter has not been specified the parser issues an informational message that notifies you that the parameter was not found and the default of 90 degrees is used. The xCalibrate tool also verifies that the actual slope angle entered intersects the top plane for this material thickness. This parameter may only be used when the diel_type parameter is set to conformal or trench.

Figure 6-67. Sidewall Slope



For more information on dielectric layer definition, see Dielectric.

Examples

```
swslope = 90
```

swstep

Parameter for layer(s): dielectric, conductor, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn

An optional conformal dielectric or conducting layer parameter that specifies the number of rectangles used to model trapezoid shapes. Used only by the Calibre xACT 3D field solver.

Syntax

swstep = *count*

Parameters

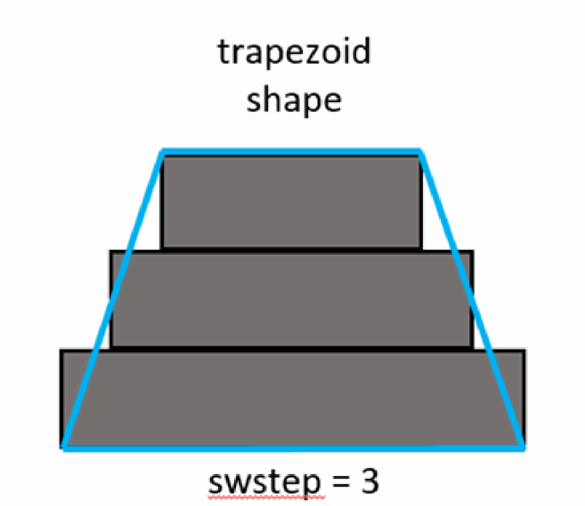
- *count*

  An integer value used to specify the number of rectangles in the trapezoid shape. The default is 2.

Description

Trapezoid shapes are modeled by a set of rectangles stacked like pyramids (Figure 6-68). The swstep parameter defines the number of rectangles used to model the trapezoid shape. A larger rectangle count means more accuracy and longer field solver run times.

Figure 6-68. swstep count of 3



Examples

```
conductor = M1 {
   thickness = 0.063
   min_width = 0.034
   min_spacing = 0.02
   tc1 = 1.565e-03
   tc2 = -3.159e-07
   extra_width = 0.012
   swstep = 3
   widths = {0.029 0.0483 0.067667 0.145 }
   spacings = {0.009 0.0149698 0.0248995 0.041415 0.068887 0.114581
               0.190584 0.317001 0.52727 1.74}
}
```

swthk

Parameter for layer(s): dielectric

An optional dielectric layer parameter used to specify the conformal coating thickness on the side wall.

Syntax

swthk = *value*

Parameters

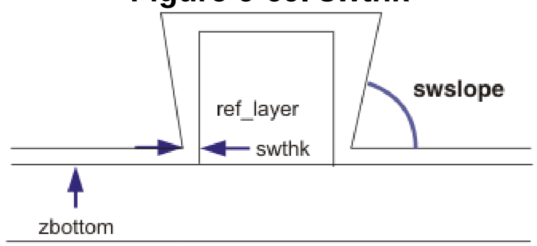
- *value*

  A floating point value used to specify the conformal coating thickness on the side wall.

Description

This swthk parameter is required when diel_type parameter is set to conformal, trench, or spacer. The value specifies the conformal coating thickness on the side wall as shown in Figure 6-69.

Figure 6-69. swthk



For more information on dielectric layer definition, see Dielectric.

Examples

```
swthk = 0.075
```

table

Type: Table Syntax

An optional structure definition keyword used to provide a unified way to input process variation information to xCalibrate.

Syntax

Syntax 1, one-dimensional table

```
table = <table_reference_name> {
   property = <table_property>
   table_type = {R | C | RC}
   dim_type = {drawn | actual}
   value_type = {absolute | delta}
   {{equation = {<equation to evaluate, optional>}} |
    {variable1 = {w1 w2  ... wn}
     value = {v(w1) v(w2) ... v(wn) }}
   }
}
```

Syntax 2, two-dimensional table

```
table = <table_reference_name> {
   property = <table_property>
   table_type = {R | C | RC}
   dim_type = {drawn | actual}
   value_type = {absolute | delta}
   bias_type = {preferred | nonpreferred}
   {{equation = {<equation to evaluate, optional>}} |
   {<variable>1 = {w1 w2  ... wn}
    <variable>2  = {s1 s2  ... sn}
    value = {v(w1,s1) v(w1,s2) ... v(w1,sn),
             ...,
             v(wn,s1) v(wn,s2) ... v(wn,sn)}
   }
}
```

Parameters

- ***table_reference_name***

  A required unique user-specified name.

- **property**

  A required parameter that specifies the table property being described. Each table property has a specific set of ordered variables associated with it.

  Valid table property types include: WIDTH, MULTI_BIAS, THICKNESS_TOP, THICKNESS_BOT, THICKNESS, GATE_FRINGE, GATE_FRINGE4, LI_GATE_FRINGE, LI1_WIDTH, LI2_WIDTH, DIELECTRIC_CONSTANCE, EXTRA_WIDTH, SIDEWALL_K, RSH, RSH_T, RHO, RHO_T, RESISTANCE, TC1, TC2, CONTACT_WIDTH, CONTACT_BIAS, VWIDTH, VLENGTH, and VRESISTANCE.

"Table Property" on page 113 shows which options are available for each table property. It also identifies which properties depict drawn or actual dimensions, or both, and what value type setting should be used.

- **table_type**

  A required parameter that must be either R, C, or RC. Note that not every keyword is supported with every table type.

- **dim_type**

  A required parameter that must be either drawn or actual. This parameter is used to specify if table property is in drawn or actual dimensions. Note that not every keyword is supported with every dimension type.

- **value_type**

  A required parameter that must be either absolute or delta. Note that not every keyword is supported with every value type.

- bias_type

  An optional parameter used to specify whether the actual width values described in the table are for the preferred or non-preferred direction. This parameter may only be used in two-dimensional width or multi_bias property tables.

- equation

  An optional parameter whose value specifies an equation to evaluate. This parameter cannot be used with variable and value parameters.

- *variable* $_1$

  A parameter that specifies variables that are affected by this table definition. Permitted values for variable include width, length, thickness, spacing, density, co_co_spacing, and gate_co_spacing. "Variable Keywords" on page 148 lists the available variable keywords and their unit value. This parameter cannot be used with the equation parameter.

- value

  An optional parameter whose value specifies the replacement values for the specified variable. This parameter cannot be used with the equation parameter.

Description

A table is typically specified in-line with a layer definition in the MIPT file, but can also be referenced using the use keyword. For more information on use keyword, see Use Directive.

Examples

```
table = my_table {
   property = width
   table_type = RC
   dim_type = drawn
   value_type = delta
   width = { 5 6 7 8 }
   value = { 0.12 0.13 0.14 0.15}
}
```

table_type

table structure parameter

A required table structure parameter used to specify how the table affects capacitance and resistance extraction calculations for the layer.

Syntax

table_type = {**R** | **C** | **RC**}

Parameters

- **R**

  Specifies that the table should be applied to resistance only.

- **C**

  Specifies that the table should be applied to capacitance only.

- **RC**

  Specifies that the table should be applied to both resistance and capacitance.

Description

A parameter that specifies how the table affects capacitance and resistance extraction calculations for the layer. Permitted values are R, C, or RC. Note that not every table_type is supported by every table property. Table 5-3 describes which table_type settings are allowed for each table property. For more information on tables, see Table Syntax.

Examples

```
table_type = RC
```

tap

Type: Tap layer

Part of the layer definition syntax, this optional layer definition keyword is used to define a tap layer. The tap keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. This layer type can be specified more than once in the MIPT file.

Syntax

tap = *layer_name* {
ztop = *top_z-coordinate*
zbottom = *bottom_z-coordinate*
thickness = *z-direction_metal_thickness*
resistivity = *resistivity_value*
bulk_min_width = *value*
bulk_resistance = *value*
hidden = <u>false</u> | true
}

Parameters

- **ztop**

  A required top z-coordinate of the tap layer.

- **zbottom**

  A required bottom z-coordinate of the tap layer.

- **thickness**

  A required layer parameter that specifies the thickness of the tap layer.

- **resistivity**

  A required parameter that specifies the tap layer resistance as rho, typically greater than or equal to 0.

- bulk_min_width

  An optional layer parameter that specifies the minimum width of the bulk layer in microns.

- bulk_resistance

  An optional layer parameter that specifies the sheet resistance of the bulk layer in ohms.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Description

A tap layer is used to connect power and ground nets to the substrate. The vertical and lateral resistance needs to be modeled for the substrate extraction model. Tap layers require the parameters for resistivity and thickness. Tap layers have placement information defined relative to the top of the substrate. This information is not used for capacitance calculations in Calibre xRC or Calibre xACT 3D. The ztop and zbottom parameters are required to establish the tap layer. For more information on tap layers, see Substrate Layer Structures.

Examples

```
tap = ntap {
   zbottom = -0.01
   ztop = 100
   thickness = 0.1
   resistivity = 11
}

substrate = substrate_layer {
   ztop = -0.18
   thickness = 0.1
   resistivity = 0.11
   eps = 2.3
}

dielectric = STI {
   zbottom = 0
   diel_type = planar
   thickness = 0.27 # hSTI
   eps = 4
}
```

tc1

Parameter for layer(s): base_via, conductor, contact, derived, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn, TSV, via

An optional parameter for layer definitions base_via, conductor, contact, derived, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn, TSV, and via that specifies the first order temperature coefficient (TC1) for resistance for the layer.

Syntax

tc1 = *value*

Parameters

- *value*

  A value that specifies the first order temperature coefficient (TC1) for resistance.

Description

Specifies the first order temperature coefficient (TC1) for resistance for the layer. When specified in a TSV layer definition, the value specifies the first order temperature coefficient of the TSV material.

The value may also be specified as a table for TC1. The nominal value you specify for tc1 may be overridden by specifying the table. For more information on tables, see Table Syntax.

Examples

```
tc1 = 0.00231
```

tc2

Parameter for layer(s): base_via, conductor, contact, derived, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn, TSV, via

An optional parameter for layer definitions base_via, conductor, contact, derived, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn, TSV, and via that specifies the second order temperature coefficient (TC2) for resistance for the layer.

Syntax

tc2 = *value*

Parameters

- *value*

  A value that specifies the second order temperature coefficient (TC2) for resistance.

Description

Specifies the second order temperature coefficient (TC2) for resistance for the layer. When specified in a TSV layer definition, the value specifies the second order temperature coefficient of the TSV material.

The value may also be specified as a table for TC2. The nominal value you specify for tc2 may be overridden by specifying the table. For more information on tables, see Table Syntax.

Examples

```
tc2 = -5.048e-07
```

temperature

Type: Global Parameters

An optional global parameter that specifies the nominal temperature for the calibration in units Celsius. The default is 25 C if you do not include this parameter in your MIPT file. It can only be specified once.

Syntax

temperature = *value*

Parameters

- *value*

  A value that represents the nominal temperature for the calibration in units Celsius.

Examples

```
temperature = 28
```

thickness

Parameter for layer(s): base, conductor, contact, device_li, dielectric, diffusion, ground, li, pcaux, poly, resistor, seed, substrate, src_drn, tap, well

A required parameter for layer definitions base, conductor, contact, device_li, dielectric, diffusion, ground, li, pcaux, poly, resistor, substrate, tap, and well that specifies the nominal metal resistance for this layer. This parameter is optional for seed and src_drn layer definitions.

Syntax

thickness = ***value***

Parameters

- ***value***

  A required positive integer or floating point value that specifies the thickness of the layer being defined.

Description

The thickness parameter represents different thickness measurements depending on the layer type.

base — the thickness of the base layer.

dielectric — the thickness of the dielectric layer in the z-direction. The thickness may be a zero value if the layer is diel_type = conformal. For a trench dielectric (diel_type = trench) the thickness must always be 0.

conductor, poly, diffusion, li — the z-direction metal thickness.

substrate, well, tap — for substrate layer types it is the thickness of the substrate. Thickness is required to establish a substrate layer. The thickness parameter can only be specified with zbottom parameter; It cannot be specified with both ztop and zbottom.

salicide — salicide thickness, deep_sd thickness.

For more information on layer definition types, see Layer Definitions.

Examples

```
thickness = 0
```

thickness_type

Parameter for layer(s): conductor, device_li, dielectric, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn

An optional parameter for layer definitions conductor, device_li, dielectric, diffusion, ground, li, pcaux, poly, resistor, seed, and src_drn, only valid and used together with the measured_from parameter, that specifies how thickness is measured. This keyword is only valid if the measured_from parameter is specified for the layer.

Syntax

thickness_type = absolute | relative

Parameters

- absolute

  Specifies that the method of thickness measurement is absolute when ztop or zbottom is defined. The default is absolute if this parameter is not specified.

- relative

  Specifies that the method of thickness measurement is relative to the layer. Use this setting when the measured_from parameter is specified, and ztop or zbottom parameters are not defined for the layer. Use this setting to override the behavior of the thickness parameter.

Examples

```
thickness_type = relative
```

top_enclosure

Parameter for layer(s): TSV

An optional TSV layer parameter used to specify the top enclosure covered by the metal "to_layer" for the through silicon via.

Syntax

top_enclosure = *value*

Parameters

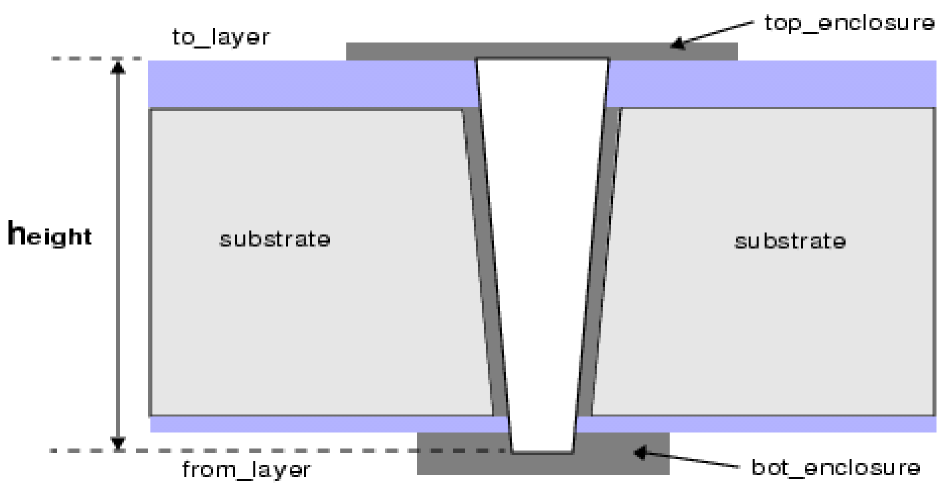
- *value*

  A floating point value that specifies the top enclosure covered by the metal "to_layer".

Description

An optional TSV parameter that specifies the top enclosure covered by the metal "to_layer" for the through silicon via. This parameter is used only in TSV layer definitions. The top_enclosure is shown in Figure 6-70.

Figure 6-70. top_enclosure of TSV Layer



For more information on the TSV layer definition, see TSV.

Examples

```
top_enclosure = 0.044
```

topthk

Parameter for layer(s): dielectric

An optional dielectric layer parameter that specifies the conformal coating thickness on top of a conductor.

Syntax

topthk = *value*

Parameters

- *value*

  A floating point value used to specify the conformal coating thickness on top of a conductor.

Description

This dielectric parameter is required when diel_type is set to conformal, trench, or spacer. The value specifies the conformal coating thickness on top of a conductor. The topthk value may be a positive or negative value as shown in Figure 6-71, Figure 6-72, and Figure 6-73.

Figure 6-71. Trench Dielectric With Positive topthk

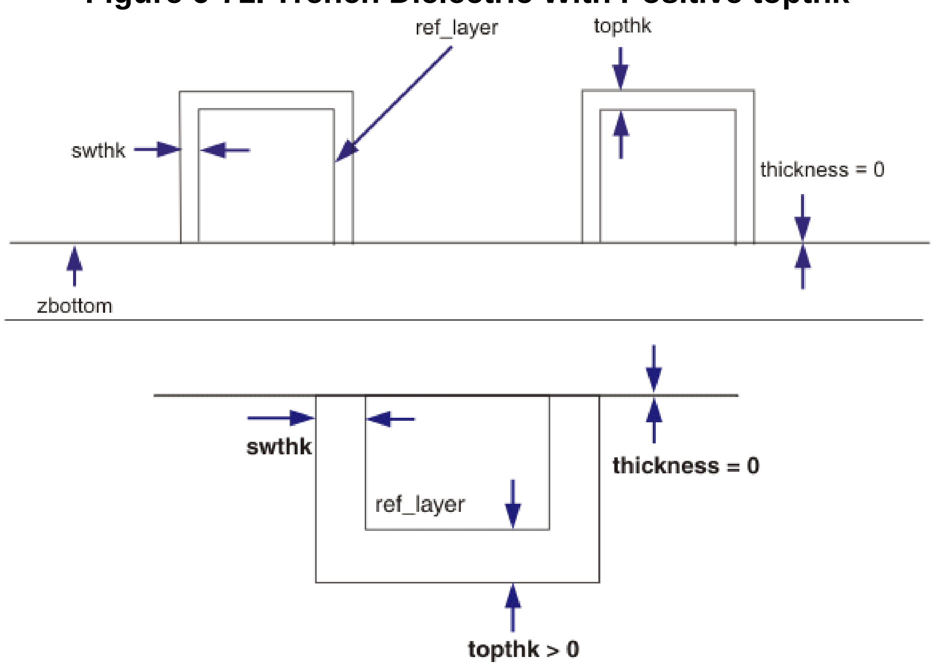


Figure 6-72. Trench Dielectric With Negative topthk

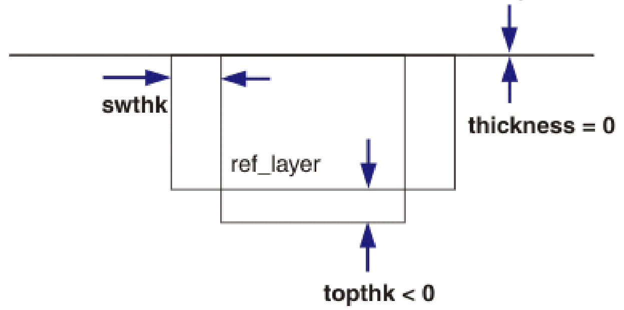
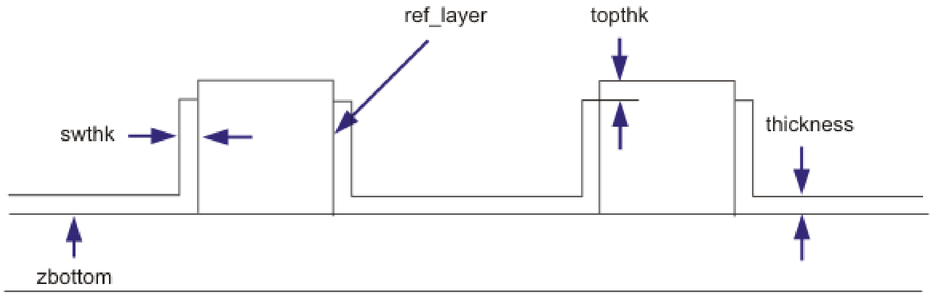


Figure 6-73. Conformal Dielectric With Negative topthk



For more information on dielectric layer definition, see Dielectric.

Examples

```
topthk = 0.02
```

trap_style

Parameter for layer(s): conductor, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, src_drn

An optional parameter for layer definitions conductor, device_li, diffusion, ground, li, pcaux, poly, resistor, seed, and src_drn, that specifies where the drawn or actual width is measured from. This parameter must be specified with the layer_bias parameter for the conductor layer. The drawn or actual width is measured in the middle of the trapezoid by averaging the top and bottom edges of the trapezoid.

Syntax

trap_style = bottom | middle | top

Parameters

- bottom

  Specifies that the drawn or actual width is measured from the bottom edge of the trapezoid.

- middle

  Specifies that the drawn or actual width is measured from the middle of the trapezoid by averaging the top and bottom edges of the trapezoid. This is the default.

- top

  Specifies that the drawn or actual width is measured from the top edge of the trapezoid.

Examples

```
trap_style = middle
```

trench_contact_extension_length

Parameter for layer(s): multigate

An optional multigate layer parameter used to specify the nominal extension of the trench contact beyond the diffusion. This parameter is used mainly for narrow devices.

Syntax

trench_contact_extension_length = *value*

Parameters

- *value*

  A floating point value that specifies the nominal extension of the trench contact beyond the diffusion.

Examples

```
trench_contact_extension_length = 8.0
```

trim

Parameter for layer(s): base_via, via

An optional layer parameter that controls how resistance is calculated for the via.

Syntax

trim = no | yes | actual | merge | bridge

Parameters

- no

  Specifies to use drawn via dimensions when calculating via resistance.

- yes

  Specifies to use trimmed via dimensions when calculating via resistance. This is the default.

- actual

  Specifies to use actual via dimensions when calculating via resistance.

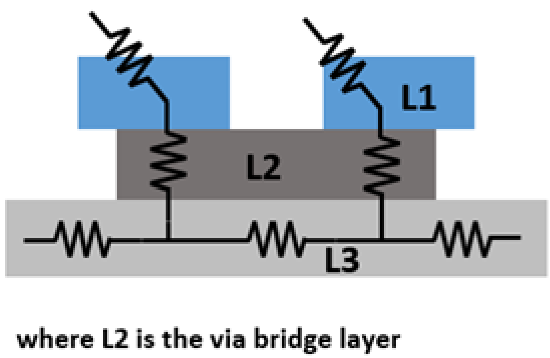
- merge

  Specifies to use the merged area of the via when calculating via resistance. Use this setting for coplanar vias.

- bridge

  Specifies to use the bridge area of a via when calculating resistance. Use this setting for small pitch redundant wire vias. A via bridge crosses two conductors completely as shown in Figure 6-74.

Figure 6-74. Via Bridge



Examples

Example 1

To use trimmed via dimensions when calculating via resistance, include the following parameter setting in your via layer definitions:

```
trim = yes
```

Example 2

To include the effects of a conductor bridge layer between vias when calculating via resistance, include the following parameter setting in your affected via layer definitions:

```
trim = bridge
```

TSV

Type: TSV layer

Part of the layer definition syntax, this optional layer definition keyword is used to define the electric and physical characteristics of a through silicon via (tsv). The tsv keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. Layers of this type can be specified multiple times in the MIPT file.

Syntax

tsv = ***layer_name*** **{**
    measured_from = ***layername*** # bottom most layer (back side)
    measured_to = ***layername*** # top most layer (front side)
    radius = ***radius_of_the_TSV***
    hollow_radius = ***radius_of_the_hollow_area***
    height = ***height_of_the_TSV***
    resistivity = ***resistivity_value_of_the_TSV_material***
    depletion_width = ***depletion_region_width***
    tc1 = *resistance_temperature1_coefficient*
    tc2 = *resistance_temperature2_coefficient*
    top_enclosure = *top_enclosure_value* # covered by the metal “to_layer”
    bot_enclosure = *bottom_enclosure_value* # covered by the metal “from_layer”
    extra_width = *extra-width_of_side_walls* # for tapered style TSV
    insulator = {{$thickness_1$, ER_1} {$thickness_2$, ER_2 }... {$thickness_n$ ER_n}}
    hidden = <u>false</u> | true
    }

Parameters

- ***layer_name***

  A required unique user-specified name.

- **measured_from**

  A required layer parameter that specifies the back side metal layer that the TSV connects to.

- **measured_to**

  A required layer parameter that specifies the front side metal layer that the TSV connects to.

- **radius**

  A required layer parameter that specifies the radius of the TSV.

- **hollow_radius**

  A required layer parameter that specifies the radius of the hollow area of the TSV.

- **height**

  A required layer parameter that specifies the height of the TSV.

- **resistivity**

  A required layer parameter that specifies the resistivity of the TSV material.

- **depletion_width**

  A required layer parameter that specifies the depletion region width for the TSV.

- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.

- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.

- top_enclosure

  An optional parameter that specifies the top enclosure covered by the metal “to_layer” for the TSV.

- bot_enclosure

  An optional parameter that specifies the bottom enclosure covered by the metal “from_layer” for the TSV.

- extra_width

  An optional layer parameter that specifies the extra-width of side walls for a tapered TSV.

- insulator

  An optional layer parameter that specifies one or more insulators for the TSVs in the process. Each insulator is described by a thickness and dielectric constant value.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Examples

```
tsv = myTSV {
   measured_from = bmet1
   measured_to = met1
   radius = 1.89
   hollow_radius = 0.87
   height = 10.3
   top_enclosure = 1.3
   bot_enclosure = 0.6
   depletion_width = 0.5
   resistivity = 8.3
}
```

tsv_model

Type: Global Parameters

An optional global parameter that specifies whether the TSV radius is retrieved from the rules (device) or the layout (layout) during calibration. It can only be specified once.

Syntax

tsv_model = <u>device</u> | layout

Parameters

- <u>device</u>

  Specifies the TSV radius is retrieved from the device. This is the default.

- layout

  Specifies the TSV radius is retrieved from the layout.

Examples

If your TSV model is layout:

```
tsv_model = layout
tsv_radius_type = hole

tsv = myTSV {
   measured_from = bmet1
   measured_to = met1
   radius = 1.89
   hollow_radius = 0.87
   hollow_diel = 1.4
   height = 40
   top_enclosure = 1.3
   bot_enclosure = 0.6
   depletion_width = 0.5
   resistivity = 0.02
}
```

tsv_radius_type

Type: Global Parameters

An optional global parameter that specifies whether to treat the radius of the TSV as metal lined or just a hole. It can only be specified once.

Syntax

tsv_radius_type = metal | hole

Parameters

- metal

  Specifies that the radius of the TSV represents the radius of the metal in the TSV. This is the default.

- hole

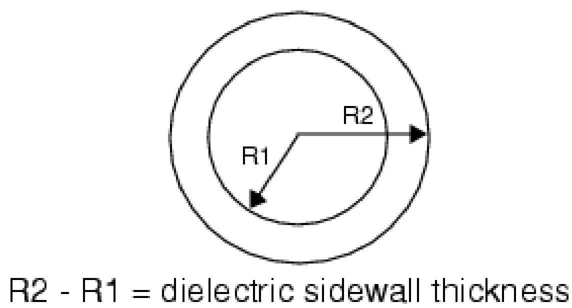
  Specifies that the radius of the TSV represents the radius of the TSV as a hole in the substrate (metal and insulator).

Description

Specifies whether to treat the radius of the TSV as metal lined or just a hole.

In Figure 6-75, R2 is the radius of the hole and R1 is the radius of the TSV when it is metal lined. The difference between R2 and R1 is the dielectric sidewall thickness.

Figure 6-75. TSV Radius



Examples

If your TSV is a hole:

```
tsv_model = layout
tsv_radius_type = hole
tsv = myTSV {
   measured_from = bmet1
   measured_to = met1
   radius = 1.89
   hollow_radius = 0.87
   hollow_diel = 1.4
   height = 40
   top_enclosure = 1.3
   depletion_width = 0.5
   resistivity = 0.1
}
```

ubump

Type: uBump layer

Part of the layer definition syntax, this optional layer definition keyword is used to define the electric and physical characteristics of a ubump. The ubump keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. Layers of this type can be specified multiple times in the MIPT file.

Syntax

ubump = *layer_name* {
measured_from = *layername* # bottom most layer
measured_to = *layername* # top most layer
min_width = *minimum_allowed_metal_width*
mid_width = *middle_width_of_the_ubump*
enclosure_down = *enclosure_with_respect_to_lower_conductor*
enclosure_up = *enclosure_with_respect_to_upper_conductor*
resistivity = *resistivity_value_of_the_ubump_material*
hidden = <u>false</u> | true
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- **measured_from**

  A required layer parameter that specifies the bottom most layer that the uBump connects to.

- **measured_to**

  A required layer parameter that specifies the top most layer that the uBump connects to.

- **min_width**

  A required parameter that specifies the minimum allowed metal width of the ubump.

- **mid_width**

  A required parameter that specifies the middle metal width of the ubump.

- **enclosure_down**

  A required parameter that specifies enclosure with respect to a lower conductor. For ubump the value of enclosure_down is the same as the value specified for enclosure_up.

- **enclosure_up**

  A required parameter that specifies enclosure with respect to an upper conductor. For ubump the value of enclosure_up is the same as the value specified for enclosure_down.

- **resistivity**

  A required layer parameter that specifies the resistivity of the ubump material.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Examples

```
ubump = my_bump {
   measured_from = WI
   measured_to = BMB
   min_width = 1.5
   mid_width = 2.0
   enclosure_up = 1.8
   enclosure_down = 1.8
   resistivity = 0.02
}
```

value

table structure parameter

A required table structure parameter used to specify a space-delimited set of values enclosed in braces that represent value variations for the property.

Syntax

value = '{'$v_{(w1)}$ $v_{(w2)}$ ... $v_{(wn)}$ '}'

Parameters

- {$v_{(w1)}$ $v_{(w2)}$ ... $v_{(wn)}$ }

  A set of space-delimited floating point values enclosed in braces ({ }), used to specify the variation values applied to associated variable in the table definition.

Description

This required parameter is the last entry of each table. The table entries for value are the absolute or delta values applied to the variation property. The end of line character of each line in the value array must be a comma ",". The adjusted data in the value arrays must conform to the corresponding minimum values defined in the MIPT file. The adjusted widths must not be smaller than the minimum width for the layer. The adjusted spacing must not be smaller than the minimum spacing for the layer. If a bias has been set in the MIPT file, an error message is generated if a table has settings that are out of range according to that bias.

The number of values that appear depends on the number of variables specified for the table. A table with only one variable is a single dimensional table. A table that specifies 2 variables is called a two dimensional table and so forth. For example, the value list for a two dimensional list looks like:

```
table = table_reference_name {
   property = table_property
   table_type = {R | C | RC}
   dim_type = {drawn | actual}
   value_type = {absolute | delta}
   equation = {<equation to evaluate, optional>}
   variable1 = {w1 w2 ... wn}
   variable2 = {s1 s2 ... sn}
   value = { v(w1,s1) v(w1,s2) ... v(w1,sn),
             ...,
             v(wn,s1) v(wn,s2) ... v(wn,sn)}
}
```

For more information on tables, see Table Syntax.

Examples

```
table = my_table {
   property = width
   table_type = RC
   dim_type = drawn
   value_type = delta
   width = { 5 6 7 8 }
   value = { 0.12 0.13 0.14 0.15}
}
```

value_type

table structure parameter

A required table structure parameter used to specify how the values provided in the table are treated.

Syntax

value_type = {absolute | delta}

Parameters

- absolute

  Specifies that the table should treat the specified values as absolute.

- delta

  Specifies that the table should treat the specified values as delta values.

Description

A required parameter that specifies how the values provided in the table are treated. Permitted types are absolute or delta. Note that not every value_type is supported by every table property. Table 5-3 describes which value_type settings are allowed for each table property. For more information on tables, see Table Syntax.

Examples

```
value_type = delta
```

variable

table structure parameter

A required table structure parameter used to specify variables that will be affected by the table definition.

Syntax

variable = ‘{’v_1 v_2 ... v_n‘}’

Parameters

- { v_1 v_2 ... v_n }

  A space delimited set of floating point values enclosed in braces that specify the values applied to the variation property. The values must be specified in ascending order (smallest to largest).

Description

The *variable* name is followed by an equal sign and a set of space delimited floating point values enclosed in braces ({ }). Permitted values for *variable* include width, length, thickness, spacing, width1, spacing1, width2, spacing2, density, co_co_spacing, gate_co_spacing, li1, li2, or corners. See Table 5-4 which lists the complete list of available variable keywords and their unit value. For more information on tables, see Table Syntax.

Examples

```
co_co_spacing = {0.045 0.08 0.2 2.5}
```

version

Type: Global Parameters

An optional global parameter that specifies the version number of the manufacturing technology (process) being used in the MIPT file. If it is a single string without spaces, the quotation marks are not required. If the string is enclosed in quotes, spaces are allowed. It can only be specified once.

Syntax

version = *version_number*

Parameters

- *version_number*

  A string value that specifies the version number of the manufacturing technology (process) being used in the MIPT file.

Examples

Example 1

```
version = 2.4
```

Example 2

```
version = "This is v2.4 of the process"
```

via

Type: Via layer

Part of the layer definition syntax, this optional layer definition keyword is used to define the electric and physical characteristics of a via. Layers of this type can be specified multiple times in the MIPT file. Vias of different sizes and different resistance may be defined between the same conducting layers.

Syntax

via = ***layer_name* {**
measured_from = ***layername*** # lower conductor layer_name
measured_to = ***layername*** # upper conductor layer_name
area = ***area_of_via***
min_width = ***minimum_allowed_metal_width***
min_spacing = ***minimum_allowed_metal_spacing***
resistance = ***resistance_per_via***
enclosure_down = ***enclosure_with_respect_to_lower_conductor***
enclosure_up = ***enclosure_with_respect_to_upper_conductor***
max_area = *maximum_via_area*
max_length = *maximum_distance_between_vias*
min_length = *length_of_shape*
tc1 = *resistance_temperature1_coefficient*
tc2 = *resistance_temperature2_coefficient*
gate_to_via_spacing_min = *minimum_allowed_gate_to_via_spacing*
extra_width = *extra-width_of_side_walls*
extrapolation = <u>yes</u> | no
ignore_caps = yes | <u>no</u>
layer_bias = *bias_actual_relative_to_drawn_via_width&length*
ronly_layers = '{'*space_delimited_list_of_r-only_layernames*'}'
parallel_to_gate = <u>no</u> | yes
hidden = <u>false</u> | true
trim = <u>no</u> | yes | actual | merge | bridge
via_r_select_up_down = <u>min</u> | max
interpolation = <u>gperarea</u> | invgperarea
}

Parameters

- ***layer_name***

  A required unique user-specified name.

- **measured_from**

  A required parameter that specifies lower conductor layer_name.

- **measured_to**

  A required parameter that specifies upper conductor layer_name.

- **area**

  A required parameter if min_width is not specified. Specifies the area of via in distance_units squared. This parameter cannot be specified with min_width parameter.

- **min_width**

  A required parameter that specifies the minimum allowed metal width if area parameter is not specified. This parameter cannot be specified with area parameter. The shape is assumed to be square, with dimensions of min_width * min_width.

- **min_spacing**

  A required parameter that specifies the minimum allowed metal spacing.

- **resistance**

  A required parameter that specifies the resistance per via or contact. The value is specified in Ohms.

- **enclosure_down**

  A required parameter that specifies enclosure with respect to a lower conductor. This value is typically 0, and defaults to 0 if not specified.

- **enclosure_up**

  A required parameter that specifies enclosure with respect to an upper conductor. This value is typically 0, and defaults to 0 if not specified.

- max_area

  An optional parameter that specifies the area threshold for large area via connections.

- max_length

  An optional parameter that specifies the maximum distance between distributed vias.

- min_length

  An optional parameter that specifies the length of the shape. This parameter may be used together with min_width parameter, but cannot be used with area parameter. The shape is assumed to be rectangular, with dimensions of min_width * min_length.

- tc1

  An optional parameter that specifies the resistance temperature1 coefficient.

- tc2

  An optional parameter that specifies the resistance temperature2 coefficient.

- gate_to_via_spacing_min

  An optional parameter that specifies the minimum allowed gate to via spacing.

- extra_width

  An optional conducting layer parameter that specifies the extra-width of side walls for a conductor with a trapezoid shape.

- extrapolation

  An optional parameter that specifies whether or not to extrapolate the variable via_resistance table parameters for the layer.

- ignore_caps

  An optional conducting layer parameter that specifies whether or not to ignore all capacitance for this layer. The default is no, if this parameter is not specified.

- layer_bias

  An optional conducting layer parameter that specifies the bias of actual width and length relative to drawn width and length of the via.

- ronly_layers

  An optional conducting layer parameter that specifies a space delimited list of r-only layer names enclosed in braces ({ }).

- parallel_to_gate

  An optional parameter, set to either yes or no, that specifies how via width and length are measured. The default is no, if this parameter is not specified.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

- trim

  An optional parameter that controls how resistance is calculated for the via layer. The default is yes, if this parameter is not specified.

- via_r_select_up_down

  An optional parameter that controls whether to use the smaller or larger resistance values from the upper and lower via layer enclosure. At least two via_resistance2 tables, where one has layers parameter set to up and the second has layers parameter set to down, and where both tables have the same via size, must be defined for the via layer when this keyword is specified. The default is min.

- interpolation

  An optional parameter that controls how the area-based tables interpolate resistance values for fixed or variable sized vias. The default is gperarea.

Examples

Example 1

This example specifies my_via with a resistance of 37 ohms between layers metal1 and metal7:

```
via = my_via {
   measured_from = metal1
   measured_to = metal7
   enclosure_down = 0.0000
   enclosure_up = 0.0000
   area = 0.75
   min_spacing = 0.040
   resistance = 37
}
```

Example 2

This example specifies via1 with a resistance of 20 ohms and via2 with a resistance of 30 ohms between layers metal1 and metal3:

```
via = via1 {
   measured_from = metal1
   measured_to = metal3
   enclosure_down = 0.0000
   enclosure_up = 0.0000
   min_width = 0.035
   min_spacing = 0.035
   resistance = 20
}

via = via2 {
   measured_from = metal1
   measured_to = metal3
   enclosure_down = 0.0000
   enclosure_up = 0.0000
   min_width = 0.035
   min_spacing = 0.035
   resistance = 30
}
```

Example 3

To define a via connection to a base layer with modeling properties max_area and max_length:

```
base = base1_r {
   zbottom = -1
   thickness = 1
}
```

```
via = b_via {
   measured_from = base1_r
   measured_to = metal1
   enclosure_down = 0.0000
   enclosure_up = 0.0000
   min_width = 0.030
   min_spacing = 0.030
   resistance = 50
   max_length = 0
   max_area = 0.004
   tc1 = 0.0000
}
```

This type of specification is useful in a PERC flow to probe bulk layers.

via_r_extrapolation

Type: Global Parameters

Optional global parameter that specifies whether or not to extrapolate the variable via_resistance table boundary values for all via and contact layers in the MIPT file. This parameter can only be specified once.

Syntax

via_r_extrapolation = yes | no

Parameters

- yes

  Specifies to extrapolate the variable via_resistance table boundary values for all via and contact layers in the MIPT file. This is the default behavior if this parameter is not specified.

- no

  Specifies to limit the upper and lower via_resistance table boundary values for all via and contact layers in the MIPT file. The variable boundaries are set by the overlap_width and overlap_length table syntax. Via dimensions are fixed.

Examples

Specify the following global setting to turn extrapolation of R table values off for all via and contact layers in your MIPT file:

```
via_r_extrapolation = no
```

via_r_select_up_down

Parameter for layer(s): via

Optional layer parameter that controls whether to use the smaller or larger resistance values from the upper and lower via layer enclosure.

Syntax

via_r_select_up_down = min | max

Parameters

- min

  Specifies to select the smaller resistance values from the up and down via_resistance2 table. This is the default.

- max

  Specifies to select the larger resistance values from the up and down via_resistance2 table.

Description

An optional parameter used to control which set of resistance values are used for vias during calibration. This parameter can only be used when via_resistance2 tables are specified for the via layer. If this parameter is specified in a via layer and no up/down via_resistance2 table is specified for the layer, calibration stops with an error.

Examples

The via layer must contain at least two via_resistance2 tables, one for layers = up and one for layers = down, when via_r_select_up_down is specified:

```
via = V1 {
   ...
   via_r_select_up_down = max
   table = resistance_table1 {
      property = via_resistance2
      table_type = R
      layers = up
      enclosure = min
      Dim_type = drawn
      value_type = absolute
      via_size = { 0.04 0.04 }
      overlap_width = { 0.060 0.061 }
      overlap_length = { 0.060 0.061 }
      value = { 10 10 10 10 }
   }
   table = resistance_table2 {
      property = via_resistance2
      table_type = R
      layers = down
      enclosure = min
      Dim_type = drawn
      value_type = absolute
      Via_size = { 0.04 0.08 }
      overlap_width = { 0.060 0.061 }
      overlap_length = { 0.060 0.061 }
      value = { 1 1  1 1 }
   }
...
}
```

via_size

table structure parameter

An optional parameter whose values specify the x,y dimensions of a square via. Can only be used in the two-dimensional via_resistance2 property table.

Syntax

via_size = { *x y* }

Parameters

- *x y*

  A pair of values enclosed in required braces ({ }) that specify the dimensions of a square via in microns.

Examples

```
table = table_up {
   property = via_resistance2
   table_type = R
   layers = up
   enclosure = min
   dim_type = drawn
   value_type = absolute
   via_size = { 0.022 0.022 }
   overlap_width = { 0.02175 }
   overlap_length = { 0.02175 }
   value = { 30.2 }
}
table = table_down {
   property = via_resistance2
   table_type = R
   layers = down
   enclosure = min
   dim_type = drawn
   value_type = absolute
   via_size = { 0.022 0.022 }
   overlap_width = { 0.02175 }
   overlap_length = { 0.02175 }
   value = { 40 }}
```

virtual

Parameter for layer(s): ground

An optional ground layer parameter that controls whether or not the ground layer is a virtual layer. Use this parameter when you want to specify a ground layer that is not in your design.

Syntax

virtual = yes | no

Parameters

- yes

  Specifies to treat the ground layer as virtual.

- no

  Specifies that the ground layer is not virtual. This is the default.

Description

Use this parameter when you want to specify a ground layer that does not exist in your design. When set to yes, the mapping and connect statements needed by extraction for the ground layer are added to the calibrated rule files by xCalibrate.

Examples

```
ground = virtual_device_gnd {
   zbottom = -0.01
   thickness = 0.1
   resistivity = 11
   virtual = yes
}
```

well

Type: Well layer

Part of the substrate layer definition syntax, this optional layer definition keyword is used to define the well layer.

Syntax

well = ***layer_name* {**
zbottom = ***bottom_z-coordinate***
thickness = ***z-direction_metal_thickness***
resistivity = ***resistivity_value***
eps = ***value***
bulk_min_width = *value*
bulk_resistance = *value*
hidden = <u>false</u> | true
}

Parameters

- **zbottom**

  A required bottom z-coordinate of the well layer.

- **thickness**

  A required layer parameter that specifies the thickness of the well layer in the z-direction.

- **resistivity**

  A required parameter that specifies the well layer resistance as rho, typically greater than or equal to 0.

- **eps**

  A required floating point value that specifies the relative permittivity (dielectric constant). The value is typically greater than or equal to 0.

- bulk_min_width

  An optional layer parameter that specifies the minimum width of the bulk layer in microns.

- bulk_resistance

  An optional layer parameter that specifies the sheet resistance of the bulk layer in ohms.

- hidden

  An optional parameter, set to either true or false, that specifies to encrypt the layer in the calibrated rule files. The default is false, if this parameter is not specified.

Description

The well keyword is followed by an equal sign, a unique user-specified layer name, and a list of parameters enclosed in required braces. Each parameter must appear on its own line. Well layers require zbottom, resistivity, eps, and thickness parameters. This layer type can be specified once in the MIPT file. Well layers may have tables and they may be coplanar with other well and substrate layers. The zbottom and thickness parameters are required to establish the well. For more information on substrate layer structures, see Substrate Layer Structures.

Examples

```
well = nwell {
   zbottom = -0.1
   thickness = 0.1
   resistivity = 11
   eps = 12
}
```

widths

Parameter for layer(s): conductor, poly, diffusion, li

An optional conducting layer parameter that specifies a list of width values used to override modeled widths for the layer. These floating point values must be enclosed in braces ({ }).

Syntax

widths = ‘{’ w_1 w_2 w_3 ... w_n ‘}’

Parameters

- ‘{’ w_1 w_2 w_3 ... w_n ‘}’

  A space delimited list of one or more floating point width values enclosed in braces ({ }).

Examples

```
conductor = M1 {
   thickness = 0.063
   hidden = {thickness}
   min_width = 0.034
   min_spacing = 0.02
   tc1 = 1.565e-03
   tc2 = -3.159e-07
   extra_width = 0.012
   swstep = 3
   widths = {0.029 0.0483 0.067667 0.145 }
   spacings = {0.009 0.0149698 0.0248995 0.041415 0.068887 0.114581
               0.190584 0.317001 0.52727 1.74}
}
```

zbottom

Parameter for layer(s): base, conductor, device_li, dielectric, diffusion, ground, li, pcaux, poly, resistor, seed, substrate, src_drn, tap, well

An optional parameter for layer definitions base, conductor, device_li, dielectric, diffusion, ground, li, pcaux, poly, resistor, seed, and src_drn that specifies the bottom z-coordinate for this layer. This parameter is required for substrate, tap, and well layer definitions. Zbottom is not valid for calibration_type RONLY.

Syntax

zbottom = *value*

Parameters

- *value*

  An absolute or relative floating point value that specifies the bottom z-coordinate. You can measure from the top of a reference layer defined by the measured_from keyword. If measured_from is not specified, zbottom is absolute (measured from z=0).

Description

By default, the bottom of the layer is placed at the top of the previously-defined layer. For most processes, you do not use zbottom. Often, it is possible to place an entire stack of layers implicitly, one on top of the other, by relying on the placement of the previous layers. Use zbottom if you cannot place layers implicitly.

If your process is under development and can potentially change, it is not recommended that you use fixed locations for the layer definitions. For example, if a layer thickness changes, then the fixed locations of all the layers above it change and you will need to recalculate all the locations. Or, if you choose to make a variation of a process that uses worst case values instead of typical, then it will be difficult to calculate the fixed locations for the variation. Use relative placement to make process modifications easier.

base — for base layers the zbottom parameter specifies the bottom z-coordinate of the base layer. It is required if and only if, ztop parameter has not be specified. If you have specified ztop you are not required to specify zbottom.

dielectric — for dielectric layers the zbottom parameter specifies the bottom z-coordinate when the dielectric layer is not defined in the specified order; that is, the dielectric layer is not placed relative to a previous layer's placement. For most processes, you do not use zbottom. If you implicitly define your layer order by listing the layers in order, bottom to top, you do not need zbottom. The zbottom parameter should not be used when defining new dielectric layers.

conductor, poly, diffusion, li — for these layer types the zbottom parameter specifies the bottom z-coordinate when the metal layer is not implicitly placed; that is, the metal layer is not placed relative to a previous layer's placement. You do not need zbottom if you implicitly define your layer order by listing the layers bottom to top in the mipt file.

substrate, well, tap — for these layer types the zbottom parameter is part of the thickness specification information. The zbottom parameter can be specified with the ztop parameter or with the thickness parameter; It cannot be specified with both ztop and thickness in the same layer definition.

Examples

Example for Base

```
zbottom = 3
```

Example for Dielectric

```
zbottom = 0.26
```

Example for Conductor, Poly, Diffusion, LI

```
zbottom = 1.01
```

ztop

Parameter for layer(s): base, conductor, device_li, dielectric, diffusion, ground, li, pcaux, poly, resistor, seed, substrate, src_drn, tap

An optional parameter for layer definitions base, conductor, device_li, dielectric, diffusion, ground, li, pcaux, poly, resistor, seed, and src_drn that specifies the bottom z-coordinate for this layer. This parameter is required for substrate and tap layer definitions. Zbottom is not valid for calibration_type RONLY.

Syntax

ztop = *value*

Parameters

- *value*

  An absolute or relative floating point value that specifies the top z-coordinate of the layer. You can measure from the top of a reference layer defined by the measured_from parameter. If measured_from parameter is not specified, ztop is absolute (measured from z=0).

Description

base — for base layers the ztop parameter specifies the top z-coordinate when the base layer is not implicitly placed; that is, the base layer is not placed relative to a previous layer's placement. For most processes, you do not use ztop if the layer order is defined from bottom to top.

dielectric — for dielectric layers the ztop parameter specifies the top z-coordinate when the dielectric layer is not implicitly placed; that is, the dielectric layer is not placed relative to a previous layer's placement. For most processes do not use ztop in dielectric layer definitions. For trench dielectrics, ztop should be set to 0. The parser verifies this condition.

conductor, poly, diffusion, li — for these layers ztop is top z coordinate for the layer. It is not placed relative to a previous layer's placement. For most processes, you do not use ztop. If you implicitly define your layer order by listing the layers bottom to top, you do not need ztop. For trenched conductors, ztop should be set to 0 and a ref_layer specified.

substrate, well, tap — for these layer types the ztop parameter is part of the thickness specification information. The ztop parameter can only be specified with the zbottom parameter; It cannot be specified with thickness.

Examples

Example for Base

```
ztop = 1.2
```

Example for Dielectric

```
ztop = 0
```

Example for Conductor, Poly, Diffusion, LI

```
ztop = 1.65
```

Chapter 7
MIPT File Utilities

xCalibrate provides utilities for translating technology files to MIPT format and verifying MIPT files (.mipt).

Interconnect Technology Format (ITF)

The interconnect technology format is a standardized format that describes the detailed modeling of interconnect parasitic effects for the technology stack. You can translate ITF files into MIPT files, or mapping and ignore files with the xCalibrate ITF Translation options.

ITF Translation

With the ITF Translation options you can perform different types of translations.

Use the ITF translation options to convert an:

- ITF file to an MIPT file (*out.mipt*).

  ```
  xcalibrate -itf2mipt2 itf_file
  ```

- ITF Mapping file to an equivalent PEX SVRF mapping and ignore file (*rules.map*).

  ```
  xcalibrate -itfmap2svrf itf_mapping_file
  ```

- ITF file with one or more ITF Mapping files to an MIPT file with embedded SVRF mapping and ignore statements (*out.mipt*).

  ```
  xcalibrate -itf2mipt2 itf_file -itfmap itf_mapping_file \
      [itf_mapping_file...]
  ```

To invoke the translator, enter the command at the shell prompt. For example:

```
xcalibrate -itf2mipt2 my.itf
```

creates an MIPT file called *out.mipt*.

If the information in the ITF file is incomplete, you can provide an auxiliary file (*itf.aux*) to fill in the missing information. By default, the translator searches the current working directory for this file and uses it if it is found. Optionally, to specify an auxiliary file, use the -aux command line option. For example:

```
xcalibrate -itf2mipt2 itf_file -aux aux_file -itfmap itf_mapping_file
```

For information on the contents of the auxiliary file, see Auxiliary File Format.

Warnings generated during conversion should be reviewed to determine if they indicate a real problem in the generated MIPT file. For example, warnings are generated during conversion for table variable values that are not in ascending order. If value order is not corrected, then the MIPT calibration stops with an error.

Table 7-1 summarizes the syntax conversion from ITF to MIPT.

Table 7-1. ITF to MIPT Syntax Conversion Table

ITF Syntax	MIPT Syntax
`air_gap_vs_spacing {` `spacings {s1 s2 s3 ... sn}` `air_gap_widths {w(s1)w(s2) ... w(sn) }` `air_gap_thicknesses {t(s1) t(s2) ... t(sn) }` `air_gap_bottom_heights {h(s1) h(s2) ... h(sn) }` `}`	`airgap = {rectangle | triangle}` `min_gap (value)` `spacing (s1 ... sn)` `width(w1 ... wn)` `thickness (t1 ... tn)` `distance_from_surface (b1 ... bn)` See airgap for details.
`area = value`	`area = value` See area for details.
`associated_conductor = conductor_name`	`ref_layer = layername` See ref_layer for details.
`background_er = value`	`background_dielectric = value` See background_dielectric for details
`bottom_thickness_vs_si_width [RESISTIVE_ONLY | CAPACITIVE_ONLY] {` `(S1,R1) (S2,R2) ... (Sn,Rn)` `}`	`table = table_name { ... }` See THICKNESS_TOP and THICKNESS_BOT Tables for details.
`CAPACITIVE_ONLY_ETCH = etch_value`	`capacitive_only_etch = value` See capacitive_only_etch for details.
`channel_er = value`	`channel_er = value` See channel_er for details.
`conductor conductor_name {` `...` `}`	`conductor = layername {` `...` `}` See conductor for details.
`conductor conductor_name {` `layer_type = diffusion` `...` `}`	`diffusion = layername {` `...` `}` See diffusion for details.
`conductor conductor_name {` `layer_type = field_poly` `...` `}`	`poly = layername {` `...` `}` See poly for details.

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

ITF Syntax	MIPT Syntax
`conductor` *`conductor_name`* `{` `layer_type = float_gate` `...` `}`	`pcaux =` *`layername`* `{` `...` `}` See pcaux for details.
`conductor` *`conductor_name`* `{` `layer_type = gate` `...` `}`	`seed =` *`layername`* `{` `...` `#default ignore parameters` `ignore_diff_intrinsic = yes` `ignore_diff_to_diff_under_poly = yes` `ignore_gate_intrinsic = yes` `ignore_gate_to_diff = yes` `...` `}` See seed for details.
`conductor` *`conductor_name`* `{` `layer_type = gate` `extensionmin = value` `...` `}`	`seed =` *`layername`* `{` `...` `extension =` *`value`* `#default ignore parameters` `ignore_diff_intrinsic = yes` `ignore_diff_to_diff_under_poly = yes` `ignore_gate_intrinsic = yes` `ignore_gate_to_diff = yes` `ignore_endcap = no` `ignore_gateext_to_diff = no` `ignore_gateext_to_diff_only = no` `...` `}` See seed and extension for details.
`conductor` *`conductor_name`* `{` `layer_type = trench_contact` `...` `}`	`device_li =` *`layername`* `{` `...` `}` See device_li for details.
`crt1 =` *`value`* `crt2 =` *`value`*	`tc1 =` *`value`* `tc2 =` *`value`* See tc1 and tc2 for details.
`crt_vs_area{` `(`*`area_1,crt1_2, crt2_1`*`)` `(`*`area_2,crt1_2, crt2_1`*`)` `...` `(`*`area_n,crt1_n, crt2_n`*`)` `}`	`table =` *`my_tc1_table`*`{ ... }` `table =` *`my_tc2_table`* `{ ... }` See TC1 and TC2 Tables for details.

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

ITF Syntax	MIPT Syntax
`crt_vs_si_width {` `(siw_1, crt1_2, crt2_1)` `(siw_2, crt1_2, crt2_1)` `...` `(siw_n, crt1_n, crt2_n)` `}`	`table = my_tc1_table { ... }` `table = my_tc2_table { ...}` See TC1 and TC2 Tables for details.
`crt_vs_si_width_2 {` `(siw_1, crt1_2, crt2_1)` `(siw_2, crt1_2, crt2_1)` `...` `(siw_n, crt1_n, crt2_n)` `}`	`table = my_tc1_table { ... }` `table = my_tc2_table { ...}` See TC1 and TC2 Tables for details.
`damage_er = value`	`damage_eps = value` See damage_eps for details.
`damage_thickness = value`	`damage_thickness = value` See damage_thickness for details.
`density_box_weighting_factor {(S1 W1) (S2 W2) (S3 W3) (S4 W4) (S5 W5)}`	density_window = $\{\{w_1\ h_1\ factor_1\ stepx_1\ stepy_1\}\ ...\ \{w_n\ h_n\ factor_n\ stepx_n\ stepy_n\}\}$ See density_window for details.
`dielectric dielectric_name {` `...` `}`	`dielectric = layername {` `...` `}` See dielectric for details.
`drop_lateral_spacing = value`	Not converted — parsed.
`extensionmin = value`	`extension = value` If the ITF conductor layer_type is gate, then the following ignore syntax is also included in the seed layer definition: `ignore_endcap = no` `ignore_gateext_to_diff = no` `ignore_gateext_to_diff_only = no` See extension and seed for details.
`ER = value`	`eps = value` See eps for details.
`er_vs_spacing {` `(s1, er1) (s2,er2) (s3,er3) ...` `}`	`table = table_name { ... }` See SIDEWALL_K Table for details.
`er_vs_si_spacing {` `(s1, er1) (s2,er2) (s3,er3) ...` `}`	`table = table_name {... }` See SIDEWALL_K Table for details.

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

<table>
<tr><th>ITF Syntax</th><th>MIPT Syntax</th></tr>
<tr><td><pre>etch = value</pre></td><td><pre>layer_bias = value</pre>
Note: the converted value for MIPT is 2 times the ITF value. See layer_bias for details.</td></tr>
<tr><td><pre>etch_vs_width_and_spacing
   [etch_from_top |
     resistive_only |
     capacitive_only]
   [parallel_to_reference |
     perpendicular_to_reference]
{
   spacings {vector(n)}
   widths {vector(m)}
   values {matrix(m,n)}
}</pre></td><td><pre>table = table_name { ... }</pre>
See WIDTH Table for details.</td></tr>
<tr><td>If the conductor layer contains:
<pre>etch_vs_width_and_spacing   { }
etch_vs_width_and_spacing   { }</pre></td><td><pre>table = metal_multi_bias_xxx {
   property = multi_bias
   table_type = RC
   dim_type = drawn
   value_type = delta
   width$_1$ = { ... }
   spacing$_1$ = { ... }
   value$_1$ = { ... }
   width$_2$ = { ... }
   spacing$_2$ = { ... }
   value$_2$ = { ... }
}</pre>
See MULTI_BIAS Table for details.</td></tr>
<tr><td>If the conductor layer contains:
<pre>etch_vs_width_and_spacing   { }
etch_vs_width_and_spacing   { }
etch_vs_width_and_spacing   { }</pre></td><td><pre>table = metal_multi_bias_xxx {
   property = multi_bias
   table_type = RC
   dim_type = drawn
   value_type = delta
   width$_1$ = { ... }
   spacing$_1$ = { ... }
   value$_1$ = { ... }
   width$_2$ = { ... }
   spacing$_2$ = { ... }
   value$_2$ = { ... }
   width$_3$ = { ... }
   spacing$_3$ = { ... }
   value$_3$ = { ... }
}</pre>
See MULTI_BIAS Table for details.</td></tr>
</table>

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

ITF Syntax	MIPT Syntax
If the conductor layer contains: etch_vs_width_and_spacing { } etch_vs_width_and_spacing { } etch_vs_width_and_spacing { } etch_vs_width_and_spacing { }	table = metal_multi_bias_*xxx* { property = multi_bias table_type = RC dim_type = drawn value_type = delta width$_1$ = { ... } spacing$_1$ = { ... } value$_1$ = { ... } width$_2$ = { ... } spacing$_2$ = { ... } value$_2$ = { ... } width$_3$ = { ... } spacing$_3$ = { ... } value$_3$ = { ... } width$_4$ = { ... } spacing$_4$ = { ... } value$_4$ = { ... } } See MULTI_BIAS Table for details.
If the conductor layer contains: etch_vs_width_and_spacing { } etch_vs_width_and_spacing CAPACITIVE_ONLY { } etch_vs_width_and_spacing RESISTIVE_ONLY { }	table = metal_multi_bias_*xxx* { property = multi_bias table_type = C dim_type = drawn value_type = delta width$_1$ = { ... } spacing$_1$ = { ... } value$_1$ = { ... } width$_2$ = { ... } spacing$_2$ = { ... } value$_2$ = { ... } } table = metal_multi_bias_*xxx* { property = multi_bias table_type = R dim_type = drawn value_type = delta width$_1$ = { ... } spacing$_1$ = { ... } value$_1$ = { ... } width$_2$ = { ... } spacing$_2$ = { ... } value$_2$ = { ... } } See MULTI_BIAS Table for details.

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

ITF Syntax	MIPT Syntax
If the conductor layer contains: etch_vs_width_and_spacing { } etch_vs_width_and_spacing { } etch_vs_width_and_spacing CAPACITIVE_ONLY { } etch_vs_width_and_spacing RESISTIVE_ONLY { }	table = metal_multi_bias_*xxx* { property = multi_bias table_type = C dim_type = drawn value_type = delta width$_1$ = { ... } spacing$_1$ = { ... } value$_1$ = { ... } width$_2$ = { ... } spacing$_2$ = { ... } value$_2$ = { ... } width$_3$ = { ... } spacing$_3$ = { ... } value$_3$ = { ... } } table = metal_multi_bias_*xxx* { property = multi_bias table_type = R dim_type = drawn value_type = delta width$_1$ = { ... } spacing$_1$ = { ... } value$_1$ = { ... } width$_2$ = { ... } spacing$_2$ = { ... } value$_2$ = { ... } width$_3$ = { ... } spacing$_3$ = { ... } value$_3$ = { ... } } See MULTI_BIAS Table for details.

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

ITF Syntax	MIPT Syntax
If the conductor layer contains: etch_vs_width_and_spacing { } etch_vs_width_and_spacing { } etch_vs_width_and_spacing { } etch_vs_width_and_spacing CAPACITIVE_ONLY { } etch_vs_width_and_spacing RESISTIVE_ONLY { }	table = metal_multi_bias_*xxx* { property = multi_bias table_type = C dim_type = drawn value_type = delta width$_1$ = { ... } spacing$_1$ = { ... } value$_1$ = { ... } width$_2$ = { ... } spacing$_2$ = { ... } value$_2$ = { ... } width$_3$ = { ... } spacing$_3$ = { ... } value$_3$ = { ... } width$_4$ = { ... } spacing$_4$ = { ... } value$_4$ = { ... } } table = metal_multi_bias_*xxx* { property = multi_bias table_type = R dim_type = drawn value_type = delta width$_1$ = { ... } spacing$_1$ = { ... } value$_1$ = { ... } width$_2$ = { ... } spacing$_2$ = { ... } value$_2$ = { ... } width$_3$ = { ... } spacing$_3$ = { ... } value$_3$ = { ... } width$_4$ = { ... } spacing$_4$ = { ... } value$_4$ = { ... } } See MULTI_BIAS Table for details.
fill_ratio = *value* fill_width = *value* fill_spacing = *value* fill_type = *floating* | *ground*	metal_fill = {*fill_ratio* *fill_spacing* *fill_width* floating | ground} See metal_fill for details.
fin_length = *value*	fin_length = *value* See fin_length for details.
fin_spacing = *value*	fin_spacing = *value* See fin_spacing for details.

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

ITF Syntax	MIPT Syntax
fin_thickness = *value*	fin_thickness = *value* See fin_thickness for details.
fin_width = *value*	fin_width = *value* See fin_width for details.
floating_gate_to_diffusion_contact_cap { gate_polygon_length {*val1...valn*} gate_to_contact_spacing {*value*} capacitance_values {*val1...valn*} }	Not converted — parsed.
from = *conductor_name* to = *conductor_name*	measured_from = *layername* measured_to = *layername* See measured_from and measured_to for details.
gate_fringe { ... }	table = *my_gate_fringe_table* {...} See GATE_FRINGE Table for details.
gate_oxide_side_t = *value*	gate_oxide_side_t = *value* See gate_poly_side_t for details.
gate_oxide_top_t = *value*	gate_oxide_top_t = *value* See gate_oxide_top_t for details.
gate_poly_extension = *value*	gate_poly_side_t = *value* See gate_poly_side_t for details.
gate_poly_top_t = *value*	gate_poly_top_t = *value* See gate_poly_top_t for details.
gate_to_diffusion_cap { contact_to_contact_spacings {*c1 c2 c3* ...} gate_to_contact_spacings {*s1 s2 s3* ...} caps_per_micron {*v*(*c1*,*s1*) *v*(*c2*,*s1*) ... *v*(*c1*,*s2*) *v*(*c2*,*s2*)...} }	table = *table_name* { ... } See GATE_FRINGE Table for details.
gate_to_diffusion_cap { gate_widths { *w1 w2* ....} gate_to_contact_spacings {*s1 s2 s3* ...} caps_per_micron {*v*(*c1*,*s1*) *v*(*c2*,*s1*) ... *v*(*c1*,*s2*) *v*(*c2*,*s2*)...} }	table = *table_name* { ... } See GATE_FRINGE_SCALE Table for details.

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

ITF Syntax	MIPT Syntax
`gate_to_diffusion_channel_cap {` `...` `}`	`table = ` *`table_name`* ` {` `  property = li_gate_fringe` `  table_type = C` `  dim_type = drawn` `  value_type = absolute` `  li`$_1$` = {...}` `  li`$_2$` = {...}` `  value = {...}` `}` See LI_GATE_FRINGE Table for details.
`global_temperature = ` *`value`*	`temperature = ` *`value`* See temperature for details
`half_node_scale_factor = ` *`value`*	`half_node_scale_factor = ` *`value`* See half_node_scale_factor for details
`ignore_fieldpoly_to_diffusion_bottom_coupling = yes | no`	Not converted — parsed.
`ild_vs_width_and_spacing {` `...` `}`	`table = ` *`table_name`* ` {` `  property = thickness_bot` `  table_type = RC` `  dim_type = drawn` `  value_type = delta` `  width = { ... }` `  spacing = { ... }` `  value = { ... }` `}` See THICKNESS_TOP and THICKNESS_BOT Tables for details.
`inductive_ground_depth = ` *`value`*	Not converted — parsed.
`is_conformal`	`diel_type = conformal | trench` See diel_type for details.
`linked_to =`	Not converted — parsed.
`measured_from = ` *`layername`*	`measured_from = ` *`layername`* See measured_from for details.
`multigate ` *`multigate_name`* ` {` `...` `}`	`multigate = ` *`layername`* ` {` `...` `}` See multigate for details.
`parallel_to_reference`	`bias_type = ` *`preferred`* See bias_type for details.

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

ITF Syntax	MIPT Syntax
`perpendicular_to_reference`	`bias_type = nonpreferred` See bias_type for details.
`process_corner`	Not converted — parsed.
`process_type`	Not converted — parsed.
`process_version`	Not converted — parsed.
`process_node`	Not converted — parsed.
`process_foundry`	`process_foundry = name` See process_foundry for details.
`raised_diffusion_etch = value`	`rsd_spacing = value` See rsd_spacing for details.
`raised_diffusion_thickness = value`	`rsd_thickness = value` See rsd_thickness for details.
`raised_diffusion_to_gate_smin = value`	`top_enclosure = value` See top_enclosure for details.
`raised_diffusion_growth = value`	`rsd_enclosure = value` See rsd_enclosure for details.
`reference_direction = vertical | horizontal`	Not converted — parsed.
`RESISTIVE_ONLY_ETCH = etch_value`	`resistive_only_etch = value` See resistive_only_etch for details.
`rho = value`	`resistivity = value` See resistivity for details.
`rho_vs_si_width_and_thickness {` `spacings {vector(n)}` `widths {vector(m)}` `values {matrix(m,n)}` `}`	`resistivity = v(t1,w1)` `table = table_name { ... }` See RHO_T Table for details.
`rpsq = value`	`resistivity = value` See resistivity for details.

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

ITF Syntax	MIPT Syntax
`rpsq_vs_si_width {` `(`*`SIW1,R1`*`) (`*`SIW2,R2`*`) ... (`*`SIWn,Rn`*`)}`	`resistivity = R1` `table = `*`table_name`*` {` `    ...` `    property = rsh` `    ...` `}` See RSH Table for details.
`rpsq_vs_width_and_spacing {` `spacings {`*`s1 s2 s3`*` ...}` `widths {`*`w1 w2 w3`*` ...}` `values {`*`v(s1,w1) v(s2,w1)`*` ... `*`v(s1,w2)`* *`v(s2,w2)`*` ...}` `}`	By default converted to: `resistivity = R1` `table = `*`table_name`*` {` `    ...` `    property = rho` `    ...` `}` See RHO Table and resistivity for details. With the -rsh command line option converted to: `resistivity = R1` `table = `*`table_name`*` {` `    ...` `    property = rsh` `    rsh_type = actual` `    ...` `}` See RSH Table and resistivity for details.
`rpv = `*`value`*	`resistance = `*`value`* See resistance for details.
`rpv_vs_area {` `(`*`area1,rpv1`*`) (`*`area2,rpv2`*`) (`*`area3,rpv3`*`)` `...}`	`table = `*`table_name`*` {` `  property = resistance` `  table_type = R` `  dim_type = drawn` `  value_type = absolute` `  width = {`w_1 w_2 `...` w_n`}` `}` See RESISTANCE Table for details.

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

ITF Syntax	MIPT Syntax
side_damage_thickness_vs_width_and_spacing { spacings {*s1 s2 s3 ...*} widths {*w1 w2 w3 ...*} values {*v(s1,w1) v(s2,w1) ... v(s1,w2) v(s2,w2) ...*} }	damage_thickness = *value* table = *table_name* { ... property = damage_thickness ... } See DAMAGE_THICKNESS Table for details.
side_tangent = *value*	extra_width = *value* See extra_width for details.
smin = *value*	min_spacing = *value* See min_spacing for details.
sw_t = *value*	swthk = *value* See swthk for details.
technology = *techname*	process = *process_name* See process for details.
thickness = *value*	thickness = *value* See thickness for details.
thickness_vs_density [RESISTIVE_ONLY | CAPACITIVE_ONLY] { *(D1 R1) (D2 R2) (D3 R3) (D4 R4)...*}	table = *table_name* { ... } See THICKNESS_TOP and THICKNESS_BOT Tables for details.
thickness_vs_width_and_spacing [RESISTIVE_ONLY | CAPACITIVE_ONLY] { spacings {*vector(n)*} widths {*vector(m)*} values {*matrix(m,n)*} }	table = *table_name* { ... } See THICKNESS Table for details.
trench_contact_extension_length = *value*	trench_contact_extension_length = *value* See trench_contact_extension_length for details.
tw_t = *value*	topthk = *value* See topthk for details.
use_si_density = yes | no	Not converted — parsed.

Table 7-1. ITF to MIPT Syntax Conversion Table (cont.)

ITF Syntax	MIPT Syntax
`width_and_spacing_from_opc {` `...` `}`	`table = table_name { ... }` See WIDTH Table for details.
`wmin = value`	`min_width = value` See min_width for details.
`vertical_resistance_vs_si_width_and_len` `gth {` `lengths {l1 l2 ... lm}` `widths {w1 w2 ... wn}` `values {v(w1,l1) v(w2,l1) ... v(wn,l1)` `v(w1,l2) v(w2,l2)` `...v(wn,l2)` `...` `v(w1,lm) v(w2,lm) ...` `v(wn,lm)}` `}`	See VERTICAL_RSH Table for details.
`via via_name {` `...` `}`	`via = via_name {` `...` `}` See via for details.

Auxiliary File Format

Use the auxiliary file (*itf.aux*) to specify global parameters, override specific syntax, or include additional layer information in the MIPT file produced with the ITF translation options.

With the exception of the ignore syntax, the auxiliary file can only contain MIPT syntax. Create and save this file in the directory you intend to run the translation from. If the *itf.aux* file exists in the current working directory, the translator automatically reads and parses the content; no command line option is needed. For information on how to run the translation, see ITF Translation.

Global Parameters are specified at the beginning of the *itf.aux* file, outside of any layer declarations. For example:

```
plate_loading = yes
li_device_model = single_li_layer
```

The layer definitions in the auxiliary file must be specified using MIPT layer syntax. For example:

```
seed = pc_par_PMOS {
   ignore_gate_to_diff = no
}
```

Use the following auxiliary file syntax to exclude specific layers from translation:

ignore *layername* [*layername*...]

where ignore is a required keyword followed by one or more layer names. For example:

```
ignore ngate
```

Translating an ITF File to MIPT

An ITF file can be converted to an MIPT file without an auxiliary file.

Prerequisites

- An xCalibrate license. See "Licensing: Parasitic Extraction Products" in the *Calibre Administrator's Guide* for details.
- A syntactically valid ITF file called *my.itf*.

Procedure

1. To convert *my.itf* file without using the auxiliary file, make sure the current working directory does not contain an auxiliary file (**.aux*).
2. Invoke the translator with the following command line:

   ```
   xcalibrate -itf2mipt2 my.itf
   ```

Results

A successful conversion run completes with no errors or warnings:

```
...
parsing succeeded
------------------
Translation succeeded
Translation Done...
```

The conversion produces an *mipt.out* file in the current working directory.

Translating an ITF File to MIPT With an Auxiliary File

To specify global parameters, override specific syntax, or include additional layer information use an auxiliary file when converting an ITF file to MIPT.

Prerequisites

- An xCalibrate license. See "Licensing: Parasitic Extraction Products" in the *Calibre Administrator's Guide* for details.

- A syntactically valid ITF file.
- An auxiliary file.

Procedure

1. For this example, the current working directory contains the *itf.aux* file. The file contains the following global MIPT parameter specification:

   ```
   li_device_model = dual_li_layers_model1
   ```

   The heading of the *my.itf* file is:

   ```
   $ TECHNOLOGY: typical
   $ SPICE model: A SPICE MODEL
   TECHNOLOGY = typical
   USE_SI_DENSITY = YES
   GLOBAL_TEMPERATURE = 25.0
   ...
   ```

2. To convert *my.itf* file and include the auxiliary file information in the generated *mipt.out* file, use either of the following command lines:

   ```
   xcalibrate -itf2mipt2 my.itf
   ```

   or

   ```
   xcalibrate -itf2mipt2 my.itf -aux itf.aux
   ```

Results

This conversion produces an *mipt.out* file with the auxiliary file annotations.

In this example, the heading information in the *out.mipt* file has been annotated with the auxiliary file contents; for example:

```
mipt_source = itf2mipt2
mipt_version = 2.0
calibration_type = all
process = typical
date = "Wed Apr 1 14 14:19:47 2015"
author = SiemensEDA
cap_unit = ff
version = "v2015.2_2.0003"
temperature = 25
plate_loading = true
li_device_model = dual_li_layers_model1
corner = typical
...
```

Note that the li_device_model line specified in the auxiliary file was included in the *out.mipt* file.

iRCX File Format

The iRCX format is a standardized technology file format introduced by the foundry, which integrates all necessary process and reliability information for each technology into a single file. You can translate iRCX files into MIPT files with the xCalibrate IRCX Translation option.

IRCX Translation

Use the xCalibrate IRCX translation option to convert an iRCX file (*filename*.ircx) to an MIPT file.

```
xcalibrate -ircx2mipt2 ircx_filename.ircx
```

To invoke the translator, enter the command at the shell prompt. For example:

```
xcalibrate -ircx2mipt2 my.ircx
```

creates an MIPT file called *my.mipt* that can then be used to generate calibrated rule files.

For more information on the -ircx2mipt2 command line option see "xcalibrate -ircx2mipt2" on page 501.

Interface Mapfile

The interface mapfile is used to create an interface MIPT file for a pair of 3DIC components. 3DIC components can be die, fanout package, or interposer.

The -interface command line option requires the *mapfile* name argument. Only one *mapfile* is allowed per component pair. If your design contains multiple dies, then a separate *mapfile* must be created for each component pair. This also means that calibration and extraction can be performed on only one component pair.

The -interface *mapfile* command line option can be used alone to generate an interface MIPT file or together with the -exec command line option to directly generate the calibrated rule files for Calibre xACT.

The interface *mapfile* specifies:

- The bottom (BOT) and top (TOP) component MIPT files. Each MIPT file contains the layer description for a component. An IRCX file can also be specified in place of the MIPT file.
- The component orientation that describes how the components are stacked.
- The component, either BOT or TOP, that contains the ground reference.
- The interface layers that are to be used in extraction.

The interface *mapfile* format is:

```
BOT = component_mipt_file
TOP = component_mipt_file
{BOT, orientation}
{TOP, orientation}
CALIBRATE ground_reference \
{(component, interface_metal_layer ) :(component, interface_metal_layer)}
```

where:

- ***component_mipt_file*** — Specifies an MIPT file or IRCX file that describes the die, fanout package, or interposer.
- ***orientation*** — Specifies the orientation of the TOP and BOT components. The ***orientation*** must be one of the following:
  - BT — bottom to top
  - TB — top to bottom
- ***ground_reference*** — Specifies the component that contains the ground reference. The ***ground_reference*** immediately follows the required CALIBRATE keyword and must be either TOP or BOT.
- (***component***,***interface_metal_layer***):(***component***,***interface_metal_layer***) — Specifies which metal layer in the component closest to the interface (point of connection) to include in the generated interface MIPT file. The top side (T) of the die is the highest metal layer. A ***component*** - ***interface_metal_layer*** comma-separated pair must be specified for each component (TOP and BOT), enclosed in parentheses, and separated by a colon.

The ground reference is determined by how the components, BOT and TOP, are stacked. Components can be stacked face-to-face or face-to-back. In a face-to-face die connection, the ground reference can be defined from either component (Figure 7-1 and Figure 7-2). For a face-to-back connection, the component (die) that interfaces with the front side in the face-to-back connection should provide the ground reference. For example, in Figure 7-3, the bottom die (BOT) has the ground reference because the top of that die is connected to the back side of the top die (TOP). In Figure 7-4, the top die (TOP) has the ground reference because the top of that die is connected to the back side of the bottom die (BOT).

Figure 7-1. Face-to-Face with Bottom Component Ground Reference

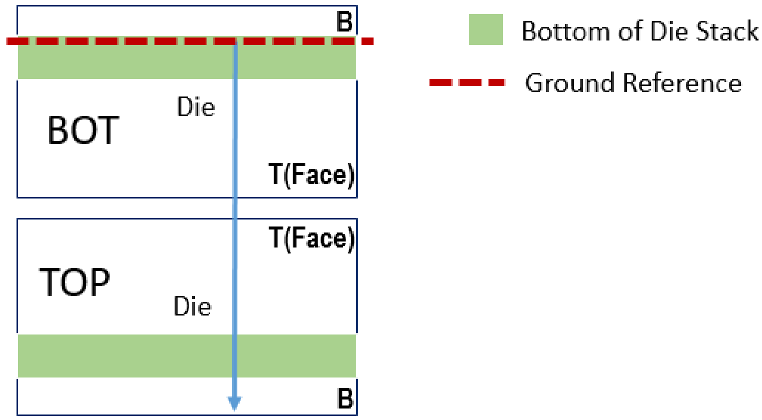


Figure 7-2. Face-to-Face with Top Component Ground Reference

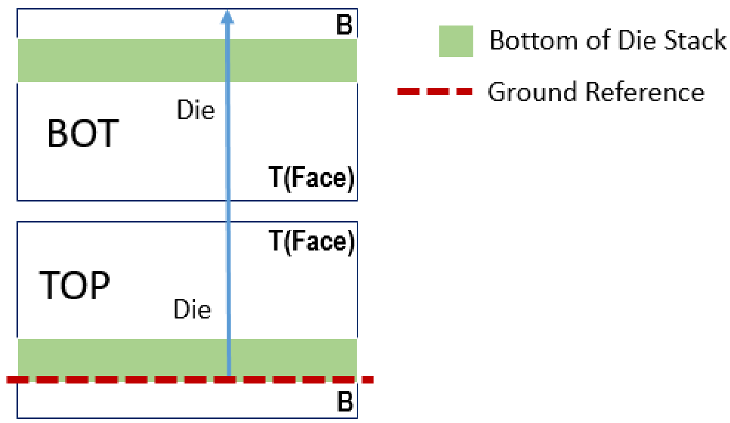


Figure 7-3. Ground Reference for Face-to-Back Bottom Front-side Interface

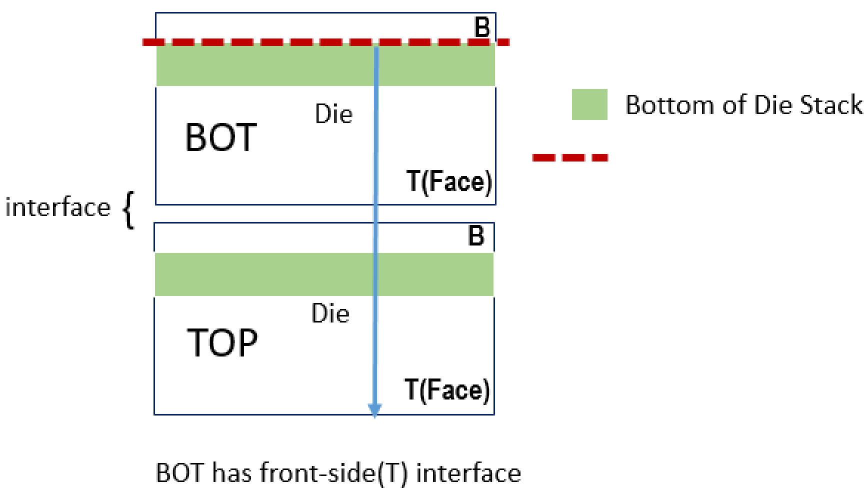
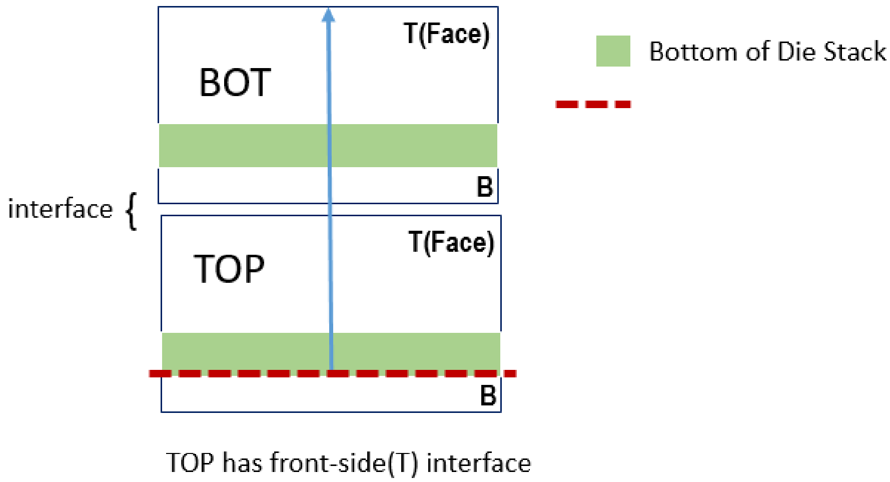
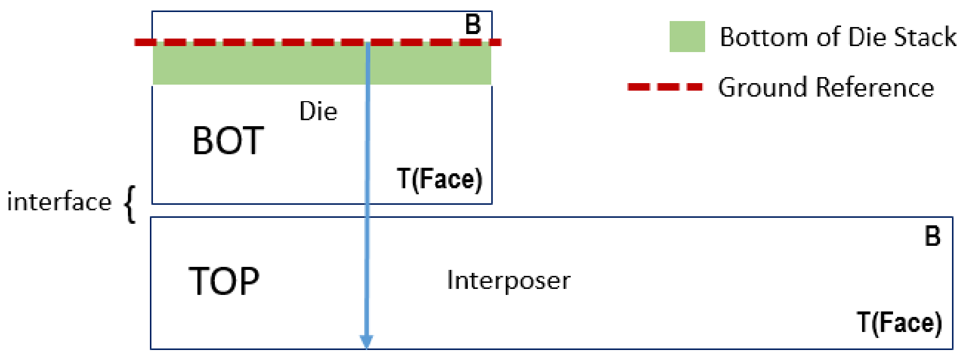


Figure 7-4. Ground Reference for Face-to-Back Top Front-side Interface



If the interfaced components are a die and interposer, then the ground reference should always be defined for the die (Figure 7-5).

Figure 7-5. Ground Reference for Face-to-Back Die-Interposer Interface



Once the component that contains the ground reference has been decided, the component orientation is determined by looking from the die towards the other component that it interfaces with. The component orientation is either BT or TB.

For example, for the face-to-face interface shown in Figure 7-1 following the blue arrow from the ground reference in the TOP die to the BOT die, the orientation syntax is:

```
{TOP,BT}
{BOT,TB}
```

The letters B and T correspond to the letters shown in the figure.

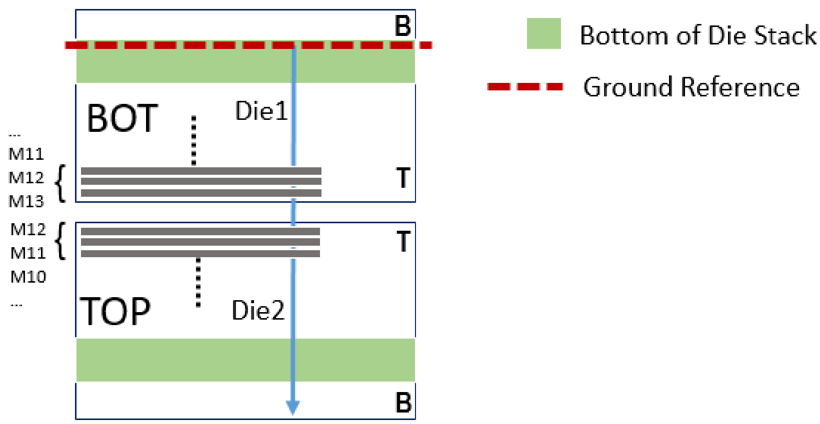
Following the blue arrow in Figure 7-3 from the ground reference in the BOT die to the TOP die, the orientation syntax is:

```
{TOP,BT}
{BOT,BT}
```

The ***interface_metal_layer*** is the metal layer from the die that is included in the interface MIPT file; this is typically the top-most metal layer in the die stack and the closest layer to the interface. You can specify additional layers for more accurate extraction. When an additional layer is specified, all layers from the specified layer to the topmost layer are included in the generated interface MIPT file.

For example, in Figure 7-6, the BOT component (Die1) has 13 metal layers in its stack and the TOP component (Die2) has 12 metal layer in its stack. For Die1 the top-most metal layer in the die stack and the closest layer to the interface between the two components is M13. For Die2 the top-most metal layer in the stack and the closest layer to the interface is M12.

Figure 7-6. Interface Metal Layer Example



The following mapfile specifies to include layers M10, M11, and M12 from Die2 and M13 from Die1 in the interface MIPT file:

```
TOP = Die2.mipt
BOT = Die1.mipt
{TOP, TB}
{BOT, BT}
CALIBRATE BOT {(TOP,M10):(BOT,M13)}
```

The following mapfile specifies to include layer M12 from Die2 and M13 from Die1 in the interface MIPT file:

```
TOP = Die2.mipt
BOT = Die1.mipt
{TOP, TB}
{BOT, BT}
CALIBRATE BOT {(TOP,M12):(BOT,M13)}
```

In the following mapfile, the ***component_mipt_file*** for the TOP is in IRCX format and BOT is in MIPT format. This syntax specifies to include layer M12 from Die2, and layers M12 and M13 from Die1 in the interface MIPT file:

```
TOP = Die2.ircx
BOT = Die1.mipt
{TOP, TB}
{BOT, BT}
CALIBRATE BOT {(TOP,M12):(BOT,M12)}
```

Verifying MIPT Structures

Applying the -check option parses an MIPT file and checks for logical inconsistencies. It is run by default when loading an MIPT file into the stack viewer.

Performing the Verification

Prior to running calibration, check your MIPT file for syntax errors with the syntax checker and view the stack structure with the stack viewer.

This procedure describes how to invoke the MIPT file checker and analyze the results.

Prerequisites

You must have a syntactically valid MIPT file and an xCalibrate license. See "Licensing: Parasitic Extraction Products" in the *Calibre Administrator's Guide* for details.

Procedure

1. You can verify the syntax by any of these methods:
   - Direct invocation:

     ```
     xcalibrate -check filename.mipt
     ```

   - Invoking the stack viewer:

     ```
     xcalibrate -view filename.mipt
     ```

2. The logic checker parses the file and any errors are written to the terminal window.

Results

A possible syntax error:

```
ERROR: XCAL_2_002: MIPT Parser Error: Technology Parser Error: bad
mipt_version value "3": must be either "1.0" or "2.0" in file name
```

The parser exits on the first error it encounters. If this occurs, the remainder of the file was not checked.

Parsing messages start with the following:

```
Note: Successfully parsed filename
MIPT checker: Problem with …
ERROR: description
…
```

If no errors are detected, the last line of the output is:

```
Note: Successfully parsed filename
```

Stack Viewer

The stack viewer shows a graphical representation of a MIPT file. You can customize the representation to display only certain measurements, as well as defined devices. It only reads MIPT format files.

Stack Viewer Invocation

To invoke the stack viewer, enter the following at a shell prompt:

```
xcalibrate -view filename.mipt
```

The filename is required; you can load a different MIPT file later using the Load File button (see Figure 7-9 on page 483).

For details on the command line syntax for the stack viewer, see “xcalibrate -view” on page 496.

Stack Viewer GUI

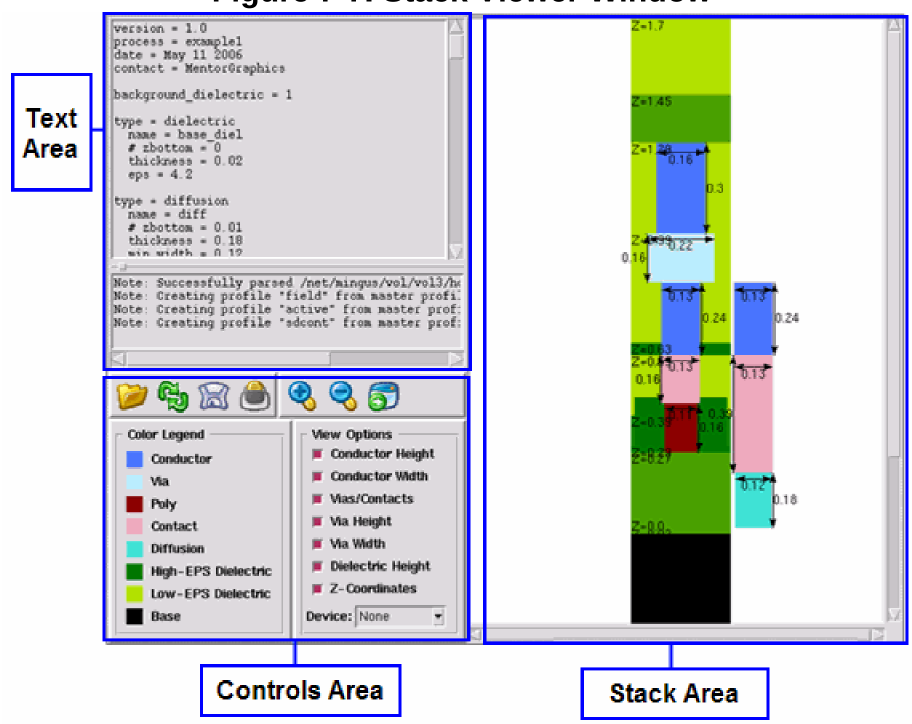
The stack viewer graphical user interface (GUI) has several features to help you visualize your technology stack.

The stack viewer window is divided into three main areas.

Text Area

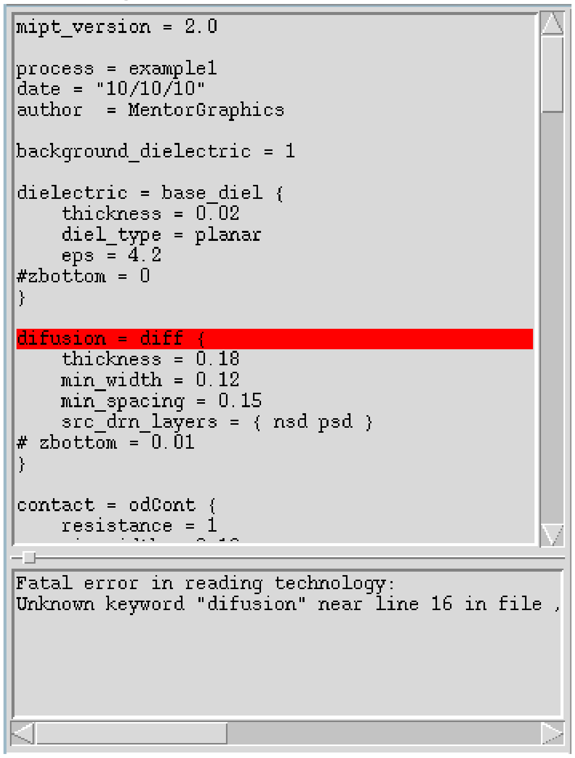
The text area displays the contents of the current MIPT file and any messages from the stack viewer. You can edit the contents of the MIPT file displayed in the text area and save the file back to disk. Once you save the file, the changes are displayed in the stack area.

Figure 7-7. Stack Viewer Window



If there are any errors in the MIPT file, they appear in the text area highlighted in red (Figure 7-8). A message describing the error appears in the message panel below the highlighted error.

Figure 7-8. MIPT File Error



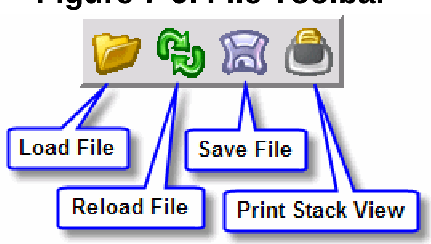
Controls Area

The controls area contains two toolbars, a color legend, and the view options panel. Use the objects in this area to control file I/O and the stack viewer window display.

File Toolbar

The file toolbar (Figure 7-9) contains buttons that enable you to load and save files and print the current view.

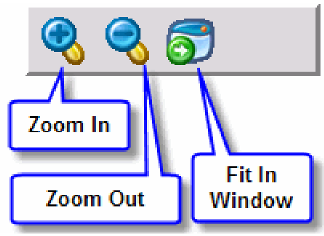
Figure 7-9. File Toolbar



Zoom Toolbar

The zoom toolbar (Figure 7-10) contains icons that enable you to change the zoom level for the stack area display.

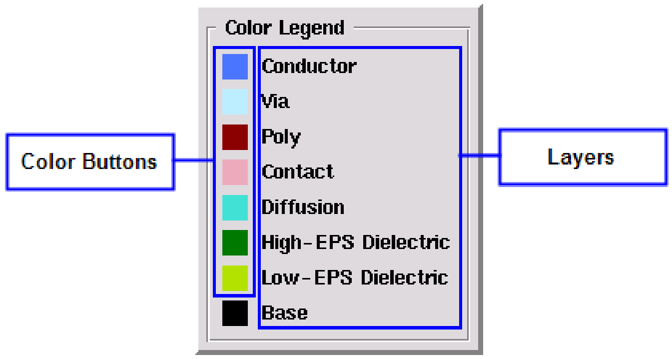
Figure 7-10. Zoom Toolbar



Color Legend

The color legend (Figure 7-11) contains color buttons that enable you to change the colors of the layer types shown in the stack area.

Figure 7-11. Color Legend Features



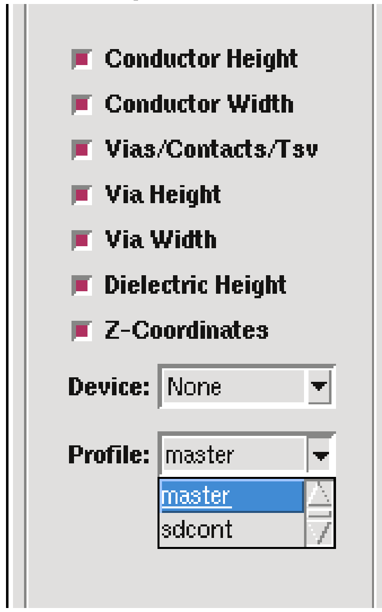
View Options Panel

The view options panel (Figure 7-12) contains buttons that enable you to toggle the visibility of measurements, coordinates, and vias and contacts in the stack area.

Figure 7-12. View Options Panel

The Profile drop list (Figure 7-13) allows you to select which profile you would like to view in the stack area. Available profiles selections can include master, sdcont, substrate, bottom, full_master ultra, and field_bottom. The selection list is context-sensitive based on the contents of the MIPT file. For example, to see field_bottom as a choice in the list, specify a field_bottom profile in the MIPT file.

Figure 7-13. View Options Panel Profile Droplist



Stack Area

The stack area contains a graphical representation of the technology stack.

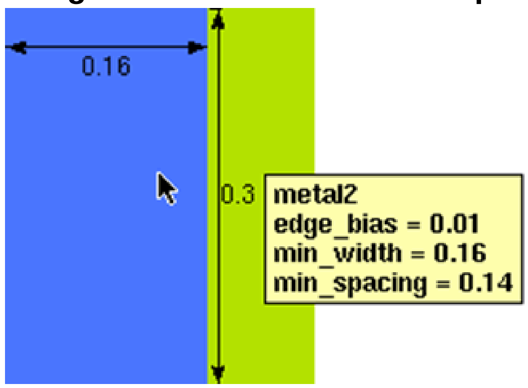
Each layer appears in the color defined in the color legend. By default, the stack viewer shows all layers in the master profile. If the MIPT file contains enough information to construct an sdcont profile, the stack area can contain additional layers such as the diffusion layer, the first metal layer, and the contact between them. The distance between the diffusion contact and the poly is exactly the value specified by the min_poly_cont_spacing attribute of the contact layer.

Layers of type via and contact are drawn as rectangles. The width of the rectangle corresponds to the minimum (min_width) attribute. The height of the rectangle corresponds to the difference in Z-coordinates between the top of the bottom conducting layer and the bottom of the top conducting layer.

Layers of type conductor, poly, and diffusion are drawn as rectangles or trapezoids, depending on whether they are defined with the extra_width attribute. Conducting layers are drawn with dimensions corresponding to their min_width and thickness attributes.

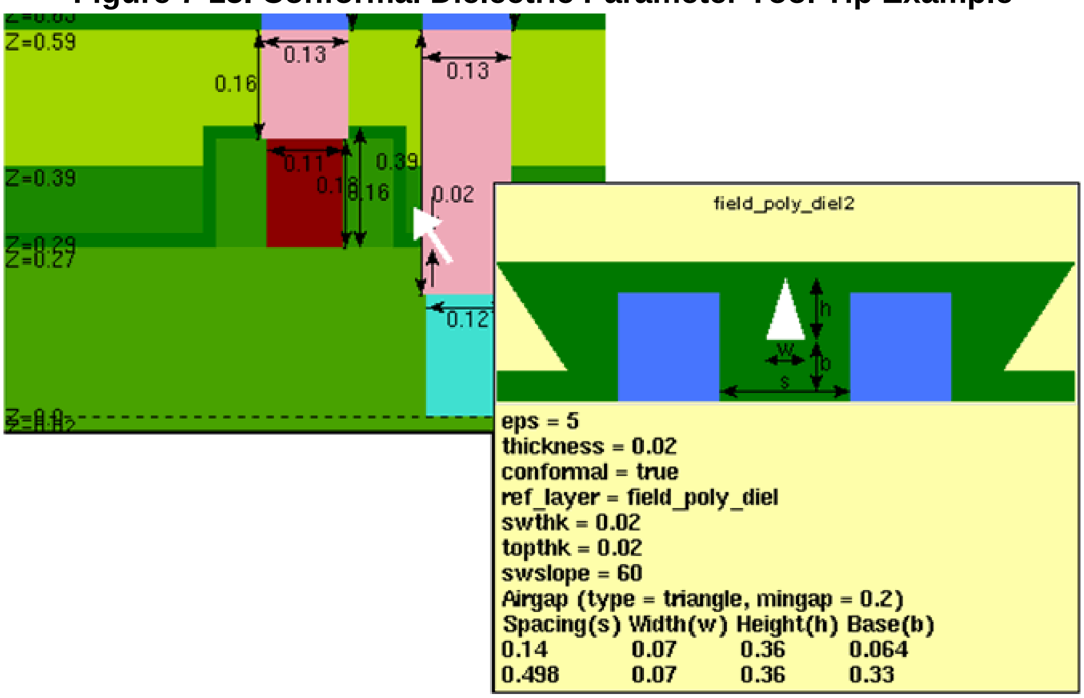
Hover the mouse pointer over a layer to bring up a tool tip (Figure 7-14) containing details about that layer. These details include the layer name, minimum width, and minimum spacing for the layer and may include other attributes as well.

Figure 7-14. Attribute Tool Tip



Some tool tips (Figure 7-15) also display graphical descriptions and include conformal dielectric parameters such as swslope, mingap and airgap.

Figure 7-15. Conformal Dielectric Parameter Tool Tip Example



Chapter 8
xCalibrate Invocation Reference

Invoke xCalibrate from the command line to create calibrated rule files.

This chapter includes sections that provide detailed information on the invocation command options used to create calibrated rule files for capacitance and resistance.

Reference Syntax

The invocation descriptions use font properties and meta-characters to document the command syntax.

For information on syntax conventions used in this chapter, refer to "Syntax Conventions" on page 21.

Setting the CALIBRE_HOME Environment Variable

The xCalibrate tool requires that the CALIBRE_HOME environment variable be set.

See the "Setting the CALIBRE_HOME Environment Variable" section of the *Calibre Administrator's Guide* for details.

Command Descriptions

The xCalibrate tool can be invoked to perform various operations.

- xcalibrate -exec — used for calibration and rule file generation.
- xcalibrate -check — used for syntax verification and logical checks.
- xcalibrate -view — used for launching the Stack Viewer.
- xcalibrate -version — used for software version check.
- xcalibrate -itf2mipt2 — used for interconnect technology format (ITF) file translation.
- xcalibrate -itfmap2svrf — used for interconnect technology format (ITF) mapping file translation.
- xcalibrate -ircx2mipt2 — used for iRCX technology format file translation.
- xcalibrate -interface — used to generate an interface MIPT file from interposer and die MIPT files using a mapfile.
- xcalibrate -cpp — used to preprocess the MIPT file using the C preprocessor directives.

xcalibrate -exec

The xCalibrate invocation command, which performs calibration to produce encrypted capacitance and resistance rule files, *rules.C*, *rules.R*, *rules.xact*, and *rules.ptf*. xCalibrate accepts MIPT or ITF technology files as input.

Usage

xcalibrate -exec
[-turbo *number_of_cpus*]
[-remote *host*[,*host*…] | -remotefile *filename*]
[-noheader]
[-tech *tech_directory*]
[[-xact3d | -xrc3d] [-tcad] | -rperc | -tsv]
[-rulefile *rule_file_name*]
[-retry *n*] [-start *start_stage*] [-end *end_stage*]
[-hidden]
[-svrf_check]
{***mipt_file_name*** [-corner "*corner_name*"] [-aux *auxiliary_file_name*] |
{**-itf** [-corner "*corner_name*"] ***itf_file_name*** [-aux *auxiliary_file_name*]
[[-separate] -itfmap *map_file_name* [*map_file_name* ...]]}}
[-interface *mapfile*]
[-interface *dfmdb_path* -top *top_filename* -bot *bot_filename*]
[-cpp]
[-smart_cal]

Arguments

- **-exec**

  A required argument used to run calibration.

- -turbo *number_of_cpus*

  An optional argument set that specifies the number of cpus to use for multi-threaded parallel processing for calibration. The *number_of_cpus* argument is a positive integer specifying the number of processors (CPUs) to use during processing. If you omit this number, the xCalibrate tool runs on the maximum available for which you have licenses. If you do not apply the -turbo option, the tool defaults to running on one processor.

  For more information on this option, refer to *Calibre Administrator's Guide*.

- -remote *host*[,*host*…]

  An optional argument set that specifies the name of one or more remote hosts on which to run calibration. For more information, refer to "-remote" in the *Calibre Administrator's Guide*.

- -remotefile *filename*

  An optional argument set that specifies the Calibre MTflex configuration *filename* which includes information about the primary and remote hosts, how to partition the run, and the launch method.

  For more information, refer to "-remotefile" in the *Calibre Administrator's Guide*.

- -noheader

  An optional argument that omits the layer stack information and hides the process variation table information from the beginning of the generated rule file.

- -tech *tech_directory*

  An optional argument set that specifies where to generate the calibration directory structure. The default is the current working directory.

- -xact3d

  An optional argument that creates rules for the Calibre xACT 3D calibration flow. Generates an xACT 3D rule file (*rules.FS*) that contains resistance rules and device definitions, and a *rules.ptf* file. The *rules.FS* file references the *rules.ptf* file. Use this argument to create the *rules.FS* file that must be included when running either the Calibre xACT 3D direct netlisting or the Calibre xACT 3D PDB flows.

- -xrc3d

  An optional argument that creates rules for the Calibre xACT 3D calibration flow. Generates an xACT 3D rule file (*rules.FS*) that contains resistance rules and device definitions, a *rules.xact* file that references the *rules.ptf* file, and a *rules.ptf* file. Include the *rules.FS* file when running the Calibre xACT 3D PDB flow. Include the *rules.xact* file when running the Calibre xACT 3D direct netlisting flow.

- -tcad

  An optional argument used with either -xact3d or -xrc3d, that specifies to calibrate layers based on the order they are defined in the MIPT file (not by their elevations).

- -rperc

  An optional argument that creates resistance only rules and can only be run on a metal only stack (no dielectrics). Generates a single resistance rule file (*rules.RO*) that also contains the capacitance rule file header.

- -tsv

  An optional argument that allows partial calibration of the 3D stack (SUBSTRATE + bottom stack description). Generates SUBSTRATE + bottom stack rule files (*rules_TSV.R* and *rules_TSV.C*) that can be combined with the top stack rule files (*rules.R* and *rules.C*) in order to create the full stack description.

- -cpp

  An optional argument that specifies to preprocess the MIPT file using C preprocessor directives. Use this option with the -exec option to preprocess the MIPT file then run calibration. For more information on how to specify the C preprocessor directives in your MIPT file, see “C Preprocessor Directives” on page 37.

- -smart_cal

  An optional argument that specifies to only recalibrate conductor layers whose parameter values have been added or modified after an initial calibration run. You must run calibration at least once before using this option in order to generate the *calibration.db* file. If the *calibration.db* file does not exist or is not found when this option is specified, calibration stops with an error.

  Use this option only when you want to make minor changes to the MIPT file and recalibrate only those changed layers. Conductor layers types that support this option are conductor, contact, device_li, diffusion, li, multigate, pcaux, poly, ronly, seed, src_drn, and via.

- -rulefile *rule_file_name*

  An optional argument set that specifies the common filename prefix for all of the rule file(s) generated by calibration. The default filename prefix is “rules”.

- -retry *n*

  An optional argument set that specifies the number of times to retry the calibration run. Use this keyword for occasional field solver failures. If *n* is not specified, the default is 10.

- -start *start_stage*

  An optional argument set that specifies the name of the stage from which to restart the calibration; for example, masterGenerateModel. The following are examples of stages that appear in the transcript:

  ```
  modInitialize
  masterGenerateModel
  masterExecModel
  sux
  gen_rules
  ```

- -end *end_stage*

  An optional argument set that specifies the name of the stage at which to end the calibration run. The default is to run to the end of the calibration.

- -hidden

  An optional argument that specifies to encrypt all layer information in the calibrated rule files.

- -svrf_check

  An optional argument used to verify the syntax of the SVRF statements generated during calibration. SVRF statements specified using the svrf_verbatim parameter are not checked.

- ***mipt_file_name*** [-aux *auxiliary_file_name*]

  A required argument that specifies the path to and name of the MIPT file.

Use the optional -aux *auxiliary_file_name* parameter set to include additional MIPT layer specification statements defined in the *auxiliary_file_name* in the calibration. For details about the auxiliary file format, see “Auxiliary File Format” on page 471.

- **-itf *itf_file_name***

  A required argument set used to specify the name of an ITF file. The ITF file is converted to an MIPT file (*out.mipt*) before calibration begins.

- -corner “*corner_name*”

  An optional argument set used to specify the process corner variation type for the MIPT file. The *corner_name* must be enclosed in quotes (“ ”). The process corner variation choices are typical, cbest, rcbest, cworst, and rcworst. This option can only be used with ITF or MIPT file types.

  The default is the process name if neither this option or the global keyword corner is specified. This parameter set takes precedence when the global keyword corner is also specified.

- -aux *auxiliary_file_name*

  An optional argument set used to specify the auxiliary file containing any missing MIPT information needed during ITF file translation. This option can only be used with ITF file types.

- [-separate] -itfmap *map_file_name* [*map_file_name* ...]

  An optional argument set used with the -itf parameter to convert an ITF file with one or more ITF mapping files to an MIPT file (*out.mipt*) with embedded SVRF mapping and ignore statements. This option can only be used with ITF file types.

  Specify the optional -separate argument just before the -itfmap argument in the command line to write the SVRF statements that would otherwise be found in the svrf_verbatim section of the generated MIPT file (*out.mipt*) into a separate mapfile called *out.map*.

  Caution

  Since the -separate argument writes the SVRF statements generated by the ITF translator to a separate file, any SVRF rules that are converted that would have been found in the svrf_verbatim section are not included in the calibrated rule files. This may cause changes in expected results.

- -interface *mapfile*

  An optional argument set that specifies to generate the calibrated rule files for a pair of 3DIC components defined in the interface *mapfile*. 3DIC components can be a die, a fanout package, or interposer. See “Interface Mapfile” on page 474 for more information.

- -interface *dfmdb_path* -top *top_filename* -bot *bot_filename*

  An optional argument set that specifies to generate the calibrated rule files for a set of 3DIC component files.

dfmdb_path — Specifies the pathname to the dfmdb directory. This directory contains the mapfile generated by Calibre 3DStack. xCalibrate searches the *dfmdb_path* and executes with the first mapfile encountered. The dfmdb directory name must use the *.dfmdb* extension. The mapfile names in the dfmdb directory must be named *xcalibrate_map* or *xcalibrate.map*.

-top *top_filename* — Specifies the name of the top (TOP) component MIPT file. An iRCX component file can also be specified in place of the MIPT component file.

-bot *bot_filename* — Specifies the name of the bottom (BOT) component MIPT file. An iRCX component file can also be specified in place of the MIPT component file.

Examples

To restart the calibration at the masterGenerateModel stage, enter the following command line:

```
xcalibrate -exec -start masterExecModel techfile.mipt
```

The calibration produces:

- a directory called *process*
- encrypted calibrated rule files, *rules.R*, *rules.C*, and *rules.xact*.

xcalibrate -check

Command used to check for logical inconsistencies and syntax errors in the MIPT or ITF file. Perform this check prior to calibration.

Usage

xcalibrate -check ***mipt_file_name***
| {**-itf** [-corner *corner_name*] ***itf_file_name***
[-aux *auxiliary_file_name*]
[[-separate] -itfmap *map_file_name* [*map_file_name* ...]]}

Arguments

- **-check**

  A required argument used to parse and check the MIPT or ITF file.

- ***mipt_file_name***

  A required argument that specifies the path to and name of the MIPT file.

- **-itf** ***itf_file_name***

  A required argument set used to specify the path to and name of an ITF file. The ITF file is converted to an MIPT file (*out.mipt*) before validation begins.

- -corner *corner_name*

  An optional argument set used to specify the process corner variation type. The process corner variation choices are typical, cbest, rcbest, cworst, and rcworst. The default is typical if this option is not specified. This option can only be used with ITF file types.

- -aux *auxiliary_file_name*

  An optional argument set used to specify the auxiliary file containing any missing MIPT information needed during ITF file processing. This option can only be used with ITF file types.

- [-separate] -itfmap *map_file_name* [*map_file_name* ...]

  An optional argument set used to convert an ITF file with one or more ITF mapping files to an MIPT file (*out.mipt*) with embedded SVRF mapping and ignore statements. This option can only be used with ITF file types.

  Use the optional -separate argument just before the -itfmap argument in the command line to write the SVRF statements that would otherwise be found in the svrf_verbatim section of the generated MIPT file (*out.mipt*) into a separate mapfile called *out.map*. Checks are only performed on the generated MIPT file (*out.mipt*). No checks are done on the separate mapfile (*out.map*).

Examples

Example 1

The following command line instructs the xCalibrate tool to parse the MIPT file, and checks for logical inconsistencies and syntax errors:

```
xcalibrate -check input.mipt
```

Example 2

The following command line instructs the xCalibrate tool to convert the ITF file to MIPT file format, then check for logical inconsistencies and syntax errors:

```
xcalibrate -check -itf input.itf
```

xcalibrate -view

Command used to invoke the stack viewer. The stack viewer shows a graphical representation of the MIPT file.

Usage

xcalibrate -view *mipt_file_name*

Arguments

- **-view**

  A required argument that invokes the xCalibrate stack viewer tool.

- ***mipt_file_name***

  A required argument that specifies the path to and name of the MIPT file.

Examples

The following command starts the stack viewer and loads the MIPT file named *input.mipt*:

```
xcalibrate -view input.mipt
```

Related Topics

Stack Viewer

xcalibrate -version

Command used to show the installed xCalibrate product version number.

Usage

xcalibrate -version

Arguments

- **-version**

  Writes the installed xCalibrate product version number to standard output.

Examples

The following statement:

```
xcalibrate -version
```

instructs the xCalibrate tool to echo the product version number to stdout. For example:

```
// xCalibrate v2014.4_34.23 Fri Jan 14 15:35:43 PST 2015
```

xcalibrate -itf2mipt2

Command used to convert an interconnect technology format (ITF) file to an MIPT file.

Usage

xcalibrate -itf2mipt2 [-corner *corner_name*] ***itf_file_name*** [-aux *auxiliary_file_name*] [[-separate] -itfmap *map_file_name* [*map_file_name* ...]] [-rsh]

Arguments

- **-itf2mipt2 *itf_file_name***

  A required argument set used to convert an ITF file to an MIPT file.

- -corner *corner_name*

  An optional argument set used to specify the process corner.variation type for the MIPT file. The process corner variation choices are typical, cbest, rcbest, cworst, and rcworst. The default is typical if this option is not specified.

- -aux *auxiliary_file_name*

  An optional argument set used to specify the auxiliary file containing any missing MIPT information needed during ITF file translation.

- [-separate] -itfmap *map_file_name* [*map_file_name* ...]

  An optional argument set used with the -itf2mipt2 to convert an ITF file with one or more ITF mapping files to an MIPT file (*out.mipt*) with embedded SVRF mapping and ignore statements.

  Use the optional -separate argument just before the -itfmap argument in the command line to write the SVRF statements that would otherwise be found in the svrf_verbatim section of the generated MIPT file (*out.mipt*) into a separate mapfile called *out.map*.

- -rsh

  An optional argument used to convert the rpsq_vs_width_vs_spacing syntax to an RSH table instead of an RHO table. This option also adds the rsh_type variable to the RSH table syntax. The rsh_type variable overrides the dim_type setting, and specifies whether to use drawn or actual values when calculating resistance. See “ITF Translation” on page 458 and “RSH Table” on page 132 for details.

Examples

Example 1

The following command line converts an ITF file to an MIPT file called *out.mipt*:

```
xcalibrate -itf2mipt2 input.itf
```

Example 2

The following command line converts an ITF file called *process1.itf* to an MIPT file called *out.mipt* using the auxiliary file called *process1_itf.aux* located in the *auxfiles* directory:

```
xcalibrate -itf2mipt2 process1.itf -aux auxfiles/process1_itf.aux
```

Example 3

The following command line converts an ITF file called *process_X.itf* to an MIPT file called *out.mipt* using the mapfile called *processX_mapfile.map*:

```
xcalibrate -itf2mipt2 process_X.itf -itfmap processX_mapfile.map
```

The converter writes any SVRF syntax generated during conversion into an svrf_verbatim section at the beginning of the generated MIPT file (*out.mipt*).

If you do not want the *out.mipt* file to contain the svrf_verbatim section, then specify the -separate argument just before the -itfmap argument in the command line:

```
xcalibrate -itf2mipt2 process_X.itf -separate -itfmap processX_mapfile.map
```

The -separate argument instructs the converter to generate an MIPT file (*out.mipt*) with the stack information and a separate mapfile (*out.map*) that contains any generated SVRF syntax.

xcalibrate -itfmap2svrf

Command used to convert an interconnect technology format (ITF) mapping file to a PEX SVRF mapping and ignore file called *rules.map*.

Usage

xcalibrate -itfmap2svrf *itf_mapping_file*

Arguments

- **-itfmap2svrf *itf_mapping_file***

  A required argument set used to convert an ITF mapping file to an equivalent PEX SVRF mapping and ignore file (*rules.map*).

Examples

The following command line converts an ITF mapping file to a PEX SVRF mapping and ignore file called *rules.map*:

```
xcalibrate -itfmap2svrf itf.map
```

xcalibrate -ircx2mipt2

Command used to convert an iRCX file (*filename*.ircx) to an MIPT file.

Usage

xcalibrate -ircx2mipt2 ***ircx_filename***

Arguments

- **-ircx2mipt2** ***ircx_filename***

  A required argument set used to convert an iRCX file to an MIPT file. The IRCX file can be encrypted or unencrypted. Translation of an encrypted iRCX file produces an encrypted MIPT file, and translation of an unencrypted iRCX file produces an unencrypted MIPT file.

Examples

The following command line converts an iRCX file called *input.ircx* to an MIPT file called *input.mipt*:

```
xcalibrate -ircx2mipt2 input.ircx
```

xcalibrate -interface

Command used to generate an interface MIPT file from interposer and die MIPT files using a mapfile.

Usage

xcalibrate -interface ***mapfile***

Arguments

- **-interface** ***mapfile***

  A required argument set used to combine the MIPT information from die and interposer MIPT files into a single MIPT file.

Description

The mapfile (***mapfile*** or *xcalibrate_map*) content specifies what layers from the die and the interposer are used.

The *xcalibrate_map* file is generated by Calibre 3DStack using the export_layout -enable_rc_deck -map_only command. The MIPT files for the die and interposer as well as the mapfile must be in the current working directory. At least one ground reference must be specified in the MIPT of the interface. For more information on the export_layout command, see "export_layout" in the *Calibre 3DStack User's Manual*.

Examples

Given the following sample from a mapfile, *xcalibrate_map*:

```
BOT = bot.mipt
TOP = top.mipt
{BOT,BT}
{TOP,BT}
CALIBRATE BOT {(BOT,M12):(TOP,M13)}
```

To create an interface MIPT file, *interface.mipt*, that contains the information from the die and interposer files specify the following command:

```
xcalibrate -interface xcalibrate_map
```

The *interface.mipt* file can then be used to generate the rule files needed by Calibre xACT using xCalibrate:

```
xcalibrate -exec interface.mipt
```

xcalibrate -cpp

Command used to preprocess the MIPT file using the C preprocessor directives.

Usage

xcalibrate -cpp ***mipt_file_name***

Arguments

- **-cpp** ***mipt_file_name***

  An required argument set that specifies to preprocess the MIPT file.

Examples

The following command line generates a preprocessed MIPT file called *myfilename_cpp.mipt*:

```
xcalibrate -cpp myfilename.mipt
```

For more information on how to specify the C preprocessor directives in your MIPT file, see "C Preprocessor Directives" on page 37.

Appendix A
Example MIPT 2.0 Files

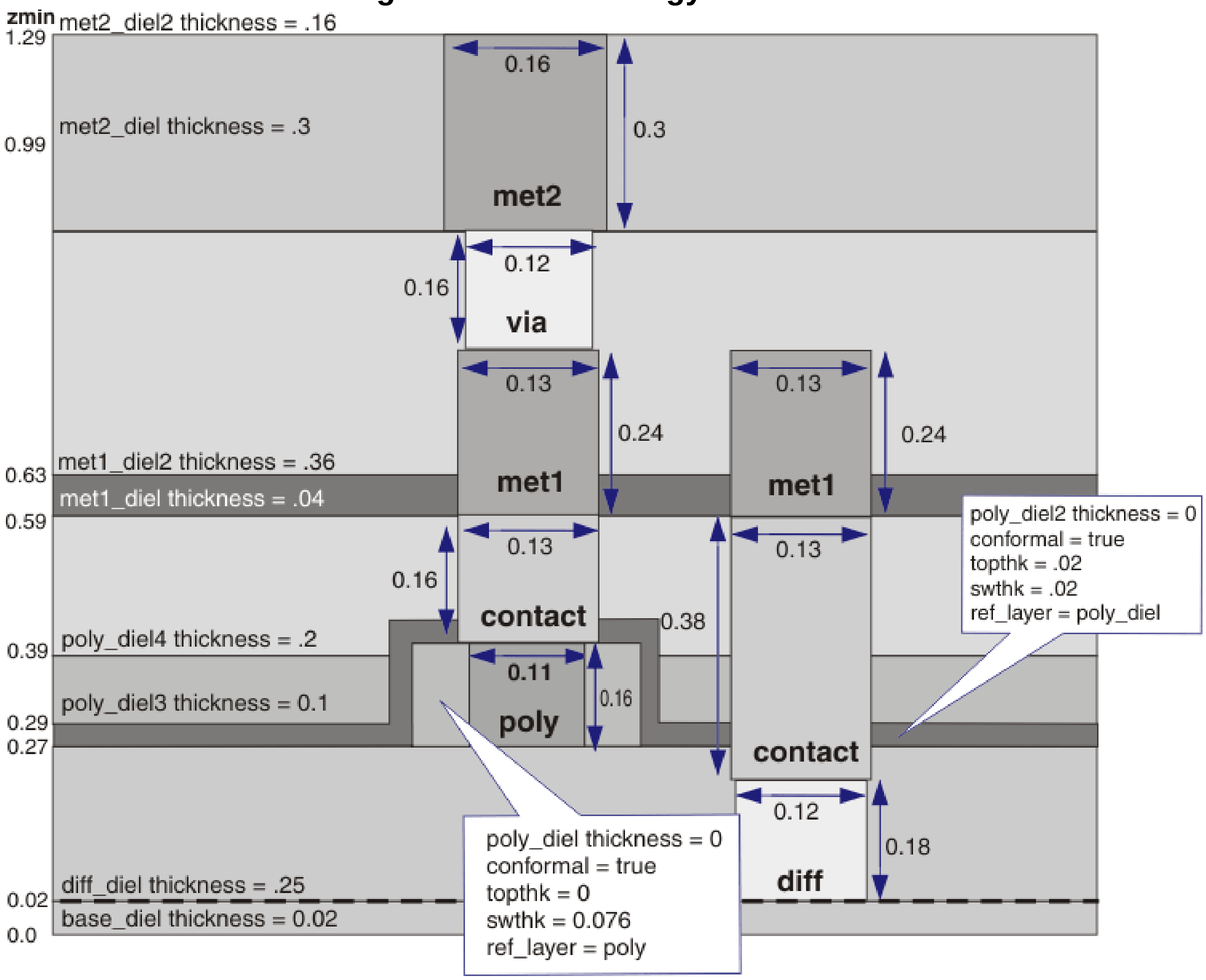
This appendix provides examples of MIPT 2.0 files.

- Example 1
- Example 2

Example 1

The figure shows a technology definition and the *.mipt* file that follows shows how the technology definition is described.

Figure A-1. Technology Definition



Note

The following MIPT file does not include the zbottom settings. The settings are not required in this case because the layers are defined in the correct order. Marking the polysilicon layer as type "poly," the diff layer as "diffusion," and including a diffusion contact allows this technology to be automatically mapped to xCalibrate's field, active, and sdcont profiles.

```
mipt_version = 2.0
process = example1
date = “10/10/10”
author = SiemensEDA
calibration_type = ALL

background_dielectric = 1


dielectric = base_diel {
# zbottom = 0
  thickness = 0.02
  diel_type = planar
  eps = 4.2
}

diffusion = diff {
# zbottom = 0.02
  thickness = 0.18
  min_width = 0.12
  min_spacing = 0.15
  r_sheet = 0.25
  src_drn_layers = {nsd psd}
  extension = 0.05
}

contact = odCont {
  resistance = 1
  min_width = 0.13
  min_spacing = 0.15
  measured_from = diff
  measured_to = metal1
  enclosure_up = 0
  enclosure_down = 0
  gate_to_cont_spacing_min = 0.12
}

dielectric = diff_diel {
  diel_type = planar
  # zbottom = 0.02
  thickness = 0.25
  eps = 4.2
}

poly = polyX {
# zbottom = 0.27
  thickness = 0.16
  min_width = 0.11
  min_spacing = 0.16
  r_sheet = 0.25
  layer_bias = -0.01
  extension = 0.04
}

contact = polyCon {
  resistance = 1.1
  min_width = 0.13
  min_spacing = 0.15
```

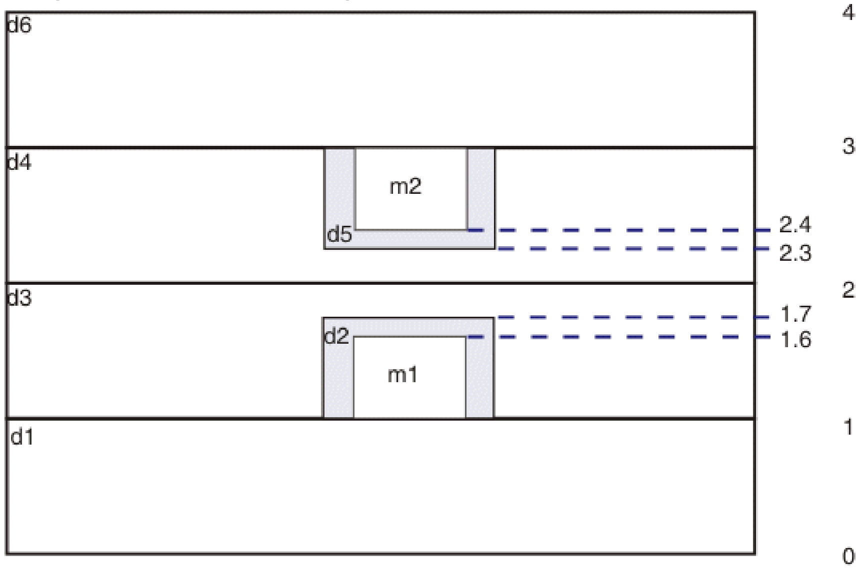
```
    enclosure_up = 0.00
    enclosure_down = 0.00
    measured_from = polyX
    measured_to = metal1
  }

  dielectric = poly_diel {
  # zbottom = 0.27
    thickness = 0
    eps = 5
    diel_type = conformal
    ref_layer = polyX
    topthk = 0.0
    swthk = 0.076
  }

  dielectric = poly_diel2 {
  # zbottom = 0.27
    thickness = 0.02
    eps = 5.5
    diel_type = conformal
    ref_layer = poly_diel
    topthk = 0.02
    swthk = 0.02
  }

  dielectric = poly_diel3 {
    diel_type = planar
    # zbottom = 0.29
    thickness = 0.1
    eps = 4.7
  }

  dielectric = poly_diel4 {
    diel_type = planar
    # zbottom = 0.39
    thickness = 0.2
    eps = 3.2
  }

  conductor = metal1 {
    r_sheet = 0.38
    # zbottom = 0.59
    thickness = 0.24
    min_width = 0.13
    min_spacing = 0.12
    layer_bias = 0.01
  }

  via = via1 {
    resistance = 2
    min_width = 0.12
    min_spacing = 0.14
    enclosure_up = 0.005
    enclosure_down = 0.005
    measured_from = metal1
    measured_to = metal2
  }
```

```
dielectric = metal1_diel {
  diel_type = planar
  # zbottom = 0.59
  thickness = 0.04
  eps = 5
}

dielectric = metal1_diel2 {
  diel_type = planar
  # zbottom = 0.63
  thickness = 0.36
  eps = 3
}

conductor = metal2 {
  r_sheet = 0.25
  # zbottom = 0.99
  thickness = 0.3
  min_width = 0.16
  min_spacing = 0.14
  layer_bias = 0.01
}

dielectric = metal2_diel {
  diel_type = planar
  # zbottom = 0.99
  thickness = 0.3
  eps = 3.8
}

dielectric = metal2_diel2 {
  diel_type = planar
  # zbottom = 1.29
  thickness = 0.16
  eps = 4.2
}

dielectric = metal2_diel3 {
  diel_type = planar
  # zbottom = 1.45
  thickness = 0.25
  eps = 3
}
```

Example 2

The following figure shows a technology definition with trench dielectrics and the MIPT file that follows shows how the technology definition is described.

Figure A-2. Technology Definition with Trench Dielectrics



The following MIPT file describes the technology definition shown in Figure A-2.

```
mipt_version = 2.0
process = example2
date = "10/10/10"
author = SiemensEDA
calibration_type = ALL

background_dielectric = 1

dielectric = d1 {
  diel_type = planar
  # zbottom = 0
  thickness = 1
  eps = 3.9
}

conductor = m1 {
  resistivity = 0.5
  # zbottom = 1
  thickness = 0.6
  min_width = 0.3
  min_spacing = 0.3
}

dielectric = d2 {
  diel_type = conformal
  # zbottom = 1
  thickness = 0
  eps = 3.8
  ref_layer = m1
  topthk = 0.1
  swthk = 0.1
}
```

```
dielectric = d3 {
  diel_type = planar
  # zbottom = 1
  thickness = 1
  eps = 2.9
}

dielectric = d4 {
  diel_type = planar
  # zbottom = 2
  thickness = 1
  eps = 3.9
}

conductor = m2 {
  resistivity = 0.25
  ztop = 0
  measured_from = d4
  thickness = 0.6
  min_width = 0.3
  min_spacing = 0.4
}

# Notice that this trench dielectric, d5, is placed after the metal
# to avoid using ztop and measured_from
dielectric = d5 {
  diel_type = trench
  # zbottom = 3
  thickness = 0
  eps = 4.1
  ref_layer = m2
  topthk = 0.1
  swthk = 0.1
}

dielectric = d6 {
  diel_type = planar
  # zbottom = 3
  thickness = 1
  eps = 4.9
}
```

Appendix B
Error and Warning Messages

Error and warning messages are generated by the xCalibrate tool when performing a syntax check or at runtime.

Error Messages

Error messages must be corrected to continue an xCalibrate run.

Error messages have the following format:

```
ERROR: error_message
```

Table B-1. xCalibrate Error Messages

Message	Possible Causes
Layer name (*layer_name*) already exists.	Two layers have been given the same name.
Incomplete layer definition for layer *layer_name* (Missing parameters are: *parameter_name*) near line *number* in file *name.mipt*	A required parameter is missing. The error lists the parameter names that must be specified.
Unknown keyword: *name* near line *number* in file *name.mipt*	An unknown keyword has been specified. Check for misspelled syntax.
MIPT Technology Checker Error: can't define "r_sheet" and "resistivity for layer "*layer_name*"	The parameters r_sheet and resistivity cannot both be specified for the same layer.
MIPT Technology Checker Error: via_r_select_up_down requires at least one up and down via_resistance2 tables with the same via size for layer *layer_name*	The via_r_select_up_down parameter can only be specified when via_resistance2 tables are specified for the via layer.
MIPT Technology Checker Error: Table (name,property) : (*table_name*, *variable_name*) for layer:*layer_name* has invalid values for *variable_name*. The values must be in ascending order	The values listed for *variable_name* are not in ascending order. The values must be listed from smallest to largest.

Table B-1. xCalibrate Error Messages (cont.)

Message	Possible Causes
Error: *name* is not a directory	The *name* specified for *dfmdb_path* for the xcalibrate -exec command line option: -interface *dfmdb_path* -top *top_filename* -bot *bot_filename* was not a directory name.
Error: XCAL_2_043: Invalid mapfile or invalid dfmdb directory *name*; add .dfmdb extension in case of dfmdb directory	The *name* specified for *dfmdb_path* for the xcalibrate -exec command line option: -interface *dfmdb_path* -top *top_filename* -bot *bot_filename* was either not a mapfile or the .dfmdb extension was missing from the dfmdb directory name.

Warning Messages

Warning messages should be reviewed to determine if they indicate a real problem in your MIPT file.

Warning messages have the following format:

```
WARNING: warning_message
```

Table B-2. xCalibrate Warning Messages

Message	Possible Causes
No SEED layers exist in the MIPT file	A poly layer is defined without an associated seed layer. Make sure the poly layer definition contains a seed_layers parameter. See seed_layers for details.
No SRC_DRN layers exist in the MIPT file	A src_drn layer is defined without an associated diffusion layer. Make sure the diffusion layer definition contains a src_drn_layers parameter. See src_drn_layers for details.
Layer *layer_name*. has no extension.	The extension parameter is missing from a diffusion, src_drn, poly, or seed layer, *layer_name*. See extension parameter for details.

Table B-2. xCalibrate Warning Messages (cont.)

Message	Possible Causes
Ignore the ignore_gateext_to_diff parameter. SVRF is generated based on ignore_endcap parameter setting.	The ignore_gateext_to_diff parameter is set to yes and ignore_endcap parameter is set to yes for the same seed or pcaux layer.
NO POLY CONTACT LAYERS IN MIPT FILE FOR *layername*.	The poly layer *layername* does not have an associated contact or via layer. Make sure the poly layer name has been specified as either the measured_from or measured_to value for a via or contact layer in your MIPT file.
"*layername*" ronly layer is referenced but not defined in MIPT file. It will inherit its properties from the parent layer "*layername*".	A *layername* that does not exist was specified in the ronly_layers parameter in the layer definition. This ronly layer will use the properties from the layer in which is was defined.
Table (name,property) : (*table_name*, sidewall_k) for layer: *layername* has invalid values for spacing. The values must be in ascending order.	For SIDEWALL_K table type only: if the values for width or spacing are not in ascending order, these values are automatically put in ascending order during calibration and this warning is generated. For all other table types, an error is generated. See "Error Messages" on page 513.
RSD of diffusion layer: "*layername*" does not touch device_li layer: "*layername*".	The bottom of device_li layer *layername* does not touch or is not aligned with the top of the RSD layer. Make sure the zbottom of the device_li layer is the same as the ztop of the RSD layer.
layer "*layername*" overlaps with the Base layer "*layername*".	The base layer is a reference layer for all layers above it. Make sure there are no layers overlapping with the base layer in the stack. See Base layer for details.
Placed Seed (*layername*) does not have dielectric conformal around it while its parent poly (*layername*) has dielectric conformal (*layername*) around it.	The placed seed layer specified by the ref_layer keyword in the parent poly layer does not have it's own conformal dielectric layer. To use the same dielectric for the placed seed layer that is specified for the parent poly, specify another conformal dielectric layer with the same properties, except set the ref_layer keyword to the placed seed layer name. See dielectric layer for an example.

Table B-2. xCalibrate Warning Messages (cont.)

Message	Possible Causes
Devices keyword is used in derived seed layer "*layername*" definition. Please move the devices keyword definition to parent layer "*layername*".	The derived layer type does not allow the devices parameter. Specify the devices parameter in the derived layer's parent layer. See devices parameter for details.
No contact is defined for src_drn layer "*layername*".	A corresponding contact layer must be defined for the src_drn layer. See src_drn layer syntax and contact layer syntax for details.
Seed layer "*layername*" is referenced in two parents: "*layername*" and "*layername*". Please remove seed layer "*layername*" from one of the parent layers.	A *layername* specified in the seed_layers parameter appears in more than one layer definition.
The parameter order in table "*tablename*" is (*index2*, *index1*), which is unexpected. Table values are functions of (*index1*, *index2*). The parameter order for table "*tablename*" is switched to (*index1*, *index2*) in this calibration.	The parameters (*index2*, *index1*) specified for the table *tablename* are not in the correct order. The parameter order will be reversed during calibration to ensure the correct output values are calculated. To avoid this warning correct the order of the parameters in *tablename*.
The parameter order in table "*tablename*" is (*index2*, *index1*), which is unexpected. Table values are functions of (*index1*, *index2*). If this is not intended, please switch the table parameter order to (*index1*, *index2*).	The parameters (*index2*, *index1*) specified for the table *tablename* are not in the expected order. To avoid this warning correct the order of the parameters in *tablename*.
ignore_diff_intrinsic is set to "NO" for layer: *layername*.	This warning is generated when the parameter ignore_diff_intrinsic is not set to the default (YES).
ignore_diff_to_diff_under_poly is set to "NO" for layer: *layername*.	This warning is generated when the parameter ignore_diff_to_diff_under_poly is not set to the default (YES).
ignore_gate_to_diff is set to "NO" for layer: *layername*.	This warning is generated when the parameter ignore_gate_to_diff is not set to the default (YES).
ignore_gateext_to_diff_only is set to "YES" for layer: *layername*.	This warning is generated when the parameter ignore_gateext_to_diff_only is not set to the default (NO).
Duplicate Vias measured from *layername* to *layername*.	This warning is generated when more than one via layer definition has the same measured_from and measured_to layer names specified.

Glossary

calibration

A 5-stage process the xCalibrate tool uses for solving target equations for their capacitance constants. The tool subsequently outputs capacitance rule file specification statements suitable for parasitic extraction by the Calibre xACT and Calibre xRC tools.

curvefitting

The process of solving target equations for their capacitance constant(s), k_n. The xCalibrate tool plots and fits simulation data with a curve representing values of k_n; this minimizes total relative error. The fourth stage of the calibration process.

field solver

A simulation tool for measuring inter-conductor electrical effects due to charged conductors in a medium in order to calculate effects such as capacitance. The xCalibrate tool calls the field solver during the third stage (simulation) of the calibration process.

layer

The basic building block of a technology file, described by physical parameters such as thickness and spacing.

process corners

The variations on a "typical" process: for instance, metal thickness may not be exactly controlled.

process variation

The statistically predictable variations within a manufacturing process.

RMOL

A high precision intentional resistor device layer, which needs to be calibrated for capacitance effects.

sensitivity-based rules

SVRF statements for capacitance and resistance calculation that incorporate functions to handle foundry-defined process corners.

technology file

A file containing the process information. You can create a technology file by setting up a MIPT file as described in this manual.

Index

— D —

— E —

— F —

— N —

— O —

— P —

— Q —

— R —

— S —

— T —

— U —

— V —

Third-Party Information

Details on open source and third-party software that may be included with this product are available in the *<your_software_installation_location>/legal* directory.