

Sentaurus™ Topography 3D User Guide

Version T-2022.03, March 2022

SYNOPSYS®

Copyright and Proprietary Information Notice

© 2022 Synopsys, Inc. This Synopsys software and all associated documentation are proprietary to Synopsys, Inc. and may only be used pursuant to the terms and conditions of a written license agreement with Synopsys, Inc. All other use, reproduction, modification, or distribution of the Synopsys software or the associated documentation is strictly prohibited.

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at <https://www.synopsys.com/company/legal/trademarks-brands.html>.

All other product or company names may be trademarks of their respective owners.

Free and Open-Source Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

www.synopsys.com

Contents

Conventions	14
Customer Support	14
<hr/>	
1. Introduction to Sentaurus Topography 3D	16
Functionality of Sentaurus Topography 3D	16
Starting Sentaurus Topography 3D	16
From Sentaurus Workbench	16
From the Command Line	17
Input Files	19
Command File	19
TDR File	19
PMC File	20
Output Files	20
Standard Output and Log File	20
TDR Files	20
PMC File	20
TCAD Sentaurus Tutorial: Simulation Projects	21
<hr/>	
2. Simulation Details	22
Sentaurus Topography 3D Computational Model	22
Computational Model When Using the Level-Set Method	23
Computational Model When Using the PMC Method.	25
Computational Model for Spin-on-Glass Deposition	25
Discretization Size and Accuracy.	25
Discretization Size and Accuracy When Using the Level-Set Method	26
Simulation Flow	27
Machines	28
Initial Structure Generation	28
Creating the Initial Structure.	28
Modifying the Initial Structure.	29
Boundary Types	30

Contents

Simulating Process Steps on 2D and 3D Structures	31
Limitations When Processing 2D Structures	33
Coordinate Systems.....	34
Wafer Coordinate System	34
Simulation Coordinate System.....	34
2D Coordinate System.....	35
Structure Tilt.....	36
Material Names and Aliases	36
Boundary Conditions	37
Boundary Conditions: reflective	38
Radiosity Method	38
Boundary Conditions: none	40
Boundary Conditions: periodic	40
Parallelization.....	40
Basic Shared-Memory Parallelization	40
Advanced Shared-Memory Parallelization Options	41
Parallelization on Distributed Processing Systems Using MPI	41
Damage Modeling in PMC Simulations	42
References.....	43
<hr/> 3. Model Descriptions	44
Level Set-Based Models	44
Modeling Fluxes and Related Physical Effects	44
Neutrals	48
Ions.....	50
Numeric Modeling	53
Orientation-Dependent Models	56
Setting Crystallographic Orientations.....	56
Continuously Rotating and Tilted Structure Modeling	57
Modeling Chemical-Mechanical Polishing Processes	58
Flux and Flux-Emulating Models	59
Flux and Flux-Emulating Deposition Models	60
Flux and Flux-Emulating Etching and Simultaneous Etching and Deposition Models	61
Deposition Models	63
Simple Deposition	63
Physical Vapor Deposition	64
Low-Pressure Chemical Vapor Deposition	64

Contents

Plasma-Enhanced Chemical Vapor Deposition	65
High-Density Plasma Deposition	65
High-Density Plasma 2 Deposition.	66
Atomic Layer Deposition	67
Electrodeposition	67
Electroplating Deposition	70
Crystallographic Orientation–Dependent Deposition	70
Etching Models.	71
Simple Etching	71
Ion-Milling.	72
Reactive Ion Etching	72
Reactive Ion Etching 2	73
High-Density Plasma Etching.	73
High-Density Plasma 2 Etching	74
Ion-Enhanced Etching	75
Wet Etching	76
Dry Etching.	76
Crystallographic Orientation–Dependent Etching	76
Simultaneous Etching and Deposition	77
Simultaneous Etching and Deposition 2.	78
Spin-on-Glass Deposition Model	79
Reaction Models	79
Feedback of Surface Reaction Products	81
Crystalline Materials, Nucleation, and Grain Growth	82
Defining Crystalline Materials.	82
Defining Monocrystalline Materials.	82
Defining Polycrystalline Materials.	83
Loading and Saving Crystalline Materials From Files	83
Plotting Crystal Orientations.	84
Boundary Structure (Using DC or VBE Method)	84
Volume Fraction Output	84
Crystal Orientation–Dependent Reactions.	85
Etching and Sputtering Reactions	85
Adsorption and Deposition Reactions	86
Nucleation and Grain Growth.	88
Diffusion	90
References.	90
<hr/> 4. Plasma Modeling	92
Simulation Modules	92

Contents

User Input Flow	92
Plasma Bulk Model	95
Particle Density and Energy Balance	95
Bulk Reactions	98
Extracting Bulk Datasets	99
Source Power Pulsing	100
Bulk Simulation With Pulsed Power	100
Sheath Simulation With Pulsed Power and Multiple Bias Sources	101
Plasma Sheath Models	102
Self-Consistent Circuit RF Sheath Model	103
Biasing Modes and Bias Pulsing	105
Analytic RF Sheath Model	108
Sheath Model for Neutral Species	109
Possible Issues and Recommendations	110
General Recommendations	110
Reaction Data	111
Negative Electron Temperature or Negative Species Density	112
Convergence of Bulk Model	113
Convergence of Circuit Sheath Model	113
Energy or Angle Resolution of IEAD Is Too Coarse	115
References	115
<hr/>	
5. Surface Charging	117
Overview of Surface Charging	117
Ion and Electron Transport Through an Electric Field	118
Ion Properties and Flux Distribution	118
Electron Flux Distribution	118
Starting Position of Charged-Particle Trajectories	119
Transfer of Impinging Charges to the Surface	119
Electric Material Properties	120
Electric Field Computation	121
Boundary Conditions	121
Poisson Solver: Guidelines	122
Choosing the Mesh Size of the Solver	122
Optimizing the Solver Speed	122
Error Handling	123
Choosing the Update Interval for the Electric Field	123

Contents

Plotting Electric Field, Potential, Charge Density, and Ion Trajectories	124
Writing Charging Datasets at the End of a Simulation	124
Writing Charging Datasets at Intermediate Time Points	124
Plotting Ion Trajectories	125
Extracting the Electric Field, Potential, and Charge Density	125
Example Project	127
References	128
<hr/>	
6. Input Commands	129
Command Syntax	129
Units	130
Syntax for Expressions	131
Summary of Available Commands	133
add_bulk_reaction	138
add_float_parameter	142
add_flux_properties	146
add_formula	149
add_int_parameter	151
add_interface_layer	153
add_ion_flux	157
add_litho_command	159
add_material	160
add_neutral_flux	163
add_reaction	164
add_reaction_properties	168
add_source_species	181
add_species	182
add_volumetric_source_species	183
define_boundary_conditions	184
define_bulk_solver	187
define_charging_model	189
define_damage	193

Contents

define_deposit_machine	195
define_electric_contact	205
define_electric_solver	206
define_etch_machine.	209
define_extraction	218
Extractions for Deposition and Etching Simulations.	218
Extractions for Plasma Simulations	229
define_iad	237
define_layout	240
define_litho_machine.	242
define_mask.	243
define_material_replacement.	247
define_model	251
define_nad	252
define_pattern_density.	254
define_pattern_density_model.	256
define_plasma_model	258
define_probability.	259
define_reactor	261
define_reflection.	269
define_shape	271
define_sheath_solver.	285
define_species_distribution	290
Syntax for Expressions	305
define_species_properties	306
define_structure	309
Loading a PMC Structure.	312
define_volumetric_species_distribution.	313
define_yield	316
deposit	322
Stopping Time-Stepping.	336
TDR Dataset Names for Fluxes	337

Contents

etch	339
Stopping Time-Stepping.....	366
Level Set Models	366
PMC Models.....	366
TDR Dataset Names for Fluxes.....	366
Variance Reduction in PMC Simulations	367
extend_structure	369
extract	375
Specifying Extraction Lines, Planes, and Pairs	376
Extraction Types.....	377
1D or 2D Cuts.....	377
Properties of Exposed Surfaces.....	378
Interface Area.....	379
Areas and Volumes of Regions and Parts	380
Region Names and Materials.....	381
Region and Part Names.....	382
Interface Position	382
Intersections With a Line	384
Intersections With a Plane	385
Shortest Distance.....	386
Shape Analysis.....	387
Cylindrical Hole Profile.....	399
Bounding Box of Materials and Regions	400
Vertical Coordinates of the Top and Bottom of the Bounding Box of Materials and Regions	400
Dimension of Structure.....	402
Damage Integral.....	402
Surface Coverage	402
fill	415
filter_structure	420
finalize_model	438
layout.....	439
let.....	440
litho	450
pattern	454
remove_material	460
save	461
Saving Structures.....	477
Saving Ion Angular Distributions	478

Contents

Saving PMC Structures	478
Saving Boundaries for PMC Structures	479
Dual-Contouring Method	480
Volume Boundary Extraction Method	480
Saving Species Distributions	481
set_material_properties	482
set_orientation	486
solve_reactor	487
transform_structure	489
truncate	491
References	495
7. Integration With Sentaurus Process and Sentaurus Interconnect	497
Introduction	497
Supported Commands and Syntax	497
Additional Supported Parameters	499
The info Parameter	499
The parameters Parameter	499
The repair Parameter	500
Different Default Behavior	500
Boundary Repair	500
Region Merging	500
Parallelization	501
Limitations and Known Issues	501
Boundary Conditions	501
Meshing Thin Layers	501
8. Rate Formula Module	502
Introduction to the Rate Formula Module	502
Model Definition	503
Defining a Model	504
Defining Fluxes	504
Specifying User-Defined Parameters	505
Defining a Rate Formula	506
Machine Configuration	507
Configuring a Machine	507

Contents

Defining a Machine and Specifying User-Defined Global Parameters	507
Specifying Flux Parameter Values	508
Specifying User-Defined Material-Dependent Parameter Values	509
RFM Commands	510
Rate Calculation	511
Syntax for Formulas and Subexpressions	511
Conditional and Relational Functions	511
Data Available for Rate Calculation	512
Example: Reimplementing and Using ionmill Etching Model	515
References	516
9. Working With Reaction Models	517
Setting Up a Simulation Using a Reaction Model	517
Integration With Other Sentaurus Topography 3D Functionality	519
Converting a Result Structure to TDR Boundary File Format	520
Example	520
10. Physical Model Interface for Etching and Deposition	522
Overview	522
Command File Interface	522
C++ Interface	523
Implementing a New Model	525
Information Available to a Model	525
Additional Input Data	526
Data Types Used in Interface	526
Error Handling	527
Compiling the Source Code	527
Using Additional Source Files or Libraries	527
Using the C++ Standard Library	528
Loading the Shared Library	528
Debugging	528
Input and Output Parameters	529
11. Known Issues and Limitations	531
Description of Known Issues and Limitations	531

Contents

A. Examples	536
Initial Structure Generation	536
Simple Trench	536
Fill.	537
Mask or Patterning.	537
Multi-Tapered 3D Shapes.	538
Deposition	539
Atomistic Layer Deposition.	539
Crystallographic Orientation–Dependent Deposition.	539
Electrodeposition	540
Electroplating Deposition	541
High-Density Plasma Deposition	541
High-Density Plasma 2 Deposition.	542
Low-Pressure Chemical Vapor Deposition.	542
Physical Vapor Deposition	543
Plasma-Enhanced Chemical Vapor Deposition	543
Simple Deposition	543
Spin-on-Glass Deposition	544
Etching.	545
Crystallographic Orientation–Dependent Etching	545
Dry Etching.	545
Wet Etching	546
Simultaneous Etching and Deposition	546
High-Density Plasma Etching.	547
High-Density Plasma 2 Etching	547
Ion-Enhanced Etching	548
Ion-Milling.	548
PMI Simple Etching	549
Reactive Ion Etching	549
Reactive Ion Etching 2.	550
Simple Etching	551
Tilt and Units	551
Simulations With 2D Structures.	552
Integration With Sentaurus Process	552

Contents

B. EAD File Format	554
Description of the EAD File Format	554
Mathematical Conventions	555
Special Cases	556
Sampling From the Distribution	556
 Glossary	 557

About This Guide

The Synopsys® Sentaurus™ Topography 3D tool is a three-dimensional simulator for evaluating and optimizing critical topography-processing steps such as etching and deposition.

The third-party software lp_solve 5.5, modified by Synopsys, forms part of Sentaurus Topography 3D. The modified source code (`lp_solve-5.5.zip`) is provided as part of the software package.

For additional information, see:

- The TCAD Sentaurus release notes, available on the Synopsys SolvNetPlus support site (see [Accessing SolvNetPlus on page 15](#))
- Documentation available on the SolvNetPlus support site

Conventions

The following conventions are used in Synopsys documentation.

Convention	Description
Bold text	Identifies a selectable icon, button, menu, or tab. It also indicates the name of a field or an option.
Courier font	Identifies text that is displayed on the screen or that the user must type. It identifies the names of files, directories, paths, parameters, keywords, and variables.
<i>Italicized text</i>	Used for emphasis, the titles of books and journals, and non-English words. It also identifies components of an equation or a formula, a placeholder, or an identifier.

Customer Support

Customer support is available through the Synopsys SolvNetPlus support site and by contacting the Synopsys support center.

Accessing SolvNetPlus

The SolvNetPlus support site includes an electronic knowledge base of technical articles and answers to frequently asked questions about Synopsys tools. The site also gives you access to a wide range of Synopsys online services, which include downloading software, viewing documentation, and entering a call to the Support Center.

To access the SolvNetPlus site:

1. Go to <https://solvnetplus.synopsys.com>.
 - a. Enter your user name and password. (If you do not have a Synopsys user name and password, follow the instructions to register.)
-

Contacting Synopsys Support

If you have problems, questions, or suggestions, you can contact Synopsys support in the following ways:

- Go to the Synopsys [Global Support Centers](#) site on www.synopsys.com. There you can find email addresses and telephone numbers for Synopsys support centers throughout the world.
 - Go to either the Synopsys SolvNetPlus site or the Synopsys Global Support Centers site and open a case (Synopsys user name and password required).
-

Contacting Your Local TCAD Support Team Directly

Send an email message to:

- support-tcad-us@synopsys.com from within North America and South America
- support-tcad-eu@synopsys.com from within Europe
- support-tcad-ap@synopsys.com from within Asia Pacific (China, Taiwan, Singapore, Malaysia, India, Australia)
- support-tcad-kr@synopsys.com from Korea
- support-tcad-jp@synopsys.com from Japan

1

Introduction to Sentaurus Topography 3D

This chapter presents an overview of the functionality of Sentaurus Topography 3D, the required input files, and the generated output files.

Functionality of Sentaurus Topography 3D

Sentaurus Topography 3D is a three-dimensional simulator for evaluating and optimizing critical topography-processing steps such as etching and deposition. Two- and three-dimensional structures composed of an arbitrary number of different materials can be handled.

The initial structure can either be created using simple geometric etching and deposition operations, or be read in from TDR files. The resulting structures can be saved in different TDR file formats and can be used for further processing.

Starting Sentaurus Topography 3D

Sentaurus Topography 3D can be invoked and used in different modes.

From Sentaurus Workbench

Sentaurus Topography 3D is fully integrated in Sentaurus Workbench. On the command line, type the following command to start Sentaurus Workbench:

```
swb
```

Inside Sentaurus Workbench, you can add Sentaurus Topography 3D to any process flow.

From the Command Line

You can start Sentaurus Topography 3D directly from the command line by specifying:

```
sptopo3d [<options>] commandfile.cmd
```

where `commandfile.cmd` is a file containing all the commands for a simulation. See [Chapter 6 on page 129](#) for detailed descriptions of commands.

Table 1 Command-line options for Sentaurus Topography 3D

Option	Description
--keep_parallel_licenses	If you specify this option, then parallel licenses remain checked out after a command using them has been executed. Specifying the <code>keep_parallel_licenses</code> parameter of the <code>let</code> command has no effect when this command-line option is used (see let on page 440). Note: This option is useful to globally change whether parallel licenses must remain checked out after a command using them has been executed without modifying command files. For example, it can be used in the tool databases of Sentaurus Workbench at the global, site, or user level as follows: <pre>set WB_tool(sptopo3d,cmd_line) "-- keep_parallel_licenses n@node@_t3d.cmd"</pre> See <i>Sentaurus™ Workbench User Guide</i> , Tool Databases.
--max_threads <i>	Sets the maximum number of threads per process to be used to execute the commands specified in the command file. Its value must be an integer greater than zero. If the number of threads specified by either the <code>num_threads</code> parameter of the <code>let</code> command or the <code>--threads</code> option exceeds the value of <code>--max_threads</code> , then the number of used threads is set equal to the value of <code>--max_threads</code> (see let on page 440 and the <code>--threads</code> entry in this table). If you do not specify this option, no limit on the number of threads per process is enforced.

Chapter 1: Introduction to Sentaurus Topography 3D

Starting Sentaurus Topography 3D

Table 1 Command-line options for Sentaurus Topography 3D (Continued)

Option	Description
--mpi-file <c>	Sets the name of the file containing a list of hosts where you can run the processes to be used to execute the commands specified in the command file. For details, see <i>TCAD Parallelization Environment Setup User Guide</i> , Creating the MPI Host File. You can specify this option only when the number of processes set by --processes is greater than 1. If you do not specify this option, all processes run on the host where Sentaurus Topography 3D is started.
--no_html_report	If you specify this option, then the etch command does not produce a report (HTML file) of a particle Monte Carlo (PMC) simulation.
--no_intermediate_extraction	If you specify this option, then the deposit and etch commands do not execute intermediate extractions and do not produce output files for such extractions.
--no_intermediate_plot	If you specify this option, then the deposit and etch commands do not calculate intermediate plots and do not produce output files for such plots.
--no_save	If you specify this option, then the save command does not produce output files.
--processes <i>	Sets the number of processes to be used for executing the commands specified in the command file. Its value must be an integer greater than zero. If you do not specify this option, one process is used.
--ssh-check yes no	Specifies whether to test all hosts used by Sentaurus Topography 3D for a silent SSH login before Sentaurus Topography 3D starts. If you do not specify this option, then all hosts are tested. You can specify this option only when the number of processes set by --processes is greater than 1.
--threads <i>	Sets the maximum number of threads per process to be used to execute the commands that support shared-memory parallelization, specified in the command file. Its value must be an integer greater than zero. Specifying the parallel and num_threads parameters of the let command has no effect when this command-line option is used (see let on page 440).

Chapter 1: Introduction to Sentaurus Topography 3D

Input Files

Table 1 *Command-line options for Sentaurus Topography 3D (Continued)*

Option	Description
--use_datex true false	When using material names defined in the <code>datexcodes.txt</code> file, the default behavior (<code>--use_datex true</code>) is that these names are translated to the canonical name for the specified material. Specify <code>--use_datex false</code> to deactivate the use of DATEX material aliases (see Material Names and Aliases on page 36).

Examples

Start Sentaurus Topography 3D on the local host using one process with four threads:

```
sptopo3d --threads 4 commandfile.cmd
```

Start Sentaurus Topography 3D using two processes on the hosts listed in the file `hostfile`, using 8 threads for each process:

```
sptopo3d --mpi-file hostfile --processes 2 --threads 8 commandfile.cmd
```

Start Sentaurus Topography 3D on the local host using one process with four threads, and specify globally that the `deposit` and `etch` commands do not calculate intermediate plots and do not produce output for such plots:

```
sptopo3d --threads 4 --no_intermediate_plot commandfile.cmd
```

Input Files

Sentaurus Topography 3D uses the following input files depending on the initial information available to start a simulation.

Command File

The command file is a text file that contains a sequence of commands that direct the simulation (see [Chapter 6 on page 129](#)). The command file can also contain Tcl commands.

TDR File

The initial structure for a simulation can be read from a TDR file containing a 2D or 3D boundary, or a 3D GC structure. The 3D boundary must have a rectangular base normal to the positive z-axis.

PMC File

The initial structure for a particle Monte Carlo (PMC) simulation can be read from a PMC file.

Output Files

The names of output files created by Sentaurus Topography 3D follow a naming convention. The base name of the output files is defined by the base name of the command file. Different extensions are used, depending on the type of the output file.

Standard Output and Log File

All the commands specified by the input file as well as the messages and output indicating the progress of Sentaurus Topography 3D are, by default, displayed on-screen. Any error or warning condition encountered is also displayed.

The log file has the extension `.log` and contains the same information as the standard output. The file should be examined after completion of a Sentaurus Topography 3D simulation.

TDR Files

Different TDR file formats can be used to save information about structures modified by a Sentaurus Topography 3D simulation. You can use the `save` command (see [save on page 461](#)), for example, to:

- Save structure boundary information at any stage of a simulation
- Save the resulting structure of a simulation that used the particle Monte Carlo method

Furthermore, you can use the `extract` command to save 1D and 2D geometric information from a structure into TDR files (see [extract on page 375](#)).

PMC File

A PMC structure can be saved to a PMC file, which can be subsequently read in to define the initial structure of a new PMC simulation.

TCAD Sentaurus Tutorial: Simulation Projects

The TCAD Sentaurus Tutorial provides various projects demonstrating the capabilities of Sentaurus Topography 3D.

To access the TCAD Sentaurus Tutorial:

1. Open Sentaurus Workbench by entering the following on the command line: `swb`
2. From the menu bar of Sentaurus Workbench, choose **Help > Training** or click  on the toolbar.

Alternatively, to access the TCAD Sentaurus Tutorial:

1. Go to the `$STROOT/tcad/current/Sentaurus_Training` directory.

The `STROOT` environment variable indicates where the Synopsys TCAD distribution has been installed.

2. Open the `index.html` file in your browser.

2

Simulation Details

This chapter provides details about simulations performed using Sentaurus Topography 3D.

Sentaurus Topography 3D Computational Model

Sentaurus Topography 3D simulates deposition, etching, and simultaneous etching and deposition processes using physical and geometric models. Except for the spin-on-glass deposition model, two simulation methods are used to implement the physical models: the level-set method and the particle Monte Carlo (PMC) method.

Some physical models can be run only using the level-set method and others can be run only using the PMC method. To the former class belong the following models:

- Rate formula module (RFM)-based models (see [Chapter 8 on page 502](#)).
- Built-in models:
 - Etching and simultaneous etching and deposition models: `crystal`, `dry`, `etchdepo`, `etchdepo2`, `hdp`, `ion_enhanced`, `ionmill`, `rie`, `rie2`, `simple`, and `wet`
 - Deposition models: `ald`, `crystal`, `electrodeposition`, `electroplating`, `hdp`, `hdp2`, `lpcvd`, `pecvd`, `pvd`, and `simple`

To the latter class belong the reaction models, that is, those defined in terms of reactions rather than rate formulas (see [add_reaction on page 164](#) and [define_model on page 251](#)).

For the spin-on-glass deposition model (named `spin_on`), the surface profile evolution is determined by solving a partial differential equation describing the motion of a fluid driven by surface tension, centrifugal forces, and evaporation (see [Spin-on-Glass Deposition Model on page 79](#)).

All Sentaurus Topography 3D models take as input the structure at the beginning of the simulated process step and produce as output the structure resulting from the simulated process step.

Chapter 2: Simulation Details

Sentaurus Topography 3D Computational Model

The internal representation of the output structure depends on the model used and might differ from the internal representation of the input structure. Sentaurus Topography 3D uses the following main internal structure representations:

- Boundary representation
- Volumetric representation

The output of geometric models and models based on the level-set method is stored internally using the boundary representation. In contrast, the output of PMC-based models is stored internally using the volumetric representation.

In this manual, for brevity, structures stored internally using the boundary representation are referred to as *boundary structures* and structures stored internally using the volumetric representation are referred to as *PMC structures*.

Computational Model When Using the Level-Set Method

When using the level-set method to simulate a model, the exposed surface (see [Boundary Types on page 30](#)) is discretized into a set of surface elements of finite extents, and the processing time is divided into discrete time steps.

At each time step, the velocity at each surface element is determined according to the processing conditions, and the surface is updated. The velocity is computed using the *rate formula*, which is specific to each model, because it expresses with a mathematical relation the deposition or etching mechanisms relevant to the modeled process.

For example, the rate formula of the LPCVD model ([Equation 17 on page 64](#)) states that the deposition rate at any surface point is proportional to the number of neutral particles hitting the surface per unit time and area, including both those reaching the surface directly from the reactor and those that are reemitted from other surface points.

At each time step, all the quantities involved in the rate formula are computed, and the velocities are determined accordingly at each surface point. Among such quantities, a crucial role is usually played by the number of particles hitting the surface per unit time and area, as described for the LPCVD model.

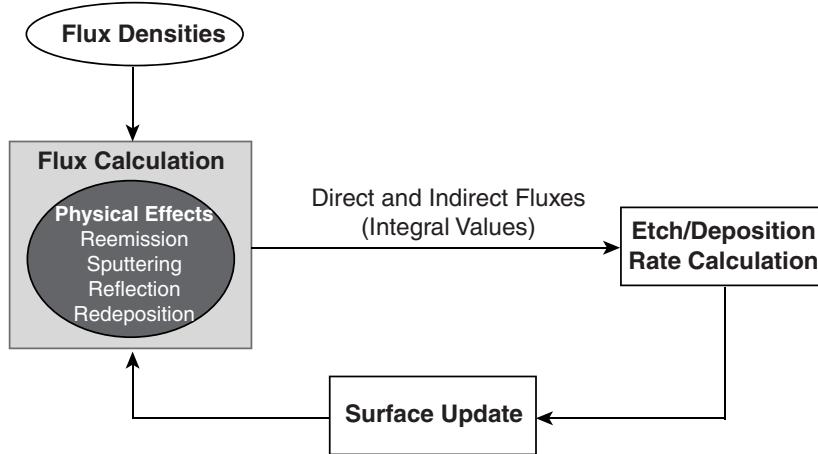
For level set-based models, Sentaurus Topography 3D uses the concept of *flux* to model the flow of particles and their interactions with the surface. In this context, a flux denotes an abstract continuum representation of one or more chemical species having a similar role in the modeled process. Therefore, *abstract* means that different chemical species (which are really involved in the process) can be included in the simulation as a single flux.

The term *continuum* refers to the fact that species are not modeled as sets of discrete particles, but by a continuum flow that is characterized by statistical properties such as the angular distribution or the energy distribution. On this basis, a flux can be considered a continuum of abstract particles.

Chapter 2: Simulation Details

Sentaurus Topography 3D Computational Model

Figure 1 Conceptual model of Sentaurus Topography 3D when using the level-set method



As explained in [Modeling Fluxes and Related Physical Effects on page 44](#), there are different kinds of flux for level set-based models:

- Fluxes with isotropic angular distribution, which are called neutrals
- Fluxes with an anisotropic angular distribution, which are called ion fluxes

The different interactions between a flux and the surface are *physical effects*. Sentaurus Topography 3D supports the physical effects of reemission for neutral fluxes, and reflection, sputtering, and deposition of sputtered material for ion fluxes.

As mentioned, a flux provides a continuum representation of a set of species. More precisely, every flux is characterized by its flux density, which describes the statistical distribution of the velocities of the particles as emitted from their source.

The scalar quantity called the *direct flux* denotes the integrated flux density on a point of the surface. The direct flux is the integral of the flux density over the visible range of the surface point, or the number of particles per unit area and time that reaches the surface at this point before having interacted with the surface.

Note:

When using the level-set method, flux densities are always normalized such that the direct flux is equal to 1 on a completely visible flat surface.

A secondary flux that results from an interaction with the surface is referred to as an *indirect flux*. For example, the number of sputtered particles per unit area and time is referred to as the *sputtered flux*.

Consequently, for each flux, there is one direct flux and possibly several indirect fluxes, according to the modeled physical effects. All these fluxes can be used in the rate formula.

Chapter 2: Simulation Details

Discretization Size and Accuracy

At each time step, all the direct and indirect fluxes involved in the rate formula are computed at each surface point. The computation of all the direct and indirect fluxes requires knowledge about the flux densities and the shape of the surface. Accordingly, when using the level-set method, the computational flow of Sentaurus Topography 3D can be described as a loop over the following three stages:

1. If the model depends on fluxes, compute the direct flux and all indirect fluxes for each flux at each surface point.
2. Compute the velocity at each surface point according to the rate formula.
3. Move the surface according to the velocities provided by the rate formula.

Computational Model When Using the PMC Method

The PMC method uses a stochastic approach to compute the time evolution of a structure. You define interactions between the plasma species and the wafer species in terms of surface reactions (see [add_reaction on page 164](#), [add_source_species on page 181](#), and [define_model on page 251](#)).

Computational Model for Spin-on-Glass Deposition

The time evolution of the film profile is determined by solving a partial differential equation describing the motion of a fluid driven by surface tension, centrifugal forces, and evaporation (see [Spin-on-Glass Deposition Model on page 79](#)).

The processing time is divided into discrete time steps, and the surface is sampled at a set of points. At each time step, the height of the film is computed at each sampling point, and the film profile is updated until the total process time is reached.

Discretization Size and Accuracy

Numeric simulation methods, in general, only produce approximations to the exact results. Usually, the quality of the approximation can be improved by increasing the computational effort. For the level-set method and the method used to solve the spin-on-glass deposition model, the required memory and CPU time are determined mainly by the spatial and time discretization.

For the PMC method, the required memory and the CPU time depend mainly on the spatial discretization, the number of simulated particles coming from the reactor, and the number of simulated reactions.

In Sentaurus Topography 3D, the `spacing` parameter of the `deposit` and `etch` commands sets the grid discretization used to compute the structure evolution (see [deposit on](#)

Chapter 2: Simulation Details

Discretization Size and Accuracy

[page 322](#) and [etch on page 339](#)). The `flux` parameter of the `define_species_distribution` command (see [define_species_distribution on page 290](#)) as well as the parameter `time` of the `etch` command (see [etch on page 339](#)) determine the number of simulated particles.

Decreasing the size of the spatial discretization increases the required memory and the CPU time because there are more grid points to store and process. When using the level-set method, due to the Courant–Friedrichs–Lewy (CFL) condition [1], the maximum time-step size must be reduced. This causes a further increase of the CPU time.

Usually when setting up an etching or a deposition simulation, it is necessary to find a compromise between the required accuracy and the limited available computational resources. A good understanding of the relationship between the discretization sizes and the resulting accuracy is important not only when setting up a simulation, but also when interpreting the results of a simulation.

You can set the spatial discretization for each coordinate axis of the computational grid. For models using the level sets, the spatial discretization along each dimension of the structure can be set. For the spin-on-glass deposition model, since the vertical dimension does not need to be discretized, only the spatial discretizations along dimensions orthogonal to the vertical can be set. You might choose the discretization in a certain direction to be much coarser than for the other directions to reduce simulation time. This is not recommended, however, for various implementation-specific reasons. It is recommended that the discretization for all axes does not vary by orders of magnitude. It has been found that a factor of three between the discretization of the axes produces good results. For models using the PMC method, the spatial discretization along all dimensions must be the same (see [deposit on page 322](#) and [etch on page 339](#)).

Discretization Size and Accuracy When Using the Level-Set Method

In general, the spatial accuracy that can be achieved when using the level-set method is of the order of the spatial discretization. Therefore, you must decide which features of the input structure and the expected output structure are important and choose the spatial discretization accordingly.

For models using the level sets, subresolution accuracy can be achieved in those parts of the simulation domain where the surface shows little variation. For example, when using a deposition model for which the rate does not depend on any property of the surface, the thickness of a deposited layer can be determined with higher accuracy than the spatial discretization size in those areas of the surface that are at a distance from the corners of the surface.

For etching, the situation is more complicated, and some additional considerations must be taken into account. Generally, several materials are etched simultaneously with different etching rates. Sentaurus Topography 3D determines the material-dependent etching rate on the exposed surface.

Chapter 2: Simulation Details

Simulation Flow

To underetch masking materials properly and to avoid certain artifacts, special treatment is necessary at the interface of two different materials. At an interface, the highest etching rate of the materials next to the interface is used. This can lead to problems with very thin layers for which the etching rate is lower than for the neighboring materials, because these thin layers can be etched away in one time-step. To avoid this, it is suggested that the spatial discretization is set to slightly less than the thickness of the thin layer.

As described in [Computational Model When Using the Level-Set Method on page 23](#), at each time step, the exposed surface (see [Boundary Types on page 30](#)) of the processed structure is discretized into surface elements of finite extents, whose sizes depend on the level-set grid spacing.

To determine the values of the material-dependent parameters involved in the rate formulas of etching models and simultaneous etching and deposition models, it is necessary to compute the material or the materials through which each element of the exposed surface passes.

Since the surface discretization depends on the level-set grid, but the material information comes from the input structure, it is possible that some surface elements pass through multiple materials of the structure. This might happen, for example, when an interface between two materials of a structure is not aligned with the level-set grid.

In such cases, the default behavior of Sentaurus Topography 3D is to assign to each surface element the material in which its centroid is contained. When better accuracy is needed, you can change this behavior by setting `region_query_accuracy=subresolution` in the `etch` command (see [etch on page 339](#)). When `region_query_accuracy=subresolution`, the etching or deposition rate of each surface element is computed by taking into account all materials through which a surface element passes.

Simulation Flow

In Sentaurus Topography 3D, independently of the used numeric method, the simulation of a physical process requires:

- A model describing the process of interest
- A structure to be processed

When a process step simulation is complete, you can write the resulting structure to a TDR file (see [Output Files on page 20](#) and [save on page 461](#)) or perform measurements on it using the `extract` command (see [extract on page 375](#)).

A model and its parameter values are specified by defining a machine, as detailed in [Machines on page 28](#). A machine can be applied to simulate a process running on any structure.

Chapter 2: Simulation Details

Simulation Flow

You can create a structure by either using Sentaurus Topography 3D commands or loading a TDR file (see [Initial Structure Generation on page 28](#) for an overview of the commands available to define the initial structure of a simulation and the related concepts).

Machines

Sentaurus Topography 3D can simulate different topography processes: deposition, etching, and lithography. Each process is represented by a machine that groups all of the parameters necessary to perform a simulation (see [define_deposit_machine on page 195](#), [define_etch_machine on page 209](#), or [define_litho_machine on page 242](#)).

Machines must be defined before their use, and a unique name for a given machine type (either deposition, etch, or lithography) must be assigned. When only one machine is defined, the name definition can be omitted and the simulator assigns the default machine name. If more than one machine of the same type is needed, a unique name must be set for each of the defined machines.

The definition of different machines with a corresponding unique name allows the creation of a machine library. A machine defined in such a library can be referenced at any time during the simulation.

Initial Structure Generation

The input structure to be processed in Sentaurus Topography 3D can be obtained in either of the following ways:

- Load a TDR boundary file, for example, `define_structure_file=input.tdr`.
- Create a structure directly in Sentaurus Topography 3D, using a combination of simple geometric etch and deposition steps.

The second option is useful to create simple input structures without using external tools, as discussed in the next sections.

Creating the Initial Structure

When a structure is created with Sentaurus Topography 3D, an initial cuboid structure must be defined. This structure constitutes the base for further etch or deposition processes. The following command creates a unit cube made of silicon (see [define_structure on page 309](#)):

```
define_structure material=Silicon point_min={0 0 0} point_max={1 1 1}
```

Chapter 2: Simulation Details

Simulation Flow

Modifying the Initial Structure

The initial cuboid region can be modified either by adding materials using the `deposit` command or by removing parts of the structure using the `etch` command.

The following commands etch a rectangular trench out of the initial cuboid defined above:

```
define_shape type=cube point_min={0 0.3 0.5} point_max={1 0.7 1.0} \
    name=etch_shape
etch shape=etch_shape
```

The first command defines a cuboid shape that overlaps the original unit cube (see [define_shape on page 271](#)). The second command uses the defined cube `etch_shape` to remove material from the unit cube (see [etch on page 339](#)).

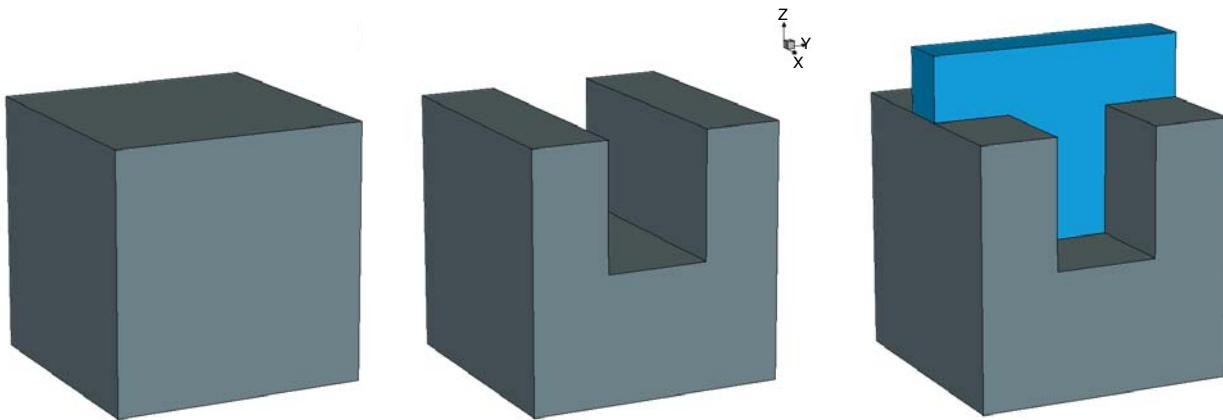
Similarly, it is possible to deposit a cuboid region:

```
define_shape type=cube point_min={0.4 0 0.5} point_max={0.6 1 1.25} \
    name=depo_shape
deposit shape=depo_shape material=Tungsten
```

The above commands define a cube, which is deposited over the trench created earlier. In the `deposit` command, the deposited material must be specified (see [deposit on page 322](#)). Parts of `depo_shape` that overlap the existing structure will not be added.

The structure that has been created could now be saved for visualization by using the command `save` (see [save on page 461](#)). It can also be used as the initial structure for physics-based etch or deposition steps. [Figure 2](#) shows the results of the commands in this section.

Figure 2 *Initial structure generation: (left) initial cuboid structure, (middle) trench etched from the initial structure, and (right) cuboid deposited on the trench*



Chapter 2: Simulation Details

Simulation Flow

Note:

When using level set-based models or the spin-on-glass deposition model, Sentaurus Topography 3D does not require the presence of any gas region on top of the initial structure. However, if the initial structure contains gas regions, gas will be present also in the final structure.

When using a reaction model based on the PMC method, where deposition is allowed, the computational domain must be sufficiently large to contain also the material that will be deposited during the simulation (see [Boundary Types on page 30](#)). This goal can be achieved by adding a gas region on top of the initial structure.

A gas region can be added to an existing structure using the `fill` command (see [fill on page 415](#)). This feature is useful also if a structure produced by Sentaurus Topography 3D must be read by other tools that require the presence of gas on top of their initial structure.

Boundary Types

As described in [Input Files on page 19](#), Sentaurus Topography 3D loads two- or three-dimensional TDR boundary files containing all the geometric and material information of the structure that must be processed.

[Figure 3 on page 31](#) (a) shows a cross section of a typical 3D structure that can be used as the *initial structure* of a simulation. The computational domain is defined as the minimum bounding box that contains the structure (see [Figure 3](#) (b)).

The upper part of the surface of the structure is called the *exposed surface* (see [Figure 3](#) (c)). When using the level-set method or the spin-on-glass deposition model, the exposed surface evolves until the end of the simulation when a final profile of it is obtained (see [Figure 3](#) (d)).

At the end of the simulation, the final exposed surface can be combined with the lateral and the bottom planes of the computational domain, resulting in the *closed surface* (see [Figure 3](#) (e)).

Using Boolean operations, a boundary representation of the final structure, hereafter referred to as the *final boundary structure* (see [Figure 3](#) (f)), which includes all material information, can be created from the initial structure (see [Figure 3](#) (a)) and the closed surface (see [Figure 3](#) (e)).

The boundary data structures – exposed surface, closed surface, and final boundary structure – can be saved in a TDR boundary file using the `save` command (see [save on page 461](#)).

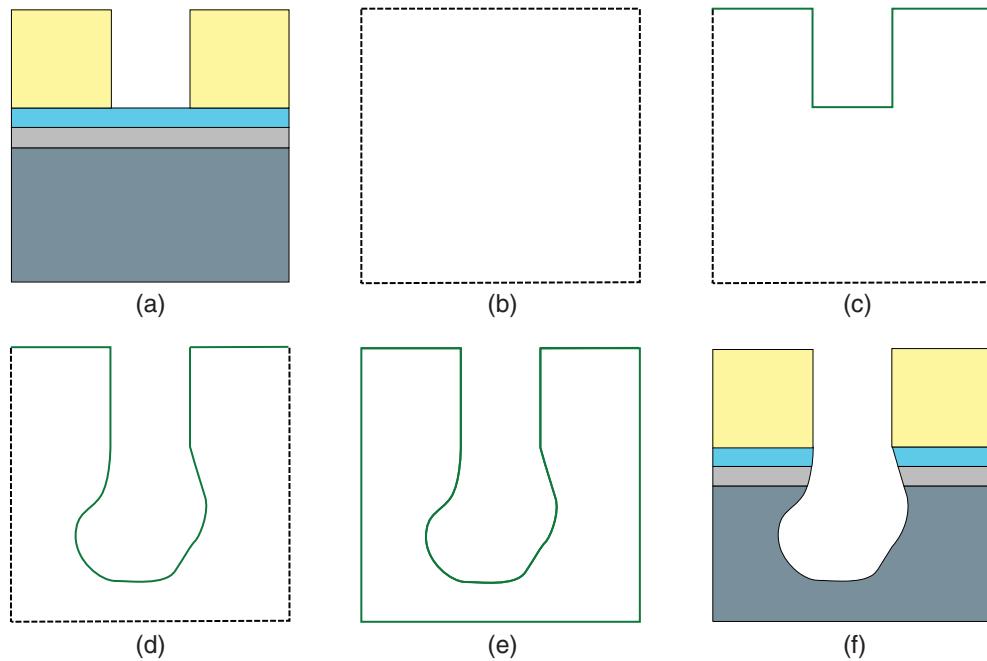
The final boundary structure contains much more information than the exposed and closed surfaces, but the Boolean operations can be computationally expensive.

Chapter 2: Simulation Details

Simulating Process Steps on 2D and 3D Structures

The final structure is necessary and must be saved when transferring the simulation results to another TCAD Sentaurus tool (for example, Sentaurus Process).

Figure 3 Cross section of a 3D structure: (a) initial structure, (b) initial computational domain, (c) initial exposed surface, (d) final exposed surface, (e) final closed surface, and (f) final structure



Simulating Process Steps on 2D and 3D Structures

When using a level set-based model or the spin-on-glass deposition model, Sentaurus Topography 3D can simulate process steps on both two-dimensional (2D) and three-dimensional (3D) structures. Two-dimensional and 3D structures can be simulated using the same physical models, with consistent results. When using the PMC simulation method, only process steps on 3D structures can be simulated.

Simulating 2D cuts of 3D structures typically is orders of magnitude faster than a full 3D simulation. Flux integration for 2D structures is performed using the radiosity method. No Monte Carlo implementation is provided.

For a full list of limitations, see [Limitations When Processing 2D Structures on page 33](#) and [Chapter 11 on page 531](#).

Sentaurus Topography 3D provides a command language that is mostly independent of the dimension of the structure to simulate.

Chapter 2: Simulation Details

Simulating Process Steps on 2D and 3D Structures

A structure is defined by the `define_structure` command (see [define_structure on page 309](#)), and there are two use cases:

- `define_structure file=<c> [name=<c>]`

Sentaurus Topography 3D reads the file specified by the parameter `file`. The last 2D or 3D TDR boundary contained in that file is used to create a structure with the name specified by the parameter `name` or with the default name `default_structure`.

- `define_structure material=<c> point_max=<v> point_min=<v> [name=<c>]`

Sentaurus Topography 3D creates a structure – a cube or a rectangle – depending on the length of the vectors used for the parameters `point_min` and `point_max`, with the name specified by the parameter `name` or with the default name `default_structure`.

Any other command either refers to an already defined structure (through its `structure` parameter) or does not operate on any structure. In this way, you can mix 2D and 3D simulations in the same input file, as shown in the following example:

```
# Define a 3D structure called 'structure_3d'. The dimension of the
# created structure is determined by the size of the 'point_min'
# and 'point_max' parameter values.
define_structure name=structure_3d material=Silicon \
    point_min={0 0 0} point_max={1 1 1}

# Define a 2D structure called 'structure_2d'. The dimension of the
# created structure is determined by the size of the 'point_min' and
# 'point_max' parameter values.
define_structure name=structure_2d material=Oxide \
    point_min={0 0} point_max={1 1}

# Define a deposition machine. The 'define_deposit_machine' command
# does not operate on any structure and the machine it defines can
# be used with both 2D and 3D structures.
define_deposit_machine anisotropy=0.8 curvature=0 material=Nitride \
    model=simple rate=1

# Simulate a deposition process for the 2D structure 'structure_2d'
# and save the result. Because the referred structure is 2D, a 2D
# simulation will be run.
deposit spacing=0.1 structure=structure_2d time=1

# Save the result of the deposition step for structure 'structure_2d'.
save structure=structure_2d

# Simulate a deposition process for the 3D structure 'structure_3d'
# and save the result. This command looks exactly like the one above.
# It only refers to a different structure. Because the referred
# structure is 3D, a 3D simulation will be run.
deposit spacing=0.1 structure=structure_3d time=1

# Save the result of the deposition step for structure 'structure_3d'.
```

Chapter 2: Simulation Details

Simulating Process Steps on 2D and 3D Structures

```
save structure=structure_3d

# Simulate another deposition process for the 3D structure 'structure_3d'
# and save the result. Because a discretization that is not uniform
# across the coordinate directions is specified, the length of the value
# of the 'spacing' parameter must match the dimension of the structure
# this command operates on. If the structure had dimension 2, an error
# will be issued by the 'deposit' command.
deposit spacing={0.1 0.5 0.025} structure=structure_3d time=1

# Save the result of the second deposition step for structure
# 'structure_3d'.
save structure=structure_3d
```

Note:

A 2D structure can be obtained as a cut of a 3D structure using the `extract` command with `type=slice` (see [extract on page 375](#)).

Limitations When Processing 2D Structures

Table 2 lists commands that are not supported or have some limitations when processing 2D structures.

Table 2 Commands not supported or not fully supported when processing 2D structures

Command	Functionality when processing 2D structures
<code>add_litho_command</code>	Sentaurus Lithography integration only works in three dimensions.
<code>define_boundary_conditions</code>	There is no support for nondefault boundary conditions for indirect flux computation for 2D structures.
<code>define_litho_machine</code>	Sentaurus Lithography integration only works in three dimensions.
<code>define_mask</code>	Masks are only supported to process 3D structures.
<code>etch</code>	Supported except for the <code>mask</code> parameter and except when using a reaction model.
<code>filter_structure</code>	Supported except for <code>type=convert_to_pmc</code> , <code>type=rediscretize_boundary</code> , and <code>type=smooth</code> .
<code>litho</code>	Sentaurus Lithography integration only works in three dimensions.
<code>pattern</code>	Patterning is only supported to process 3D structures.

Coordinate Systems

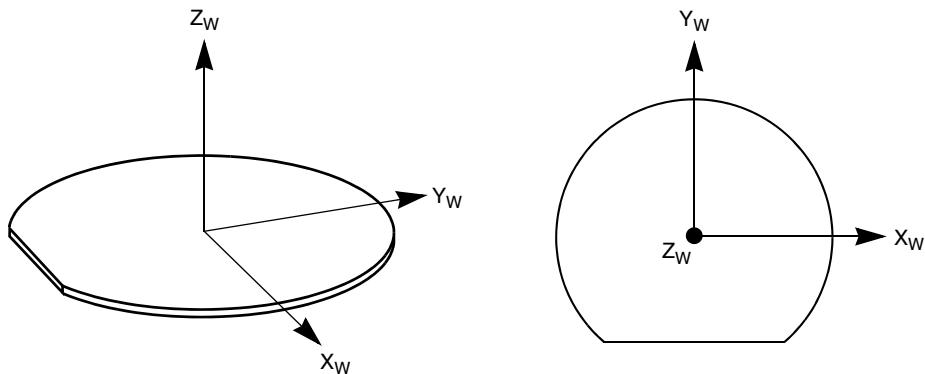
This section describes the wafer coordinate system and the simulation coordinate system.

Wafer Coordinate System

Similar to Sentaurus Process, Sentaurus Topography 3D uses a wafer coordinate system to describe the crystal orientation of the wafer. The wafer coordinate system is a right-handed coordinate system in which the x-axis is parallel to the wafer flat, and the y-axis is perpendicular to the wafer flat. The z-axis is perpendicular to the wafer surface (see [Figure 4](#)).

The tilt and rotation of the structure are defined with respect to the wafer coordinate system (see [Structure Tilt on page 36](#)).

Figure 4 Wafer coordinate system



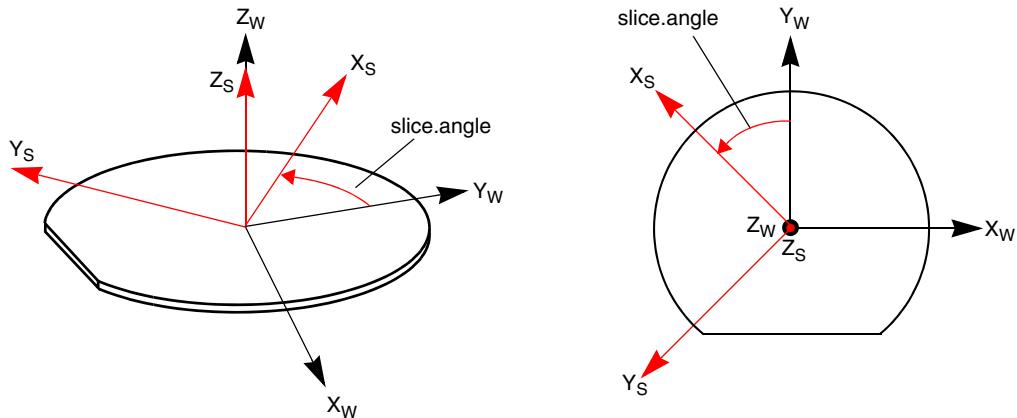
Simulation Coordinate System

In contrast to the simulation coordinate system of Sentaurus Process, in the simulation coordinate system of Sentaurus Topography 3D, the x-axis and y-axis lie in the wafer plane, and the z-axis is perpendicular to the wafer surface. The x-axis of the simulation coordinate system is rotated with respect to the y-axis of the wafer coordinate system by the slice angle (see [Figure 5](#)).

Chapter 2: Simulation Details

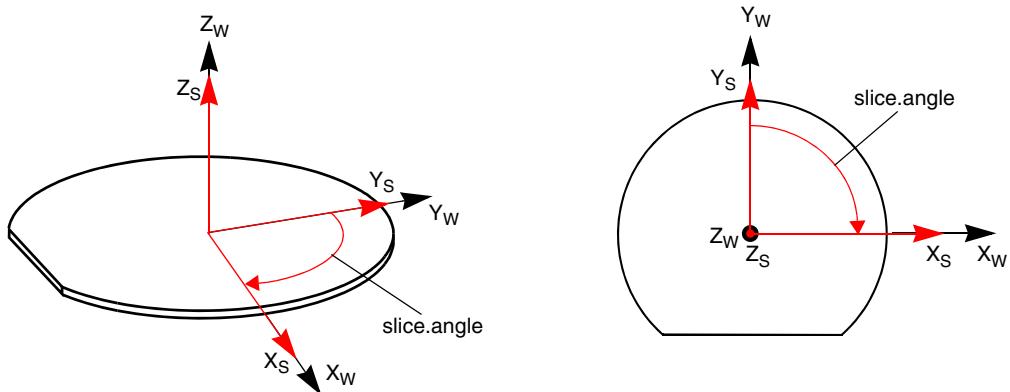
Coordinate Systems

Figure 5 *Simulation coordinate system with slice.angle = 45°*



As in Sentaurus Process, the default value of the slice angle is -90° (see [Figure 6](#)).

Figure 6 *Simulation coordinate system when using default value of slice.angle (-90°)*



The simulation coordinate system is used to specify the coordinates of the simulated structure and all input and output coordinates, except for the crystal orientation.

2D Coordinate System

The simulation coordinate system (x/y) used for 2D structures is oriented in such a way that the positive y-axis points upwards in a device. Therefore, the y-coordinate of the simulation coordinate system used for 2D structures corresponds to the z-coordinate of the simulation coordinate system used for 3D structures.

Structure Tilt

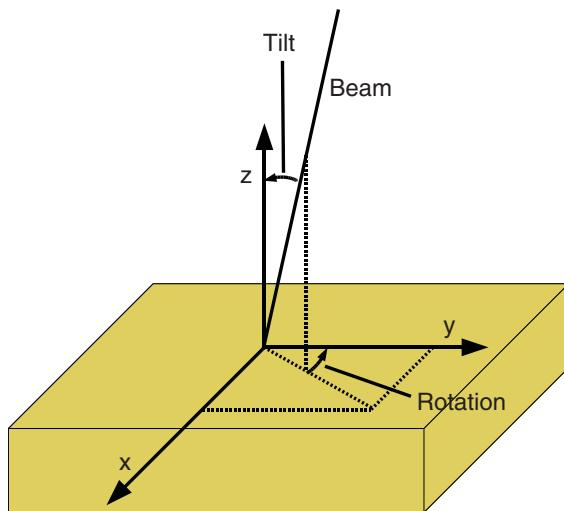
When a machine is defined using either the `define_deposit_machine` command or the `define_etch_machine` command, you can specify the tilt of the structure by setting the parameters `tilt` and `rotation`, which represent the tilt and rotation (or twist) angles as shown in [Figure 7](#).

The tilt is the angle between the beam and the z-axis of the wafer coordinate system. The rotation is defined as the angle between the following two planes:

1. The plane that contains the beam and the z-axis of the wafer coordinate system.
2. The yz plane of the wafer coordinate system.

The tilt angle is positive when measured from the beam axis to the z-axis; while the rotation angle is positive when measured from plane 1 to plane 2 as shown in [Figure 7](#).

Figure 7 Definition of tilt and rotation angles



Material Names and Aliases

By default, when using material names defined in the `datexcodes.txt` file, these names are translated to the canonical name for the specified material. For example, when specifying PolySilicon, it is translated to the canonical name PolySi. Similarly, the material name Resist is translated to Photoresist.

Although using DATEX material aliases can be helpful in large simulation setups where different material names are used, which traditionally refer to the same material, it also can

Chapter 2: Simulation Details

Boundary Conditions

make some simulation setups more difficult to understand. Therefore, the use of DATEX material aliases can be deactivated with the command-line option `--use_datex false`.

Boundary Conditions

Similar to other types of simulation, topography simulations usually handle only a small part of a much larger structure. Typical reasons for this are limited computational resources or a limited area of interest. When limiting the simulation domain to a small part of a larger structure, it is necessary to use boundary conditions to take into account how the parts outside the simulation domain influence the simulation domain.

In general, for some quantities, it is sufficient if the boundary conditions model the immediate vicinity of the simulation domain and, in general, highly accurate results can be obtained. For nonlocal effects, much larger parts outside the simulation domain have a significant influence. This makes the definition of boundary conditions that lead to highly accurate results much more difficult.

For a plasma etching or deposition process, the ion or neutral flux arriving at a point on the surface of the wafer is such a nonlocal effect and can depend on parts of the surface that do not belong to the actual area of interest. Depending on the particular structure, it might still be possible to define boundary conditions that allow you to restrict the simulation domain to the area of interest and to achieve highly accurate results. Unfortunately, this is not always possible.

In cases where limiting the simulation domain to the area of interest and achieving high accuracy is not possible, it is necessary to choose the simulation domain larger than the area of interest.

When increasing the simulation domain, you need to find a trade-off between the improved accuracy in the area of interest and the increase in the required computational resources.

When simulating etching or deposition processes, different boundary conditions are available in Sentaurus Topography 3D: `reflective`, `none`, and `periodic`. The default for the boundary conditions at the sidewalls of the simulation domain is `reflective`, which implements reflective boundary conditions.

The `define_boundary_conditions` command specifies the boundary conditions to use when processing a structure (see [define_boundary_conditions on page 184](#)). However, you can specify periodic boundary conditions only when processing structures using PMC-based models. Periodic boundary conditions are activated automatically whenever you use a model that utilizes the RFM function `pad_pressure()` (see [Data Available for Rate Calculation on page 512](#)) or when processing tilted structures using PMC-based models, as clarified in the following.

For tilted structures, at sidewalls that are not parallel to the tilt vector, boundary conditions of type `reflective` cannot be used. Therefore, another type of boundary condition must be

Chapter 2: Simulation Details

Boundary Conditions

applied at those sidewalls, independently of the boundary conditions that were specified for the sidewalls using the `define_boundary_conditions` command. When using level set-based models, the boundary condition type `none` is always enforced on such sidewalls; whereas, periodic boundary conditions are applied when using PMC-based models.

Boundary Conditions: reflective

In general, with reflective boundary conditions, it is assumed that the sidewalls of the simulation domain are symmetry planes. When checking the visibility of the plasma source or other parts of the surface in a specific direction, a ray is started in this direction and the first intersection with the plasma source, the surface, or the sidewalls of the simulation domain is calculated. If the plasma source or the surface is hit, the process stops. If a sidewall of the simulation domain is hit, the ray is reflected and the next intersection is calculated.

In theory, this corresponds to an infinitely extended surface and gives the correct result if the sidewalls of the simulation domain are symmetry planes. In practice, it is necessary to limit the number of reflections and, thereby, to reduce the size of the extended surface that is taken into account. As long as this limit is high enough, it will not have a significant influence on the result.

When calculating the indirect flux using the radiosity method, it is necessary to solve an equation system. The elements of the system matrix depend on the material properties of the surface and on the geometry of the surface.

Radiosity Method

The geometric interaction between a pair of surface elements is described by the form factor. The form factor must be calculated for all pairs of elements of the original surface. However, it is also necessary to calculate the form factors between elements of the original surface and elements that are part of the extended surface, which was created by reflecting the original surface at the sidewall of the simulation domain where the boundary condition type is `reflective`. At corners where both adjacent sidewalls have the boundary type `reflective`, the surface must be extended for the corner area. This is performed by reflecting the original surface at each of these sidewalls, and the form factors between these elements and the elements of the original surface are calculated.

The complexity of calculating the form factors between elements of the original surface and elements of the extended surface increases rapidly with the number of reflections that were necessary to create the surface element. Therefore, the extended surface is limited to the direct neighbors at the sidewalls and at the corners of the simulation domain.

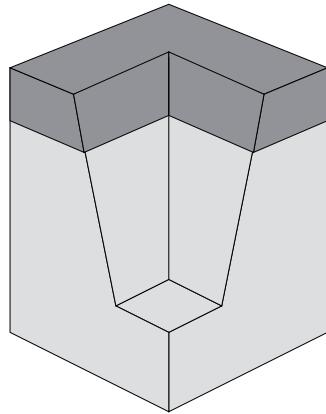
Therefore, the accuracy of the indirect flux calculated using the radiosity method strongly depends on the characteristic of the surface within the simulation domain. For example, if the surface inside the simulation domain represents a quarter or a half of a hole (see

Chapter 2: Simulation Details

Boundary Conditions

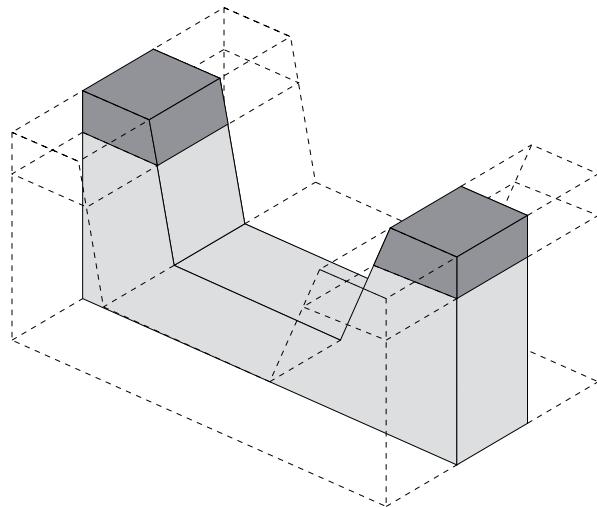
[Figure 8](#)), the interaction of a surface element inside the simulation domain will be limited to other surface elements inside the simulation domain and to elements of the extended surface that were created by one reflection. In this case, the reflective boundary conditions describe the real situation very well, and the result will be highly accurate.

Figure 8 Quarter of a hole



If the surface inside the simulation domain represents the narrow cross section of a trench (see [Figure 9](#)), surface elements much further away than one reflection can have a significant influence. In this case, the result can contain large errors, and it is recommended to use the 2D mode for this kind of structure.

Figure 9 Trench with extended surface at the front and rear sidewalls; the surface is extended at the left and right sidewalls and at the corners but, for this structure, these extensions have no influence on the result and, therefore, are not shown



Chapter 2: Simulation Details

Parallelization

Boundary Conditions: none

When using the boundary condition type `none`, there is no reflection and no calculation of the next intersection when the ray hits the sidewall of the simulation domain. Boundary conditions of type `none` are not supported when using the PMC method.

When calculating the direct flux, the rays are started from the surface. If a ray hits a sidewall for which the boundary condition type is `none`, the process stops and the plasma source is considered to be visible.

When calculating the indirect flux, the original surface is not extended at sidewalls with the boundary condition type `none`. Therefore, there is no contribution to the indirect flux from elements that would be located in this area.

Boundary Conditions: periodic

When using periodic boundary conditions, the processed structure is supposed to repeat infinitely many times along the direction where periodic boundary conditions are applied.

Periodic boundary conditions can be applied only to a pair of parallel sidewalls and not to a single sidewall. You can specify boundary conditions of type `periodic` only when simulating structures with the PMC method. Periodic boundary conditions are always used for simulating tilted structures with the PMC method as well as for RFM models that utilize the RFM function `pad_pressure()`.

Parallelization

Sentaurus Topography 3D can use multiple CPU cores or CPUs to accelerate simulations on shared-memory computers and on distributed processing (DP) systems. Distributed computation is implemented using the message passing interface (MPI) [2].

Shared-memory parallelization (SMP) on the nodes of a DP system is also supported.

Basic Shared-Memory Parallelization

To activate multithreaded computation, use the command:

```
let parallel=true
```

The number of threads is determined automatically, depending on the hardware resources, the optimum number of threads for the algorithm used, and the number of parallel licenses available. This is the recommended SMP mode.

Advanced Shared-Memory Parallelization Options

Advanced users can set a specific number of threads per process using the command:

```
let num_threads=<n>
```

where `<n>` is an integer greater than zero. This forces Sentaurus Topography 3D to use the number of threads specified for each process. The engine tries to check out the necessary number of parallel licenses.

You can also set the number of threads per process using the `--threads` command-line option (see [From the Command Line on page 17](#)). When doing so, the number of threads specified using the `parallel` and `num_threads` parameters of the `let` command are ignored.

You can use the `--max_threads` command-line option to set the maximum number of threads per process.

If insufficient licenses are available, the behavior of the simulator is controlled by the `parallel_license` parameter of the `let` command (see [let on page 440](#)).

Parallelization on Distributed Processing Systems Using MPI

Sentaurus Topography 3D supports parallelization on distributed processing (DP) systems using the message passing interface (MPI) [2].

To activate parallelization on DP systems, specify a number of processes greater than 1 with the `--processes` command-line option when starting Sentaurus Topography 3D (see [From the Command Line on page 17](#)).

You can specify the hosts where to start the processes using the `--mpi-file` command-line option (see [From the Command Line on page 17](#)).

For details, see *TCAD Parallelization Environment Setup User Guide*, Parallelization Using the Message Passing Interface, and Running TCAD Sentaurus Tools on a Cluster.

In Sentaurus Topography 3D, only PMC simulations and the flux integration using the Monte Carlo method for level set-based models (see [Modeling Fluxes and Related Physical Effects on page 44](#)) can be parallelized on DP systems.

In particular, you can distribute the execution of the following commands over different processes, which can run either on the same host or on the hosts of a cluster:

- `deposit` command, when issued with `engine=monte_carlo`
- `etch` command, when issued with `engine=monte_carlo`
- `etch` command, when issued with `method=pmc`

Each process executing these commands can use multiple CPU cores or CPUs of the host where it is running, that is, Sentaurus Topography 3D supports SMP on each process of a DP system. See [Basic Shared-Memory Parallelization on page 40](#) and [Advanced Shared-Memory Parallelization Options on page 41](#).

Note:

For best performance, the total number of threads requested on each node must not exceed the number of CPU cores available on the node.

When running on a cluster, the speedup is limited by the performance of the slowest node and by the speed and bandwidth of the network connections between machines. Therefore, for best performance, use cluster machines with uniform performance and with fast interconnections between them.

Damage Modeling in PMC Simulations

Subsurface damage has been observed during plasma etching for many different material systems. This damage is attributed to high-energy ions impacting the surface during the etching process, resulting in a variety of changes to the damaged material such as amorphization of crystalline materials, formation of dangling bonds, and mixing of atoms and materials.

A simple analytic model is available to track this damage, which increments a damage field with a Gaussian pulse for each ion impact. This model is based on the Hobler damage model [3] and can be specified for any combination of incoming ion and target material.

The addition of each ion results in the damage field being incremented by the following formula, where y is the primary direction, and x_1 and x_2 are the lateral directions:

$$\begin{aligned} f_{\text{damage}} &= f_p(y) \times f_l(x_1, x_2) \\ f_p(y) &= \frac{\text{total_damage}}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y - R_p)^2}{2\sigma^2}} \\ f_l(x_1, x_2) &= \frac{1}{2\pi\sigma_l^2} e^{-\frac{x_1^2 + x_2^2}{2\sigma_l^2}} \end{aligned} \tag{1}$$

where:

- R_p is the peak of the profile and is given by the parameter `projected_range`.
- σ is the spread of the profile in the primary direction and is given by the parameter `sigma`.

Chapter 2: Simulation Details

References

- σ_l is the spread of the profile in the lateral direction and is given by the parameter `lateral_sigma`.
- `total_damage` is the integral of damage over all space.

For details, see [define_damage on page 193](#).

The damage field can be stored in a TDR file by using the `save type=transfer` command. Sentaurus Visual can read this file to visualize the structure. An integral of the damage is available through the `extract type=damage_integral` command. The integration domain can be limited with the parameters `point_min` and `point_max`.

References

- [1] R. Courant, K. Friedrichs, and H. Lewy, “On the Partial Difference Equations of Mathematical Physics,” *IBM Journal*, vol. 11, no. 2, pp. 215–234, 1967.
- [2] For information about the MPI Forum, go to <https://www mpi-forum.org/>.
- [3] G. Hobler and S. Selberherr, “Two-Dimensional Modeling of Ion Implantation Induced Point Defects,” *IEEE Transactions on Computer-Aided Design*, vol. 7, no. 2, pp. 174–180, 1988.

3

Model Descriptions

This chapter describes the fundamental theory and assumptions of the physical models used in Sentaurus Topography 3D. In particular, level set-based models, the spin-on-glass deposition model, and the reaction models are addressed.

Level Set-Based Models

This section discusses level set-based models.

Modeling Fluxes and Related Physical Effects

[Figure 10 on page 45](#) shows an overview of the physical processes relevant to the process models implemented in Sentaurus Topography 3D based on the level-set method.

The source of reactants (in most cases, a plasma) is considered to consist of two kinds of particle: neutrals and ions. Neutrals and ions are assumed to be independent and do not interact with each other. However, they can interact with the surface in various ways, as described in the following.

Moreover, for all plasma processes, the pressure in the reaction chamber is assumed to be very low. Therefore, the mean free path length of ions and neutrals is much larger than the feature size.

Fluxes in Sentaurus Topography 3D are normalized such that a flux integrated on an unshadowed flat surface is equal to one. There are two different kinds of flux:

- Fluxes with an angular distribution that is typically isotropic

These fluxes are suitable for modeling electrically neutral species. Therefore, they will be called neutral fluxes in the following.

It is assumed that neutrals travel in a straight path to the surface without interacting with other particles. On the surface, neutrals react with a certain probability, and deposit or etch material. If they do not react, they are isotropically reemitted from the surface. The reemitted particles can react elsewhere or can be reemitted again.

Chapter 3: Model Descriptions

Level Set-Based Models

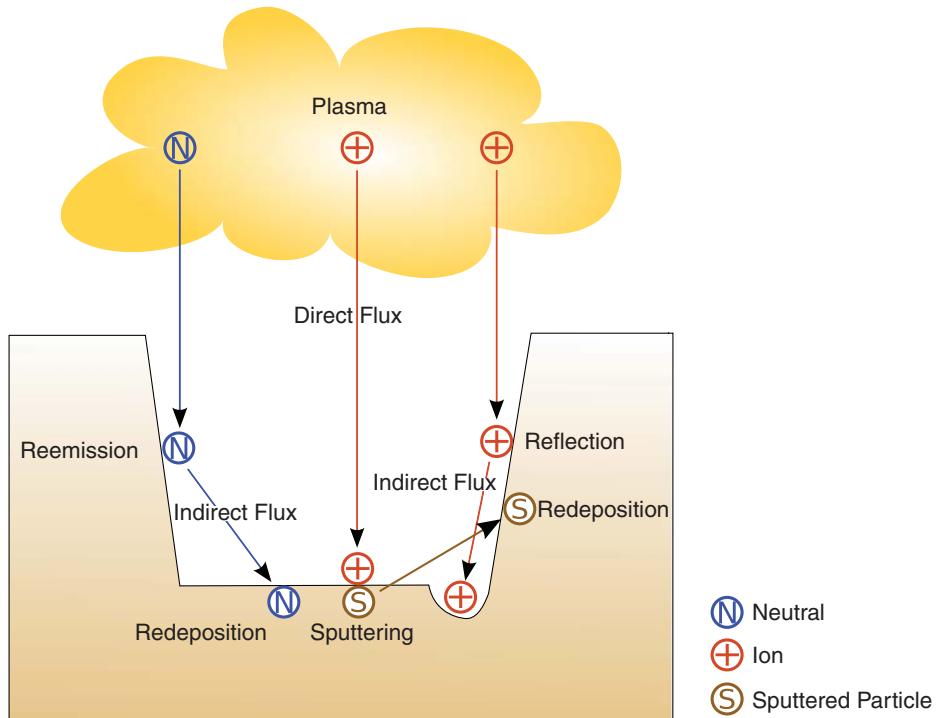
- Fluxes with an anisotropic angular distribution

These fluxes are suitable for modeling charged particles. Therefore, they will be called ion fluxes in the following.

Ions are assumed to travel in a straight path to the surface. Depending on the impinging angle on the surface, ions can then react, or be reflected, or sputter away surface material.

In the case of sputtering or reaction, it is assumed that the ion is consumed. The sputtered particles generate an indirect flux, which can be redeposited elsewhere on the surface. Depending on the material that is being sputtered, the sputtered particles can have an angular distribution with the symmetry axis either normal to the surface or in the direction of the reflected impacting ion.

Figure 10 Physical effects available in Sentaurus Topography 3D for modeling plasma processes



Ion fluxes can have arbitrarily shaped angular distributions. Accordingly, to use a model involving ion fluxes, the angular distributions of each ion flux must be specified (see [define_iad on page 237](#)). Ion angular distributions are properties only of the particles and the processing conditions, and they do not depend on the particular structure under process.

Chapter 3: Model Descriptions

Level Set-Based Models

On the other hand, the distributions of neutral fluxes are supposed to be uniform by default. Therefore, no distribution needs to be defined explicitly for neutral fluxes. However, if needed, this is possible (see [define_nad on page 252](#)).

Neutral and ion fluxes are also different in terms of the physical effects they support.

For neutral fluxes, only reemission is taken into account, which means that a part of the neutrals reaching the surface reacts and sticks to the surface; whereas, the remainder is reemitted isotropically. This effect is characterized by the sticking coefficient according to [Equation 3 on page 48](#).

For each neutral flux, a sticking coefficient must be provided to allow the computation of the reemitted flux. The total number of neutral particles reaching a surface point per unit area and time, taking into account both those directly arriving from the plasma and those reemitted from the surface, is referred to as the *total flux*.

For ion fluxes, it is possible to take into account three physical effects: reflection, sputtering, and deposition of the sputtered material. If reflection is activated, ions can be reflected specularly at the surface of the structure. The reflection probability depends on both the incident angle of the particles and the surface material. As previously mentioned, reflection probabilities are a property not only of the ion flux being reflected, but also of the target material.

Accordingly, to use a model involving reflection, the reflection probability of each ion flux must be known for each material involved in the simulation. The number of particles per unit area and time reaching a surface point after being reflected by all the other surface points is referred to as the *reflected flux*. For built-in models, only the first reflection of ions by the surface is taken into account in the definition of the reflected flux.

If sputtering is taken into account, the number of particles removed from the surface per unit area and time by ions, referred to as the *sputtered flux*, is computed by Sentaurus Topography 3D. The yield function, which gives the number of sputtered atoms per incoming ion as a function of the ion incident angle, is used to specify the process, and it depends on both the properties of the incoming ions and the target material. For this reason, the yield function of each ion flux included in the model must be provided by users for all the materials involved in the structure under process.

Finally, to model reemission of sputtered material, the sticking coefficient, the angular distribution of the sputtered material, and the type of the sputter emission must be provided. For the sputter type `diffuse`, the symmetry axis of the angular distribution of the sputtered material is given by the surface normal. For the sputter type `reflective`, the symmetry axis is defined by the direction of the specular reflection of the incoming ion.

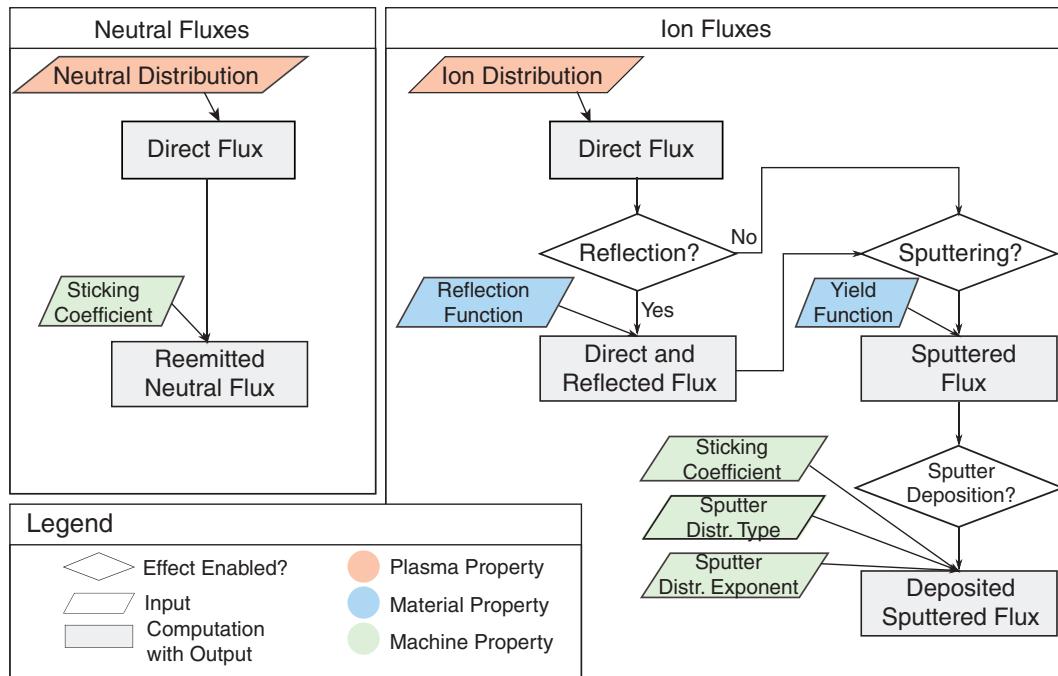
Therefore, the computation of direct and indirect fluxes might require scalar values (such as the sticking coefficient and the sputter type) as well as some functions, for example, ion angular distribution, yield, and reflection functions. Angular distributions are assumed to be only plasma dependent; whereas, yield and reflection functions are specific to both the particles and the target material.

Chapter 3: Model Descriptions

Level Set-Based Models

Figure 11 summarizes the indirect fluxes available in Sentaurus Topography 3D and the information you must provide to compute each of them.

Figure 11 Direct and indirect fluxes and their relationships with physical effects



When a physical effect is activated for a flux, Sentaurus Topography 3D computes the corresponding indirect flux. However, if an indirect flux does not appear in the rate formula, it will have no effect on the evolution of the surface, even if it is computed by Sentaurus Topography 3D. In other words, the definition and the configuration of a flux specify which direct and indirect fluxes are computed at each time step, but the way they are used is defined only by the rate formula. This is important for user-defined models using the rate formula module (RFM) because, for these kinds of model, users are responsible for adding the appropriate fluxes, activating the required physical effects, and defining the rate formula (see [Chapter 8 on page 502](#)).

For built-in models, neutral particles are lumped and modeled as belonging to one flux. The same assumption is made for ion particles in the built-in models. For RFM models, an arbitrary number of ion and neutral fluxes can be used. Each of them can have different properties. [Neutrals on page 48](#) and [Ions on page 50](#) discuss the physical effects in more detail.

Deposition or etching rates for built-in models are obtained by multiplying the normalized flux with the deposition or etching rate measured on an unshadowed flat surface.

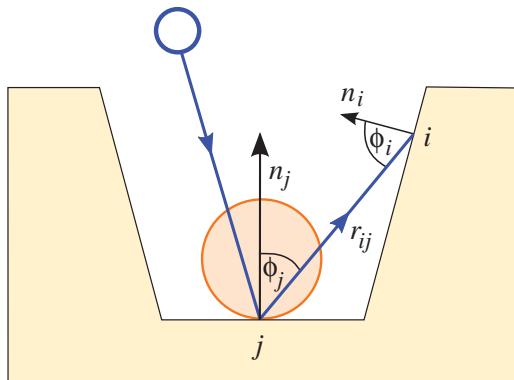
Note:

Level set-based models do not support energy-dependent fluxes.

Neutrals

Usually, a neutral flux is characterized by an isotropic source distribution that is modeled as $\cos\theta$. Neutral particles emitted from the source and arriving at a surface point j can directly react on the point j or can be reemitted several times from the surface and react at a surface point i (see [Figure 12](#)).

Figure 12 Reemission of neutrals



The sticking coefficient is defined as the reaction probability with the surface:

$$\sigma_j = \frac{\Gamma_{\text{reaction},j}}{\Gamma_{\text{neutral},j}} = 1 - \frac{\Gamma_{\text{re-emitted},j}}{\Gamma_{\text{neutral},j}} \quad (2)$$

where:

- $\Gamma_{\text{neutral},j}$ is the total incoming neutral flux at surface element j .
- $\Gamma_{\text{reaction},j}$ is the fraction of the incoming flux that reacts.
- $\Gamma_{\text{re-emitted},j}$ is the fraction of the incoming flux that is reemitted.

The sticking coefficient σ_j is not a constant on the whole surface. It depends on the material at the surface point j but not on the neutral species, and it varies between 0 and 1.

The total neutral incoming flux on a surface point i can be written as the summation of the direct neutral flux plus all the contributions due to reemissions of the surrounding points j [1]:

$$\Gamma_{\text{neutral},i} = \Gamma_{\text{direct},i} + \sum_{j \neq i} (1 - \sigma_j) g_{ij} \Gamma_{\text{neutral},j} \quad (3)$$

where:

- $\Gamma_{\text{direct},i}$ is the direct flux arriving from the source at the point i .
- g_{ij} is the *form factor* that accounts for how much of the reemitted flux from the point j arrives at the point i .

Chapter 3: Model Descriptions

Level Set-Based Models

The form factor depends on the surface geometry and the reemission angular distribution. Assuming that neutrals are reemitted with an isotropic angular distribution [1] (see [Figure 12 on page 48](#)):

$$g_{ij} = \int_{A_j} \frac{\cos\phi_i \cos\phi_j}{\pi r_{ij}^2} V_{ij} dA_j \quad (4)$$

where:

- dA_j is the area of the infinitesimal emitter.
- ϕ_i is the angle between the incoming particle direction and the surface normal.
- ϕ_j is the angle between the emitted particle direction and the normal of the surface of the emitter.
- r_{ij} is the distance between the two surface elements i and j .
- V_{ij} is the mutual visibility matrix, defined as:

$$V_{ij} = \begin{cases} 0 & i \text{ and } j \text{ are not mutually visible} \\ 1 & i \text{ and } j \text{ are mutually visible} \end{cases} \quad (5)$$

[Equation 3](#) is one row of a system of equations that must be solved to find the unknown neutral fluxes $\Gamma_{\text{neutral},i}$ at each surface point.

Analytic Angular Distribution

A good approximation to measured neutral angular distributions (NADs) can be obtained by defining the NAD analytically as:

$$f(\theta) = A \cos^m(\theta) \quad (6)$$

where:

- θ is the angle between the vertical and the incoming neutral direction.
- m is a user-defined parameter (exponent) that describes the anisotropy of the distribution.
- A is a constant that is determined by normalizing the integrated neutral flux on a flat unshadowed surface.

The default NAD is obtained from [Equation 6](#) with $m = 1$.

The flux normalization implies that the normalized flux is dimensionless. Consequently, the total etching or deposition rate can be obtained by multiplying the integrated and normalized flux on a surface element by the etching or deposition rate of a flat unshadowed surface.

Chapter 3: Model Descriptions

Level Set-Based Models

User-Defined Neutral Angular Distribution

The flux models of Sentaurus Topography 3D allow the definition of user-defined NADs, which can be obtained by measurement or from plasma simulations. Arbitrary NADs can be defined or loaded using the `define_nad` command (see [define_nad on page 252](#)). NADs are defined in a tabular format, and linear interpolation is used between data points.

User-defined NADs will be normalized internally when used in flux models.

Ions

The ion flux is characterized by a directional angular source distribution. The source distribution is either modeled as $\cos^m\theta$, where the exponent m controls the flux anisotropy, or specified as an arbitrary ion angular distribution (IAD), using the command `define_iad` (see [define_iad on page 237](#)) and the parameter `iad` in the respective etching or deposition models.

Ions can react on the surface and can etch or deposit. They also can sputter some material from the substrate or be reflected by vertical walls producing the microtrenching phenomenon at the bottom of a trench.

Analytic Ion Angular Distribution

A good approximation to measured IADs can be obtained by defining the IAD analytically according to [Equation 6](#).

As for neutrals, the flux normalization implies that the normalized flux is dimensionless. Consequently, the total etching or deposition rate can be obtained by multiplying the integrated and normalized flux on a surface element by the etching or deposition rate of a flat unshadowed surface.

User-Defined Ion Angular Distribution

The flux models of Sentaurus Topography 3D allow the definition of user-defined IADs, which can have been obtained by measurement or from plasma simulations. Arbitrary IADs can be defined or loaded using the `define_iad` command (see [define_iad on page 237](#)). The IADs are defined in a tabular format, and linear interpolation is used between data points.

User-defined IADs will be normalized internally when used in flux models.

Sputtering

High-energy ions can sputter some substrate material. The sputter etch rate depends greatly on the impact angle θ_{im} of the ions (the angle between the normal of the surface element and the incoming ion direction; see [Figure 13 on page 51](#)).

Chapter 3: Model Descriptions

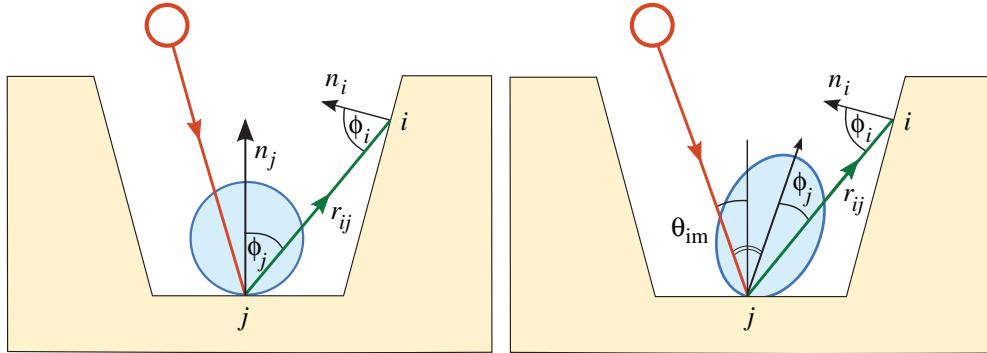
Level Set-Based Models

This dependency is expressed by the yield function:

$$\gamma(\theta_{\text{im}}) = s_1 \cos \theta_{\text{im}} + s_2 \cos^2 \theta_{\text{im}} + s_4 \cos^4 \theta_{\text{im}} \quad (7)$$

where s_1 , s_2 , and s_4 are the sputtering coefficients.

Figure 13 Sputtering with the symmetry axis of the angular distribution (left) along the surface normal and (right) along the reflected impacting ion direction



In Sentaurus Topography 3D, the sputtered flux is evaluated under the assumption of a narrow angular distribution (a large coefficient m in the angular distribution) for which it is possible to write:

$$\Gamma_{\text{sputter},j} = \gamma(\theta_{\text{im}}) \Gamma_{\text{ion}} \quad (8)$$

where Γ_{ion} is the total direct ion flux, which is normalized. For the sputtered flux Γ_{sputter} to be normalized such that the total sputtered flux from an unshadowed flat surface is unity, it is necessary that $\gamma(0) = 1$. Using this constraint, a relevant condition on the sputtering coefficients is obtained:

$$s_1 + s_2 + s_4 = 1 \quad (9)$$

This means that for the definition of the yield function $\gamma(\theta_{\text{im}})$, only two parameters are required, and Sentaurus Topography 3D uses the parameters s_1 and s_2 .

The sputtered material can be redeposited as well. It is assumed that the redeposition process occurs with a probability $\sigma_{\text{redeposit}}$, and no further reemissions are considered, that is, the remaining $1 - \sigma_{\text{redeposit}}$ sputtered material is considered to be volatile. The redeposition flux is:

$$\Gamma_{\text{redeposition},i} = \sigma_{\text{redeposit}} \sum_{j \neq i} v_{ij} \Gamma_{\text{sputter},j} \quad (10)$$

Chapter 3: Model Descriptions

Level Set-Based Models

where v_{ij} is another form factor that accounts for how much of the sputtered material from the point j arrives at the point i :

$$v_{ij} = \int_{A_j} (m+1) \frac{\cos\phi_i \cos^m \phi_j}{\pi r_{ij}^2} V_{ij} dA_j \quad (11)$$

where ϕ_j now has a general meaning as the angle between the emitted particle direction and the symmetry axis of the angular distribution of the sputtered material (see [Figure 13 on page 51](#)).

The differences between g_{ij} and v_{ij} are:

- g_{ij} is evaluated under the assumptions that the reemission of neutrals can be approximated with an isotropic angular distribution. The symmetry axis of the angular distribution is the surface normal, and the exponent of the cosine distribution is one.
- v_{ij} is evaluated under more general assumptions. Depending on the surface, the angular distribution of the sputtered material can be either:
 - Diffuse – The sputtered particles have no *memory* of the impact direction, and the axis of the distribution is normal to the surface element (see [Figure 13 \(left\)](#)).
 - Reflective – The sputtered particles keep some momentum of the impacting ion, and the axis of the distribution has a preferential direction, which is the reflected incoming ion direction (see [Figure 13 \(right\)](#)).

In both cases (diffuse and reflective), an exponent can be assigned to the angular distribution.

Reflection

Low-energy ions with a large incident angle have a larger probability of being reflected by walls (see [Figure 14](#)). This is an important phenomenon that contributes to microtrenching at the bottom of sidewalls. The probability that an ion is reflected is modeled according to [2]:

$$P_{\text{reflection}}(\theta_{\text{im}}) = \min \left\{ 1, k \left[\frac{1}{2\pi} + \left(\frac{\pi}{4} - \frac{1}{3} \right) \frac{1}{\left(\frac{\pi}{2} - \theta_{\text{im}} \right)^2} + \frac{5}{\pi^3} \left(\frac{\pi}{2} - \theta_{\text{im}} \right) \right] \right\} \quad (12)$$

where k is a constant that depends on the mass and atomic number of the incoming ion and the wall material, and on the ion energy:

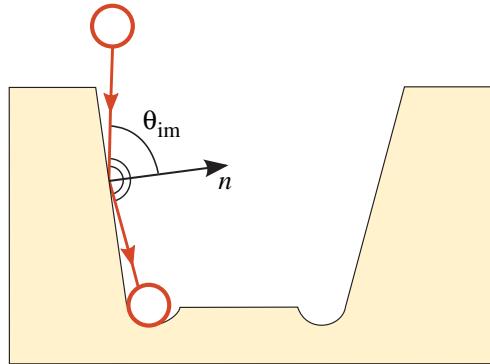
$$k \propto \frac{1}{E_{\text{ion}}^2} \quad (13)$$

The constant k can be set for each material in the structure using the parameter `reflection` in the `add_material` command and the `define_deposit_machine` command (see [add_material on page 160](#) and [define_deposit_machine on page 195](#)).

Chapter 3: Model Descriptions

Level Set-Based Models

Figure 14 Ion reflection process: ions with a large impact angle can be reflected and react at the bottom of a trench producing microtrenching



Numeric Modeling

The number of incoming (neutral or ion) particles on a surface element is calculated by integrating the various contributions to the total flux. Sentaurus Topography 3D offers two different numeric methods to perform the flux integration: the radiosity method and the Monte Carlo method.

You set the integration method with the parameter `engine` in the `etch` and `deposit` commands. Some models do not support both methods. Furthermore, models that do not perform any flux integration do not need or support these two methods.

Note:

For 2D structures, the only supported flux integration method is the radiosity method.

Table 3 Deposition and etching models: Support for numeric integration methods

Model	Radiosity support	Monte Carlo support
Deposition		
ald	Yes	Yes
crystal	No	No
electrodeposition	No	No
electroplating	No	No
hdp	Yes	Yes
hdp2	Yes	Yes

Chapter 3: Model Descriptions

Level Set-Based Models

Table 3 Deposition and etching models: Support for numeric integration methods

Model	Radiosity support	Monte Carlo support
lpcvd	Yes	Yes
pecvd	Yes	Yes
pvd	Yes	Yes
simple	No	No
Etching		
crystal	No	No
dry	Yes	Yes
etchdepo	Yes	Yes
etchdepo2	Yes	Yes
hdp	Yes	Yes
hdp2	Yes	Yes
ion_enhanced	Yes	Yes
ionmill	No	No
rie	Yes	Yes
rie2	Yes	Yes
simple	No	No
wet	No	No

Radiosity Method

The radiosity method is a numeric method that solves [Equation 3 on page 48](#) by evaluating the direct flux Γ_{direct} and then by inverting the linear system matrix to find the unknown fluxes at each surface point.

The linear system matrix scales with the square of the number of surface elements and, therefore, it can become prohibitively large for simulations with small level-set grid spacing. In addition, inverting the system matrix does not lend itself well to parallelization and,

Chapter 3: Model Descriptions

Level Set-Based Models

therefore, simulations can use more wallclock time than the Monte Carlo method on multicore machines.

The radiosity method is used by default. It is activated explicitly using `engine=radiosity` in the `etch` and `deposit` commands.

Note:

The following limitations apply:

- The only supported flux integration method for 2D structures is the radiosity method.
- The boundary condition type `reflective` is not supported for the indirect flux calculation for 2D structures.
- The radiosity method does not provide accurate results for machines with a tilt angle greater than approximately 70° for 3D structures.

Monte Carlo Method

In the Monte Carlo method, neutral or ion particles are sent randomly from the top of the simulation domain to the structure. These particles interact with the surface and can be adsorbed or reemitted, and can sputter some material. The particles can be reemitted several times from the surface depending on the sticking coefficient.

The Monte Carlo method uses less memory than the radiosity method because there is no large system matrix to invert. The simulation of individual particles scales well on multiple cores and DP systems. Therefore, the Monte Carlo method shows significant speedup on multicore machines when run in parallel and on DP systems (see [Parallelization on page 40](#)).

You activate the Monte Carlo method using `engine=monte_carlo` in the `etch` and `deposit` commands.

Note:

When using the Monte Carlo method, be careful with the following:

- Due to its stochastic nature, the Monte Carlo method introduces some numeric noise. Therefore, the results of the radiosity method and Monte Carlo method do not always match exactly. The parameter `integration_samples` can be used to increase the number of samples. This increases simulation accuracy at a higher computational cost.
- There is no Monte Carlo implementation for 2D structures. Only the radiosity method is available to perform flux integration with 2D structures.
- The Monte Carlo method can be activated only for models that perform flux integration. See [Table 3 on page 53](#) for a list of supported models.

Orientation-Dependent Models

Sentaurus Topography 3D supports two methods for orientation-dependent etching or deposition modeling when using the level-set method: the built-in `crystal` models and the function `directional_value()` in the rate formula module (RFM) (see [Chapter 8 on page 502](#)).

The RFM function `directional_value()` is used to introduce an orientation dependency into the rate formula of an RFM model. This function has three arguments that are the values of an arbitrary quantity for the directions $<100>$, $<110>$, and $<111>$, respectively. Depending on the direction of the normal of a surface element, an appropriately interpolated value of this quantity is returned and can be used in the rate calculation (see [Data Available for Rate Calculation on page 512](#)).

For each region of a structure, the crystal orientation can be defined. When creating a new region with the `define_structure`, `deposit`, or `etch` command, the crystal orientation can be defined with the parameters `flat_orientation` and `vertical_orientation` (see [define_structure on page 309](#), [deposit on page 322](#), and [etch on page 339](#)).

For structures that were loaded from a TDR file and do not yet contain crystal orientations, or to change the orientation of a region, the `set_orientation` command can be used (see [set_orientation on page 486](#)).

Crystal orientations are always specified with respect to the wafer coordinate system (see [Wafer Coordinate System on page 34](#)).

Setting Crystallographic Orientations

When using a model with orientation-dependent rates, the crystallographic orientations must be defined with respect to the wafer coordinate system.

The slice angle, which defines how the simulated structure is oriented with respect to the wafer coordinate system, can be set only with the `define_structure` command (see [Simulation Coordinate System on page 34](#)). For a newly created structure, the slice angle is defined with the parameter `slice_angle`. If a structure is loaded from a TDR file, the slice angle is read from that file, but the value of the slice angle can be overwritten with the parameter `slice_angle` of the `define_structure` command.

The flat orientation and the vertical orientation of a region can be set with the following commands:

- `define_structure` (only if it is not used to load a structure from a TDR file)

This command is used to set the crystallographic orientation of a newly created structure (see [define_structure on page 309](#)).

- `deposit` (see [deposit on page 322](#))

Chapter 3: Model Descriptions

Level Set-Based Models

- `etch` (only when using a model for simultaneous etching and deposition; see [etch on page 339](#))
- `set_orientation` (see [set_orientation on page 486](#))

This command is used to set the crystallographic orientation of any region of a structure.

Note:

The parameters `flat_orientation` and `vertical_orientation` of the `define_structure` command are optional (without a default value). Unless specified, no crystallographic orientation will be set for a newly created region.

The parameters `flat_orientation` and `vertical_orientation` of the `etch` and `deposit` commands are mandatory if the crystallographic orientation is required by the model, namely, for the `crystal` deposition model and RFM models that use the `directional_value()` function in the rate formula. Otherwise, these two parameters are optional (without a default value).

Continuously Rotating and Tilted Structure Modeling

You can model etching, deposition, and simultaneous etching and deposition processes for tilted structures with a continuously increasing value of their rotation angle (see [Structure Tilt on page 36](#)).

This feature is available only for rate formula module (RFM) models and reaction models, and is activated by setting `rotation=continuous` in the `define_deposit_machine` command or the `define_etch_machine` command (see [define_deposit_machine on page 195](#) and [define_etch_machine on page 209](#)).

The simulation of processes involving continuously rotating structures is based on the assumption that the structure rotation period is much smaller than the process time.

To simulate such processes, Sentaurus Topography 3D internally recasts the actual problem into an equivalent one, where the structure of interest is not tilted and does not rotate, and the species of the model (see [add_ion_flux on page 157](#), [add_neutral_flux on page 163](#), and [add_source_species on page 181](#)) have angular distributions different from those of the original problem. Each species of the equivalent problem has the same effect on the not tilted and nonrotating structure as the original species has on the tilted and continuously rotating structure, within a margin of error that depends only on the original angular distribution and on the tilt angle. In the following, such a discrepancy is referred to as the *equivalence error* for a species.

The equivalence error of each species is measured by a number in the range [0, 0.5]. You can set the maximum-tolerated equivalence error for all distributions of a model (parameter `maximum_error` of the `define_deposit_machine` or `define_etch_machine` command). The actual equivalence error of each species of the model is written to the log file, when the machine is used with the `deposit` or `etch` command. If the actual equivalence error for any

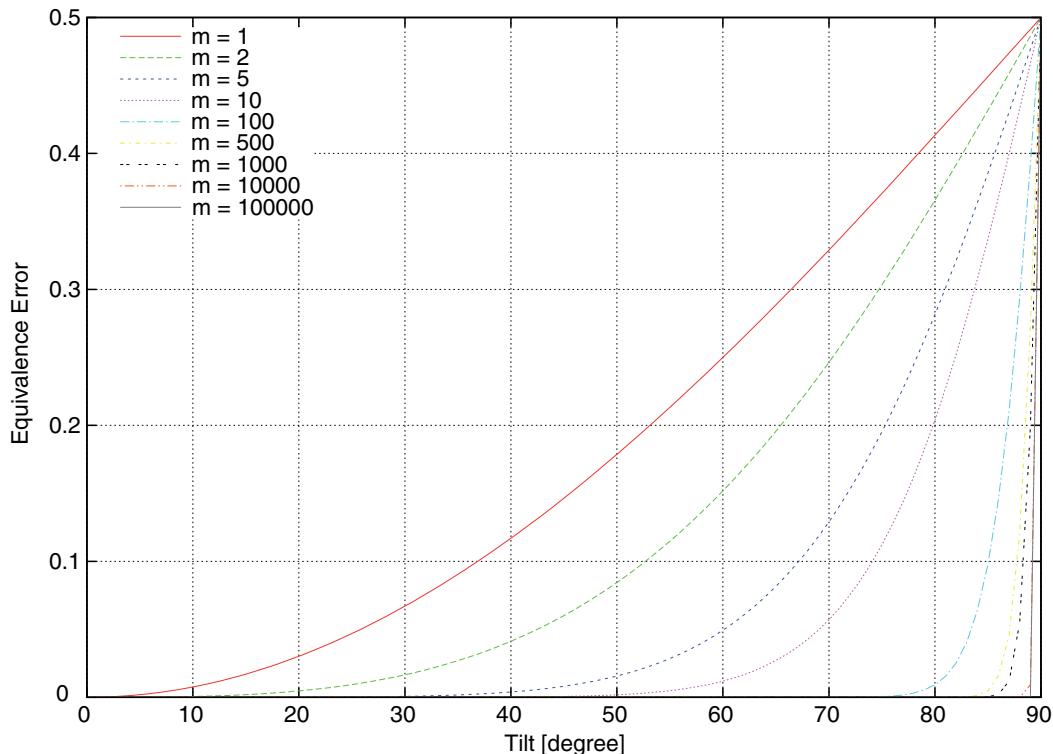
Chapter 3: Model Descriptions

Level Set-Based Models

species of the model is greater than the one specified with the `maximum_error` parameter, an error will be issued.

[Figure 15](#) shows the values of the equivalence error for species having an analytic angular distribution with different values of the exponent m (see [Equation 6 on page 49](#)) for different tilt angles of the structure.

Figure 15 Equivalence error for species having an analytic angular distribution with different values of the exponent for different tilt angles of the structure



As can be seen from [Figure 15](#), the equivalence error increases as the tilt angle increases; whereas, for a given tilt angle, the equivalence error decreases as the angular distribution becomes more focused around the source axis. The trend can be observed for species with angular distributions other than that of [Equation 6](#).

Modeling Chemical-Mechanical Polishing Processes

When using the level-set method, Sentaurus Topography 3D allows you to model chemical-mechanical polishing (CMP) processes at feature scale, using the RFM function `pad_pressure()` (see [Data Available for Rate Calculation on page 512](#)).

Chapter 3: Model Descriptions

Level Set-Based Models

The RFM function `pad_pressure()` returns the pressure produced by a deformable pad in contact with the simulated structure using a contact mechanics-based model [3].

In the implemented model, the pad is characterized by its Young's modulus and Poisson ratio as well as by a Laplacian distribution of the heights of its asperities. The interaction between the pad asperities and the substrate is modeled according to the laws governing Hertz contacts [3].

The Young's modulus of the pad and its Poisson ratio are set with the parameters `pad_young_modulus` and `pad_poisson_ratio` of the `define_etch_machine` command, respectively. Whereas, the standard deviation of the height distribution of the pad asperities is determined from the value of the parameter `pad_roughness` of the `define_etch_machine` command. The pressure applied to the pad is given by the value of the parameter `applied_pressure` of the `define_etch_machine` command (see [define_etch_machine on page 209](#)).

The pressure distribution over the substrate is computed using an iterative method to solve the nonlinear integral equation of the model.

The maximum number of iterations that the solver can perform as well as the solution accuracy below which the solver stops are set with the parameters `pad_pressure_max_iterations` and `pad_pressure_accuracy` of the `etch` command, respectively (see [etch on page 339](#)).

Flux and Flux-Emulating Models

This section presents an overview of the flux and flux-emulating level set-based models available in Sentaurus Topography 3D. Flux and flux-emulating models require flux computations or approximate flux quantities using purely geometric quantities, respectively. See [Deposition Models on page 63](#) and [Etching Models on page 71](#) for detailed descriptions of each model.

Chapter 3: Model Descriptions

Level Set-Based Models

Flux and Flux-Emulating Deposition Models

[Table 4](#) provides an overview of all flux and flux-emulating deposition models available in Sentaurus Topography 3D.

Table 4 Flux and flux-emulating deposition models

Model	simple	pvd	lpcvd	pecvd	hdp	hdp2
Machine parameters	anisotropy curvature rate	(exponent iad) rate	(nad neutral_ exponent) rate sticking	anisotropy (exponent iad) (nad neutral_ exponent) rate sticking	anisotropy (exponent iad) (nad neutral_ exponent) rate redeposition s1, s2 sputter_rate sticking	anisotropy (exponent iad) (nad neutral_ exponent) rate redeposition reflection s1, s2 sputter_ exponent sputter_rate sputter_type sticking
Neutral flux with exponent or NAD			Yes	Yes	Yes	Yes
Ion flux with exponent or IAD		Yes		Yes	Yes	Yes
Reemission and redeposition			Yes	Yes	Yes	Yes
Sputtering and redeposition of sputtered material					Yes	Yes
Diffusive or reflective sputtering						Yes

Chapter 3: Model Descriptions

Level Set-Based Models

Flux and Flux-Emulating Etching and Simultaneous Etching and Deposition Models

[Table 5](#) and [Table 6](#) on page 62 give an overview of etching, and simultaneous etching and deposition, flux and flux-emulating models available in Sentaurus Topography 3D.

Table 5 Flux and flux-emulating etching models

Model	simple	ionmill	rie	rie2	hdp	hdp2	ion_enhanced
Machine parameters			(exponent iad)	(exponent iad) (nad neutral_exponent)	(exponent iad)	(exponent iad) (nad neutral_exponent)	(exponent iad) (nad neutral_exponent)
Material parameters	anisotropy curvature rate	anisotropy rate s1, s2	anisotropy rate	anisotropy rate reflection sticking	anisotropy rate s1, s2	anisotropy rate reflection s1, s2 sputter_rate sticking	anisotropy desorption_rate rate reflection s1, s2 sticking
Isotropic etch rate	Yes	Yes	Yes		Yes		
Neutral flux with exponent or NAD				Yes		Yes	Yes
Uni-directional ion flux	Yes	Yes					
Ion flux with exponent or IAD			Yes	Yes	Yes	Yes	Yes
Sputtering		Yes			Yes	Yes	Yes
Reemission				Yes		Yes	Yes
Ion reflection				Yes		Yes	Yes

Chapter 3: Model Descriptions

Level Set-Based Models

Table 6 Flux and flux-emulating simultaneous etching and deposition models

Model	dry	etchdepo	etchdepo2
Machine parameters	deposit_material (nad neutral_exponent) rate sticking	deposit_material (exponent iad) rate sticking	deposit_material (exponent iad) (nad (neutral_etch_exponent neutral_exponent)) rate sticking
Material parameters	rate s1, s2	rate reflection s1, s2 sputter_type	anisotropy desorption_rate rate reflection s1, s2 sticking
Isotropic etch rate			
Neutral flux with exponent or NAD	Yes		Yes
Unidirectional ion flux			
Ion flux with exponent or IAD		Yes	Yes
Sputtering	Yes	Yes	Yes
Diffusive or reflective sputtering		Yes	Yes
Reemission	Yes		Yes
Redeposition		Yes	
Ion reflection		Yes	Yes

Deposition Models

This section discusses deposition models.

Simple Deposition

Set `model=simple` in the `define_deposit_machine` command to define a simple deposition machine (see [define_deposit_machine on page 195](#)).

The deposition rate is evaluated as the contribution of the following components:

- An isotropic deposition
- A directional deposition
- A curvature-dependent term

The resulting rate at a surface point (x, y, z) can be written as:

$$R(x, y, z) = \begin{cases} R_0[(1 - A) + H(x, y, z)A\vec{v} \cdot \vec{n}(x, y, z)](1 - k\kappa(x, y, z)) & ** \\ 0 & \text{Otherwise} \end{cases} \quad (14)$$

** If (x, y, z) does not lie on the surface enclosing a void or if `exposed_only=false` in the `deposit` command

where:

- A is the anisotropy factor, parameter `anisotropy`.
- $H(x, y, z)$ is a function that accounts for point shadowing in the vertical direction, defined as:

$$H(x, y, z) = \begin{cases} 0 & \text{vertically shadowed region} \\ 1 & \text{vertically unshadowed region} \end{cases} \quad (15)$$

- k is the curvature factor; parameter `curvature`.
- $\kappa(x, y, z)$ is the surface curvature at the point (x, y, z) .
- $\vec{n}(x, y, z)$ is the unit vector normal to the surface at the point of coordinate (x, y, z) .
- R_0 is the deposition rate on a completely flat surface; parameter `rate`.
- \vec{v} is the unit vector normal to the horizontal plane.

The limiting values of $A = 0$ and $A = 1$ correspond to a pure isotropic and a pure directional deposition model.

Chapter 3: Model Descriptions

Level Set-Based Models

The pure isotropic deposition model mimics a chemical vapor deposition (CVD) where reactants, or reactive intermediates, have a very low sticking coefficient, resulting in a uniform concentration of reactants along the surface irrespective of the geometric configuration.

The pure directional deposition model corresponds to a physical vapor deposition (PVD) process where deposition occurs in only one direction. Atoms in the vapor stream that impinge on the surface of the structure are always parallel to the vector \vec{v} . No material deposition occurs in the shadowed regions (see [Equation 14](#) and [Equation 15](#)).

The curvature term has the effect of decreasing the deposition at convex surfaces and increasing the deposition at concave surfaces, smoothing the surface as it evolves.

Physical Vapor Deposition

Set `model=pvd` in the `define_deposit_machine` command to define a physical vapor deposition (PVD) machine (see [define_deposit_machine on page 195](#)).

The PVD model applies to processes where the deposition involves pure physical processes rather than chemical reactions. The particle flux is characterized by a $\cos^m\theta$ angular distribution, where m is set by the parameter `exponent` or a user-defined IAD (the parameter `iad`).

The sticking coefficient is equal to one, that is, each incoming particle deposits on the surface. The deposition rate R_0 on an unshadowed flat surface is set with the parameter `rate`.

The total deposition rate is evaluated as:

$$R = R_0 \Gamma \quad (16)$$

where Γ is the integrated flux on the considered surface point.

Low-Pressure Chemical Vapor Deposition

Set `model=lpcvd` in the `define_deposit_machine` command to define a low-pressure chemical vapor deposition (LPCVD) machine (see [define_deposit_machine on page 195](#)).

In the LPCVD model, the vapor flux of chemical precursors is simulated by neutrals having, by default, an isotropic angular flux distribution. The distribution of the neutral flux can be set either by using the parameter `neutral_exponent` or by specifying a user-defined NAD (the parameter `nad`). The incoming particles can either react on the surface and be deposited, with a probability given by the sticking coefficient (parameter `sticking`), or be reemitted as described in [Neutrals on page 48](#). The reemitted flux is distributed isotropically. The deposition rate R_0 on an unshadowed flat surface is set with the parameter `rate`. The total deposition rate is evaluated as:

$$R = R_0 \Gamma_{\text{neutral}} \quad (17)$$

Chapter 3: Model Descriptions

Level Set-Based Models

with Γ_{neutral} evaluated from [Equation 3 on page 48](#).

Plasma-Enhanced Chemical Vapor Deposition

Set `model=pecvd` in the `define_deposit_machine` command to define a plasma-enhanced chemical vapor deposition (PECVD) machine (see [define_deposit_machine on page 195](#)).

The incoming fluxes to the surface are characterized by two components: a thermally driven LPCVD precursor and ion-induced deposition precursors. The parameter `anisotropy` sets the ratio between the anisotropic ion rate and the total deposition rate R_0 (parameter `rate`). The ion flux can be defined to have a $\cos^m\theta$ angular distribution, where m is set by the parameter `exponent`, or can be a user-defined IAD (the parameter `iad`). By default, the neutral flux has an isotropic distribution. It can be defined to have a $\cos^m\theta$ angular distribution, where m is set by the parameter `neutral_exponent` or can be a user-defined NAD (the parameter `nad`).

The total deposition rate is evaluated as:

$$R = (1 - A)R_0\Gamma_{\text{neutral}} + AR_0\Gamma_{\text{ion}} \quad (18)$$

High-Density Plasma Deposition

Set `model=hdp` in the `define_deposit_machine` command to define a high-density plasma (HDP) deposition machine (see [define_deposit_machine on page 195](#)).

In the HDP deposition model, two simultaneous mechanisms are competing: the deposition and the etching by physical sputtering due to high-energy ions. The neutral flux can either react and be deposited with a probability given by the sticking coefficient σ_j (see [Equation 2 on page 48](#)), the parameter `sticking`, or be reemitted as described in [Neutrals on page 48](#).

Ions are deposited on the surface and, at the same time, can induce a flux of sputtered particles. The sputtering etch rate is highly dependent on the impact angle, and it is described by a yield function (see [Equation 7 on page 51](#)), in which s_1 and s_2 are set with the parameters `s1` and `s2`, respectively.

It is assumed that the sputtered material has an isotropic angular distribution. The amount of redeposition on the surface $\sigma_{\text{redeposition}}$ is set by the parameter `redeposition`.

The total rate on an unshadowed flat surface is given by:

$$R_{\text{flat}} = R_0 - R_{\text{sputter}} \quad (19)$$

where:

- R_0 is the deposition rate on an unshadowed flat surface, without sputtering; parameter `rate`.
- R_{sputter} is the sputtering rate; parameter `sputter_rate`.

Chapter 3: Model Descriptions

Level Set-Based Models

Note:

The model assumes a net deposition, that is, $R_0 > R_{\text{sputter}}$ must hold for flat surfaces.

The total deposited material is the sum of the deposition due to neutrals (direct and indirect flux), plus the deposition due to ions, plus the sputtered material that redeposits, minus the amount of locally sputtered material. Finally, for nonzero anisotropy, the deposition rate is:

$$R = (1 - A)R_0\Gamma_{\text{neutral}} + AR_0\tilde{\Gamma}_{\text{ion}} + R_{\text{sputter}}(\tilde{\Gamma}_{\text{redeposition}} - \tilde{\Gamma}_{\text{sputter}}) \quad (20)$$

where A is the anisotropy coefficient; parameter [anisotropy](#).

For anisotropy equal to zero, the deposition rate is:

$$R = R_0\Gamma_{\text{neutral}} \quad (21)$$

The fluxes Γ_{neutral} , $\tilde{\Gamma}_{\text{redeposition}}$, and $\tilde{\Gamma}_{\text{sputter}}$ are evaluated as described in [Neutrals on page 48](#) and [Ions on page 50](#).

High-Density Plasma 2 Deposition

Set `model=hdp2` in the `define_deposit_machine` command to define a high-density plasma 2 (HDP2) deposition machine (see [define_deposit_machine on page 195](#)).

The HDP2 model extends the HDP model. It takes ion reflection into account and allows you to specify the sputter type and sputter exponent.

For nonzero anisotropy, the deposition rate is:

$$R = (1 - A)R_0\Gamma_{\text{neutral}} + AR_0\tilde{\Gamma}_{\text{ion}} + R_{\text{sputter}}(\tilde{\Gamma}_{\text{redeposition}} - \tilde{\Gamma}_{\text{sputter}}) \quad (22)$$

For anisotropy equal to zero, the deposition rate is:

$$R = R_0\Gamma_{\text{neutral}} \quad (23)$$

[Equation 22](#) has the same structure as [Equation 20](#), but the factors $\tilde{\Gamma}_{\text{ion}}$ and $\tilde{\Gamma}_{\text{redeposition}}$ for the ion flux and the redeposition, respectively, are different.

$\tilde{\Gamma}_{\text{ion}}$ takes ion reflection into account as described in [Reflection on page 52](#). When the parameter `reflection` is set to 0, $\tilde{\Gamma}_{\text{ion}}$ is equal to Γ_{ion} in [Equation 20](#).

$\tilde{\Gamma}_{\text{redeposition}}$ can take diffuse or reflective reemission of the sputtered material into account. By setting `sputter_type=diffuse` and `sputter_exponent=1`, $\tilde{\Gamma}_{\text{redeposition}}$ is equal to $\Gamma_{\text{redeposition}}$ in [Equation 20](#).

As in the HDP model, the HDP2 model considers the process with two competing simultaneous mechanisms: the deposition and the etching by physical sputtering due to high-energy ions. The neutral flux can either react and be deposited with a probability given by the parameter `sticking` or be reemitted. Ions are deposited on the surface and, at the

Chapter 3: Model Descriptions

Level Set-Based Models

same time, can induce a flux of sputtered particles. The angular dependency of the sputter rate is given by the yield function as defined in [Equation 7](#) by the parameters s_1 and s_2 .

The main difference with respect to the HDP model is that the angular distribution of the sputtered material, diffuse or reflective, can be set by users (see [Ions on page 50](#)). The ion reflection index (see [Reflection on page 52](#)) is set with the parameter `reflection`.

The rate evaluation for the HDP2 model is the same as that described in [Equation 20](#).

Atomic Layer Deposition

Set `model=ald` in the `define_deposit_machine` command to define an atomic layer deposition (ALD) machine (see [define_deposit_machine on page 195](#)).

The ALD model is an empirical model that incorporates self-limited growth behavior with a calibration parameter (`conformality`) to control the conformality of the deposited layer. Self-limiting growth is achieved by a rate model that empirically describes the self-limiting saturated growth behavior, which is typical for ALD.

Electrodeposition

Set `model=electrodeposition` in the `define_deposit_machine` command to define an electrodeposition machine (see [define_deposit_machine on page 195](#)).

The model takes into account the following species: plating species, inhibitors, and accelerators. They are assumed to diffuse from the bulk of the plating bath to the surface of the structure under processing, with their concentrations remaining constant at a specified distance above the top of the structure.

Inhibitors and accelerators competitively adsorb on the surface of the structure. Their adsorption kinetics are determined by their local concentration, their adsorption rates, and the local surface curvature. The additives surface interaction model also includes displacement of inhibitors by accelerators and surface saturation effects. The local surface coverage of inhibitors and accelerators, together with the local concentration of the species being plated, determine the deposition rate of the plating species.

You can set up an oscillating overpotential between two values with a given duty cycle to simulate pulse-plating conditions.

The deposition rate R of the plating species at each surface element is determined by the local surface coverage of inhibitors and accelerators, by the local concentration of the species being plated, and by the overpotential, as specified by the following equations [\[4\]](#)[\[5\]](#)[\[6\]](#):

$$R = \frac{\Omega}{zF} \frac{C_{\text{depo}}}{C_{\text{depo}}^{\text{ref}}} J \quad (24)$$

Chapter 3: Model Descriptions

Level Set-Based Models

where:

$$J = \omega[DC J_{\text{on}} + (1 - DC)J_{\text{off}}] \quad (25)$$

$$J_{\text{on}} = J_0(1 - \vartheta_{\text{accel}} - \vartheta_{\text{inhib}})g(\alpha, \eta_{\text{on}}) + J_0^{\text{accel}}\vartheta_{\text{accel}}g(\alpha_{\text{accel}}, \eta_{\text{on}}) + J_0^{\text{inhib}}\vartheta_{\text{inhib}}g(\alpha_{\text{inhib}}, \eta_{\text{on}}) \quad (26)$$

$$J_{\text{off}} = J_0(1 - \vartheta_{\text{accel}} - \vartheta_{\text{inhib}})g(\alpha, \eta_{\text{off}}) + J_0^{\text{accel}}\vartheta_{\text{accel}}g(\alpha_{\text{accel}}, \eta_{\text{off}}) + J_0^{\text{inhib}}\vartheta_{\text{inhib}}g(\alpha_{\text{inhib}}, \eta_{\text{off}}) \quad (27)$$

$$g(a, \eta) = \exp\left(-a \frac{z_\alpha F}{RT} \eta\right) - \exp\left((1 - a) \frac{z_\alpha F}{RT} \eta\right) \quad (28)$$

and:

- Ω is the molar volume of the plating species (parameter `depo_molar_volume`).
- DC is the duty cycle of the electrode overpotential (parameter `duty_cycle`). When its value is 1, pulse plating is deactivated.
- z is the number of electrons transferred during the deposition of an ion of the plating species (parameter `z`).
- $F = 96485.33289$ C/mol is the Faraday constant.
- ω is equal to 1 if the current surface element does not belong to a void and is equal to 0 otherwise.
- C_{depo} is the local concentration of the plating species. Its value is computed by the simulator assuming that the concentration of the plating species remains constant to the value specified by the `depo_bulk_concentration` parameter at the distance from the top of the structure given by the `bulk_distance` parameter. The diffusivity of the plating species can be specified with the `depo_diffusivity` parameter. When `depo_diffusivity` is not specified, the concentration of the plating species is assumed to be equal to its bulk value all over the surface. C_{depo} depends on both the electrode overpotential and its duty cycle.
- $C_{\text{depo}}^{\text{ref}}$ is the reference concentration of the plating species at which the exchange current densities are measured (parameter `depo_reference_concentration`).
- J_0 is the exchange current density for a surface where no additives were adsorbed, when the concentration of the plating species is $C_{\text{depo}}^{\text{ref}}$ (parameter `exchange_current_density` parameter).
- J_0^{accel} is the exchange current density for a surface fully covered by accelerators, when the concentration of the plating species is $C_{\text{depo}}^{\text{ref}}$ (`exchange_current_density_acc` parameter).
- J_0^{inhib} is the exchange current density for a surface fully covered by inhibitors, when the concentration of the plating species is $C_{\text{depo}}^{\text{ref}}$ (`exchange_current_density_inh` parameter).

Chapter 3: Model Descriptions

Level Set-Based Models

- ϑ_{accel} and ϑ_{inhib} are the local accelerator and inhibitor surface coverages, respectively. Their values are computed by the simulator and depend on the following quantities (among others):
 - Accelerator adsorption rate (parameter `acc_adsorption_rate`).
 - Inhibitor adsorption rate (parameter `inh_adsorption_rate`).
 - Accelerator saturation surface concentration (parameter `acc_saturation_surface_concentration`).
 - Inhibitor saturation surface concentration (parameter `inh_saturation_surface_concentration`).
 - Inhibitor displacement rate by accelerators (parameter `inh_displacement_rate`).
 - Local concentration of accelerators. This value is computed assuming that the accelerator concentration remains constant to the value specified by the `acc_bulk_concentration` parameter at the distance from the top of the structure given by the `bulk_distance` parameter. The diffusivity of accelerators can be specified with the `acc_diffusivity` parameter. When `acc_diffusivity` is not specified, the concentration of accelerators is assumed to be equal to its bulk value all over the surface.
 - Local concentration of inhibitors. This value is computed assuming that the inhibitor concentration remains constant to the value specified by the `inh_bulk_concentration` parameter at the distance from the top of the structure given by the `bulk_distance` parameter. The diffusivity of inhibitors can be specified with the `inh_diffusivity` parameter.
 - Local surface curvature.
- α , α_{accel} , and α_{inhib} are the transfer coefficients (parameters `alpha`, `alpha_acc`, and `alpha_inh`, respectively).
- z_α is the number of electrons transferred during the rate-determining elementary reaction (parameter `z_alpha`).
- $R = 8.3144598 \text{ J/mol/K}$ is the gas constant.
- T is the absolute temperature of the electroplating cell (parameter `temperature`).
- η_{on} is the electrode overpotential during the *on* time of the pulse period when pulse plating is either activated or deactivated (parameter `overpotential`).
- η_{off} is the electrode overpotential during the *off* time of the pulse period when pulse plating is activated (parameter `overpotential_off`).

Electroplating Deposition

Set `model=electroplating` in the `define_deposit_machine` command to define an electroplating machine (see [define_deposit_machine on page 195](#)).

This model applies to the electrodeposition of copper. The deposition rate is proportional to the ratio of accelerators to inhibitors on the surface. The accelerators are conserved as the surface evolves. For trenches, accelerators in the trench accumulate while the trench closes, leading to an accelerated surface growth speed in the trench, an effect that is commonly referred to as *superfilling*.

It is assumed that, at the beginning of a simulation, there is a starting coverage of accelerators distributed on the exposed surface. The initial distribution of accelerators either can be uniform, which is the default, or can be set to be linearly dependent on the depth of the surface. The parameter `delta` describes the derivative of the accelerator coverage with respect to the vertical direction. This parameter is used to emulate an unevenly distributed initial accelerator coverage, where the coverage in the depths of a trench is lower than on the top surface.

In each time step, the surface evolves; whereas, the deposition rate R at each surface element is proportional to the ratio of accelerator coverage C_{accel} and inhibitor coverage C_{inhib} at the surface element:

$$R \propto R_0 \frac{C_{\text{accel}}}{C_{\text{inhib}}} \quad (29)$$

where:

- R_0 is the deposition rate (parameter `rate`).
- C_{accel} is the accelerator coverage on the surface. The total number of accelerators is conserved over time.
- C_{inhib} is the inhibitor coverage on the surface. It is assumed that inhibitors are constantly replenished from the electrolyte solution and, therefore, inhibitors are distributed uniformly in space, and the coverage remains constant in time.

Crystallographic Orientation–Dependent Deposition

Set `model=crystal` in the `define_deposit_machine` command to define a crystallographic orientation–dependent deposition machine (see [define_deposit_machine on page 195](#)).

The deposition rate is determined by the orientation of the exposed surface with respect to the lattice of its bulk material, which is assumed to be cubic. More precisely, the deposition rate at any point of the exposed surface is computed as the spherical barycentric interpolation [7], along its normal direction, of the deposition rates along the $\langle 100 \rangle$, $\langle 110 \rangle$

Chapter 3: Model Descriptions

Level Set-Based Models

and <111> crystallographic directions (parameters `rate_100`, `rate_110`, and `rate_111` of the `define_deposit_machine` command, respectively).

The crystallographic orientation of the deposited material must be defined with the parameters `flat_orientation` and `vertical_orientation` of the `deposit` command. The model does not require the other regions of the structure to have crystallographic orientations defined. See [Orientation-Dependent Models on page 56](#), [Setting Crystallographic Orientations on page 56](#), and [set_orientation on page 486](#).

Moreover, unlike the other deposition models, this model makes it possible to simulate selective deposition processes, as the list of materials on which deposition will occur can be specified using the `selective_materials` parameter of the `define_deposit_machine` command.

Etching Models

This section discusses etching models.

Simple Etching

Set `model=simple` in the `define_etch_machine` command to define a simple etching machine (see [define_etch_machine on page 209](#)).

The etch rate is evaluated as the contribution of the following different components:

- An isotropic etch
- A directional etch
- A curvature-dependent term

The resulting rate at the surface point (x, y, z) for the material m can be written as:

$$R_m(x, y, z) = \begin{cases} R_{0m}[(1 - A_m) + H(x, y, z)A_m \hat{v} \cdot \hat{n}(x, y, z)](1 + k_m \kappa(x, y, z)) & ** \\ 0 & \text{Otherwise} \end{cases} \quad (30)$$

** If (x, y, z) does not lie on the surface enclosing a void or if `exposed_only=false` in the `etch` command

where:

- A_m is the anisotropy factor of material m (parameter `anisotropy`).
- $H(x, y, z)$ is a function that accounts for shadowing, defined in [Equation 15](#).
- k_m is the curvature factor for material m (parameter `curvature`).

Chapter 3: Model Descriptions

Level Set-Based Models

- $\kappa(x, y, z)$ is the surface curvature in the point (x, y, z) .
- $\hat{n}(x, y, z)$ is the unit vector normal to the surface at the point of coordinate (x, y, z) .
- R_{0m} is the etching rate on a completely flat surface for material m (parameter `rate`).
- \hat{v} is the unit vector normal to the horizontal plane.

Plasma-assisted processes from reactive or nonreactive ions impinging upon the surface of the structure usually result in a directional etching. The directional component is evaluated only for unshadowed regions (see [Equation 15](#) and [Equation 30](#)).

The curvature term has the effect of increasing etching at convex surfaces and decreasing etching at concave surfaces, smoothing the surface as it evolves.

Ion-Milling

Set `model=ionmill` in the `define_etch_machine` command to define an ion-mill etching machine (see [define_etch_machine on page 209](#)).

In this model, etching is due to high-energy (unidirectional) ions impacting the surface. The sputtering etch rate is evaluated using the yield function defined in [Equation 8](#):

$$R_m(x, y, z) = R_{0m}[(1 - A_m) + H(x, y, z)A_m\gamma_m(\theta_{im})] \quad (31)$$

where:

- A_m is the anisotropy factor of material m (parameter `anisotropy`).
- $\gamma_m(\theta_{im})$ is the value of the yield function for the material m and the ion impact angle θ_{im} .
- $H(x, y, z)$ is the function that accounts for the surface point shadowing as defined in [Equation 15](#).
- R_{0m} is the etching rate on a completely flat surface for material m (parameter `rate` in the `add_material` command).

Reactive Ion Etching

Set `model=rie` in the `define_etch_machine` command to define a reactive ion etch (RIE) machine (see [define_etch_machine on page 209](#)).

The etching process has two contributions: an isotropic etch and an anisotropic etch. The isotropic etch emulates the wet (chemical) etch process, while the anisotropic etch emulates the plasma etch process.

The anisotropic etch rate is proportional to the incoming ion flux. The ion flux Γ_{ion} either has an angular distribution $\cos^m\theta$, where m is set by the parameter `exponent`, or is a user-defined IAD (the parameter `iad`).

Chapter 3: Model Descriptions

Level Set-Based Models

The etch rate R_{0m} on an unshadowed flat surface for the material m is set with the parameter `rate`.

The total etch rate is the sum of these two etch rates:

$$R_m = (1 - A_m)R_{0m} + A_m R_{0m}\Gamma_{\text{ion}} \quad (32)$$

where A is the anisotropy coefficient; parameter `anisotropy`.

Reactive Ion Etching 2

Set `model=rie2` in the `define_etch_machine` command to define a reactive ion etch 2 (RIE2) machine (see [define_etch_machine on page 209](#)).

The RIE2 model extends the RIE model by improving both the neutral and the ion modeling. The RIE2 model considers the etching process as the result of the following contributions:

- Chemical reactions induced by radicals
- Desorptions from the surface induced by ions

The evaluation of the neutral flux Γ_{neutral} is performed considering the structure shadowing and the multiple reemissions as described in [Neutrals on page 48](#) (see [Equation 3](#)). The sticking coefficient is material dependent and must be set during the material definition with the parameter `sticking`. By default, the neutral flux has an isotropic distribution. It can be defined to have a $\cos^m\theta$ angular distribution, where m is set by the parameter `neutral_exponent` or can be a user-defined NAD (the parameter `nad`).

Ion reflection can be included by setting the parameter `reflection` as described in [Reflection on page 52](#). The ion etch rate is proportional to the incoming ion flux. The ion flux has an angular distribution $\cos^m\theta$, where m is set by the parameter `exponent`, or is a user-defined IAD (the parameter `iad`).

The etch rate R_{0m} on an unshadowed flat surface for the material m is set with the parameter `rate`.

The etch contribution due to neutrals and ions is:

$$R_m = (1 - A_m)R_{0m}\Gamma_{\text{neutral}} + A_m R_{0m}\Gamma_{\text{ion}} \quad (33)$$

where A_m is the anisotropy coefficient for material m (parameter `anisotropy`). The flux Γ_{neutral} is evaluated as described in [Neutrals on page 48](#).

High-Density Plasma Etching

Set `model=hdp` in the `define_etch_machine` command to define a high-density plasma (HDP) etching machine (see [define_etch_machine on page 209](#)).

As in the RIE process, the etch rate has two contributions: an isotropic etch and an anisotropic etch. The total etch rate is the sum of these two contributions; the parameter

Chapter 3: Model Descriptions

Level Set-Based Models

anisotropy defines the ratio between the anisotropic etch rate and the total rate. The isotropic etch emulates the wet (chemical) etch process. The anisotropic etch emulates the plasma etch process that, contrary to the RIE model, is due mainly to sputtering. The sputtering yield function is defined as in [Equation 7](#). The anisotropic etch models the plasma etch process where the anisotropic etch rate is proportional to the incoming ion flux. The isotropic part consists of two terms: an isotropic etch term and a term that is proportional to the incoming ion flux. The latter is only active if the surface element is shadowed by another surface element in the vertical direction; whereas, the former is only active if, in the vertical direction, there is no shadowing (see [Equation 15](#)).

The total etch rate is:

$$R_m = (1 - A_m)R_{0m}[H(x, y, z) + (1 - H(x, y, z))\Gamma_{\text{ion}}] + A_mR_{0m}\gamma_m(\theta_{\text{im}})\Gamma_{\text{ion}}H(x, y, z) \quad (34)$$

where:

- A_m is the anisotropy factor of material m (parameter `anisotropy`).
- R_{0m} is the etching rate on a completely flat surface for the material m (parameter `rate` in the `add_material` command).
- $H(x, y, z)$ is the function that accounts for the surface point shadowing as defined in [Equation 15](#).

The ion flux has an angular distribution $\cos^m\theta$, where m is set by the parameter `exponent`, or is a user-defined IAD (the parameter `iad`).

High-Density Plasma 2 Etching

Set `model=hdp2` in the `define_etch_machine` command to define a high-density plasma 2 (HDP2) etching machine (see [define_etch_machine on page 209](#)).

The HDP2 model extends the HDP model by implementing the features already discussed in [Reactive Ion Etching 2 on page 73](#) for the RIE2 model: neutral flux calculation including reemissions and ion reflections.

The HDP2 model considers etching as the result of the following contributions:

- Chemical reactions induced by radicals
- Desorptions from the surface induced by ions
- Sputtering

The total rate is given by:

$$R_m = (1 - A_m)R_{0m}\Gamma_{\text{neutral}} + A_mR_{0m}\Gamma_{\text{ion}} + A_mR_{\text{sputter}, m}\gamma_m(\theta_{\text{im}})\Gamma_{\text{ion}} \quad (35)$$

Chapter 3: Model Descriptions

Level Set-Based Models

where:

- A_m is the anisotropy coefficient for the material m (parameter `anisotropy`).
- $R_{\text{sputter},m}$ is the sputtering rate for the material m (parameter `sputter_rate`).
- R_0 is the etch rate on a flat unshadowed surface when there is no sputtering (parameter `rate`).
- Γ_{neutral} is the neutral flux that takes shadowing and multiple reemissions into account as described in [Neutrals on page 48](#) (see [Equation 3](#)). The sticking coefficient σ_j is material dependent and must be set during the material definition with the parameter `sticking`.

Ion-Enhanced Etching

Set `model=ion_enhanced` in the `define_etch_machine` command to define an ion-enhanced etching machine. (see [define_etch_machine on page 209](#))

The ion-enhanced etching model considers etching as the result of the combined effects of neutrals and ions. Neutrals are deposited on the surface (neutral flux Γ_{neutral}). Then, either these neutrals are removed by ions or they dissociate from the surface through thermal desorption. Where the layer of neutrals is removed, chemical etching occurs.

The etching due to ions is modeled as a sputter etching process, where the sputter yield function is defined as in [Equation 7](#).

In this model, the total etch rate is not considered as the linear summation of neutrals and ions etch rates, but:

$$R_m = \frac{1}{\frac{1}{(1 - A_m) R_{0m} \Gamma_{\text{neutral}}} + \frac{1}{A_m^2 R_{0m} \gamma_m(\theta_{im}) \Gamma_{\text{ion}} + R_{\text{des},m}}} \quad (36)$$

where:

- $R_{\text{des},m}$ is the thermal desorption rate for material m (parameter `desorption_rate` in the `add_material` command).
- A_m is the anisotropy of material m (parameter `anisotropy` in the `add_material` command).
- R_{0m} is the etch rate on a flat surface of material m (parameter `rate` in the `add_material` command).

This model has the following properties:

- If one of the fluxes is zero, then the overall etch rate is zero since both neutrals and ions are required to etch the material.

Chapter 3: Model Descriptions

Level Set-Based Models

- The etch rate becomes saturated when one component becomes too large relative to the other.

The neutral flux is evaluated as in the RIE2 and HDP2 models by considering the shadowing and the reemission probability on the surface. Ion reflections to the sidewalls can also be considered by setting the parameter `reflection`.

Wet Etching

Set `model=wet` in the `define_etch_machine` command to define a wet etching machine (see [define_etch_machine on page 209](#)).

This model assumes that the etchant concentration is constant at a specified distance above the top of the structure, and that the transport of the etchant to the surface of the structure occurs by diffusion. At the surface of the structure, the model assumes that the etchant is consumed by a first-order reaction.

Dry Etching

Set `model=dry` in the `define_etch_machine` command to define a dry etching machine (see [define_etch_machine on page 209](#)).

This model considers the combination of sputter etching and deposition processes. The etch process is induced by unidirectional high-energy ions that sputter the material of the structure (see [Ion-Milling on page 72](#)). At the same time, an LPCVD process is considered (see [Low-Pressure Chemical Vapor Deposition on page 64](#)).

The total deposition etch rate is evaluated as:

$$R_m = R_{\text{deposition}} \Gamma_{\text{neutral}} - H(x, y, z) R_{\text{ion-mill}, m} \gamma_m(\theta_{\text{im}}) \quad (37)$$

where:

- $R_{\text{deposition}}$ is the deposition rate on a completely flat surface (parameter `rate` in the `define_etch_machine` command).
- $R_{\text{ion-mill}, m}$ is the etching rate on a completely flat surface for material m (parameter `rate` in the `add_material` command).
- $H(x, y, z)$ is the function that accounts for the surface point shadowing as defined in [Equation 15 on page 63](#).

Crystallographic Orientation–Dependent Etching

Set `model=crystal` in the `define_etch_machine` command to define a crystallographic orientation–dependent etching machine (see [define_etch_machine on page 209](#)).

Chapter 3: Model Descriptions

Level Set-Based Models

The etching rate is determined by the orientation of the exposed surface with respect to the lattice of its bulk material, which is assumed to be cubic. More precisely, the etching rate at any point of the exposed surface is computed as the spherical barycentric interpolation [7], along its normal direction, of the deposition rates along the <100>, <110> and <111> crystallographic directions (parameters `rate_100`, `rate_110`, and `rate_111` of the `define_etch_machine` command, respectively).

Compared to other built-in etching models, only one material (specified with the parameter `etchable_material` of the `define_etch_machine` command) can be etched using this model.

The crystallographic orientation of the material to be etched is expected to be set. If there are several regions with the material to be etched, their crystallographic orientations must be all set and they must be identical. See [Orientation-Dependent Models on page 56](#), [Setting Crystallographic Orientations on page 56](#), and [set_orientation on page 486](#).

Simultaneous Etching and Deposition

Set `model=etchdepo` in the `define_etch_machine` command to define a simultaneous etching and deposition machine (see [define_etch_machine on page 209](#)).

This model considers the etching process due to sputtering induced by high-energy ions and the redeposition of this sputtered material. At the bottom of the trench, the polymer is usually removed by ion bombardment; while on the sidewalls, it accumulates forming a thin layer. This layer can inhibit the etching process.

The ion flux has an angular distribution $\cos^m\theta$, where m is set by the parameter `exponent`, or is a user-defined IAD (the parameter `iad`).

The deposition process is simulated considering that the incoming material flux $\Gamma_{\text{sputter deposition},i}$ at point i is the sum of the integrated sputtered flux $\Gamma_{\text{sputter},j}$, which is sputtered from point j and arriving at point i , plus the reemitted flux from all the visible points j :

$$\Gamma_{\text{sputter deposition},i} = \sum_{j \neq i} g_{ij} \Gamma_{\text{sputter},j} + (1 - \sigma) \sum_{j \neq i} g_{ij} \Gamma_{\text{sputter deposition},j} \quad (38)$$

Here, $\Gamma_{\text{sputter},j}$ is induced by the ion flux Γ_{ion} (see [Equation 8](#)).

Ion reflection at the sidewalls also can be considered by setting the parameter `reflection`. The angular distribution of sputtered material can be set with the parameters `sputter_type` and `exponent`.

Finally, the total rate is evaluated as:

$$R_m = R_{\text{deposition}} \Gamma_{\text{sputter deposition}} - R_{\text{etch, } m} \Gamma_{\text{sputter}} \quad (39)$$

Chapter 3: Model Descriptions

Level Set-Based Models

where:

- $R_{\text{deposition}}$ is the deposition rate on a completely flat surface (parameter `rate` in the `define_etch_machine` command).
- $R_{\text{etch},m}$ is the etching rate on a completely flat surface for material m (parameter `rate` in the `add_material` command).
- $\Gamma_{\text{sputter deposition}}$ is the indirect flux of sputtered material, as described in [Equation 38](#).

A positive total rate R_m indicates a net deposition; a negative total rate indicates a net etching.

Note:

In contrast to other models, in the `etchdepo` model, the sputter flux emitted from a point is set to zero if the sputter rate for the corresponding material is exactly zero. The sputter rate for a material can be zero either because it has been specified explicitly with the `add_material` command or because the properties of the material have not been specified with the `add_material` command.

Simultaneous Etching and Deposition 2

Set `model=etchdepo2` in the `define_etch_machine` command to define a simultaneous etching and deposition 2 machine (see [define_etch_machine on page 209](#)).

This model considers the combined effect of etching in the ion-enhanced regime and the deposition due to polymerization. As in the ion-enhanced model, the ion flux removes the layer of atoms formed by the chemical reactions and damages the surface that favors chemical etching. The polymer deposits on the surface as in an LPCVD process and is removed with the ion-milling mechanism.

The etching due to ions is modeled as a sputter etching process, where the sputter yield function is defined as in [Equation 7](#).

The polymer also dissociates thermally, with a rate specified by the parameter `desorption_rate`. At the bottom of the trench, the polymer is usually removed by ion bombardment; while on the sidewalls, it accumulates forming a thin layer. This layer can inhibit the etching process. The etch rates and fluxes are evaluated as in the ion-enhanced model; while the polymer deposition rate is evaluated as in an LPCVD process.

The total deposition rate for all materials except the deposited material is:

$$R_m = R_{\text{deposition}} \Gamma_{\text{neutral depo}} - \frac{1}{(1 - A_m) R_{\text{etch},m} \Gamma_{\text{neutral etch}}} + \frac{1}{A_m R_{\text{etch},m} \gamma_m(\theta_{\text{im}}) \Gamma_{\text{ion}} + R_{\text{des},m}} \quad (40)$$

The total deposition rate for the deposited material is:

$$R_m = R_{\text{deposition}} \Gamma_{\text{neutral depo}} - R_{\text{etch},m} \gamma_m(\theta_{\text{im}}) \Gamma_{\text{ion}} \quad (41)$$

Chapter 3: Model Descriptions

Spin-on-Glass Deposition Model

where:

- $R_{des,m}$ is the thermal desorption rate for material m (parameter `desorption_rate` in the `add_material` command).
- A_m is the anisotropy for material m (parameter `anisotropy` in the `add_material` command).
- $\Gamma_{neutral\ etch}$ is the total flux of neutrals, computed with a material-dependent sticking coefficient (specified with the `add_material` command). Its angular distribution can be specified with either the parameter `neutral_etch_exponent` or the `define_nad [species=neutral]` command.
- $\Gamma_{neutral\ depo}$ is the total flux of neutrals, computed with a material-independent sticking coefficient (specified with the `define_etch_machine` command). Its angular distribution can be specified with either the parameter `neutral_exponent` or with the `define_nad [species=neutral]` command.
- Γ_{ion} is the ion flux.

Spin-on-Glass Deposition Model

The `spin_on` model is used to determine the spin-on material profile over a given topographic substrate. The profile evolution of the film is computed by solving the Navier–Stokes equations under the lubrication theory for Newtonian fluids [8][9][10].

Given the initial thickness of the spin-on material (parameter `initial_thickness`), the profile evolution of the film is computed taking into account the capillarity and centrifugal forces acting on it. In addition, the film is supposed to evaporate during the process (parameter `evaporation_rate`). The spin-on material is characterized by its viscosity (parameter `viscosity`), its density (parameter `density`), and its surface tension with the surrounding fluid (parameter `surface_tension`).

The centrifugal force acting on the simulated structure is supposed to be constant over the simulation domain, and it is determined by the angular velocity of the wafer (parameter `angular_velocity`), the position of the simulated structure on the wafer (parameters `angular_position` and `radial_distance`), and the film density.

Reaction Models

Reaction models provide a very powerful and flexible means of describing the physical and chemical effects that occur on a wafer surface due to interaction with plasma species. Basic effects such as adsorption, reemission, sputtering, and chemical etching are described by surface reactions, using a syntax that is similar to that used for chemical reactions. Each reaction defines the interaction of a single gas-phase species with one or more surface

Chapter 3: Model Descriptions

Reaction Models

species, along with the products that might result from this interaction. Reaction products can be emitted from the surface and eventually interact with other parts of the wafer surface, or they can be defined as being volatile, meaning that they have no effect in the simulation.

In the definition of a reaction model, unlike the definition of a level set-based model, there is no distinction between ion and neutral species. Indeed, a reaction model consists of a set of source species and a set of reactions. However, different properties (for example, angular distributions) can be assigned to each source species to effectively model both the neutral and the charged particles present in the reactor.

In addition, whereas for the level set-based models, the fluxes of the source species are always assumed to be normalized, reaction models require the specification of the absolute fluxes of the source species.

A reaction model is defined using the `define_model` command (see [define_model on page 251](#)). A source species is defined using the `add_source_species` command (see [add_source_species on page 181](#)); whereas, reactions are defined using the `add_reaction` command (see [add_reaction on page 164](#)).

The angular distribution of each source species can have an arbitrary shape and must be specified with the `define_species_distribution` command (see [define_species_distribution on page 290](#)). However, the angular distributions of the source species are not part of any model. Only when defining a machine with the `define_etch_machine` command, a reaction model is bound to the angular distributions of its source species (see [define_etch_machine on page 209](#)).

As detailed in [add_reaction on page 164](#), a reaction states a rule to transform a set of reactants into a set of products. Each reaction must contain one reactant species coming from the reactor, which must be either a source species or a species produced as a result of another reaction.

When the reactants become available, reactions are executed with a probability specified with the `add_reaction_properties` command (see [add_reaction_properties on page 168](#)). In the general case, the reaction probability depends on the angle between the traveling direction of the incoming species and the normal to the surface where the incoming species collides with the structure as well as on the energy of the incoming particle. The `define_probability` command allows you to define an energy-dependent and angle-dependent reaction probability (see [define_probability on page 259](#)). Since the execution of a reaction depends on a random decision, it is possible that no reaction occurs even if the reactants are available. When the reactants are available, but no reaction is actually executed, the incoming species is either stopped or reemitted in the reactor, as specified by the `default_event` parameter of the `define_species_properties` command (see [define_species_properties on page 306](#)).

Feedback of Surface Reaction Products

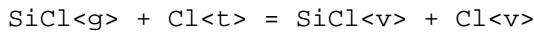
During a topography-processing step, a significant amount of surface reaction products is produced, which desorb from the wafer and interact with the plasma source [11]. In the following, these surface reaction products are referred to as *byproducts*.

A reaction model can include different types of plasma feedback interaction:

- Depletion of plasma radicals: The feedback of byproducts into the plasma leads to a consumption of plasma radicals (*loading effect*), which might result, for example, in a local decrease of the etch rate.
- Enhancement of plasma radicals: The feedback of byproducts triggers the emission of new plasma particles, which might lead, for example, to a local increase of the etch rate.

You can model both feedback effects, depletion and enhancement, in reaction models by defining feedback reactions at the plasma source (the *top plane* $\langle t \rangle$), which is located at a user-defined height above the substrate.

To define a feedback reaction between a surface reaction product (for example, $\text{SiCl}\langle g \rangle$) and a plasma bulk species in the top plane (for example, $\text{Cl}\langle t \rangle$), you can define the following reaction:



This reaction corresponds to an annihilation of the plasma radical Cl , resulting in a decrease of the Cl concentration in the plasma bulk.

An enhancement of the particle flux emitted by the plasma source can be achieved by defining the following reaction:



where $\langle t \rangle$ is a placeholder for any plasma bulk species. This reaction represents a generic reaction between an incoming surface reaction product ($\text{SiCl}\langle g \rangle$) and a plasma bulk species (represented by the placeholder $\langle t \rangle$) at the top plane.

See [add_reaction on page 164](#), [add_reaction_properties on page 168](#), and [define_etch_machine on page 209](#) for details on the usage of the plasma feedback model.

Crystalline Materials, Nucleation, and Grain Growth

This section discusses crystalline materials, nucleation, and grain growth.

Defining Crystalline Materials

Crystalline and polycrystalline materials such as polysilicon are commonly used in modern industrial applications. While amorphous materials have a disordered atomic structure, crystals possess a highly symmetric and repeating pattern. In Sentaurus Topography 3D PMC, the only supported crystal structure is the diamond cubic lattice, which is the most common crystal structure in topography simulation applications.

Defining Monocrystalline Materials

To define a crystalline material, its lattice type, lattice constant, and orientation in space must be specified:

```
set_material_properties material=<c> type=crystalline  
  crystal_type=diamond flat_orientation=<v> \  
    vertical_orientation=<v> [lattice_constant=<n>]
```

The parameters `flat_orientation` and `vertical_orientation` are needed to define the orientation of the crystal lattice in space:

- `flat_orientation` sets the Miller indices of the crystal plane, which is perpendicular to the y-axis (wafer flat) of the wafer coordinate system.
- `vertical_orientation` sets the Miller indices of the crystal plane, which is perpendicular to the z-axis (vertical axis) of the wafer coordinate system.

Example 1 Define a structure consisting of crystalline silicon:

```
define_structure material=Silicon point_min={-1 -1 0} \  
  point_max={3 1 10}  
  
set_material_properties material=Silicon type=crystalline \  
  crystal_type=diamond \  
  flat_orientation={1 1 0} vertical_orientation={0 0 1}
```

Example 2 Set the crystal properties only for a subregion of the structure:

```
define_structure region=region1 material=Silicon \  
  point_min={-1 -1 0} point_max={3 1 10}  
  
set_material_properties region=region1 type=crystalline \  
  crystal_type=diamond \  
  flat_orientation={1 1 0} vertical_orientation={0 0 1}
```

Defining Polycrystalline Materials

Besides pure crystalline materials, polycrystalline materials can also be modeled, consisting of many small crystals (called crystallites or grains) of different orientations. You can specify the average size of the grains (parameter `average_grain_size`) as well as the preferred orientation, which is defined by a symmetry axis (`preferred_vertical_orientation`) and an angular spread around this axis (`vertical_orientation_spread`). The crystal grains are created randomly from the angular distribution around the preferred vertical orientation.

Example Create a polycrystalline region consisting of grains with an average size of 10 nm and a random crystal orientation around the preferred vertical direction:

```
set_material_properties region=region1 type=polycrystalline \
    crystal_type=diamond \
    preferred_vertical_orientation={0 0 1} \
    vertical_orientation_spread=10 \
    average_grain_size=10<nm>
```

For details about the syntax, see [set_material_properties on page 482](#).

Note:

For polycrystalline materials:

- The command `set_material_properties` converts the material properties of the structure at the time when it is called (similarly to the `filter_structure` command). It does not define material properties for materials that are created at a later time, for example, during a PMC deposition simulation (see [Adsorption and Deposition Reactions on page 86](#) for how to specify the crystal properties of PMC reaction products).
- If periodic boundary conditions are specified for the structure, then the generated grains are also periodic.

Loading and Saving Crystalline Materials From Files

When saving a structure to a TDR file, the morphology of the materials (amorphous, crystalline, or polycrystalline) and the crystal orientations are saved to the file as well. This allows you to reload the crystal properties when reading in the file.

Example 1 Save the crystal properties of silicon to a TDR file:

```
define_structure material=Silicon point_min={ -1 -1 0 } \
    point_max={ 3 1 10 }

set_material_properties material=Silicon type=crystalline \
    crystal_type=diamond \
    flat_orientation={1 1 0} vertical_orientation={0 0 1}

save file=file1.tdr
```

Chapter 3: Model Descriptions

Crystalline Materials, Nucleation, and Grain Growth

Example 2 Load the crystal properties of silicon from the TDR file saved in Example 1:

```
define_structure file=file1.tdr
```

When saving a boundary structure, extracted from the PMC structure using the DC or VBE method, you can choose to save different crystal orientations as different regions. For details, see [save on page 461](#).

Note:

Crystal properties cannot be saved to the GC format (`save type=gc`).

Plotting Crystal Orientations

This section discusses how to plot crystal orientations.

Boundary Structure (Using DC or VBE Method)

The easiest way to visualize the crystalline properties of a structure is to save the structure with `save type=dc` or `save type=vbe`. You can also specify `grain_regions=true` to save each crystal orientation as a separate region in the output (see [Figure 16 \(left\)](#)). This makes it possible to color each crystal orientation independently:

```
save file=file1.tdr type=dc dc_version=2 grain_regions=true
```

or:

```
save file=file1.tdr type=vbe grain_regions=true
```

Note:

When using the VBE method with `grain_regions=true`, only up to 240 different crystal orientations are supported (this corresponds to 240 grains in a structure).

Volume Fraction Output

Another way to visualize crystal orientations is to write out the volume fractions:

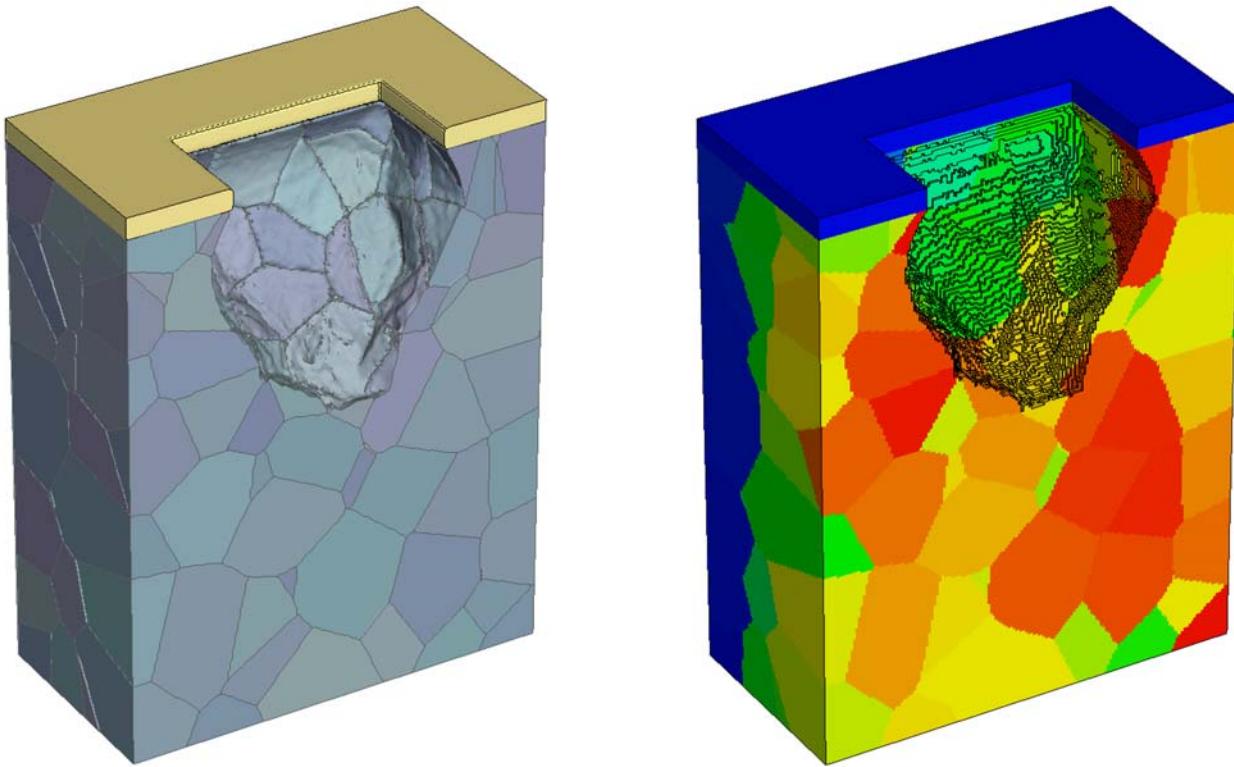
```
save file=file1.tdr type=volume_fractions
```

This command creates a TDR file containing a dataset `solid_number` for each material in the structure. The solid number is a unique ID for each crystal orientation contained in the structure (`solid_number=0` is reserved for amorphous material). When plotting the solid number datasets in Sentaurus Visual, each crystal orientation will be shown in a different color (see [Figure 16 \(right\)](#)).

Chapter 3: Model Descriptions

Crystalline Materials, Nucleation, and Grain Growth

Figure 16 (Left) Boundary structure containing polycrystalline material, with different grain orientations represented by different regions and (right) volume fraction plot (colors correspond to the solid number dataset)



Crystal Orientation–Dependent Reactions

This section discusses reactions that are dependent on the crystal orientation.

Etching and Sputtering Reactions

In general, the etch reactivity on crystalline surfaces is affected by the crystal orientation of the exposed surface plane. To model crystal orientation–dependent effects, you can define crystal orientation–dependent reaction probabilities in the `add_reaction_properties` command.

Due to the symmetry properties of the crystal, it suffices to define the reaction probabilities on the main crystal planes. For example, for a diamond crystal, you must specify the reaction probabilities only on three independent planes (1 0 0), (1 1 0), and (1 1 1) (`parameters diamond_p_100, diamond_p_110, and diamond_p_111`). From those values, the reaction probabilities on all other surface planes (for example, (1 1 3), (2 1 3) ...) are computed by interpolation.

Chapter 3: Model Descriptions

Crystalline Materials, Nucleation, and Grain Growth

Example Consider a PMC etch model for crystalline silicon. The etch probability depends on the crystal orientation of the exposed surface and is specified for the three main crystal planes:

```
define_structure material=Silicon point_min={-1 -1 0} \
    point_max={3 1 10}
set_material_properties material=Silicon type=crystalline \
    crystal_type=diamond \
    flat_orientation={1 1 0} vertical_orientation={0 0 1}
.

.

.

define_model name=m description="Etching crystalline Silicon"
add_source_species model=m name=F
add_reaction model=m name=R expression="F<g> + Silicon<s> = Silicon<v>"
finalize_model model=m
.

.

.

add_reaction_properties reaction=R p=1 diamond_p_100=1 \
    diamond_p_110=0.7 diamond_p_111=0.1
```

Notes:

- In addition to the crystalline probabilities `diamond_p_i j k`, you must define the probability for amorphous surfaces `p`, since the structure might, in general, contain amorphous material as well.
- If more than one reaction is defined on the same surface material, then the sum of all probabilities must be less than or equal to 1 on each crystal plane:

$$\sum_{r \in \{\text{reactions with the same surface reactant}\}} p_{ijk, r} \leq 1 \quad \forall ijk \in \{(100), (110), (111)\}$$

- If the sum of the probabilities is less than one, then there exists a probability that no reaction is executed upon a surface collision. In that case, the default event is executed, which is defined in the `define_species_properties` command (see [define_species_properties on page 306](#)).

For details about the syntax, see [add_reaction_properties on page 168](#).

Adsorption and Deposition Reactions

This section describes adsorption and deposition of crystalline material on crystalline surfaces. For adsorption or deposition on amorphous substrates, see [Nucleation and Grain Growth on page 88](#).

Chapter 3: Model Descriptions

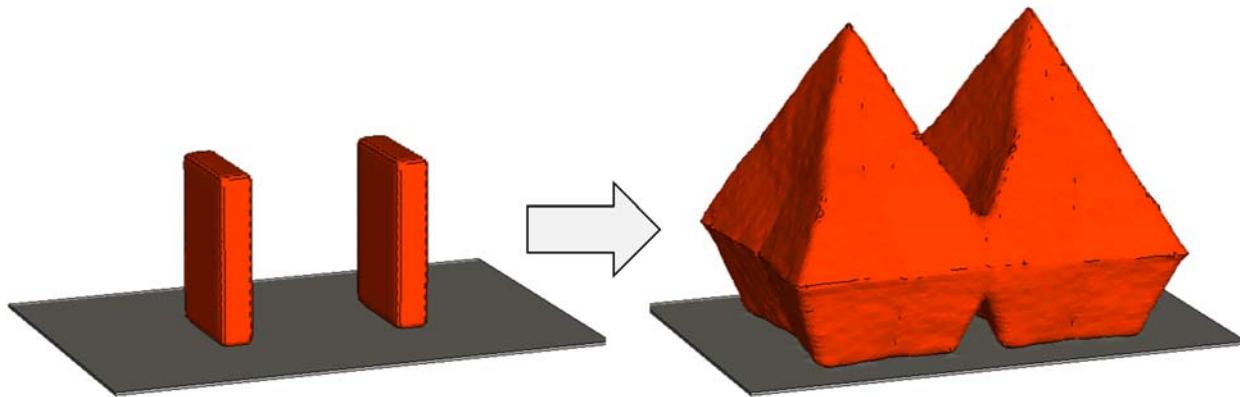
Crystalline Materials, Nucleation, and Grain Growth

To simulate the adsorption or deposition of material on a crystalline surface, you must specify whether the deposited layer is amorphous or crystalline by using the parameter `product_morphology` in the `add_reaction_properties` command:

- To deposit amorphous material on a crystalline surface, set `product_morphology=amorphous`.
- To deposit crystalline material on a crystalline surface, adopting the same crystal orientation as the surface, set `product_morphology=same_as_surface_reactant`.

In analogy to etching and sputtering reactions, the probability for adsorption or deposition depends on the crystal orientation of the exposed surface plane. The resulting anisotropic deposition rates can lead to characteristic crystal shapes with sharp edges and facets (see [Figure 17](#)). To model orientation-dependent adsorption or deposition rates, you must provide the reaction probabilities on the main crystal planes (parameters `diamond_p_100`, `diamond_p_110`, and `diamond_p_111`).

Figure 17 Deposition on crystalline pillars: the resulting shape shows characteristic sharp edges and facets



Example 1 Deposit crystalline PolySi on a (polycrystalline) PolySi surface, taking over the crystal properties (crystal type and orientation) of the surface:

```
define_model name=m description="Grain growth model"
add_source_species model=m name=Silane
add_reaction model=m name=R \
    expression="Silane<g> + PolySi<s> = PolySi<s> + PolySi<b>"
finalize_model model=m
.
.
.
add_reaction_properties reaction=R p=1 \
    product_morphology=same_as_surface_reactant \
    diamond_p_100=0.5 diamond_p_110=0.8 diamond_p_111=0.2
```

Chapter 3: Model Descriptions

Crystalline Materials, Nucleation, and Grain Growth

Example 2 Adsorb amorphous silicon a-Si on a polycrystalline PolySi surface:

```
define_model name=m description="Adsorption of amorphous Silicon"
add_source_species model=m name=Silane
add_reaction model=m name=R \
    expression="Silane<g> + PolySi<s> = a-Si<s>"
finalize_model model=m
.
.
.
add_reaction_properties reaction=R p=1 product_morphology=amorphous \
    diamond_p_100=0.5 diamond_p_110=0.8 diamond_p_111=0.2
```

Nucleation and Grain Growth

The adsorption or deposition of crystalline material (for example, polycrystalline silicon) on top of an amorphous substrate (for example, graphite) is commonly known as *crystal film growth*. In general, the morphology of the deposited film depends on the process conditions and can be amorphous (disordered), crystalline (oriented), or polycrystalline (multiple grains). Crystal film growth is a complex physical process that consists of many simultaneous atomistic effects, including adsorption, desorption, nucleation, grain growth, and coalescence (see, for example, [13], Chapter 8.5).

You can use a simplified model to describe crystal film growth by using only two parameters (see [Figure 18](#)):

- The parameter `p_nucleation` sets the probability that a gaseous reactant sticks stably to the surface (and, therefore, becomes a so-called adatom). This probability implicitly describes the net effect of the atomistic processes of adsorption, desorption, and nucleation. By default, the crystal type of the deposited nucleus is diamond, and the crystal orientation is chosen randomly by the simulator.
- The parameter `p_island_growth` sets the probability that a new gaseous reactant, which is adsorbed near an existing grain, attaches to that grain. This parameter implicitly describes adatoms sticking to the boundary of grain islands.

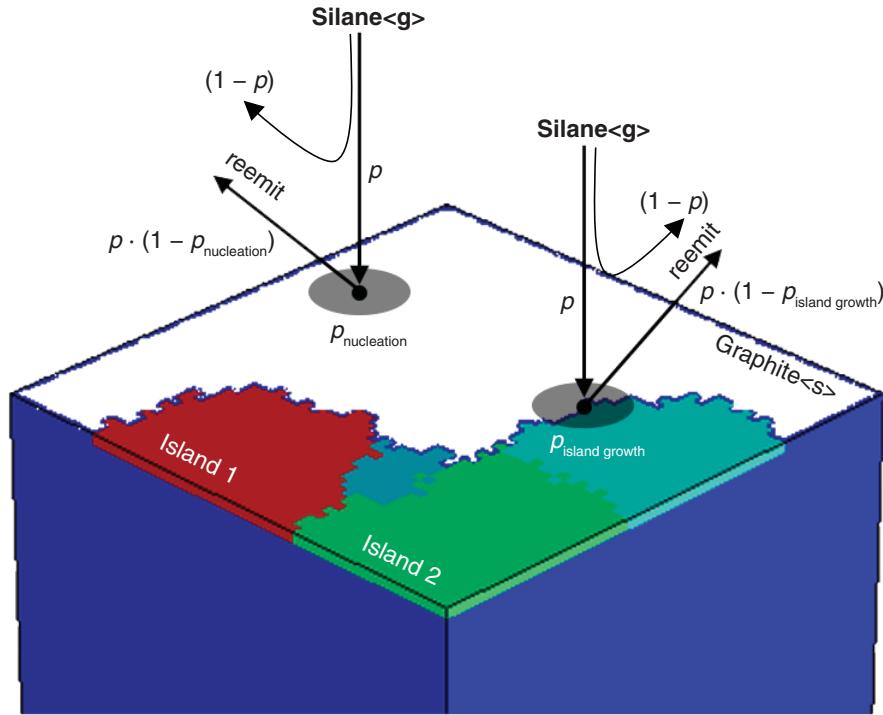
Example Grow a crystalline PolySi film on an amorphous graphite substrate (see [Figure 18](#)):

```
define_model name=m description="Nucleation and grain growth model"
add_source_species model=m name=Silane
add_reaction model=m name=R \
    expression="Silane<g> + Graphite<s> = PolySi<s> + Graphite<b>"
finalize_model model=m
.
.
.
add_reaction_properties reaction=R p=1 \
    product_morphology=crystalline crystal_type=diamond \
    p_nucleation=1e-6 p_island_growth=1
```

Chapter 3: Model Descriptions

Crystalline Materials, Nucleation, and Grain Growth

Figure 18 Nucleation and grain growth model: If Silane $\langle g \rangle$ hits the amorphous Graphite $\langle s \rangle$ surface in a free area, a new crystal nucleus with a random crystal orientation is created with probability $p \cdot p_{\text{nucleation}}$ (where p is the overall probability that the reaction is executed). On the other hand, if Silane $\langle g \rangle$ hits the Graphite $\langle s \rangle$ surface near an existing PolySi $\langle s \rangle$ island, it attaches to that island with probability $p \cdot p_{\text{island growth}}$, taking over the crystal properties of the island. If no nucleation or attachment occurs, then Silane $\langle g \rangle$ is reemitted or discarded (depending on the default event defined for Silane $\langle g \rangle$).



Notes:

- You can use `product_morphology=crystalline` only together with a nucleation model, that is, together with the nucleation parameters `p_nucleation` and `p_island_growth`.
- The probability of nucleation is typically orders of magnitude smaller than the probability of island growth. This is explained by the underlying atomistic processes: The probability of an adatom forming a new stable island is much lower than the probability of it sticking to a preexisting island. The ratio of `p_nucleation` to `p_island_growth` determines how many grains are deposited on the surface; therefore, you can control the average grain size by varying these parameters.
- After some time, the entire surface will be covered by a thin film of grains. To further grow these grains in height, you need to define a deposition reaction (see [Adsorption and Deposition Reactions on page 86](#)).

Diffusion

Surface diffusion is available for deposition reactions and can be specified separately for each reaction, using the `add_reaction_properties method=diffusion` command. The implementation uses the random walk method to model diffusion. The following methods are implemented:

- Pure random walk (`biased_random_walk=last_step`)

Choose this method to model a pure random walk with a fixed number of steps given by parameter `num_diffusion_steps`.

- Biased random walk (`biased_random_walk=always`)

Choose this method to model the effects of energetic barriers such as surface roughness. The number of executed diffusion steps might be less than `num_diffusion_steps`, depending on the surface roughness.

Example

Specify that the silane particle, after hitting the surface, will execute a five-step random walk along the surface until it reaches a suitable surface site to which to stick:

```
define_model name=m \
    description="Surface diffusion of PolySi on Graphite"
add_source_species model=m name=Silane
add_reaction model=m name=R \
    expression="Silane<g> + Graphite<s> = PolySi<s> + Graphite<b>"
finalize_model model=m
.
.
.
add_reaction_properties reaction=R p=1 method=diffusion \
    biased_random_walk=always num_diffusion_steps=5
```

There are additional numeric parameters to control the algorithm to minimize surface roughness (see [add_reaction_properties on page 168](#)). These parameters are set to reasonable default values.

References

- [1] J. Li, *Topography Simulation of Intermetal Dielectric Deposition and Interconnection Metal Deposition Processes*, PhD thesis, Stanford University, Stanford, CA, USA, March 1996.
- [2] T. Mizuno *et al.*, “Analytical Model for Oblique Ion Reflection at the Si Surface,” *IEEE Transactions on Electron Devices*, vol. 35, no. 12, pp. 2323–2327, 1988.

Chapter 3: Model Descriptions

References

- [3] J. J. Vlassak, “A model for chemical–mechanical polishing of a material surface based on contact mechanics,” *Journal of the Mechanics and Physics of Solids*, vol. 52, no. 4, pp. 847–873, 2004.
- [4] J. A. Sethian and Y. Shan, “Solving partial differential equations on irregular domains with moving interfaces, with applications to superconformal electrodeposition in semiconductor manufacturing,” *Journal of Computational Physics*, vol. 227, no. 13, pp. 6411–6447, 2008.
- [5] R. Akolkar and U. Landau, “Mechanistic Analysis of the “Bottom-Up” Fill in Copper Interconnect Metallization,” *Journal of the Electrochemical Society*, vol. 156, no. 9, pp. D351–D359, 2009.
- [6] T. P. Moffat *et al.*, “Curvature enhanced adsorbate coverage mechanism for bottom-up superfilling and bump control in damascene processing,” *Electrochimica Acta*, vol. 53, no. 1, pp. 145–154, 2007.
- [7] T. Langer, A. Belyaev, and H.-P. Seidel, “Spherical Barycentric Coordinates,” in *Eurographics Symposium on Geometry Processing*, Cagliari, Sardinia, Italy, pp. 81–88, June 2006.
- [8] S. Hirasawa *et al.*, “Analysis of Drying Shrinkage and Flow Due to Surface Tension of Spin-Coated Films on Topographic Substrates,” *IEEE Transactions on Semiconductor Manufacturing*, vol. 10, no. 4, pp. 438–444, 1997.
- [9] L. M. Peurung and D. B. Graves, “Film Thickness Profiles over Topography in Spin Coating,” *Journal of the Electrochemical Society*, vol. 138, no. 7, pp. 2115–2124, 1991.
- [10] L. E. Stillwagon and R. G. Larson, “Leveling of thin films over uneven substrates during spin coating,” *Physics of Fluids A*, vol. 2, no. 11, pp. 1937–1944, 1990.
- [11] G. Cunge *et al.*, “Ion flux composition in HBr/Cl₂/O₂ and HBr/Cl₂/O₂/CF₄ chemistries during silicon etching in industrial high-density plasmas,” *Journal of Vacuum Science & Technology B*, vol. 20, no. 5, pp. 2137–2148, 2002.
- [12] C. Kittel, *Introduction to Solid State Physics*, Hoboken, New Jersey: John Wiley & Sons, 8th ed., 2005.
- [13] A. Groß, *Theoretical Surface Science, A Microscopic Perspective*, Berlin: Springer, 2nd ed., 2009.

4

Plasma Modeling

This chapter describes plasma modeling in Sentaurus Topography 3D.

Simulation Modules

A plasma model consists of two simulation modules: a global plasma bulk model and a sheath model.

The plasma bulk model computes the ion, neutral, and electron densities and the electron temperature in the plasma bulk region for given plasma equipment parameters, such as RF power, pressure, and inlet gas flows.

The sheath model takes as input the species densities and the electron temperature computed by the plasma bulk model, as well as the electrode parameters (such as RF bias power, RF bias voltage, and RF bias current), and computes the energy and angular distributions of the ions and neutrals impinging on the wafer surface.

User Input Flow

[Figure 19 on page 93](#) shows the input flow of a plasma model.

You create a plasma model by using the `define_plasma_model` command, which sets the bulk and sheath model types to be used for the plasma simulation (see [define_plasma_model on page 258](#)).

In addition, a plasma model consists of a set of neutral and ion species, and a list of reactions between them. You can add species to the model by using the `add_species` command (see [add_species on page 182](#)). Reactions between species are added by using the `add_bulk_reaction` command (see [add_bulk_reaction on page 138](#)).

Note:

All species names occurring in reaction expressions must have been previously defined in the `add_species` command.

Chapter 4: Plasma Modeling

User Input Flow

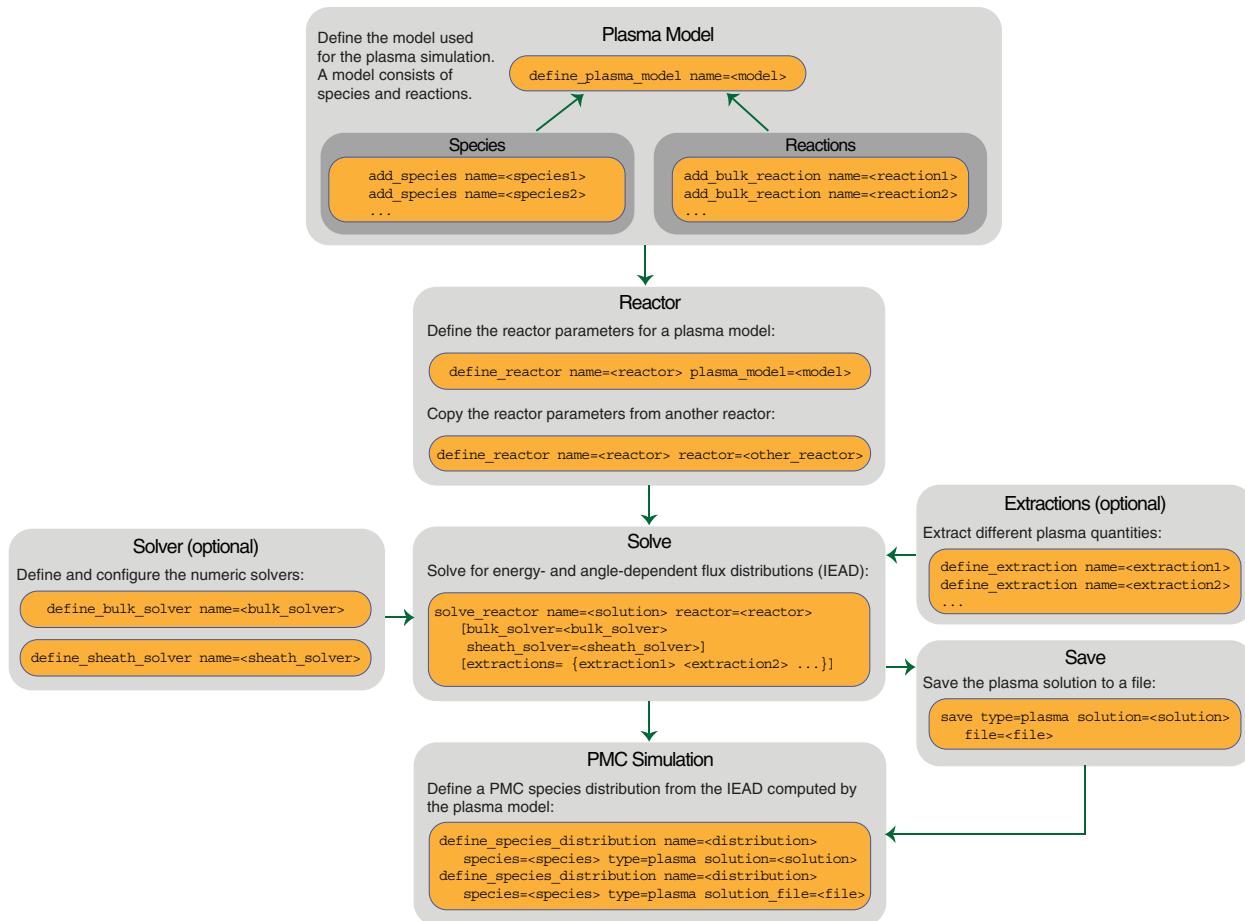
For example:

```
define_plasma_model name=M bulk_model_type=global \
sheath_model_type=circuit

add_species plasma_model=M name=Ar mass=39.948<amu> charge=0
add_species plasma_model=M name=Ar+ mass=39.948<amu> charge=+1

add_bulk_reaction plasma_model=M name=r \
expression="Ar + e- = Ar+ + 2e-" \
rate_coefficient_type=arrhenius a=2.3e-14 b=0.6329 c=16.0627 \
energy_transfer=15.76<eV>
```

Figure 19 Input flow of a plasma model



The reactor parameters (such as power, gas flow, and pressure) to be used in the simulation must be specified in the `define_reactor` command (see [define_reactor on page 261](#)). Reactor parameters can also be initialized as a copy from another reactor object (see the

Chapter 4: Plasma Modeling

User Input Flow

following `solve_reactor` example). When initializing a reactor from another reactor, you can modify specific reactor parameters in the same command. For example:

```
define_reactor name=R plasma_model=M type=icp radius=15<cm> \
    height=8<cm> power=1000<W> pressure=10<mTorr> \
    inlet_gas_flow= {{Ar 40<sccm>}} gas_temperature=400<K> \
    rf_bias_frequency=3.6<MHz> rf_bias_power=40<W>

define_reactor name=R2 reactor=R

define_reactor name=R3 reactor=R power=500<W> gas_temperature=300<K>
```

Finally, the plasma simulation is executed by the `solve_reactor` command, which computes the energy- and angle-dependent flux distributions of the species defined in the plasma model (see [solve_reactor on page 487](#)). The energy-angular flux distributions can be used directly in a PMC simulation, or they can be saved to a file. You can define a PMC species distribution directly after the plasma simulation by calling `define_species_distribution` with the name of the plasma solution (as specified in the `solve_reactor` command), or you can load the distribution from a file containing the plasma solution.

Example 1

```
solve_reactor name=S reactor=R

define_species_distribution name=sd species=Ar+ type=plasma solution=S
```

Example 2

```
solve_reactor name=S reactor=R

save type=plasma solution=S file=solution.plasma

define_species_distribution name=sd species=Ar+ type=plasma \
    solution_file=solution.plasma
```

If you want to analyze the plasma simulation further, then you can extract different dataset types such as the species densities in the plasma bulk, the electron temperature, the reaction statistics in the plasma bulk, the convergence residuals, the electron energy distribution, and the sheath characteristics (potential waveform, sheath currents, sheath capacitance, flux, electric field, and so on). You can define several extraction quantities by calling the `define_extraction` command multiple times with different names (see [define_extraction on page 218](#)). Finally, you must pass a list of extraction names to the `solve_reactor` command. For example:

```
define_extraction name=ex1 type=plasma bulk_model_type=global \
    quantities={state residuals reactions}

define_extraction name=ex2 type=plasma sheath_model _type=circuit \
    quantities={energy_distribution wall_potential sheath_width}
```

Chapter 4: Plasma Modeling

Plasma Bulk Model

...

```
solve_reactor name=S reactor=R extractions={ex1 ex2 ... }
```

For a complete list of the available extraction quantities, see [Table 35 on page 234](#).

Optionally, you can configure the numeric solvers used to compute the solution in the `solve_reactor` command. The bulk solver can be configured by calling the command `define_bulk_solver`, and the sheath solver is specified in the `define_sheath_solver` command. If you do not specify a solver explicitly, the default solvers are used (see [define_bulk_solver on page 187](#) and [define_sheath_solver on page 285](#)).

Plasma Bulk Model

The plasma bulk model describes the complex physical processes in the plasma bulk region, which is governed by reactions between electrons and gas particles. Depending on the operating conditions of the plasma reactor (such as applied power and pressure), high-energetic collisions between plasma particles are triggered, producing energetic ions and other byproducts. At the boundary of the plasma bulk region, ions escape into the sheath region at a certain rate, where they are accelerated by the sheath potential towards the structure.

The main characteristics of the plasma bulk region are described by particle densities and the electron temperature. In the global plasma bulk model, the plasma species in the reactor are assumed to be well mixed, such that the species densities and temperatures can be treated as spatially uniform.

You can use the global plasma bulk model by setting `bulk_model_type=global` in the `define_plasma_model` command (see [define_plasma_model on page 258](#)).

Particle Density and Energy Balance

Unless otherwise stated, the input parameters mentioned in this section refer to the command `define_reactor` (see [define_reactor on page 261](#)).

The governing equations of the plasma bulk model are the particle continuity equations for the species densities and the energy conservation equation for the electron temperature.

The continuity equation for the i -th species is given by [1][2]:

$$\frac{\partial n_i(t)}{\partial t} = R_{\text{gain}, i}(t) - R_{\text{loss}, i}(t) + Q_{\text{inlet}, n} - n_i(t)Q_{\text{outlet}}(t) - n_i(t)v_{\text{loss}, i}(t) \quad (42)$$

Chapter 4: Plasma Modeling

Plasma Bulk Model

where:

- t denotes the time.
- $n_i(t)$ is the volume-averaged particle number density of species i .
- $R_{\text{gain}, i}(t)$ is the density increase by production of species i in the reactions. The total density gain rate by all reactions is given by:

$$R_{\text{gain}, i}(t) = \sum_{r \in \{\text{reactions}\}} R_{\text{gain}, i, r}$$

where $R_{\text{gain}, i, r}$ is the density gain rate by reaction r :

$$R_{\text{gain}, i, r} = \begin{cases} R_r, & \text{if species } i \text{ is a product of reaction } r \\ 0, & \text{else} \end{cases} \quad (43)$$

and the reaction rate of reaction r is:

$$R_r = k_r(T_e) \prod_{j \in \{\text{reactants of } r\}} n_j \quad (44)$$

Here:

- $k_r(T_e)$ denotes the rate coefficient associated with reaction r , which depends on the electron temperature T_e .
- n_j are the reactant densities of reaction r .
- $R_{\text{loss}, i}(t)$ is the density decrease by consumption of species i in the reactions. The total density loss rate by all reactions is given by:

$$R_{\text{loss}, i}(t) = \sum_{r \in \{\text{reactions}\}} R_{\text{loss}, i, r}$$

where $R_{\text{loss}, i, r}$ is the density loss rate by reaction r :

$$R_{\text{loss}, i, r} = \begin{cases} R_r, & \text{if species } i \text{ is a reactant of reaction } r \\ 0, & \text{else} \end{cases}$$

- $Q_{\text{inlet}, n}$ is the source term by the gas inlet (specified by the `inlet_gas_flow` parameter).
- $Q_{\text{outlet}}(t)$ is the loss term by the gas outlet.

Chapter 4: Plasma Modeling

Plasma Bulk Model

- $v_{\text{loss}, i}(t)$ is the loss rate of species i escaping into the sheath at the plasma bulk boundary.

All quantities in [Equation 42](#) are spatially averaged over the volume of the plasma bulk.

The plasma bulk is modeled as a cylindrical region of radius R_{bulk} (parameter `radius`) and height L_{bulk} (parameter `height`) in the inner of the plasma chamber. As suggested in [\[3\]](#), the effective dimensions R_{bulk} and L_{bulk} of the plasma bulk are related to the actual dimensions of the plasma chamber R and L by the scaling factors h_R and h_L :

$$\begin{aligned} R_{\text{bulk}} &= R \cdot \sqrt{h_L} \\ L_{\text{bulk}} &= L \cdot \frac{h_R}{\sqrt{h_L}} \end{aligned} \quad (45)$$

The scaling factors h_R and h_L depend on the mean free path λ of the ions:

$$h_R = 0.8 \left(4 + \frac{R}{\lambda} \right)^{-0.5}$$

$$h_L = 0.86 \left(3 + \frac{L}{2\lambda} \right)^{-0.5}$$

These factors reduce to $h_R = 0.4$ and $h_L = 0.5$ in the collisionless limit, which is suitable for inert gases [\[4\]](#).

The total pressure is computed by the thermodynamic relation [\[4\]](#):

$$p(t) = \sum_{\substack{i \in \text{ion} \\ \text{and neutrals}}} n_i(t) k_B T_{\text{gas}} \quad (46)$$

where:

- The sum runs over all ions and neutral species.
- $n_i(t)$ is the density of species i .
- k_B is the Boltzmann constant.
- T_{gas} denotes the gas temperature, specified by the `gas_temperature` parameter.

The pressure inside the plasma chamber remains constant at the user-specified value in the `pressure` parameter. This is done by a pressure controller, which automatically calibrates the pumping speed at the outlet, such that the total pressure remains at the user-defined fixed value. The characteristic retardation of the pressure controller can be adjusted by the parameter `pressure_relaxation_time`.

Chapter 4: Plasma Modeling

Plasma Bulk Model

To prevent nonphysical negative values, reasonable lower cutoff values are defined for all solution variables (ion, neutral and electron density, and electron temperature). You can adjust these values by using the parameters `min_density`, `min_electron_density`, `min_electron_energy`, and `min_density_relaxation_time`.

A key assumption in the modeling of the plasma bulk region is the charge-neutrality condition. Since the number of positively charged particles in the plasma bulk region is assumed to equal the number of negatively charged particles, the total charge equals zero.

For the electron density $n_e(t)$, this implies [5]:

$$n_e(t) = \sum_{i \in \text{ions}} q_i \cdot n_i(t) \quad (47)$$

where q_i is the charge of ion i , measured in units of the unit charge e .

To ensure the correct treatment of the energy balance in the plasma bulk region, the model simultaneously solves a power balance equation, given by [1][3][6][7]:

$$P_{\text{abs}} = P_e(t) + P_{\text{reaction}}(t) + P_{\text{loss}}(t) \quad (48)$$

where:

- P_{abs} is the total power absorbed by the system. The total absorbed power is assumed to be proportional to the time average of the RF power applied to the reactor (specified by the `power` parameter):

$$P_{\text{abs}} = \alpha \langle P_{\text{rf}}(t) \rangle_t$$

where $\alpha \in [0, 1]$ is the power absorption coefficient (specified by the parameter `power_absorption_coefficient`).

- $P_e(t)$ is the power absorbed by the bulk electrons.
- $P_{\text{reaction}}(t)$ is the power transferred in plasma bulk reactions.
- $P_{\text{loss}}(t)$ is the power lost into the reactor walls.

When calling the `solve_reactor` command, the governing equations (Equation 42, Equation 46, Equation 47, and Equation 48) are solved simultaneously until a stationary state is reached. The initial values for the species densities and the electron temperature are found automatically. However, you can define your own initial values by using the `density` and `electron_temperature` parameters.

Bulk Reactions

The plasma bulk model supports user-defined plasma bulk reaction sets. Bulk reactions can be defined by the `add_bulk_reaction` command, which takes the reaction equation, the

Chapter 4: Plasma Modeling

Plasma Bulk Model

reaction rate, and the energy transferred during the reaction (see [add_bulk_reaction on page 138](#)).

The reaction equation (specified by the `expression` parameter) consists of two reactant species, and one or more reaction products. For example:

```
expression= "Ar + e- = Ar+ + 2e-"
```

While no stoichiometric coefficients other than 1 are allowed on the reactant side, the stoichiometric coefficients of the reaction products can be arbitrary positive integers.

The rate at which each reaction is executed in the plasma bulk region is specified by a rate coefficient $k_r(T_e)$. The rate coefficient depends on the electron temperature T_e by either an Arrhenius law (`rate_coefficient_type=arrhenius`):

$$k_r(T_e) = a \cdot \left(\frac{T_e}{T_{\text{ref}}}\right)^b \cdot \exp\left(-\frac{c}{T_e}\right) \quad (49)$$

or a generalized Arrhenius law (`rate_coefficient_type=general_arrhenius`):

$$k_r(T_e) = \exp\left(a + b \ln T_e + \frac{c}{T_e} + \frac{d}{T_e^2} + \frac{e}{T_e^3}\right) \quad (50)$$

where the coefficients a , b , c , d , and e are listed in reaction tables (for example, see [\[2\]\[8\]](#)).

In the previous example reaction, an Ar neutral is transformed into an Ar^+ ion. During this reaction, the density of Ar reduces by an amount proportional to $n_{\text{Ar}} n_e k_r(T_e)$, while the density of Ar^+ increases by the same amount.

In addition, a certain amount of energy is consumed by this reaction, which is specified by the `energy_transfer` parameter. By convention, the energy transfer is positive for endothermal reactions and negative for exothermal reactions. Accordingly, the amount of energy consumed or produced by reactions leads to a decrease or an increase of the electron temperature by the power balance equation ([Equation 48](#)).

You can save and plot a rate coefficient, which you defined using the `add_bulk_reaction` command (see [add_bulk_reaction on page 138](#)), as a function of the electron temperature, by calling the `save plasma_model=<c> quantity=rate_coefficient` command (see [save on page 461](#)).

Extracting Bulk Datasets

You can further analyze the plasma bulk simulation by saving and plotting various physical datasets.

Chapter 4: Plasma Modeling

Plasma Bulk Model

By calling the `define_extraction type=plasma bulk_model_type=global quantities=<1>` command, you can extract:

- `quantities=electron_energy_distribution`: The steady-state energy distribution $\varepsilon_e(E)$ of the electron in the bulk
- `quantities=reactions`: The reaction rate R_r (see [Equation 44](#)) of each bulk reaction r as a function of time
- `quantities=residuals`: The convergence residuals of the solution variables n_i , n_e , and T_e in [Equation 42](#), [Equation 47](#), and [Equation 48](#) as a function of time
- `quantities=species_balance`: For each species i , the time-integrated rate of production (or consumption) in each reaction r (see [Equation 43](#)):

$$\Delta R_{i,r} = \int_0^T (R_{\text{gain},i,r} - R_{\text{loss},i,r}) dt$$

- `quantities=state`: The ion or neutral densities n_i , the electron density n_e , the electron temperature T_e , the electron energy density $\varepsilon_e = 3/2n_e T_e$, the pressure p , and the absorbed power P_{abs} as a function of time
- `quantities=waveforms`: When simulating pulsed powers: The periodic steady-state waveforms of the ion or neutral densities n_i , the electron density n_e , the electron temperature T_e , and the absorbed power P_{abs} as a function of time for one period of the power pulse

The time-dependent quantities can be written to a file at regular time intervals (parameter `file_update_step`) while the simulation is still running. You can also define the sampling rate at which the data points are measured (parameter `time_step` or `extraction_step`). The datasets can be written to a TDR file (`output_type=tdr`) or to a CSV file (`output_type=csv_file`) or both files. You can reduce the number of datasets written by applying filters to the set of species names or reaction equations (parameters `species`, `species_pattern`, `reactions`, and `expression_pattern`). See [define_extraction on page 218](#) for details.

Source Power Pulsing

The latest plasma reactors support not only single-frequency RF biasing, but also dual and triple frequencies (for example, 60 MHz, 10 MHz, 2 MHz), to better control energetic ion bombardment onto the wafer. While the low frequency controls the shape of the ion energy distribution, the high frequency affects the level of ionization in the plasma bulk and can result in a modulation of the ion energy distribution [9].

Bulk Simulation With Pulsed Power

To simulate power pulsing, you can define a time-dependent function expression for the source power (parameter `power_waveform` in the `define_reactor` command; see

Chapter 4: Plasma Modeling

Plasma Bulk Model

[define_reactor on page 261](#)). The power function is assumed to be periodic in time with a period given by the parameter `power_period`. By default, the power function is sampled using 100 points per period, which can be changed by using the `num_waveform_samples` parameter.

For information about the syntax for function expressions, see [Syntax for Expressions on page 131](#).

Example 1 Set the input power to a square pulse with an on-value of 2000 W, an off-value of 200 W, a period of 1 ms (corresponding to a pulse frequency of 1 MHz), and a duty ratio of 20%:

```
define_reactor ... \
    power_waveform="square_pulse( t<s>, 2000, 200, 1e-6, 0.2 )" \
    power_period=1e-6<s>
```

Example 2 Set the input power to a sinusoidal wave with a DC power of 1600 W, an amplitude of 400 W, and a frequency of 1 MHz (corresponding to a period of 1 ms):

```
define_reactor ... power_waveform="1600 + 400*sin(2*M_PI*1e6*t)" \
    power_period=1e-6<s>
```

Tip:

To improve convergence when simulating pulsed powers, it is advisable to not start the bulk simulation from scratch, but to initialize the plasma state (densities and electron temperature) from the solution of an equivalent continuous-wave power simulation using the time-averaged power.

To initialize the state from a previous solution, see the `solution` or `solution_file` parameter in the `define_reactor` command (see [define_reactor on page 261](#)).

When solving a plasma model with a pulsed power waveform, the steady-state species densities and the electron temperature are periodic waveforms. The waveforms can be saved and visualized by defining an extraction with `type=plasma` and `quantities=waveforms` (see [define_extraction on page 218](#)). If instead you want to have the species densities and the electron temperature time-averaged, then you must call the `define_bulk_solver` command with `time_average_solution=true` and pass the name of the bulk solver to the `solve_reactor` command (see [solve_reactor on page 487](#)).

Sheath Simulation With Pulsed Power and Multiple Bias Sources

The species density waveforms and the electron temperature waveform computed by the bulk model serve as input for the sheath model. Internally, the sheath solver needs to find the least common multiple of the periods of the different input waveforms (densities and electron temperature from the bulk model) and of the bias sources applied to the sheath. You can control the tolerance used to find the common period by using the parameter `frequency_snapping_period` in the `define_sheath_solver` command (see

Chapter 4: Plasma Modeling

Plasma Sheath Models

[define_sheath_solver on page 285](#)). In addition, you can increase the maximum number of periods considered by using the parameter `max_num_waveform_periods`.

When passing time-dependent density and electron temperature waveforms to the sheath model, the solution of the sheath model (the flux) is a time-dependent waveform as well. By default, however, the flux waveform is returned as time averaged and can be used to define a (time-independent) flux distribution for a PMC simulation (see [define_species_distribution on page 290](#)). If instead you want to work with a time-dependent flux waveform, then you must set `time_average_solution=false` in the `define_sheath_solver` command and pass the name of the sheath solver to the `solve_reactor` command (see [solve_reactor on page 487](#)).

Note:

The distribution of energies and angles of the flux is always averaged over one period of the flux waveform. Only the total flux (that is, the integral of the energy- and angle-dependent distribution) is time dependent.

If you pass a time-dependent flux distribution to the PMC simulator, then you can optionally specify the sampling time step used for the flux distribution in the PMC simulator:

```
define_species_distribution name=<c> type=plasma species=<c>
(solution=<c> | solution_file=<c>) [sampling_time_step=<n>]
```

You can visualize the time-dependent flux by saving the parameters of the species distribution to a TDR file (see [save on page 461](#)):

```
save species_distribution=<c> output_type=parameters time=<n>
```

Plasma Sheath Models

While the plasma bulk region is assumed to be spatially uniform, charge neutral, and homogeneous, there exists a region with a high electrical potential drop between the plasma bulk and the electrode: the so-called plasma sheath region.

In this region, ions accelerate strongly from the plasma bulk region towards the wafer surface. For the modeling of industrial plasma etching processes, it is essential to know the energy and angular distribution of the particles reaching the wafer surface. While the distribution in the plasma bulk region is isotropic in the thermodynamic equilibrium, the distribution changes considerably due to the acceleration of ions across the sheath layer.

You select the sheath models by using the `sheath_model_type` parameter in the command `define_plasma_model` (see [define_plasma_model on page 258](#)).

The next sections present different sheath models that predict the energy and angular distributions of ions and neutrals at the surface. In addition, unless otherwise stated, input parameters refer to the `define_reactor` command (see [define_reactor on page 261](#)).

Self-Consistent Circuit RF Sheath Model

A self-consistent dynamic model of the RF-biased sheath has been proposed by Edelberg and Aydil [10]. You select this model by setting `sheath_model_type=circuit` in the command `define_plasma_model`.

The circuit RF sheath model consists of two modules, which are coupled self-consistently:

- The first module is an ion transport model, which describes the charge transport in the sheath by solving the ion continuity equation, the ion momentum equation, and the Poisson equation for the sheath potential.
- The second module is an equivalent circuit model, which describes the sheath by a parallel electrical circuit consisting of a diode, a current source, and a capacitor connected to the applied RF bias source [10].

In the model, it is assumed that the sheath is infinite in the directions parallel to the electrode surface, such that the system reduces to a one-dimensional problem in the z-direction orthogonal to the surface. The model is suitable for industrial plasma reactors, which typically operate at high bias frequencies (in the MHz range) and low pressures (in the mTorr range). This implies that particles crossing the sheath can be assumed to be collisionless, as their mean free path (several millimeters) is typically much larger than the thickness of the sheath (several hundred micrometers).

Based on the sheath thickness $d_s(t)$ computed by the sheath model, the equivalent circuit model predicts the time-dependent waveform of the sheath potential at the electrode $V_w(t)$, which serves as a boundary condition for the Poisson equation. From the solution of the Poisson equation, a new value for the sheath thickness $d_s(t)$ is obtained, and the procedure is repeated self-consistently until the wall potential $V_w(t)$ converges to a periodic steady-state waveform (see [Figure 20 on page 104](#)).

To further analyze the simulation results for the circuit sheath model, you can save and visualize various datasets such as the wall potential $V_w(t)$, the sheath thickness $d_s(t)$, the currents through the sheath (applied RF bias current, electron current, total ion current, and capacitive displacement current), the sheath capacitance, the electric field at the wall, the fluxes, and the energy distributions of the ions or neutrals passing the sheath. For details about the datasets, see also [10]. You can extract datasets of interest by calling the `define_extraction` command with the parameters `type=plasma` and `sheath_model_type=circuit` (see [define_extraction on page 218](#)).

After the sheath potential profile at the wall $V_w(t)$ has converged, the energy distribution of the ions crossing the sheath potential is computed from the wall potential [10].

Chapter 4: Plasma Modeling

Plasma Sheath Models

As for the analytic sheath model, [Equation 68](#), the full ion energy and angular distribution (IEAD) for ion species i is computed from the ion energy distribution (IED) by:

$$\text{IEAD}_i(E, \theta, \phi) \propto \text{IED}_i(E) \cdot \frac{1}{\cos^2 \theta} \cdot \exp\left(-\frac{E}{T_i} \tan^2 \theta\right) \quad (51)$$

where:

- $\theta \in [0, \pi/2]$ is the polar angle.
- $\phi \in [0, 2\pi]$ is the azimuth angle.
- E is the ion energy in eV.
- T_i is the ion temperature in eV, which is assumed to equal the neutral gas temperature in the plasma bulk region (given by the `gas_temperature` parameter).

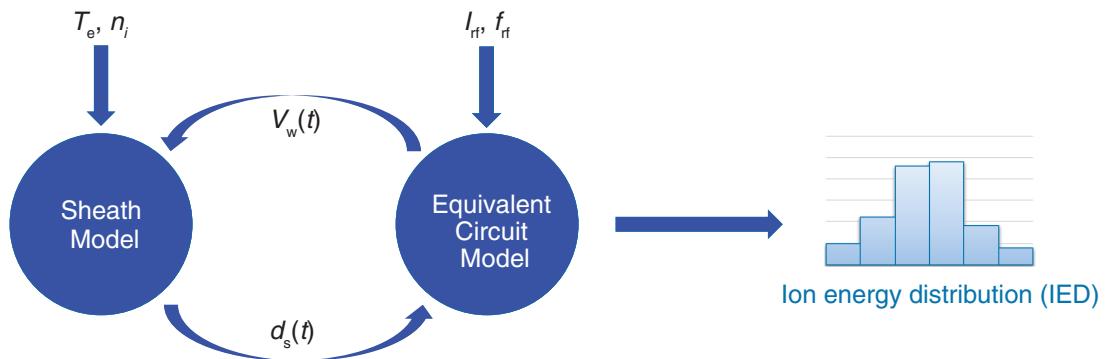
See [\[11\]](#) for details on the derivation of the ion angular distribution.

The IEAD satisfies the normalization condition:

$$\int_{E_{\min}}^{E_{\max}} dE \int_0^{2\pi} d\phi \int_0^{\pi/2} \sin \theta d\theta \text{IEAD}_i(E, \theta, \phi) = \Gamma_i \quad (52)$$

where Γ_i denotes the total flux of ion species i .

Figure 20 Flow chart of the model by Edelberg and Aydil used to predict the ion energy distribution (IED) of ions impinging on an RF-biased electrode, showing the self-consistent coupling between the sheath model and the equivalent circuit model. T_e and n_i denote the electron temperature and ion densities, I_{rf} and f_{rf} denote the RF bias current and frequency, and $V_w(t)$ and $d_s(t)$ denote the wall potential and sheath thickness, respectively.



Biasing Modes and Bias Pulsing

The model supports different biasing modes, which can be specified in the command `define_reactor` for a circuit sheath model (see [define_reactor on page 261](#)):

- [Current-Driven Mode](#)
- [Voltage-Driven Mode](#)
- [Power-Driven Mode](#)

Current-Driven Mode

In this mode, you can set the RF bias current $I(t)$ through the sheath as input to the sheath model. The resulting RF bias voltage $V_w(t)$ (wall potential) at the electrode is computed self-consistently to satisfy the current balance equation [10].

CDM1 Apply a sinusoidal RF bias current:

$$I(t) = I_{rf} \cdot \sin(2\pi f_{rf} t) \quad (53)$$

where I_{rf} is the amplitude of the bias current (parameter `rf_bias_current`), and f_{rf} is the bias frequency (parameter `rf_bias_frequency`).

CDM2 Apply multiple RF bias current sources with multiple frequencies:

$$I(t) = I_1(t) + I_2(t) + \dots = I_1 \cdot \sin(2\pi f_1 t) + I_2 \cdot \sin(2\pi f_2 t) + \dots \quad (54)$$

where $\{I_i\}$ are the amplitudes and $\{f_i\}$ are the frequencies. You can set these parameters by passing a list of values to the parameters `rf_bias_current` and `rf_bias_frequency`, respectively.

CDM3 Define an arbitrary time-dependent function expression for the RF bias current:

$$I(t) = f(t) \quad (55)$$

where the function $f(t)$ must be specified as a string in the `bias_current_waveform` parameter. (For information about the syntax for function expressions, see [Syntax for Expressions on page 131](#).) The bias current waveform is assumed to be periodic in time with a period given by parameter `bias_current_period`. By default, the bias current function is sampled using 100 points per period, which can be changed by using the parameter `num_waveform_samples`.

Voltage-Driven Mode

In this mode, you can set the RF voltage $V_w(t)$ (wall potential) applied to the electrode as input to the sheath model. The resulting RF bias current $I(t)$ through the sheath is computed self-consistently to satisfy the current balance equation [10].

Chapter 4: Plasma Modeling

Plasma Sheath Models

VDM1 Apply a sinusoidal RF bias voltage:

$$V_w(t) = V_{DC} + V_{AC} \cdot \sin(2\pi f_{rf} t) \quad (56)$$

where V_{DC} and V_{AC} are the DC part and AC part of the wall potential, respectively, and f_{rf} is the bias frequency. You can only set V_{AC} and f_{rf} (parameters `rf_bias_voltage` and `rf_bias_frequency`) since the DC part is computed self-consistently to satisfy the current balance equation (for details, see [12]).

VDM2 Apply multiple RF bias voltage sources with multiple frequencies:

$$\begin{aligned} V_w(t) &= V_{DC} + V_1(t) + V_2(t) + \dots \\ &= V_{DC} + V_1 \cdot \sin(2\pi f_1 t) + V_2 \cdot \sin(2\pi f_2 t) + \dots \end{aligned} \quad (57)$$

where V_{DC} is the DC part of the sheath potential, $\{V_i\}$ are the AC amplitudes per frequency and $\{f_i\}$ are the frequencies. You can set parameters $\{V_i\}$ and $\{f_i\}$ by simply passing a list of values to parameters `rf_bias_voltage` and `rf_bias_frequency`, respectively. V_{DC} is determined self-consistently following [12].

VDM3 Define an arbitrary time-dependent function expression for the RF bias voltage:

$$V_w(t) = f(t) \quad (58)$$

where the function $f(t)$ must be specified as a string in the `wall_potential_waveform` parameter. (For information about the syntax for function expressions, see [Syntax for Expressions on page 131](#).) The bias voltage waveform is assumed to be periodic in time with a period given by parameter `wall_potential_period`. By default, the wall potential function is sampled using 100 points per period, which can be changed by the `num_waveform_samples` parameter.

Power-Driven Mode

In this mode, you can set the RF bias power P_{rf} applied to the electrode as input to the sheath model. The power-driven mode is a special case of the other biasing modes: instead of specifying the current or voltage explicitly, you set the power absorbed by the system. This implicitly determines the RF bias current $I(t)$ and the wall potential $V_w(t)$ through the relation:

$$P_{rf} = -\frac{1}{\tau} \int_0^\tau V_w(t) \cdot I(t) \, dt \quad (59)$$

where $\tau = 1/f_{rf}$ is the period of the wall potential $V_w(t)$ and of the bias current $I(t)$.

PDM1 Specify the RF bias power for a sinusoidal RF bias current (see **CDM1**):

$$P_{rf} = -\frac{1}{\tau} \int_0^\tau V_w(t) \cdot I_{rf} \sin(2\pi f_{rf} t) \, dt \quad (60)$$

Chapter 4: Plasma Modeling

Plasma Sheath Models

You must set `bias_mode=current_driven` and specify the power P_{rf} (parameter `rf_bias_power`) and the frequency (parameter `rf_bias_frequency`). The current amplitude I_{rf} and the wall potential waveform $V_w(t)$ are computed self-consistently to match the given RF bias power P_{rf} .

PDM2 Specify the RF bias power for a sinusoidal RF bias voltage (see **VDM1**):

$$P_{\text{rf}} = -\frac{1}{\tau_{\text{rf}}} \int_0^{\tau_{\text{rf}}} (V_{\text{DC}} + V_{\text{AC}} \cdot \sin(2\pi f_{\text{rf}} t)) \cdot I(t) \, dt \quad (61)$$

You must set `bias_mode=voltage_driven` and specify the power P_{rf} (parameter `rf_bias_power`) and the frequency (parameter `rf_bias_frequency`). The voltages V_{DC} and V_{AC} and the current waveform $I(t)$ are computed self-consistently to match the given RF bias power P_{rf} .

PDM3 Specify the RF bias power for multiple RF bias current sources (see **CDM2**):

$$P_{\text{rf}} = -\frac{1}{\tau} \int_0^{\tau} V_w(t) \cdot (I_1 \cdot \sin(2\pi f_1 t) + I_2 \cdot \sin(2\pi f_2 t) + \dots) \, dt \quad (62)$$

You must set `bias_mode=current_driven` and provide a list of frequencies $\{f_i\}$ (parameter `rf_bias_frequency`) and a list of power values per frequency $\{P_i\}$ (parameter `rf_bias_power`), where the power per frequency is defined by:

$$P_i = -\frac{1}{\tau} \int_0^{\tau} V_w(t) \cdot I_i \sin(2\pi f_i t) \, dt \quad (63)$$

As in **PDM1**, the current amplitudes $\{I_i\}$ and the wall potential waveform $V_w(t)$ are computed self-consistently to match the given powers per frequency $\{P_i\}$.

PDM4 Specify the RF bias power for multiple RF bias voltage sources (see **VDM2**):

$$P_{\text{rf}} = -\frac{1}{\tau} \int_0^{\tau} (V_{\text{DC}} + V_1 \cdot \sin(2\pi f_1 t) + V_2 \cdot \sin(2\pi f_2 t) + \dots) \cdot I(t) \, dt \quad (64)$$

You must set `bias_mode=voltage_driven` and provide a list of frequencies $\{f_i\}$ (parameter `rf_bias_frequency`) and a list of power values per frequency $\{P_i\}$ (parameter `rf_bias_power`), defined by:

$$P_i = -\frac{1}{\tau} \int_0^{\tau} V_i \cdot \sin(2\pi f_i t) \cdot I(t) \, dt \quad (65)$$

Analogously to **PDM3**, the voltages V_{DC} and $\{V_i\}$ and the current waveform $I(t)$ are computed self-consistently to match the given powers per frequency $\{P_i\}$.

Chapter 4: Plasma Modeling

Plasma Sheath Models

Note:

In the power-driven mode, the voltage and current waveforms are computed self-consistently by an iterative scheme. You can control the initial values of the iterative scheme by specifying the parameter `initial_rf_current` or `initial_wall_potential` in the `define_sheath_solver` command for `sheath_model_type=circuit` (see [define_sheath_solver on page 285](#)).

To monitor the convergence towards the target bias power, you can save and visualize the solution variables ($\{I_i\}$ or $\{V_i\}$) and the target powers (P_{rf} and $\{P_i\}$) as a function of the iteration number (see `define_extraction` with quantities `rf_bias_power_iterations`, `rf_bias_current_iterations`, or `rf_bias_voltage_iterations`).

To monitor the convergence errors as a function of the iteration, specify the quantities `rf_bias_power_residuals` and `wall_potential_residuals`.

If there are convergence issues, then you can decrease the values of parameter `power_relaxation_constant`, `max_rf_current_step`, or `max_rf_voltage_step` in the `define_sheath_solver` command. By changing the value of `power_tolerance`, you can influence when the iterative scheme is considered to have converged to the target power value.

Analytic RF Sheath Model

Another RF sheath model is the analytic model by Benoit-Cattin and Bernard [13]. You select it by setting `sheath_model_type=analytic` in the `define_plasma_model` command (see [define_plasma_model on page 258](#)).

The analytic RF sheath model predicts a bimodal ion energy distribution (IED) with two symmetric energy peaks, which are separated by a certain energy width ΔE . It is assumed that the sheath is one-dimensional in the z-direction orthogonal to the surface, and that the plasma reactor is operated at high bias frequencies (in the MHz range) and low pressures (in the mTorr range), such that collisions between particles in the sheath can be neglected.

The RF-modulated electric potential between the sheath edge at the plasma bulk boundary and the electrode on the wafer surface is described by:

$$V(t) = V_{DC} + V_{AC} \sin(2\pi f_{rf} t) \quad (66)$$

where:

- V_{DC} is the DC part of the sheath potential (specified by the `dc_voltage` parameter).
- V_{AC} is the AC part of the sheath potential (specified by the `ac_voltage` parameter).
- f_{rf} is the RF bias frequency (specified by the `rf_bias_frequency` parameter).

Chapter 4: Plasma Modeling

Plasma Sheath Models

With the given sheath potential, the IED of ion i is computed to be [13][14]:

$$\text{IED}_i(E) \propto \frac{1}{\sqrt{1 - \frac{4(E - V_{\text{DC}})^2}{\Delta E_i^2}}} \quad (67)$$

where E denotes the ion energy in eV, and ΔE_i is the width of the IED in eV.

Given the IED, the full energy and angular distribution (IEAD) of ion species i reads:

$$\text{IEAD}_i(E, \theta, \phi) \propto \text{IED}_i(E) \cdot \frac{1}{\cos^2 \theta} \cdot \exp\left(-\frac{E}{T_i} \tan^2 \theta\right) \quad (68)$$

where:

- $\theta \in [0, \pi/2]$ is the polar angle.
- $\phi \in [0, 2\pi]$ is the azimuth angle.
- E is the ion energy in eV.
- T_i is the ion temperature in eV, which is assumed to equal the neutral gas temperature in the plasma bulk (given by the `gas_temperature` parameter).

See [11] for details on the derivation of the ion angular distribution.

The IEAD satisfies the normalization condition:

$$\int_{E_{\min}}^{E_{\max}} dE \int_0^{2\pi} d\phi \int_0^{\pi/2} \sin \theta d\theta \text{IEAD}_i(E, \theta, \phi) = \Gamma_i \quad (69)$$

where Γ_i denotes the total flux of ion species i .

For capacitively coupled plasma discharges (`reactor_type=ccp`), instead of specifying the sheath potential itself, you can define the RF voltage, $V_{\text{rf}} \sin(2\pi f_{\text{rf}} t)$, applied between the two parallel plates of the discharge. According to [6], the AC and DC sheath potentials are related to the applied RF voltage by:

$$V_{\text{DC}} = \frac{V_{\text{AC}}}{2} \approx 0.83 \frac{V_{\text{rf}}}{2} \quad (70)$$

The applied RF voltage can be specified by the following input parameters:

- `rf_bias_frequency` defines the RF bias frequency.
- `rf_voltage` defines the amplitude of the RF voltage waveform.

Sheath Model for Neutral Species

Both the circuit sheath model (`sheath_model_type=circuit`) and the analytic sheath model (`sheath_model_type=analytic`) treat neutral species in the same way.

Neutral plasma particles are not affected by the sheath potential, which implies that their energy and angular distribution at the electrode, after crossing the sheath, is the same as in the plasma bulk region. It is assumed that the energy distribution (ED) of neutral species n in the plasma bulk region is given by a Maxwell–Boltzmann distribution:

$$\text{ED}_n(E) = \frac{2\Gamma_n}{T_n^{3/2}\sqrt{\pi}} \exp\left(-\frac{E}{T_n}\right) \quad (71)$$

where:

- E is the energy in eV.
- $\Gamma_n = n_n v_n$ is the total flux of neutral n , where:
 - n_n is the plasma bulk density of neutral n .
 - $v_n = \sqrt{8eT_n/\pi m_n}$ is the thermal velocity of neutral n , where m_n is the mass of neutral n (specified in the `add_species` command; see [add_species on page 182](#)).
- T_n is the temperature of neutral n in eV (given by the `gas_temperature` parameter of the `define_reactor` command; see [define_reactor on page 261](#)).

Assuming an isotropic angular distribution, the full energy and angular distribution (EAD) of neutral species n reads:

$$\text{EAD}_n(E, \theta, \phi) \propto \text{ED}_n(E) \cdot \cos\theta \quad (72)$$

where $\theta \in [0, \pi/2]$ is the polar angle. The EAD satisfies the normalization condition:

$$\int_{E_{\min}}^{E_{\max}} dE \int_0^{2\pi} d\phi \int_0^{\pi/2} \sin\theta d\theta \text{EAD}_n(E, \theta, \phi) = \Gamma_n \quad (73)$$

Possible Issues and Recommendations

This section presents recommendations and possible issues that might be encountered when working with plasma models.

General Recommendations

The following are general recommendations:

1. Check that the plasma model operates in a suitable parameter range, as follows:
 - `ac_voltage` ∈ [10 V, 500 V]
 - `dc_voltage` ∈ [10 V, 500 V]
 - `densities` ∈ [10^{14} m^{-3} , 10^{21} m^{-3}]

Chapter 4: Plasma Modeling

Possible Issues and Recommendations

- electron_temperature ∈ [1 eV, 10 eV]
 - gas_temperature ∈ [200 K, 400 K]
 - inlet_gas_flow ∈ [1 sccm, 100 sccm]
 - power ∈ [100 W, 2500 W]
 - pressure ∈ [10^{-2} mTorr, 100 mTorr]
 - rf_bias_current ∈ [1 A, 50 A]
 - rf_bias_frequency ∈ [0.1 MHz, 100 MHz]
 - rf_bias_power ∈ [1 W, 200 W]
 - rf_voltage ∈ [10 V, 500 V]
2. Check whether all parameters are specified in the correct physical units. Reaction rate coefficients (a, b, c, d, e, and energy_threshold) can be specified for the electron temperature in eV or K.

Common Mistakes

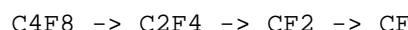
The following are common mistakes that can occur when setting up plasma models:

- rf_bias_frequency=3e6<MHz> instead of rf_bias_frequency=3<MHz>
- mass=6e-26 instead of mass=40 (default unit: <amu>)
- pressure=10 instead of pressure=10<mTorr> (default unit: <Pa>)
- gas_temperature=30 instead of gas_temperature=303.15 (default unit: <K>)

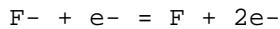
Reaction Data

The plasma bulk reaction datasets must be consistent and sufficiently complete to recover meaningful physical results. This means:

- Every reactant species should have been produced by another process. It must be a product of another reaction, or it must be specified as an inlet gas species (inlet_gas_flow parameter).
- For every meta-stable reaction byproduct, there should be another reaction that transforms the byproduct into a more stable product. For example:
 - Molecules in excited states, for example: Ar*
 - Nonatomic molecules, which can further split up, for example:



- For every negatively charged reaction product, there should be another reaction that transforms the negative ion into a neutral species. For example:



In addition, a common source of error is incorrect reaction rate coefficients. Incorrect numbers, signs, or physical units in reaction rate coefficients (parameters `a`, `b`, `c`, `d`, `e`, and `energy_threshold`), even if occurring only in a single reaction, can significantly change the result or lead to an overall failure of the model (typically in a negative density or negative electron temperature).

Negative Electron Temperature or Negative Species Density

You might encounter a negative electron temperature or negative species density during the bulk simulation. This error occurs in the plasma bulk model.

Possible solutions are:

- Use different initial conditions for the species densities (parameter `density` in the `define_reactor` command) or for the electron temperature (`electron_temperature` parameter).

Recommendations:

- Initialize most neutral or negatively charged species with zero density.
- The density n_i of the inlet gases (parameter `inlet_gas_flow`) must correspond to the given pressure p (parameter `pressure`) and gas temperature T_{gas} (`gas_temperature` parameter), that is:

$$\sum_{i \in \text{inlet gases}} n_i = \frac{p}{k_B T_{\text{gas}}}$$

- Do not specify the densities of the positive ions. Let the model chose them automatically.
- Configure the numeric parameters of the bulk solver, using [define_bulk_solver on page 187](#).

Recommendations:

- When simulating pulsed powers, decrease `pressure_relaxation_time` or `min_density_relaxation_time` to prevent undershooting toward negative values. You can also increase the cutoff parameters `min_density`, `min_electron_density`, or `min_electron_energy_density`.
- Decrease the initial step size of the solver (parameter `step_size`).

- Decrease the stepper errors (parameters `abs_error` and `rel_error`).
 - Vary the `stationary_state_tolerance` parameter.
3. See also [Reaction Data on page 111](#).

Convergence of Bulk Model

If your plasma bulk model takes a long time to converge, consider the following:

- Monitor the progress of the bulk simulation by writing out the bulk state and the convergence residuals (quantities `state` and `residuals` in [define_extraction on page 218](#)) at regular file update steps (parameter `file_update_step`). By monitoring the bulk state, you can check whether the solver proceeds.
- When using pulsed powers, initialize the reactor state with the steady-state solution of an equivalent continuous-wave power simulation, using the time-average value of the power waveform (parameter `solution` or `solution_file` in [define_reactor on page 261](#)).
- Increase the threshold for convergence, parameter `stationary_state_tolerance` in the `define_bulk_solver` command, to see whether the system converges at all.

Note:

Increasing the convergence threshold can lead to nonphysical results.

- Change the interval at which convergence is checked (parameter `convergence_check_step`).
- Consider increasing the value of parameters `pressure_relaxation_time` and `density_relaxation_time` in [define_reactor on page 261](#).
- Slightly change the process conditions to see whether the system converges in a test simulation under different conditions. If yes, then you can use the steady-state solution of the test simulation as an initial state for the actual simulation. See parameter `solution` or `solution_file` in [define_reactor on page 261](#).

Convergence of Circuit Sheath Model

The circuit sheath model might not converge to the specified RF bias power or might not converge to a self-consistent solution.

If the progress is slow during the iterative scheme, then consider the following:

- Monitor the progress of the sheath simulation by writing out the RF bias power, current, or voltage as a function of the iteration (quantities `rf_bias_power_iterations`, `rf_bias_current_iterations`, or `rf_bias_voltage_iterations` in the

Chapter 4: Plasma Modeling

Possible Issues and Recommendations

`define_extraction` command; see [define_extraction on page 218](#)). To monitor the convergence errors as a function of the iteration, use the quantities `rf_bias_power_residuals` and `wall_potential_residuals`.

- Increase the number of iterations, by using the `max_rf_bias_power_iterations`, `max_sheath_iterations`, `max_sheath_width_iterations`, and `max_wall_potential_iterations` parameters in the `define_sheath_solver` command.
- Optimize the relaxation toward the target bias power by using the `power_relaxation_constant`, `max_rf_current_step`, or `max_rf_voltage_step` parameter in the `define_sheath_solver` command.
- Increase the value of `power_tolerance` to increase the threshold for convergence toward the target bias power.
- Slightly change the initial values of the sheath model, by using the parameters `initial_rf_current`, `initial_sheath_width`, and `initial_wall_potential`.

If the sheath model stops with an error:

- In the simulation log file, analyze the densities of the positive ions computed by the bulk model. The sum of the densities of all positive ions should be in the range [10^{14} m^{-3} , 10^{18} m^{-3}]. If ion densities are too low, then increase the reactor parameter `power` or `pressure`. If ion densities are too high, then decrease `power` or `pressure`.
- If you use time-dependent waveforms for the densities and the electron temperature as input to the sheath model (resulting from a plasma bulk simulation with pulsed power), then you can instead use the time-averaged values by setting `time_average_solution=false` in the `define_bulk_solver` command of your bulk model.
- If you use a user-defined function expression for the bias source applied to the sheath (`parameter wall_potential_waveform` or `bias_current_waveform`), then you can increase the number of samples used to discretize the function by using the parameter `num_waveform_samples`.

If the progress is slow during computation of the energy- and angle-dependent flux distribution (EAD):

- Reduce the resolution of the EAD by decreasing `num_angle_samples`, `num_energy_samples`, and `num_inv_cdf_histogram_bins` in the `define_sheath_solver` command.
- If using the Monte Carlo IED solver (`ied_solver=monte_carlo`), then reduce the number of samples (parameter `num_monte_carlo_samples` in the `define_sheath_solver` command).

Energy or Angle Resolution of IEAD Is Too Coarse

The energy or angle resolution of the IEAD might be too coarse.

To resolve this, increase the values of the following parameters in the command `define_sheath_solver`:

- `num_angle_samples`
 - `num_energy_samples`
 - `num_ied_histogram_bins`
 - `num_inv_cdf_histogram_bins`
-

References

- [1] D. C. Kwon *et al.*, “A self-consistent global model of solenoidal-type inductively coupled plasma discharges including the effects of radio-frequency bias power,” *Journal of Applied Physics*, vol. 109, no. 7, p. 073311, 2011.
- [2] G. Kokkoris *et al.*, “A global model for C₄F₈ plasmas coupling gas phase and wall surface reaction kinetics,” *Journal of Physics D: Applied Physics*, vol. 41, no. 19, p. 195211, 2008.
- [3] C. Lee and M. A. Lieberman, “Global model of Ar, O₂, Cl₂, and Ar/O₂ high-density plasma discharges,” *Journal of Vacuum Science & Technology A*, vol. 13, no. 2, pp. 368–380, 1995.
- [4] M. Meyyappan and T. R. Govindan, *SAMPR: A Computer Code for Simple Analysis of Materials Processing Reactors*, NASA Research Publication 1402, Ames Research Center, Moffett Field, CA, USA, April 1997.
- [5] R. Zorat *et al.*, “Global model of a radiofrequency H₂ plasma in DENISE,” *Plasma Sources Science and Technology*, vol. 9, no. 2, pp. 161–168, 2000.
- [6] M. A. Lieberman and A. J. Lichtenberg, *Principles of Plasma Discharges and Materials Processing*, Chapters 6 and 11, Hoboken, New Jersey: John Wiley & Sons, 2nd ed., 2005.
- [7] S. Ashida, C. Lee, and M. A. Lieberman, “Spatially averaged (global) model of time modulated high density argon plasmas,” *Journal of Vacuum Science & Technology A*, vol. 13, no. 5, pp. 2498–2507, 1995.
- [8] G. Kokkoris *et al.*, “A global model for SF₆ plasmas coupling reaction kinetics in the gas phase and on the surface of the reactor walls,” *Journal of Physics D: Applied Physics*, vol. 42, no. 5, p. 055209, 2009.

Chapter 4: Plasma Modeling

References

- [9] Y. Zhang and M. J. Kushner, "Control of ion energy and angular distributions in dual-frequency capacitively coupled plasmas through power ratios and phase: Consequences on etch profiles," *Journal of Vacuum Science & Technology A*, vol. 33, no. 3, p. 031302, 2015.
- [10] E. A. Edelberg and E. S. Aydil, "Modeling of the sheath and the energy distribution of ions bombarding rf-biased substrates in high density plasma reactors and comparison to experimental measurements," *Journal of Applied Physics*, vol. 86, no. 9, pp. 4799–4812, 1999.
- [11] L. L. Raja and M. Linne, "Analytical model for ion angular distribution functions at rf biased surfaces with collisionless plasma sheaths," *Journal of Applied Physics*, vol. 92, no. 12, pp. 7032–7040, 2002.
- [12] Z.-L. Dai, X. Xu, and Y.-N. Wang, "A self-consistent hybrid model of a dual frequency sheath: Ion energy and angular distributions," *Physics of Plasmas*, vol. 14, no. 1, p. 013507, 2007.
- [13] P. Benoit-Cattin and L.-C. Bernard, "Anomalies of the Energy of Positive Ions Extracted from High-Frequency Ion Sources. A Theoretical Study," *Journal of Applied Physics*, vol. 39, no. 12, pp. 5723–5726, 1968.
- [14] E. Kawamura *et al.*, "Ion energy distributions in rf sheaths; review, analysis and simulation," *Plasma Sources Science and Technology*, vol. 8, no. 3, pp. R45–R64, 1999.

5

Surface Charging

This chapter describes surface charging in PMC simulations.

Overview of Surface Charging

An important phenomenon commonly observed in high aspect ratio plasma etching is surface charging inside the feature [1]. During plasma etching, the surface is constantly exposed to charged particles from the plasma, such as ions and electrons, impinging on the surface. At the surface, some of the ions and electrons transfer their charge to the surface. If the surface is a dielectric, then the charges cannot flow away and build up in time. Every surface charge acts as a source of an electric field, which deflects the trajectories of subsequent ions and electrons entering the feature [1][2].

The consequences of surface charge-up are significant: common artifacts attributed to the charge-up effect are sporadic bending and twisting of holes and trenches in dielectrics, notching and microtrenching at the feature bottom, and even early etch stop [1][2][3] [4][5][6].

The charge-up feature in Sentaurus Topography 3D PMC aims to model the surface charging effect during feature etching and deposition processes. The model consists of different parts:

- Ion and electron transport through an electric field (see [Ion and Electron Transport Through an Electric Field on page 118](#))
- Transfer of impinging charges to the surface (see [Transfer of Impinging Charges to the Surface on page 119](#))
- Modeling electric material properties (dielectrics, conductors, and applied electric potentials) (see [Electric Material Properties on page 120](#))
- Computation of the electric field caused by surface charges and applied electric potentials (see [Electric Field Computation on page 121](#))

Ion and Electron Transport Through an Electric Field

This section explains ion and electron transport through an electric field.

Ion Properties and Flux Distribution

When moving through an electric field, charged particles encounter a Coulomb force influencing the trajectory of the moving charge [2]. Depending on the local strength and orientation of the electric field, flying charges can be accelerated or decelerated, deflected sideways, or even repelled back into the plasma [2][5]. The influence of the electric field on the trajectories of charged particles depends on the charge, mass, and velocity of the particles.

Therefore, to simulate charged particle sources, first an energy- and angle-dependent flux distribution ([define_species_distribution command](#); [define_species_distribution on page 290](#)) must be specified, from which the initial velocities and directions of the source particles are sampled. Second, the mass and charge of the flux species must be specified by calling the [define_species_properties command](#) (see [define_species_properties on page 306](#)):

```
define_species_properties name=<c> species=<c> mass=<n> charge=<n>
```

The mass must be entered in atomic mass units, and the charge in units of the unit charge.

Electron Flux Distribution

While the main surface-changing reactions such as sputtering, etching, and deposition are driven by heavy ions and neutral radicals, charge-up is strongly influenced by electrons [1][2][3]. The flux and species properties for electrons do not need to be defined explicitly. When using the PMC charge-up model, electrons are simulated by default with thermal energies of 1 eV and with an isotropic flux distribution, $f(\theta) = 1/\pi\Gamma_e \cos(\theta)$, where the total electron flux Γ_e is implicitly set equal to the total ion flux to satisfy the overall charge neutrality condition [1][3][7].

A user-defined electron flux distribution is input by providing a list of exponent parameters $\{m_i\}$ and flux values $\{\Gamma_i\}$ for a generalized electron angular distribution:

$$f(\theta) = \frac{1}{2\pi} \sum_i \Gamma_i \cdot (m_i + 1) \cdot \cos^{m_i}(\theta)$$

See the parameters `electron_exponents` and `electron_fluxes` in the `define_charging_model` command (see [define_charging_model on page 189](#)).

Chapter 5: Surface Charging

Transfer of Impinging Charges to the Surface

In addition, for user-defined electron distributions, the total electron flux $\Gamma_e = \sum_i \Gamma_i$ is rescaled to match the total ion flux (charge neutrality condition). This rescaling can be switched off by setting `enforce_charge_neutrality=false` in the `define_charging_model` command.

Starting Position of Charged-Particle Trajectories

Since charged particles move through an electric field, the starting position of the charged particles can affect the simulation results. To change the vertical position of the plane from which the charged particles start, use parameter `source_position` in the `define_etch_machine` command (see [define_etch_machine on page 209](#)).

Note:

The source position must lie inside the simulation domain.

To extend the simulation domain in the vertical direction, you can add a layer of Gas by using the `fill material=Gas thickness=<n>` command or the parameter `top_gas_thickness` in the `etch` or `filter_structure` command.

Transfer of Impinging Charges to the Surface

At the surface, incoming ions and electrons transfer their charges to the surface. During an etching or deposition process, the surface charges move with the dynamically evolving surface. To control the amount of charges transferred to the surface, you can set the `charging_factor` parameter in the `define_charging_model` command (see [define_charging_model on page 189](#)).

By calibrating this parameter, you can effectively control the strength of the charge-up effect in your simulation. This allows you to calibrate the model to experimental data using a single `charging_factor` parameter. Reasonable values for the `charging_factor` lie in the order of 1. Alternatively, you can explicitly set the dose of charges deposited on the surface (parameter `charge_dose`), which gives you full control over the charge-up. The charge dose is defined as the number of charges per area of a flat reference surface.

To improve the statistics and smoothness of the results, you can optionally increase the number of samples sent to the surface during a charge-up simulation (parameter `num_samples_per_cell`). Internally, when increasing the number of samples, the charge per sample is reduced proportionally such that the physical total charge stays invariant. Equivalently, the charge per sample can be reduced (parameter `charge_scaling`), which will result in an up-scaling of the number of samples to keep the total charge fixed. (Charge scaling is simply a numeric optimization to smooth the distribution of charges along the surface. It affects neither the ion and electron trajectories nor the charge density used for the electric field computation.)

Electric Material Properties

For a realistic evaluation of the electric field induced by surface charges, it is essential to know the electric properties of the materials. On dielectric or insulating surfaces, incoming charges are trapped and remain at localized positions [1]. On conducting surfaces, on the other hand, deposited charges flow away and spread immediately over the entire conductor surface. Even if materials are not exposed to the surface, polarization and capacitance effects inside these materials can still influence the electric field in the gas phase due to the long-range nature of the electrostatic force. Therefore, the electric properties of each material in the structure must be defined, regardless of whether or not the material is exposed to the surface.

Different types of medium can be defined using the `define_species_properties` command:

- A dielectric medium (by specifying its relative permittivity):

```
define_species_properties name=<c> species=<c> permittivity=<n>
```

- A conductor (by specifying any nonzero conductivity):

```
define_species_properties name=<c> species=<c> conductivity=<n>
```

Note:

The PMC charge-up model supports only ideal conductors. Any material with a nonzero conductivity is treated as an ideal conductor, that is, the value of the conductivity is not relevant for the electric field computation.

By default, conducting materials are treated as floating materials, which means that their electric potential is not fixed, but depends on their environment. To model a conductor with a fixed potential (for example, a grounded or contacted conductor), a constant electric potential can optionally be applied; it remains constant during the entire simulation even if the conductor surface is hit by incoming charges. To fix the potential on a conducting material, a (virtual) point-like contact at any location in the inside of the conductor must be specified:

```
define_electric_contact name=<c> position=<v> potential=<n>
```

See [define_electric_contact on page 205](#). Then, the name of the contact must be passed to the `define_charging_model` command.

Note:

For both floating and contacted conductors, the potential inside the conductor is spatially constant (general property of an ideal conductor). If two conducting materials touch, then they are automatically treated as a single joined conductor with a common spatially constant potential. If two floating conductors merge, then their total charges are summed. If a floating conductor and a contacted conductor

Chapter 5: Surface Charging

Electric Field Computation

merge, then the floating conductor acquires the fixed potential of the contacted conductor. If two conductors with different fixed potentials merge, then an error message is generated, stating that an electric shortcut has been detected.

The domain boundaries can also behave like conductors, for example, if the bottom boundary with Dirichlet boundary conditions is grounded at a fixed potential of 0 V.

Electric Field Computation

The electric field caused by the surface charges is updated at regular time intervals for the entire simulation domain. The update times can be set by the parameter `electric_field_update_interval` in the `etch` command (see [etch on page 339](#)).

At every electric field update, the main PMC simulation stops, and the new electric field is computed. The computation of the electric field requires solving the Poisson equation for the given charge distribution, inhomogeneous material properties, and boundary conditions (BC). This is performed by the electric field solver module called GRC.

Boundary Conditions

The BC for the electric field must be specified on all boundary planes of the simulation domain:

- On the lateral boundary planes at $x = x_{\min}$, $x = x_{\max}$, $y = y_{\min}$, and $y = y_{\max}$, the BC are adopted automatically from the specified structure (see [define_boundary_conditions on page 184](#)).

In the case of reflective BC, Neumann BC are used for the electric field, $\vec{E} \cdot \hat{n} = 0$, where \hat{n} denotes the surface normal of the boundary plane. In the case of periodic BC, the electric field also satisfies periodic BC:

$$\vec{E}(x_{\min}, y, z) = \vec{E}(x_{\max}, y, z), \vec{E}(x, y_{\min}, z) = \vec{E}(x, y_{\max}, z)$$

- On the bottom and top boundary planes at $z = z_{\min}$ and $z = z_{\max}$, three types of BC can be specified by using the parameters `bottom_bc` and `top_bc` in the `define_charging_model` command:

- `dirichlet`: Sets the electric potential ϕ on that boundary plane to a spatially constant value. The boundary potential value is set with the parameters `bottom_potential` and `top_potential` (default: 0 V).
- `neumann`: Sets the normal component of the electric field, $\vec{E} \cdot \hat{n}$, to a fixed value. By default, $\vec{E} \cdot \hat{n} = 0$ V/m, but the right-hand side value can be changed with the parameters `bottom_electric_field` and `top_electric_field`.

Chapter 5: Surface Charging

Electric Field Computation

- floating: The value of the spatially constant potential is not fixed on the boundary plane, but computed by the solver. This BC allows you to model a floating conducting plane.

The BC at the bottom and top of the simulation domain can be chosen independently of each other, but you must set the value of the electric potential in at least one point to obtain a unique solution of the Poisson equation. A common practice is to use Dirichlet BC with $\phi=0$ V on the bottom plane and Neumann BC with $\vec{E} \cdot \hat{e}_z = E_z$ on the top plane, where E_z is the vertical component of the time-averaged electric field in the plasma sheath.

E_z can be either extracted from a plasma sheath simulation (see plasma extraction quantity `wall_electric_field_z`; see [Table 35](#)) or approximated by:

$$E_z = -\frac{|V_{DC}|}{d_s}$$

where V_{DC} is the DC part of the sheath potential, and d_s is the sheath thickness. Assuming typical values $V_{DC} = 100\text{--}1000$ V and $d_s = 1\text{--}10$ mm, the electric field E_z lies in the range from 10^4 to 10^6 V/m.

Poisson Solver: Guidelines

This section provides guidelines for working with the Poisson solver.

Choosing the Mesh Size of the Solver

Different optional parameters can be adjusted in the `define_electric_solver` command to optimize the speed, robustness, and accuracy of the Poisson solver. The most important parameter is `grc_grid_refinement_ratio`, which controls the mesh size of the Poisson solver in units of the PMC spacing.

For large domain sizes, to reduce computation time, you should successively increase the grid refinement ratio (preferably in powers of 2) until a reasonable balance between speed and accuracy is found. This might require carefully checking the consistency of the simulation results.

Optimizing the Solver Speed

Different solvers and preconditioning methods are available and can be selected by using the parameters `grc_solver` and `grc_preconditioner`. Depending on the problem, simulation speed is optimized by choosing a different solver or preconditioner or both.

The internal convergence thresholds of the iterative linear solver (parameters `grc_solver_abs_error` and `grc_solver_rel_error`) can also be configured for optimization. The absolute error is defined as the residuum of the linear equation system. If

Chapter 5: Surface Charging

Electric Field Computation

the absolute residuum becomes smaller than `grc_solver_abs_error`, then the solver is considered to have converged and the solution is returned. Otherwise, the relative error is computed, which is defined as the residual of the current iteration divided by the residual of the initial linear equation system. If the relative error becomes smaller than `grc_solver_rel_error`, then the solver is considered to have converged.

Note:

By default, `grc_solver_abs_error=0`, that is, the absolute error check is switched off. This is a reasonable starting point for solver optimization. To speed up convergence, you can increase `grc_solver_abs_error` slightly, while carefully checking the consistency of the simulation results.

Error Handling

Depending on the problem, the linear solver might fail to find a solution. In that case, by default, the computation restarts automatically with different solver and preconditioner settings (the automatic solver switch can be deactivated by using `grc_optimize_solver=false`).

The maximum number of solver iterations (`grc_max_num_iterations`) can be configured. If the number of iterations exceeds the maximum number of iterations, then an error message is generated. In that case, increase `grc_max_num_iterations`.

Choosing the Update Interval for the Electric Field

When simulating charge-up during a PMC etching or deposition simulation, a critical issue is how to choose the time interval at which the electric field is recomputed (parameter `electric_field_update_interval` in the `etch` command; see [etch on page 339](#)).

Theoretically, in an etching process or a deposition process, the electric field changes after every geometric change of the surface, which is after every executed PMC surface reaction. Practically, a reasonable balance needs to be found between accuracy and speed. A reasonable choice for the update time interval is the time span needed to etch or deposit a layer of one PMC spacing. An estimate for this time span can be obtained, for example, from the measured etching rate in a test simulation without charging.

To speed up the electric field update, it is reasonable to recompute the electric field only if the distribution of surface charges has changed significantly. Therefore, as an optimization, a threshold for the change of the surface charges (parameter `electric_field_update_accuracy` in the `etch` command) can optionally be set. If the change of the surface charges is less than `electric_field_update_accuracy`, then the electric field is not recomputed. The criterion for an electric field update is defined by:

$$\left(\int_{\Omega} (\rho(\vec{r}, t) - \rho(\vec{r}, t - \Delta t))^2 d^3 r \right)^{1/2} > \varepsilon \cdot \left(\int_{\Omega} \rho^2(\vec{r}, t) d^3 r \right)^{1/2} \quad (74)$$

Chapter 5: Surface Charging

Plotting Electric Field, Potential, Charge Density, and Ion Trajectories

where $\rho(\vec{r}, t)$ denotes the charge density at time t , Δt is the electric field update interval, ϵ is the electric field update accuracy, and the integration is over the entire simulation domain Ω .

Plotting Electric Field, Potential, Charge Density, and Ion Trajectories

You can save datasets either at the end of a PMC simulation or during a PMC simulation (as intermediate plots at regular time intervals).

Available datasets are the electric field (vector field), the electric potential (scalar field), the charge density (scalar field), and the ion trajectories (1D lines).

Writing Charging Datasets at the End of a Simulation

To save a dataset at the end of a simulation, call the command:

```
save type=<c> [point_min=<v>] [point_max=<v>] [file=<c>]
```

where `type` is any of `electric_field`, `electric_potential`, or `charge_density`. For large domain sizes, the loading time of saved TDR files decreases by restricting the domain to the volume of interest using the parameters `point_min` and `point_max`.

Caution:

Ion trajectories cannot be plotted in the `save` command. To plot trajectories, use intermediate plots.

Writing Charging Datasets at Intermediate Time Points

To save intermediate plots while the simulation is running, call the `etch` command with the following parameters:

```
etch ... plot_type=<l> plot_interval=<n>
```

where `plot_type` is a list of datasets, for example, `electric_field`, `electric_potential`, `charge_density`, and `trajectories`. Note that, for charge-up-related datasets, the `plot_interval` must be a multiple of the `electric_field_update_interval`.

As an alternative to a regular plot interval, you can define a list of plot time points by using the parameter `plot_times`:

```
etch ... plot_type=<l> plot_times=<v> [plot_times_unit=<c>]
```

Chapter 5: Surface Charging

Extracting the Electric Field, Potential, and Charge Density

Analogously to `plot_interval`, each time point in `plot_times` must be a multiple of the `electric_field_update_interval`. To change the unit of the plot times, use the parameter `plot_times_unit`.

Before updating the file, you can use the parameter `plot_file_update_interval` to accumulate several plots. A common practice to avoid large output files is to write each intermediate plot to a separate file – one file for each time point – by setting `split_plots=true`.

Similar to the `save` command, the domain of the intermediate plots can be restricted to a sub-volume by using the parameters `point_min` and `point_max`.

Plotting Ion Trajectories

For charge-up simulations, it can be instructive to plot the trajectories of charged ions on their way from the plasma sheath toward the structure through the electric field. To plot the trajectories, simply add `trajectories` to the list of plot types in the `etch` command.

By default, 50 ion trajectories are plotted with random starting positions along the PMC source plane. If more than one ion flux is defined, then the species type of each trajectory is sampled according to the flux ratio of the ions. To change the number of trajectories, use the parameter `num_trajectories` of the `etch` command. Alternatively, the precise starting positions of the ion trajectories on the source plane can be defined as a list of xy pairs in the parameter `trajectory_starting_positions`.

When loading the trajectory plots in Sentaurus Visual, the trajectories can be colorized to see the local ion velocity, kinetic energy, or charge along each trajectory.

Note:

The plotted trajectories are computed only for visualization purposes and are not connected to the actual PMC simulation. In particular, the parameters `num_trajectories` and `trajectory_starting_positions` have no influence on the actual PMC simulation.

Extracting the Electric Field, Potential, and Charge Density

Different quantities can be extracted as a function of time, such as:

- Charge density for each bulk species in units of [e/nm^3]
- Net charge density of all species in units of [e/nm^3]
- Electric field (x-, y-, z-components, and the absolute value) in units of [V/m]
- Electric potential in units of [V]

Chapter 5: Surface Charging

Extracting the Electric Field, Potential, and Charge Density

All these quantities can be extracted in different formats:

- Scalar results (summed or averaged over a specified volume)
- Two-dimensional results (on an axis-aligned plane)
- Three-dimensional results (on a 3D bounding box)

The results are written to a TDR file, CSV file, or Tcl variable.

To define the extraction type, call the `define_extraction` command with `type=pmc_data` and provide a list of datasets (`charge_density`, `electric_field`, or `electric_potential`) in the parameter `quantities` (see [define_extraction on page 218](#)):

```
define_extraction name=<c> type=pmc_data quantities=<l>
```

The `define_extraction` command supports many other optional parameters:

- When extracting `charge_density`, you can specify a list of species names for which you want to extract the charge density (parameter `species`). By default, the charge density of **all** species is extracted.
- For large domain sizes, you can speed up the loading of saved TDR files by restricting the extraction domain to a sub-volume, using the parameters `point_min` and `point_max`.
- By default, the mean value of each quantity is extracted as a function of time, averaged over the entire simulation domain (or, if specified, over the sub-volume defined by `point_min` and `point_max`). Alternatively, the maximum or minimum value, standard deviation, and sum or mean of the absolute values (parameter `operator=mean|sum|max|min|stddev|sumabs|meanabs`) can be computed.
- Three-dimensional grid output can be produced for all quantities by setting `grid_output=true`.
- The quantities on a 2D cutplane, specified by a normal vector (parameter `axis`), and a point on the plane (parameter `position`) can be extracted.
- The extraction results can be saved in a Tcl variable by using the parameter `tcl_output_variable`.
- In addition to a TDR file, you can save the extraction results to a CSV file by specifying the parameter `csv_file`.
- The default grouping of the time-dependent datasets stored in a TDR file, which will affect the dataset grouping in Sentaurus Visual, can be changed. By default, datasets are labeled by a `pmc_data` tag. When setting `tdr_file_dataset_grouping=quantity`, datasets will be instead labeled by the quantity name in Sentaurus Visual.

Chapter 5: Surface Charging

Example Project

Finally, you must register the name of the defined extraction in the `etch` command and specify a regular time interval at which the data is extracted:

```
etch ... extraction=<c> extraction_interval=<n>
```

In analogy to intermediate plots, a list of extraction time points can be provided instead:

```
etch ... extraction=<c> extraction_times=<v> [extraction_times_unit=<c>]
```

In addition, several plots can be accumulated before updating the file by using the parameter `extraction_file_update_interval`.

Note:

As for intermediate plots, the extraction time points must be a multiple of the `electric_field_update_interval`.

Example Project

The following code shows the essential parts of a command file for a charge-up simulation:

1. Flux distribution: Specify the energy distribution of ion fluxes. For realistic energy distributions, use the plasma model.

```
define_species_distribution name=sd species=I flux=1e-4 \
    angle_spread=0.5 energy_min=10 energy_max=100
```

2. Species properties: For ion fluxes, mass and charge must be specified. For insulating materials, specify the permittivity. For conductors, specify a conductivity > 0. The exact value is not important.

```
define_species_properties name=sp species=I mass=40 charge=1
define_species_properties name=sp species=Photoresist \
    permittivity=4.2
define_species_properties name=sp species=Aluminum conductivity=1
```

3. Charging model and electric solver: Use these commands to define properties of the charging model and of the electric field solver.

```
define_charging_model name=cm charging_factor=1 \
    bottom_bc=dirichlet bottom_potential=0 \
    top_bc=neumann top_electric_field=-1e5
```

```
define_electric_solver name=es grc_grid_refinement_ratio=4
```

4. Etch machine: Add the charging model to the etch machine.

```
define_etch_machine model=mo species_distribution=sd \
    species_properties=sp charging_model=cm
```

Chapter 5: Surface Charging

References

5. Etch command: Specify the electric solver and the field update interval in the `etch` command.

```
etch spacing=0.001 time=1 method=pmc electric_solver=es \
electric_field_update_interval=0.1
```

References

- [1] J. C. Arnold and H. H. Sawin, "Charging of pattern features during plasma etching," *Journal of Applied Physics*, vol. 70, no. 10, pp. 5314–5317, 1991.
- [2] G. S. Hwang and K. P. Giapis, "On the origin of the notching effect during etching in uniform high density plasmas," *Journal of Vacuum Science & Technology B*, vol. 15, no. 1, pp. 70–87, 1997.
- [3] K. P. Giapis and G. S. Hwang, "Pattern-Dependent Charging and the Role of Electron Tunneling," *Japanese Journal of Applied Physics*, vol. 37, no. 4B, pp. 2281–2290, 1998.
- [4] M. A. Vyvoda, M. Li, and D. B. Graves, "Hardmask charging during Cl₂ plasma etching of silicon," *Journal of Vacuum Science & Technology A*, vol. 17, no. 6, pp. 3293–3307, 1999.
- [5] T. Kinoshita, M. Hane, and J. P. McVittie, "Notching as an example of charging in uniform high density plasmas," *Journal of Vacuum Science & Technology B*, vol. 14, no. 1, pp. 560–565, 1996.
- [6] J. Matsui *et al.*, "The effect of topographical local charging on the etching of deep-submicron structures in SiO₂ as a function of aspect ratio," *Applied Physics Letters*, vol. 78, no. 7, pp. 883–885, 2001.
- [7] M. Wang and M. J. Kushner, "High energy electron fluxes in dc-augmented capacitively coupled plasmas. II. Effects on twisting in high aspect ratio etching of dielectrics," *Journal of Applied Physics*, vol. 107, no. 2, p. 023309, 2010.

6

Input Commands

This chapter describes the input commands of Sentaurus Topography 3D.

Command Syntax

All commands in Sentaurus Topography 3D follow the general structure:

```
command parameter1=value1 [parameter2=value2]
```

where `parameter1` and `parameter2` are parameters that are set to `value1` and `value2`, respectively.

Note:

Brackets denote optional parameters.

The type of parameter values is denoted by a character in angle brackets as follows:

- `` – Boolean (this is always either `true` or `false`)
- `<c>` – character string
- `<l>` – list of strings
- `<n>` – numeric value
- `<v>` – vector of numbers

Vectors are denoted using the Tcl array syntax (braces). For example, `{0 0 1}` is a vector normal to the horizontal plane.

- `<w>` – list of vectors of numbers

A vertical bar denotes exclusive options, such that either one parameter or another is expected by a command. For example:

```
command parameter1=value1 / parameter2=value2
```

Chapter 6: Input Commands

Command Syntax

Parentheses indicate grouping of parameters. Parentheses are *not* part of the syntax. For example:

```
[ (extraction_interval=<n> | \
    extraction_times=<v> [extraction_times_unit=<c>]) \
    [extraction_file_update_interval=<n>] \
    [force_intermediate_extraction=<b>]]
```

For examples, see [Appendix A on page 536](#).

Units

Sentaurus Topography 3D supports a set of default units (second column in [Table 7](#)) that is used whenever no unit is specified for a parameter in the command file.

You can use different units by specifying them in angle brackets after the numeric value. For example:

```
command parameter1=value1<unit1> [parameter2=value2<unit2>]
```

For example, the default unit for a rate is <um/min>. In the command file, you can also specify any of the measurement units listed in [Table 7](#) for velocity.

Table 7 Supported units in Sentaurus Topography 3D

Variable	Default		Other possible values	
	Unit	Symbol	Unit	Symbol
Angle	degree	<deg>	radian	<rad>
Angular velocity	revolution per minute	<rpm>	radian per second	<rad/s>
Density	gram per cubic centimeter	<g/cm ³ >	kilogram per cubic meter	<kg/m ³ >
Energy	electron volt	<eV>	joule	<J>
Length	micrometer	<um>	meter	<m>
			nanometer	<nm>
Surface tension	dyne per centimeter	<dyn/cm>	Newton per meter	<N/m>
Time	minute	<min>	second	<s>

Chapter 6: Input Commands

Command Syntax

Table 7 *Supported units in Sentaurus Topography 3D (Continued)*

Variable	Default		Other possible values	
	Unit	Symbol	Unit	Symbol
Velocity	micrometer per minute	<um/min>	meter per second	<m/s>
			micrometer per second	<um/s>
			nanometer per minute	<nm/min>
			nanometer per second	<nm/s>
Viscosity	poise	<poise>	Pascal second	<Pa*s>

Syntax for Expressions

Note:

Expressions are supported only if the GNU compiler gcc is installed on the system where Sentaurus Topography 3D is running and is globally available there.

Sentaurus Topography 3D supports the following expression types:

- Time-dependent expressions
- Position-dependent expressions
- Angle-dependent expressions
- Energy-dependent expressions

Their syntax consists of the following components:

- Numeric constants containing the decimal point
- Arithmetic operators: +, -, *, /
- Mathematical constants:
 - `M_E` (denotes the value of the Euler number)
 - `M_PI` (denotes the value of π)
- Mathematical functions: `fabs(u)`, `sin(u)`, `cos(u)`, `tan(u)`, `asin(u)`, `acos(u)`, `atan(u)`, `atan2(v,u)`, `sinh(u)`, `cosh(u)`, `tanh(u)`, `exp(u)`, `log(u)`, `log10(u)`, `pow(u,v)`, `sqrt(u)`, `ceil(u)`, `floor(u)`, `fmod(u,v)`

The argument of the trigonometric functions must be given in radians.

Chapter 6: Input Commands

Command Syntax

- Relational functions:

```
high_pass(u, v, w), higher_pass(u, v, w)  
low_pass(u, v, w), lower_pass(u, v, w)
```

For definitions of these functions, see [Conditional and Relational Functions on page 511](#).

- `square_pulse(u, v_on, v_off, period, duty_cycle)`
 - `square_pulse` defines a square pulse function, that is, a periodic function with a period `period`. In the first part of each period, its value is `v_on`. In the second part of each period, its value is `v_off`. The length of the first part of the period is specified by the product of the duty cycle `duty_cycle` and the period `period`. `square_pulse` is defined such that a period starts when `u=0`.
 - `period` must have a positive value, and `duty_cycle` must belong to the [0,1] range.
- `square_pulse_arg_on(u, period)`
 - It returns the difference between `u` and the beginning of the period of a square pulse of period `period` to which `u` belongs.
 - `period` must have a positive value.
- `square_pulse_arg_off(u, period, duty_cycle)`
 - It returns the difference between `u` and the largest beginning of the second part of the period of a square pulse of period `period` and duty cycle `duty_cycle`, not larger than `u`.
 - `period` must have a positive value, and `duty_cycle` must belong to the [0,1] range.
- The conditional expression "condition ? u : v", returning `u` if `condition` is true, and `v` otherwise
- Parentheses
- For time-dependent expressions:
 - The string `t` denotes the numeric value of the current simulation time `t` in minutes.
 - The string `t<unit>` denotes the numeric value of the current simulation time `t` in the specified unit (possible values for `<unit>` are `min` and `s`).
- For position-dependent expressions:
 - The strings `u`, `v`, and `w` denote the numeric value of the x-, y-, and z-coordinates of a point in micrometers, respectively.
 - The strings `u<unit>`, `v<unit>`, and `w<unit>` denote the numeric value of the x-, y-, and z-coordinates of a point in the specified unit (possible values for `<unit>` are `m`, `um`, and `nm`), respectively.

Chapter 6: Input Commands

Command Syntax

- For angle-dependent expressions, the string `theta` denotes the angle in radians between the normal to the surface at the collision point and the direction of the incoming particle.
- For energy-dependent expressions, the string `E` denotes the energy in electron volts of the incoming particle.

The arguments `duty_cycle`, `period`, `v_off`, `v_on`, `u`, `v`, and `w` of these functions are themselves expressions.

A condition is an expression that can also contain relational operators:

`<`, `<=`, `>`, `>=`, `==`, `!=`

Summary of Available Commands

Table 8 *Summary of Sentaurus Topography 3D commands*

Command	Function
<code>add_bulk_reaction</code>	Adds a new plasma bulk reaction to a specified plasma model.
<code>add_float_parameter</code>	Adds a new user-defined parameter for floating-point values to the specified rate formula module (RFM) model.
<code>add_flux_properties</code>	Specifies the values of the parameters for a flux associated with an RFM model.
<code>add_formula</code>	Defines a formula for calculating the deposition or etching rate for an RFM model.
<code>add_int_parameter</code>	Adds a new user-defined parameter for integer values to the specified RFM model.
<code>add_interface_layer</code>	Adds a layer across the interface between two materials, where a material-dependent parameter can take position-dependent values, and specifies the spatially dependent values of that parameter in that layer.
<code>add_ion_flux</code>	Adds a new ion flux to the specified RFM model.
<code>add_litho_command</code>	Adds a Sentaurus Lithography Tcl command.
<code>add_material</code>	Specifies the values of the parameters of an etch machine for a particular material.
<code>add_neutral_flux</code>	Adds a new neutral flux to the specified RFM model.

Chapter 6: Input Commands

Command Syntax

Table 8 Summary of Sentaurus Topography 3D commands (Continued)

Command	Function
add_reaction	Adds a reaction to a reaction model.
add_reaction_properties	Specifies the values of the parameters for a reaction.
add_source_species	Adds a species to the source of a reaction model.
add_species	Adds a new plasma species to a specified plasma model.
add_volumetric_source_species	Adds a species to the volumetric source of a reaction model.
define_boundary_conditions	Specifies the boundary conditions to be used when processing a structure.
define_bulk_solver	Defines the numeric parameters of the solver used to simulate the plasma bulk.
define_charging_model	Defines a charging model.
define_damage	Defines the parameters of the damage model for PMC simulations.
define_deposit_machine	Defines a machine for a built-in or an RFM deposition model.
define_electric_contact	Defines an electric contact.
define_electric_solver	Defines an electric solver.
define_etch_machine	Defines a machine for an etching model, or a simultaneous etching and deposition model, or a deposition reaction model.
define_extraction	Defines extractions that can be used during a deposition step or an etching step.
define_iad	Defines an ion angular distribution (IAD).
define_layout	Reads a GDSII, an OASIS®, or a TCAD layout file, which can then be used to create logical masks with the define_mask command.
define_litho_machine	Defines a machine for a lithographic process.
define_mask	Defines a mask to be used in an etch step or a patterning step.

Chapter 6: Input Commands

Command Syntax

Table 8 Summary of Sentaurus Topography 3D commands (Continued)

Command	Function
define_material_replacement	Defines a material replacement map.
define_model	Starts the definition of a new RFM model or a new reaction model.
define_nad	Defines a neutral angular distribution (NAD).
define_pattern_density	Defines a new pattern density object or extends the definition of an existing pattern density object.
define_pattern_density_model	Defines a defines a new pattern density model.
define_plasma_model	Defines a new plasma model.
define_probability	Defines an energy- and angle-dependent probability that can be used in a reaction model.
define_reactor	Defines the reactor parameters for a plasma model.
define_reflection	Defines the properties of a new reflection function.
define_shape	Defines a new shape for a geometric etch or deposit step.
define_sheath_solver	Defines the numeric parameters of the solver used to simulate the plasma sheath.
define_species_distribution	Defines the angular distribution of a source species of a reaction model.
define_species_properties	Defines the properties of a species used in a reaction model.
define_structure	Defines the initial 2D or 3D boundary structure.
define_volumetric_species_distribution	Defines the spatial distribution of a volumetric source species of a reaction model.
define_yield	Defines the properties of a new yield function.
deposit	Performs a deposition process step on a structure.
etch	Performs an etching process step or a simultaneous etching and deposition process step on a structure, or performs a deposition step based on a reaction model.

Chapter 6: Input Commands

Command Syntax

Table 8 Summary of Sentaurus Topography 3D commands (Continued)

Command	Function
extend_structure	Extends a structure by mirroring it, or by copying and shifting it.
extract	Extracts properties from a structure.
fill	Fills up the structure with one or more materials.
filter_structure	Executes Boolean operations, or converts a boundary structure to a PMC structure, or converts a PMC structure to a boundary structure, or creates a copy of an existing structure, or decimates the number of surface elements, or merges several regions into one region, or creates a structure with a different surface discretization, or removes parts that are disconnected or that fit user-specified criteria, or removes a region from a structure, or renames a region of a structure, or replaces the material of the regions of a structure, or smooths a structure.
finalize_model	Indicates that the definition of a model is completed.
layout	Queries the properties of a layout that has been created with the <code>define_layout</code> command.
let	Defines global settings for Sentaurus Topography 3D.
litho	Performs a lithography simulation and adds a resist region.
pattern	Performs a patterning step on a structure.
remove_material	Removes a specified material from a structure.
save	Saves a structure, or an ion angular distribution (IAD), or a probability function, or a reflection function, or a neutral angular distribution (NAD), or pattern density-related functions, or a yield function to a TDR file, or saves a species distribution to a TDR file or to a text file, or saves a PMC structure to a PMC file, or saves a grid containing the volume fractions of the species in a PMC structure to a TDR file.
set_material_properties	Sets or changes the properties of materials.
set_orientation	Sets or changes the crystal orientation of a region.
solve_reactor	Executes the plasma model solvers.

Chapter 6: Input Commands

Command Syntax

Table 8 Summary of Sentaurus Topography 3D commands (Continued)

Command	Function
transform_structure	Performs an axis-mapping transformation.
truncate	Truncates a structure.

Chapter 6: Input Commands

add_bulk_reaction

add_bulk_reaction

This command adds a new plasma bulk reaction to a specified plasma model.

Note:

If you specify an `add_bulk_reaction` command with the name of a plasma bulk reaction that already exists, then this command is ignored.

Syntax

```
add_bulk_reaction a=<n> b=<n> c=<n> [d=<n> e=<n>] \
    energy_transfer=<n> expression=<c> name=<c> \
    plasma_model=<c> rate_coefficient_type=<c> [T_ref=<n>]
```

Table 9 Parameters of `add_bulk_reaction` command

Parameter	Type	Description	Default [Range]	Unit
a	Number	Sets an Arrhenius law coefficient. The unit is as follows: <ul style="list-style-type: none">• none: For reactions of <code>rate_coefficient_type=general_arrhenius</code>• 1/s: For one-body reactions of <code>rate_coefficient_type=arrhenius</code>• m³/s: For two-body reactions of <code>rate_coefficient_type=arrhenius</code>	none [0, +∞[none, or 1/s, or m ³ /s
b	Number	Sets an Arrhenius law coefficient for any value of the <code>rate_coefficient_type</code> parameter.	none]-∞, +∞[none
c	Number	Sets an Arrhenius law coefficient: <ul style="list-style-type: none">• For <code>rate_coefficient_type=arrhenius</code>, the unit is eV.• For <code>rate_coefficient_type=general_arrhenius</code>, no unit is specified for the coefficient.	none]-∞, +∞[none or eV
d	Number	Sets an Arrhenius law coefficient for <code>rate_coefficient_type=general_arrhenius</code> .	none]-∞, +∞[none
e	Number	Sets an Arrhenius law coefficient for <code>rate_coefficient_type=general_arrhenius</code> .	none]-∞, +∞[none

Chapter 6: Input Commands

add_bulk_reaction

Table 9 Parameters of add_bulk_reaction command (Continued)

Parameter	Type	Description	Default [Range]	Unit
energy_transfer	Number	Sets the energy transferred during the reaction. The following sign conventions apply: <ul style="list-style-type: none"> Positive for endothermal reactions Negative for exothermal reactions 	none]-∞, +∞[eV
expression	Character	Sets the expression of the reaction added to the plasma model.	none	none
name	Character	Sets the name of the reaction. The name is used as a unique label for this reaction.	none	none
plasma_model	Character	Sets the name of the plasma model to which the reaction is added. It must be already defined by define_plasma_model on page 258 .	none	none
rate_coefficient_type	Character	Sets the mathematical form of the rate coefficient. Options are: <ul style="list-style-type: none"> arrhenius: $a\left(\frac{T_e}{T_{ref}}\right)^b \exp\left(-\frac{c}{T_e}\right)$ general_arrhenius: $\exp\left(a + b \ln T_e + \frac{c}{T_e} + \frac{d}{T_e^2} + \frac{e}{T_e^3}\right)$ where T_e is the electron temperature in eV. 	none	none
T_ref	Number	Sets an Arrhenius law coefficient for rate_coefficient_type=arrhenius.	1 [0, +∞[eV

Plasma Bulk Reactions

A plasma bulk reaction specifies a rule to transform a set of reactants into a set of products. A reaction expression must be given in the following format:

<R1> = n1<P1> + n2<P2> + ...

or:

<R1> + <R2> = n1<P1> + n2<P2> + ...

Chapter 6: Input Commands

add_bulk_reaction

where:

- <R1> and <R2> are the species names of the reactants (that is, the state before the reaction). The number of reactants must be one or two.
- <P1>, <P2>, ... are the species names of the reaction products (that is, the state after the reaction). The number of products is arbitrary.
- n1, n2, ... are the stoichiometric coefficients of the reaction products and must be positive integers. If you do not specify the stoichiometric coefficient, the default value of 1 is assumed.
- Each species name used in a reaction expression must have been previously defined within the specified model by the add_species command.
- The left-hand side (reactants) and the right-hand side (products) of the reaction expression are separated by an equal sign (=) with space on either side of the equal sign, and the different species on either side of the expression are separated by a plus sign (+) with space on either side of the plus sign.
- The species name e- is reserved for electrons, which are implicitly defined in the plasma model and, therefore, do not have to be defined explicitly by the add_species command.

Note:

Rate coefficients must be given in the form of an Arrhenius law:

$$r = a \langle m^3/s \rangle \cdot \left(\frac{T_e}{1 \langle eV \rangle} \right)^b \exp\left(-\frac{c \langle eV \rangle}{T_e}\right)$$

where the parameters a , b , and c are given for the electron temperature T_e in eV, and $T_{ref} = 1 \text{ eV}$. If, instead, you want to specify the parameters a , b , and c for the electron temperature T_e in K, then you must set $T_{ref} = 1 \text{ K}$:

$$r = a \langle m^3/s \rangle \cdot \left(\frac{T_e}{1 \langle K \rangle} \right)^b \exp\left(-\frac{c \langle K \rangle}{T_e}\right)$$

If you want to plot the rate coefficients, then you can use the save command to write the rate coefficient curve as a function of the electron temperature to a TDR file.

Examples

If the reaction coefficients are given for T_e in units of eV:

```
add_bulk_reaction plasma_model=M name=r1 \
    expression="Ar + e- = Ar+ + 2e-" \
    rate_coefficient_type=arrhenius energy_transfer=16<eV> \
    a=2e-14<m^3/s> b=0.6 c=16<eV> T_ref=1<eV>
```

Chapter 6: Input Commands

add_bulk_reaction

If data is given for T_e in kelvin instead of eV:

```
add_bulk_reaction plasma_model=M name=r1 \
expression="Ar + e- = Ar+ + 2e-" \
rate_coefficient_type=arrhenius energy_transfer=16<eV> \
a=6e-17<m^3/s> b=0.6 c=186400<K> T_ref=1<K>
```

Chapter 6: Input Commands

add_float_parameter

add_float_parameter

This command adds a new user-defined parameter for floating-point values to the specified RFM model.

For parameters defined as globally valid:

- You must specify the parameter `default` when `optional=true`.
The value specified with `default` is used when `optional=true` and the parameter has not been set explicitly with the `define_deposit_machine` or `define_etch_machine` command (see [define_deposit_machine on page 195](#) and [define_etch_machine on page 209](#)).
- You must not specified the parameter `default` when `optional=false`.

For material-dependent parameters, you must set the parameter `default` independently of the value of the `optional` parameter. The value specified with `default` is used for materials for which parameters have not been specified with the `add_material` command. The value is also used if `optional=true` and the parameter has not been set explicitly with the `add_material` command (see [add_material on page 160](#)).

Syntax

For deposition:

```
add_float_parameter default=<n> model=<c> name=<n> quantity=<c> \
[description=<c>] [max=<n>] [min=<n>] [optional=<c>]
```

For etching and simultaneous etching and deposition:

```
add_float_parameter default=<n> model=<c> name=<n> quantity=<c> \
scope=<c> [description=<c>] [max=<n>] [min=<n>] [optional=<c>]
```

Table 10 Parameters of add_float_parameter command

Parameter	Type	Description	Default	Unit
default	Number	Sets the default value to assign to the new parameter for unknown materials and also if the new parameter is optional.	none	Default unit of quantity set by quantity parameter
description	Character	Describes the new parameter.	empty string	none

Chapter 6: Input Commands

add_float_parameter

Table 10 Parameters of add_float_parameter command (Continued)

Parameter	Type	Description	Default	Unit
max	Number	Sets the maximum value of the new parameter. Use the <code>max</code> and <code>min</code> parameters to limit the valid range of the new parameter.	∞	Default unit of quantity set by quantity parameter
min	Number	Sets the minimum value of the new parameter. Use the <code>max</code> and <code>min</code> parameters to limit the valid range of the new parameter.	$-\infty$	Default unit of quantity set by quantity parameter
model	Character	Sets the RFM model to which the new parameter is added. Note: The <code>model</code> parameter must specify an RFM model, that is, the <code>type</code> parameter must have been set in the <code>define_model</code> command for that model (see define_model on page 251).	none	none
name	Character	Sets the name of the new parameter. When <code>scope=global</code> , these values cannot be used: <code>applied_pressure</code> , <code>deposit_material</code> , <code>iad</code> , <code>material</code> , <code>maximum_error</code> , <code>model</code> , <code>name</code> , <code>pad_poisson_ratio</code> , <code>pad_roughness</code> , <code>pad_young_modulus</code> , <code>reflection</code> , <code>rotation</code> , <code>tilt</code> , <code>yield</code> When <code>scope=material_dependent</code> , the value <code>material</code> cannot be used.	none	none

Chapter 6: Input Commands

add_float_parameter

Table 10 Parameters of add_float_parameter command (Continued)

Parameter	Type	Description	Default	Unit
optional	Boolean	<p>Specifies whether the new parameter is optional when the model is used.</p> <p>Note: In general, do not make a parameter optional. Instead, simplify a model to such a degree that all user-defined parameters are required.</p>	false	none
quantity	Character	<p>Sets the physical quantity of the new parameter value. The expression for the rate defined with the <code>add_formula</code> command must have the dimension of a velocity (see add_formula on page 149). Options are:</p> <ul style="list-style-type: none"> • angle (default unit: degree) • dimensionless (default unit: 1) • energy (default unit: eV) • length (default unit: μm) • time (default unit: minute) • velocity (default unit: μm/minute) 	none	none
scope	Character	<p>Sets how the new parameter is defined. Options are:</p> <ul style="list-style-type: none"> • global: New parameter is valid globally. • material_dependent: New parameter is material dependent. <p>The scope parameter applies only to etching and simultaneous etching and deposition.</p>	none	none

Chapter 6: Input Commands

add_float_parameter

Examples

Add a parameter for a deposition model:

```
define_model description="Advanced CVD model" name=depo_CVD \
    type=deposit

add_float_parameter default=12 model=depo_CVD name=tilt \
    quantity=angle description="Irrational tilt angle" min=5 max=45
```

The `model` parameter is dependent on specifying the `type` parameter in the `define_model` command beforehand. In this example, since `quantity=angle`, the default unit for the `default`, `max`, and `min` parameters is degree.

Add a parameter for an etching model:

```
define_model description="Modified RIE model" name=etch_rie \
    type=etch
add_float_parameter default=2 model=etch_rie name=sim_time \
    quantity=time scope=global
```

The `model` parameter is dependent on specifying the `type` parameter in the `define_model` command beforehand. In this example, since the new parameter is for an etching model, the `scope` parameter is required.

Chapter 6: Input Commands

add_flux_properties

add_flux_properties

This command specifies the parameters of a flux associated with an RFM model. There are different forms of the command for neutral and ion fluxes.

Syntax

For neutral fluxes, you must specify the value of the sticking coefficient. For deposition models, the sticking coefficient is material independent (otherwise, it is material dependent):

```
add_flux_properties flux=<c> sticking=<n> [machine=<c>]  
add_flux_properties flux=<c> material=<c> sticking=<n> [machine=<c>]
```

If `sputter_deposition` has been activated for an ion flux during the model definition (see [add_ion_flux on page 157](#)), then the values for the parameters `sputter_exponent` (exponent of the angular distribution of the emission of the sputtered material), `sputter_type` (the type of emission of the sputtered material), and `sticking` (the sticking coefficient of the reemitted material) must be specified. Otherwise, these parameters must not be specified.

For ion fluxes, in deposition models, the flux parameters are material independent; whereas, for etching and simultaneous etching and deposition, they are material dependent:

```
add_flux_properties flux=<c> [machine=<c>] [sputter_exponent=<n>] \  
[sputter_type=<c>] [sticking=<n>]  
  
add_flux_properties flux=<c> material=<c> [machine=<c>] \  
[sputter_exponent=<n>] [sputter_type=<c>] [sticking=<n>]
```

The `flux` parameter takes the name that was given to a flux in the model definition with the `add_neutral_flux` or `add_ion_flux` command.

Note:

If `sputter_exponent`, `sputter_type`, or `sticking` has been fixed in the model definition (given a constant value), then you cannot specify the parameter again with the `add_flux_properties` command.

Chapter 6: Input Commands

add_flux_properties

Table 11 Parameters of add_flux_properties command

Parameter	Type	Description	Default [Range]	Unit
flux	Character	Sets the name of the flux to configure.	none	none
machine	Character	Sets the name of the machine for which the flux is configured. Note: The <code>machine</code> parameter must specify a machine using an RFM model, that is, the <code>type</code> parameter must have been set in the <code>define_model</code> command for that model (see define_model on page 251).	default_machine	none
material	Character	Sets the material for which the properties are specified.	none	none
sputter_exponent	Number	Sets the exponent used to characterize the angular distribution $\cos^m\theta$ of the sputtered material. The value must be an integer.	none [1, ∞[none
sputter_type	Character	Sets the sputter reemission type of the model (see add_ion_flux on page 157). Options are: <ul style="list-style-type: none">• diffuse• reflective	none	none
sticking	Number	Sets the sticking coefficient.	none [0,1]	none

Examples

```
# Define a model with sputter deposition activated that fixes
# the sputter exponent and type, but not the sticking.
define_model name=m type=deposit description=""

add_ion_flux model=m name=i energy=independent reflection=false \
    sputtering=true sputter_deposition=true sputter_exponent=1 \
    sputter_type=diffuse
```

Chapter 6: Input Commands

add_flux_properties

```
add_formula model=m expression="sputter_depo_flux(i)-direct_flux(i)"

finalize_model model=m

# ...
# When model is used:
define_deposit_machine model=m material=Oxide ...

# OK: Parameter sticking is required because sputter deposition is
# switched on in the model definition.
add_flux_properties model=m flux=i sticking=0.9

# Error: sputter_exponent has a constant value of 1 and cannot be
# specified again here when model is used.
add_flux_properties model=m flux=i sputter_exponent=100 sticking=0.9
```

Chapter 6: Input Commands

add_formula

add_formula

This command defines the formula for calculating the deposition or etching rate. To make it easier to develop and maintain an RFM model, you can define subexpressions and use them in the definition of subsequent subexpressions and the main expression for the rate.

For etching and simultaneous etching and deposition, there is a material independent form and material dependent form of the `add_formula` command:

- The material-independent form defines a default rate formula. The default rate formula is used for all materials for which no specific rate formula has been defined.
- The material-dependent form defines a rate formula for a specific material.

[Rate Calculation on page 511](#) describes the data and functions available for defining the formula for the rate. In addition, previously defined subexpressions can be used as part of the new rate formula.

Syntax

For deposition and for default rate formulas (material-independent form):

```
add_formula model=<c> name=<c> subexpression=<c> [unit=<c>]  
add_formula expression=<c> model=<c> [unit=<c>]
```

For material-dependent rate formulas:

```
add_formula material=<c> model=<c> name=<c> subexpression=<c> [unit=<c>]  
add_formula expression=<c> material=<c> model=<c> [unit=<c>]
```

Table 12 Parameters of `add_formula` command

Parameter	Type	Description	Default	Unit
expression	Character	Sets the formula used to calculate the rate.	none	none
material	Character	Sets the material for which the formula is used.	none	none
model	Character	Sets the RFM model to which the formula is added. Note: The <code>model</code> parameter must specify an RFM model, that is, the <code>type</code> parameter must have been set in the <code>define_model</code> command for that model.	none	none

Chapter 6: Input Commands

add_formula

Table 12 Parameters of *add_formula* command (Continued)

Parameter	Type	Description	Default	Unit
name	Character	Sets the name of the subexpression.	none	none
subexpression	Character	Sets the formula used to calculate the subexpression.	none	none
unit	Character	Sets the unit of the expression or subexpression. Options are: <ul style="list-style-type: none">• nm min^-1• nm s^-1• um min^-1• um s^-1	um min^-1	none

Chapter 6: Input Commands

add_int_parameter

add_int_parameter

This command adds a new user-defined parameter for integer values to the specified RFM model.

For parameters defined as globally valid:

- You must specify the parameter `default` when `optional=true`.
The value specified with `default` is used when `optional=true` and the parameter has not been set explicitly with the `define_deposit_machine` or `define_etch_machine` command (see [define_deposit_machine on page 195](#) and [define_etch_machine on page 209](#)).
- You must not specified the parameter `default` when `optional=false`.

For material-dependent parameters, you must set the parameter `default` independently of the value of the `optional` parameter. The value specified with `default` is used for materials for which parameters have not been specified with the `add_material` command. The value is also used if `optional=true` and the parameter has not been set explicitly with the `add_material` command (see [add_material on page 160](#)).

Syntax

For deposition:

```
add_int_parameter default=<n> model=<c> name=<c> \
[description=<c>] [max=<n>] [min=<n>] [optional=<c>]
```

For etching and simultaneous etching and deposition:

```
add_int_parameter default=<n> model=<c> name=<c> scope=<c> \
[description=<c>] [max=<n>] [min=<n>] [optional=<c>]
```

Table 13 Parameters of `add_int_parameter` command

Parameter	Type	Description	Default [Range]	Unit
default	Number	Sets the default value to assign to the new parameter for unknown materials and also if the new parameter is optional.	none	none
description	Character	Describes the new parameter.	empty string	none
max	Number	Sets the maximum value of the new parameter. Use the <code>max</code> and <code>min</code> parameters to limit the valid range of the new parameter.	2147483647 [-2147483648, 2147483647]	none

Chapter 6: Input Commands

add_int_parameter

Table 13 Parameters of add_int_parameter command (Continued)

Parameter	Type	Description	Default [Range]	Unit
min	Number	Sets the minimum value of the new parameter. Use the <code>max</code> and <code>min</code> parameters to limit the valid range of the new parameter.	-2147483647 [-2147483648, 2147483647]	none
model	Character	Sets the RFM model to which the integer parameter is added. Note: The <code>model</code> parameter must specify an RFM model, that is, the <code>type</code> parameter must have been set in the <code>define_model</code> command for that model (see define_model on page 251).	none	none
name	Character	Sets the name of the new parameter. When <code>scope=global</code> , these values cannot be used: <code>applied_pressure</code> , <code>deposit_material</code> , <code>iad</code> , <code>material</code> , <code>maximum_error</code> , <code>model</code> , <code>name</code> , <code>pad_poisson_ratio</code> , <code>pad_roughness</code> , <code>pad_youth_modulus</code> , <code>reflection</code> , <code>rotation</code> , <code>tilt</code> , <code>yield</code> When <code>scope=material_dependent</code> , the <code>material</code> value cannot be used.	none	none
optional	Boolean	Specifies whether the new parameter is optional when the model is used. Note: In general, do not make a parameter optional. Instead, simplify a model to such a degree that all user-defined parameters are required.	false	none
scope	Character	Sets how the new parameter is defined. Options are: <ul style="list-style-type: none">• <code>global</code>: New parameter is valid globally.• <code>material_dependent</code>: New parameter is material dependent. The <code>scope</code> parameter applies only to etching and simultaneous etching and deposition.	none	none

Chapter 6: Input Commands

add_interface_layer

add_interface_layer

This command adds a layer across the interface between two different materials, where a numeric material-dependent parameter can take position-dependent values, and specifies the spatially dependent values of that parameter in that layer.

Note:

The `add_interface_layer` command can be used only for machines using level set-based models. This command cannot be used for deposition machines or for machines using simultaneous etching and deposition models.

When using a built-in model, the `add_interface_layer` command is not supported for the sticking and reflection parameters.

This command uses the initial structure to determine the spatially dependent parameter values and has different versions.

Syntax

For a linear grading of the specified parameter across the interface between its bulk values, which are specified with the `add_material` command:

```
add_interface_layer material1=<c> material2=<c> parameter=<c> \
thickness1=<n> thickness2=<n> [machine=<c>]
```

For a piecewise linear grading of the specified parameter across the interface between its bulk values, which are specified with the `add_material` command:

```
add_interface_layer material1=<c> material2=<c> parameter=<c> \
table1=<v> table2=<v> \
[machine=<c>] [table1_distance_unit=<c>] [table2_distance_unit=<c>]
```

Table 14 Parameters of `add_interface_layer` command

Parameter	Type	Description	Default [Range]	Unit
machine	Character	Sets the name of the machine to which an interface layer is added.	default_machine	none
material1	Character	Sets the name of the first material of the interface. It must be a material different from the material specified with <code>material2</code> .	none	none
material2	Character	Sets the name of the second material of the interface. It must be a material different from the material specified with <code>material1</code> .	none	none

Chapter 6: Input Commands

add_interface_layer

Table 14 Parameters of add_interface_layer command (Continued)

Parameter	Type	Description	Default [Range]	Unit
parameter	Character	Sets the name of the material-dependent parameter for which an interface layer is specified. The name must be one of the material-dependent parameter names of the machine specified by machine. For RFM models, it must be one of the parameters defined using the add_float_parameter command.	none	none
table1	Vector	Sets the tabular format of the piecewise linear scaling of the parameter value in the region of space occupied by the material specified by material1 (see Tabular Format on page 155).	none	none
table1_distance_unit	Character	Sets the unit of distances listed in table1. Options are: <ul style="list-style-type: none"> • nm • m • um 	um	none
table2	Vector	Sets the tabular format of the piecewise linear scaling of the parameter value in the region of space occupied by the material specified by material2 (see Tabular Format on page 155).	none	none
table2_distance_unit	Character	Sets the unit of distances listed in table2. Options are: <ul style="list-style-type: none"> • nm • m • um 	um	none
thickness1	Number	Sets the thickness of the interface layer in the region of space occupied by the material specified by material1.	none [0 , ∞[µm

Chapter 6: Input Commands

add_interface_layer

Table 14 Parameters of add_interface_layer command (Continued)

Parameter	Type	Description	Default [Range]	Unit
thickness2	Number	Sets the thickness of the interface layer in the region of space occupied by the material specified by material2.	none [0, ∞[μm

Tabular Format

The parameters `table1` and `table2` require a vector of numbers that fulfill the following conditions (the convention that the first vector element has index 0 is understood):

- The number of elements of the vector must be even and not less than 4.
- Vector elements with an even index represent a distance from the interface and must be nonnegative, unique, and sorted in ascending order.
- Vector elements with an odd index represent a scaling factor for the corresponding bulk value of the parameter specified with the `add_material` command and must be nonnegative.
- The first vector element (that is, the first distance) must be 0.
- The last vector element (that is, the last scaling factor) must be 1.

Examples

The following commands define an etch machine using the `simple` model, where the `rate` parameter values vary linearly across the interfaces between silicon and oxide:

```
define_etch_machine model=simple
add_material material=Silicon rate=1. anisotropy=0.5 curvature=0.
add_material material=Oxide rate=0.1 anisotropy=0.5 curvature=0.
add_material material=Nitride rate=0.5 anisotropy=0.5 curvature=0.
add_interface_layer material1=Silicon material2=Oxide parameter=rate \
    thickness1=0.1 thickness2=0.2
```

In particular:

- The value of the `rate` parameter is 1 μm/minute for points that are located in a region occupied by material `Silicon` and with a distance from a silicon–oxide interface greater than 0.1 μm.
- The value of the `rate` parameter is 0.1 μm/minute for points that are located in a region occupied by material `Oxide` and with a distance from a silicon–oxide interface greater than 0.2 μm.

Chapter 6: Input Commands

add_interface_layer

- The value of the `rate` parameter decreases linearly from 1 $\mu\text{m}/\text{minute}$ to 0.1 $\mu\text{m}/\text{minute}$ as you move along a segment that crosses the silicon–oxide interface orthogonally, starting from a point located in the silicon region at 0.1 μm from that interface and up to a point located in the oxide region at 0.2 μm from that interface.
- The value of the `rate` parameter is 0.5 $\mu\text{m}/\text{minute}$ for points located in a region occupied by material `Nitride`.

The following commands define an etch machine using an RFM model, where the `R` parameter values vary linearly across the interfaces between silicon and oxide:

```
define_model name=m type=etch description="isotropic RFM etch model"
add_float_parameter name=R model=m quantity=velocity \
    scope=material_dependent default=0
add_formula model=m expression="-R()"
finalize_model model=m

define_etch_machine model=m
add_material material=Silicon R=1
add_material material=Oxide R=2
add_interface_layer parameter=R material1=Silicon material2=Oxide \
    table1={0 0.1 1 0.5 2 1} table1_distance_unit=nm \
    table2={0 0.1 1 0.2 1.5 1} table2_distance_unit=nm
```

In particular:

- The value of the `R` parameter is 1 $\mu\text{m}/\text{minute}$ for points that are located in a region occupied by material `Silicon` and with a distance from a silicon–oxide interface greater than 2 nm.
- The value of the `R` parameter is 2 $\mu\text{m}/\text{minute}$ for points that are located in a region occupied by material `Oxide` and with a distance from a silicon–oxide interface greater than 1.5 nm.
- The value of the `R` parameter changes in a piecewise linear way from 1 $\mu\text{m}/\text{minute}$ to 2 $\mu\text{m}/\text{minute}$ as you move along a segment that crosses the silicon–oxide interface orthogonally, starting from a point located in the silicon region at 2 nm from that interface and up to a point located in the oxide region at 1.5 nm from that interface.
- The values of the `R` parameter on the silicon side of the interface layer are determined by multiplying the bulk value of the `R` parameter (that is, 1 $\mu\text{m}/\text{minute}$) by the piecewise linear scaling factor specified by the parameter `table1`.
- The values of the `R` parameter on the oxide side of the interface layer are determined by multiplying the bulk value of the `R` parameter (that is, 2 $\mu\text{m}/\text{minute}$) by the piecewise linear scaling factor specified by the parameter `table2`.

Chapter 6: Input Commands

add_ion_flux

Note:

When using the `etch` command with a machine having interface layers defined, it is recommended to set the `spacing` parameter such that the spatial changes of the parameter values can be properly resolved.

add_ion_flux

This command adds a new ion flux to the specified RFM model.

If you activate redeposition of the sputtered material, you can use the optional parameters `sputter_exponent`, `sputter_type`, and `sticking` to define constant values for the exponent of the angular distribution of the emission of the sputtered material, the type of emission of the sputtered material, and the sticking coefficient of the reemitted material, respectively. If any of these parameters is defined with the `add_ion_flux` command, its value cannot be changed with a subsequent `add_flux_properties` command.

Syntax

```
add_ion_flux energy=<c> model=<c> name=<c> reflection=<b> \
    sputter_deposition=<b> sputtering=<b> \
    [sputter_exponent=<n>] [sputter_type=<c>] [sticking=<n>]
```

Table 15 Parameters of `add_ion_flux` command

Parameter	Type	Description	Default [Range]	Unit
energy	Character	Selects whether the reflection and sputtering are energy dependent when performing flux integration, depending on the kinetic energy of the ion species. Options are: <ul style="list-style-type: none">• dependent• independent	independent	none
model	Character	Sets the RFM model to which the ion flux is added. Note: The <code>model</code> parameter must specify an RFM model, that is, the <code>type</code> parameter must have been set in the <code>define_model</code> command for that model.	none	none

Chapter 6: Input Commands

add_ion_flux

Table 15 Parameters of add_ion_flux command (Continued)

Parameter	Type	Description	Default [Range]	Unit
name	Character	Sets the name to be given to the ion flux. This name is used to reference the ion flux when configuring the flux properties using the add_flux_properties command and when accessing the flux values in the formula for calculating the etching and deposition rates.	none	none
reflection	Boolean	Specifies whether ion reflection is taken into account.	false	none
sputter_deposition	Boolean	Specifies whether to calculate the redeposition of sputtered material. Note: This parameter must be specified if sputtering=true. Otherwise, it must not be specified.	none	none
sputter_exponent	Number	Sets the distribution exponent of the sputtered material.	none [1, ∞[none
sputter_type	Character	Sets the sputter reemission type of the model. Options are: <ul style="list-style-type: none">• diffuse• reflective	none	none
sputtering	Boolean	Specifies whether to take sputtering into account.	false	none
sticking	Number	Sets the sticking coefficient.	none [0, 1]	none

Chapter 6: Input Commands

add_litho_command

add_litho_command

This command adds a Sentaurus Lithography Tcl command to a lithography machine identified by its name. A lithography machine must be created with the `define_litho_machine` command before using the `add_litho_command` command. The `machine` parameter specifies the name of the machine to which the Tcl command is added.

Note:

The `add_litho_command` command cannot be used when you start Sentaurus Topography 3D with a value greater than 1 for the command-line option `--processes` (see [From the Command Line on page 17](#)).

Users are responsible for specifying the correct parameters depending on the specified Sentaurus Lithography Tcl command.

In general, material and simulation parameters should be specified in the SLO file or the Sentaurus Lithography material database. The `add_litho_command` command should be used mainly for varying a limited number of parameters, for example, in a parameterized Sentaurus Workbench project.

Syntax

```
add_litho_command machine=<c> <slitho command> <slitho arguments>
```

Table 16 Parameters of add_litho_command command

Parameter	Type	Description	Default	Unit
machine	Character	Sets the name of the lithography machine. You can use this name in subsequent <code>litho</code> commands for creating a resist region by a lithography simulation.	none	none
<slitho command>	Character	Sets the Sentaurus Lithography Tcl command.	none	none
<slitho arguments>	Character	Sets the arguments to a Sentaurus Lithography Tcl command, without the Sentaurus Lithography connection handle.	none	none

Chapter 6: Input Commands

add_material

add_material

This command defines the material-dependent properties of an etch machine (`define_etch_machine`), and it can be used several times for the same etch machine to configure the properties of different materials. A material cannot be configured with the `add_material` command more than once.

The machine to be configured is referenced with the `machine` parameter that must match one of the names of the previously defined etch machines.

Note:

The `add_material` command is not supported by etching machines having `model=crystal` because it is not needed. The properties of the material etched by an etching machine having `model=crystal` are set using the `define_etch_machine` command. The crystallographic orientation of the regions containing the material etched by an etching machine having `model=crystal` can be set with the command `set_orientation`. See [Orientation-Dependent Models on page 56](#), [Crystallographic Orientation-Dependent Etching on page 76](#), [define_etch_machine on page 209](#), and [set_orientation on page 486](#).

The `add_material` command is not supported by etching machines using reaction models because it is not needed. In fact, reaction models use reactions to specify material-dependent behavior. The properties of reactions are set using the `add_reaction_properties` command (see [add_reaction_properties on page 168](#)).

Syntax

Simple etching:

```
add_material anisotropy=<n> curvature=<n> material=<c> rate=<n> \
[machine=<c>]
```

Dry etching:

```
add_material material=<c> rate=<n> s1=<n> s2=<n> [machine=<c>]
```

Wet etching:

```
add_material material=<c> rate=<n> [deactivation_rate=<n>] \
[density=<n>] [machine=<c>]
```

Simultaneous etching and deposition (etchdepo):

```
add_material material=<c> rate=<n> s1=<n> s2=<n> [machine=<c>] \
[sputter_type=<c>] [reflection=<n>]
```

Chapter 6: Input Commands

add_material

Simultaneous etching and deposition 2 (etchdepo2):

```
add_material anisotropy=<n> material=<c> rate=<n> s1=<n> s2=<n> \
sticking=<n> [desorption_rate=<n>] [machine=<c>] [reflection=<n>]
```

High-density plasma etching:

```
add_material anisotropy=<n> material=<c> rate=<n> s1=<n> s2=<n> \
[machine=<c>]
```

High-density plasma 2 etching:

```
add_material anisotropy=<n> material=<c> rate=<n> s1=<n> s2=<n> \
sputter_rate=<n> sticking=<n> [machine=<c>] [reflection=<n>]
```

Ion-enhanced etching:

```
add_material anisotropy=<n> material=<c> rate=<n> s1=<n> s2=<n> \
sticking=<n> [desorption_rate=<n>] [machine=<c>] [reflection=<n>]
```

Ion-milling:

```
add_material anisotropy=<n> material=<c> rate=<n> s1=<n> s2=<n> \
[machine=<c>]
```

Reactive ion etching:

```
add_material anisotropy=<n> material=<c> rate=<n> [machine=<c>]
```

Reactive ion etching 2:

```
add_material anisotropy=<n> material=<c> rate=<n> sticking=<n> \
[machine=<c>] [reflection=<n>]
```

RFM model:

```
add_material material=<c> [machine=<c>] ...
```

Table 17 Parameters of add_material command

Parameter	Type	Description	Default [Range]	Unit
anisotropy	Number	Sets the anisotropy coefficient for the current material.	none [0, 1]	none
curvature	Number	Sets the curvature coefficient for the current material.	none [0, 0.1]	µm
deactivation_rate	Number	Sets the rate at which etchants are deactivated when reaching the surface.	0 [0, ∞[µm min ⁻¹

Chapter 6: Input Commands

add_material

Table 17 Parameters of add_material command (Continued)

Parameter	Type	Description	Default [Range]	Unit
density	Number	Sets the volume density of the current material.	1 [0, ∞[mol cm ⁻³
desorption_rate	Number	Sets the thermal desorption rate for the material.	0 [0, ∞[μm min ⁻¹
machine	Character	Sets the machine to which the command is applied.	default_machine	none
material	Character	Sets the material.	none	none
rate	Number	Sets the etching rate for the current material.	none [0, ∞[μm min ⁻¹
reflection	Number	Sets the reflection parameter used to evaluate the reflection probability [1].	0 [0, 1]	none
s1	Number	Sets the first sputter coefficient.	none	none
s2	Number	Sets the second sputter coefficient.	none	none
sputter_rate	Number	Sets the sputter rate.	none [0, ∞[μm min ⁻¹
sputter_type	Character	Sets the angular distribution type of the sputtered material. The value reflective is only supported for the radiosity method for flux integration. Options are: <ul style="list-style-type: none">• diffuse• reflective	diffuse	none
sticking	Number	Sets the sticking coefficient.	none [0, 1]	none

Chapter 6: Input Commands

add_neutral_flux

Examples

This command configures the material-dependent parameters of the previously defined machine etchmachine for the material Silicon, with rate of 0.5 µm/minute, an anisotropy coefficient of 0.3, and a curvature coefficient of 0.05 µm:

```
add_material machine=etchmachine material=Silicon anisotropy=0.3 \
curvature=0.05 rate=0.5
```

Note:

For materials that are contained in the initial TDR structure, whose properties have not been defined with the add_material command, the etch rate is zero.

add_neutral_flux

This command adds a new neutral flux to the specified RFM model.

Syntax

```
add_neutral_flux model=<c> name=<c> [sticking=<n>]
```

Table 18 Parameters of add_neutral_flux command

Parameter	Type	Description	Default [Range]	Unit
model	Character	Sets the RFM model to which the neutral flux is added. Note: The model parameter must specify an RFM model, that is, the type parameter must have been set in the define_model command for that model.	none	none
name	Character	Sets the name of the neutral flux. This name is used to reference it when configuring the flux properties using the add_flux_properties command and when accessing the flux values in the formula for calculating the etching and deposition rates.	none	none
sticking	Number	Sets the sticking coefficient. This parameter can define a constant for the sticking coefficient. If you set this parameter, the sticking coefficient for this flux cannot be changed with a subsequent add_flux_properties command.	none [0, 1]	none

Chapter 6: Input Commands

add_reaction

add_reaction

This command defines a reaction of a reaction model.

Syntax

```
add_reaction expression=<c> model=<c> name=<c>
```

Table 19 Parameters of add_reaction command

Parameter	Type	Description	Default	Unit
expression	Character	Sets the expression of the reaction added to the model.	none	none
model	Character	Sets the name of the model to which the reaction is added. Note: The <code>model</code> parameter must specify a reaction model, that is, the <code>type</code> parameter must have been set in the <code>define_model</code> command for that model.	none	none
name	Character	Sets the name of the reaction.	none	none

A reaction specifies a rule to transform a set of reactants into a set of products. The transformation specified by the `add_reaction` command is irreversible.

A reaction expression consists of two reaction states:

- The first reaction state describes the state *before* the reaction (specifying the *reactants*).
- The second reaction state describes the state *after* the reaction (specifying the *products*).

Reaction states are separated by the equal sign (=).

Each reaction state consists of one or more chemical species terms, which are separated by the plus sign (+). A chemical species term consists of a chemical symbol and a species-type modifier (see [Table 20](#)).

Chapter 6: Input Commands

add_reaction

Table 20 Valid species-type modifiers

Modifier	Description
	Bulk species
<g>	Gaseous species
<p>	Sputtered product species that is tracked by the simulator
<q>	Sputtered product species that is not tracked by the simulator
<r>	Reflected product
<s>	Surface species
<t>	Top plane species (plasma source species)
<v>	Product that is not tracked by the simulator

Limitations

The following limitations apply:

- Each reaction must have exactly two reactants: one with the <g> species-type modifier and one with the <s> or <t> species-type modifier.
- Except for top plane reactions, each reaction is allowed to have, at most, one product that goes into the reactor and that is tracked. The species-type modifiers that denote products that go into the reactor and that are tracked are <g>, <p>, and <r>.

Example of Adsorption Reaction

The following command adds a reaction named r1 to the model reaction_model:

```
add_reaction model=reaction_model \
    expression="F<g> + Silicon<s> = SiF<s>" name=r1
```

According to the specified expression, when an atom of species F coming from the reactor reacts with an atom of Silicon on the surface of the processed structure, a SiF molecule is produced. The effect of such a reaction is that a SiF molecule is adsorbed.

Chapter 6: Input Commands

add_reaction

Example of Etching Reactions

The following commands add two reactions `r2` and `r3` to the model `reaction_model`:

```
add_reaction model=reaction_model \
    expression="Ar<g> + Silicon<s> = Silicon<v>" name=r2
add_reaction model=reaction_model \
    expression="Cl<g> + Silicon<s> = SiCl<g>" name=r3
```

According to reaction `r2`, when an atom of species `Ar` coming from the reactor reacts with an atom of `Silicon` on the surface of the processed structure, that silicon atom is removed without being tracked by the simulator.

According to reaction `r3`, when an atom of species `Cl` coming from the reactor reacts with an atom of `Silicon` on the surface of the processed structure, a `SiCl` molecule is produced and it goes into the reactor and is tracked by the simulator. The net effect of this reaction is that a silicon atom is etched from the structure.

Example of Deposition Reactions

The following command adds a reaction named `r4` to the model `reaction_model`:

```
add_reaction model=reaction_model \
    expression="SiH2<g> + Silicon<s> = Silicon<s> + Silicon<b> + H2<v>" \
    name=r4
```

According to the specified expression, when a molecule of species `SiH2` coming from the reactor reacts with an atom of `Silicon` on the surface of the processed structure, a `Silicon` atom goes into the bulk and a new `Silicon` atom is deposited on the surface of the structure. Moreover, a `H2` molecule is produced, but it is not tracked by the simulator. Therefore, from the perspective of the simulation, the following command is equivalent to the previous one:

```
add_reaction model=reaction_model \
    expression="SiH2<g> + Silicon<s> = Silicon<s> + Silicon<b>" name=r4
```

However, including the chemical species term `H2<v>` makes the reaction clearer and more readable.

Example of Sputtering Reactions

The following commands add two reactions `r5` and `r6` to the model `reaction_model`:

```
add_reaction model=reaction_model \
    expression="I<g> + Silicon<s> = Silicon<p>" name=r5
add_reaction model=reaction_model \
    expression="Ar<g> + Photoresist<s>= Photoresist<q>" name=r6
```

According to reaction `r5`, when an atom of species `I` coming from the reactor reacts with an atom of `Silicon` on the surface of the processed structure, a `Silicon` atom is sputtered

Chapter 6: Input Commands

add_reaction

and tracked by the simulator. Since the sputtered atom is tracked, it can be deposited somewhere if the model contains a proper reaction for silicon deposition.

According to reaction `r6`, when an atom of species `Ar` coming from the reactor reacts with an atom of `Photoresist` on the surface of the processed structure, a `Photoresist` atom is sputtered off the structure. However, the removed atom is not tracked by the simulator. As a consequence, this reaction models the sputtering of `Photoresist` by `Ar`, but it cannot be used if the deposition of the sputtered `Photoresist` must be modeled.

Example of Reflection Reaction

The following command adds a reaction named `r7` to the model `reaction_model`:

```
add_reaction model=reaction_model \
    expression="I<g> + Nitride<s> = I<r> + Nitride<s>" name=r7
```

According to reaction `r7`, when an atom of species `I` coming from the reactor reacts with an atom of `Nitride` on the surface of the processed structure, the `Nitride` atom is left untouched; whereas, the `I` atom is reflected.

Example of Top Plane Reactions

The following command adds a plasma depletion reaction between gaseous surface reaction byproducts and a top plane species in the plasma source:

```
add_reaction model=reaction_model \
    expression="SiCl<g> + I<t> = SiCl<v> + I<v>" name=r8
```

This reaction corresponds to an annihilation of the plasma radical `I<t>`, resulting in a decrease of the `I<t>` concentration in the plasma bulk.

The following command adds a plasma enhancement reaction between gaseous surface reaction byproducts and a generic top plane species:

```
add_reaction model=reaction_model \
    expression="SiCl<g> + <t> = 2Cl<g>" name=r9
```

This reaction represents a generic reaction between a surface reaction byproduct (`SiCl<g>`) and a generic plasma bulk particle (represented by the placeholder `<t>`) at the top plane, triggering the emission of two new plasma particles (`2Cl<g>`) and, therefore, leading to an enhancement of the plasma particle flux.

Chapter 6: Input Commands

add_reaction_properties

add_reaction_properties

This command specifies the parameters of a reaction.

Syntax

For reflection reactions (see [add_reaction on page 164](#)):

```
add_reaction_properties (p=<n> | probability=<c>) reaction=<c> \
[<crystal_parameters>] [<energy_parameters>] \
[machine=<c>] [max_product_coverage=<n>] [reflection_exponent=<n>]
```

For sputtering reactions (see [add_reaction on page 164](#)):

```
add_reaction_properties (p=<n> | probability=<c>) reaction=<c> \
[<crystal_parameters>] [<energy_parameters>] \
[machine=<c>] \
[sputter_exponent=<n>] \
[(sputter_gamma=<n> [sputter_isotropic_ratio=<n>] \
[sputter_preferential_angle=<n>]) | sputter_type=<c>]
```

For adsorption or deposition reactions with surface diffusion (see [add_reaction on page 164](#) and [Diffusion on page 90](#)):

```
add_reaction_properties method=diffusion reaction=<c> \
(p=<n> | probability=<c>) reaction=<c> \
[alpha=<n>] \
[biased_random_walk=<c>] \
[<crystal_parameters>] [<energy_parameters>] \
[machine=<c>] [max_product_coverage=<n>] \
[num_diffusion_steps=<n>] \
[redistribution_factor=<n>]
```

For top plane reactions with gaseous reaction product (see [Example of Top Plane Reactions on page 167](#)):

```
add_reaction_properties reaction=<c> (p=<n> | probability=<c>) \
[machine=<c>] [product_distribution=<c>]
```

For other reactions:

```
add_reaction_properties (p=<n> | probability=<c>) reaction=<c> \
[<crystal_parameters>] [<energy_parameters>] \
[machine=<c>] [max_product_coverage=<n>]
```

Chapter 6: Input Commands

add_reaction_properties

Table 21 Parameters of add_reaction_properties command

Parameter	Type	Description	Default [Range]	Unit
activation_energy	Number	Sets the value of the activation energy parameter E_a in Equation 75 .	0 [0, ∞[eV
alpha	Number	Controls the strength of the bias during a biased random walk. You can specify this parameter only for method=diffusion.	1 [0, ∞)	none
biased_random_walk	Character	Specifies whether to execute the full random walk for the given number of diffusion steps (option last_step) or to terminate the random walk automatically when a suitable deposition site is found (option always). See Diffusion on page 90 for details. You can specify this parameter only for method=diffusion.	always	none
<crystal_parameters>	Character	See Crystal Parameters on page 175 .	none	none
crystal_type	Character	Sets the crystal type of the deposited crystalline material. This parameter can only be set for deposition reactions with product_morphology=crystalline. The only supported option is diamond.	none	none
diamond_activation_energy_<hkl>	Number	Sets the value of the activation energy parameter E_a in Equation 75 , when the solid reactant has a diamond crystal structure and its exposed surface is parallel to an (h k l) crystal plane, where <hkl> is either (1 0 0), (1 1 0), or (1 1 1).	0 [0, ∞[eV

Chapter 6: Input Commands

add_reaction_properties

Table 21 Parameters of add_reaction_properties command (Continued)

Parameter	Type	Description	Default [Range]	Unit
diamond_energy_exponent_<hkl>	Number	Sets the value of the exponent parameter m in Equation 76 , when the solid reactant has a diamond crystal structure and its exposed surface is parallel to an $(h k l)$ crystal plane, where $<hkl>$ is either $(1\ 0\ 0)$, $(1\ 1\ 0)$, or $(1\ 1\ 1)$.	none] $0, \infty[$	none
diamond_energy_reference_<hkl>	Number	Sets the value of the parameter E_{ref} in Equation 76 , when the solid reactant has a diamond crystal structure and its exposed surface is parallel to an $(h k l)$ crystal plane, where $<hkl>$ is either $(1\ 0\ 0)$, $(1\ 1\ 0)$, or $(1\ 1\ 1)$.	none] $0, \infty[$	eV
diamond_energy_threshold_<hkl>	Number	Sets the value of the parameter E_{th} in Equation 76 , when the solid reactant has a diamond crystal structure and its exposed surface is parallel to an $(h k l)$ crystal plane, where $<hkl>$ is either $(1\ 0\ 0)$, $(1\ 1\ 0)$, or $(1\ 1\ 1)$.	none] $0, \infty[$	eV
diamond_p_<hkl>	Number	Sets the energy- and angle-independent value of the reaction probability to use when the solid reactant has a diamond crystal structure and its exposed surface is parallel to an $(h k l)$ crystal plane, where $<hkl>$ is either $(1\ 0\ 0)$, $(1\ 1\ 0)$, or $(1\ 1\ 1)$.	none] $0, 1]$	none
diamond_probability_<hkl>	Character	Sets the name of the probability function defined with the <code>define_probability</code> command to use when the solid reactant has a diamond crystal structure and its exposed surface is parallel to an $(h k l)$ crystal plane, where $<hkl>$ is either $(1\ 0\ 0)$, $(1\ 1\ 0)$, or $(1\ 1\ 1)$.	none	none

Chapter 6: Input Commands

add_reaction_properties

Table 21 Parameters of add_reaction_properties command (Continued)

Parameter	Type	Description	Default [Range]	Unit
<energy_parameters>	Character	See Activation Energy on page 176 .	none	none
energy_exponent	Number	Sets the value of the exponent parameter m in Equation 76 .	none]0, ∞[none
energy_reference	Number	Sets the value of the parameter E_{ref} in Equation 76 .	none]0, ∞[eV
energy_threshold	Number	Sets the value of the parameter E_{th} in Equation 76 .	none]0, ∞[eV
machine	Character	Sets the name of the machine for which the reaction is configured.	default_machine	none
max_product_coverage	Number	Sets the maximum surface coverage of the reaction product with the <s> species-type modifier that allows the reaction to be executed. This parameter can be specified only if there is exactly one reaction product with the <s> species-type modifier.	1 [0, 1]	none
method	Character	If this parameter is set to diffusion, then it allows surface diffusion (see Diffusion on page 90). You can specify this parameter only for adsorption or deposition reactions.	none	none
num_diffusion_steps	Number	Sets the number of steps of the random walk along the surface during deposition reactions. You can specify this parameter only for method=diffusion.	1 [0, ∞)	none
p	Number	Sets the energy- and angle-independent value of the reaction probability to use when the solid reactant is amorphous.	none [0, 1]	none

Chapter 6: Input Commands

add_reaction_properties

Table 21 Parameters of add_reaction_properties command (Continued)

Parameter	Type	Description	Default [Range]	Unit
p_island_growth	Number	Sets the probability that the gaseous reactant sticks to a preexisting island and takes over the crystal orientation of that island.	none [0, 1]	none
p_nucleation	Number	Sets the probability that the gaseous reactant sticks to the surface and forms a new island with a random crystal orientation.	none [0, 1]	none
probability	Character	Sets the name of the probability function defined with the define_probability command to use when the solid reactant is amorphous.	none	none
product_distribution	Character	Sets the name of the species distribution of the gaseous reaction product.	The species distribution defined within the specified machine or, if the distribution of the species is undefined, an isotropic distribution (exponent=1)	none
product_energy_factor	Number	Sets the ratio between the energy of any reaction products with the <g>, <p>, or <r> species-type modifier and the energy of the gaseous reactant of the reaction being specified.	1 [0, 1]	none
product_energy_max	Number	Sets the maximum energy of the products with the <g>, <p>, or <r> species-type modifier of the reaction being specified, when they are assumed to have a uniform energy distribution.	none]0, ∞[eV

Chapter 6: Input Commands

`add_reaction_properties`

Table 21 Parameters of `add_reaction_properties` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
<code>product_energy_min</code>	Number	Sets the minimum energy of the products with the <code><g></code> , <code><p></code> , or <code><r></code> species-type modifier of the reaction being specified, when they are assumed to have a uniform energy distribution.	none]0, ∞[eV
<code>product_morphology</code>	Character	<p>Sets the material type of the adsorbed or deposited species. Options are:</p> <ul style="list-style-type: none"> • <code>amorphous</code> • <code>crystalline</code> • <code>same_as_surface_reactant</code> <p>Note: You can use the <code>crystalline</code> option only when modeling nucleation, that is, in combination with the parameters <code>p_nucleation</code> and <code>p_island_growth</code> (see Nucleation and Grain Growth on page 88).</p>	amorphous	none
<code>reaction</code>	Character	Sets the name of the reaction to configure.	none	none
<code>redistribution_factor</code>	Number	<p>Sets the percentage of reactions for which the random walk is executed.</p> <p>You can specify this parameter only for <code>method=diffusion</code>.</p>	1 [0, 1]	none
<code>reflection_exponent</code>	Number	Sets the exponent of the angular distribution $\cos^m\theta$ of the reflected species. If omitted, all particles are reflected along the direction specular to the incoming one with respect to the surface normal.	none [1, 2147483647]	none

Chapter 6: Input Commands

`add_reaction_properties`

Table 21 Parameters of `add_reaction_properties` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
<code>sputter_exponent</code>	Number	Sets the exponent of the angular distribution $\cos^m\theta$ of the sputtered species.	1 Energy-dependent expression, evaluating to $[0, \infty[$	none
<code>sputter_gamma</code>	Number	Sets the value of γ in Equation 77 that controls the main direction of the emission of the sputtered material. You can use this parameter only if you do not use <code>sputter_type</code> . Note: Setting <code>sputter_gamma=0</code> and <code>sputter_exponent=1</code> is equivalent to <code>sputter_type=diffuse</code> .	none $]-\infty, \infty[$	none
<code>sputter_isotropic_ratio</code>	Number	Sets the ratio of particles that are sputtered isotropically for small incidence angles. It can be used only if you specify <code>sputter_preferential_angle</code> .	0 Energy-dependent expression, evaluating to $[0, 1]$	none
<code>sputter_preferential_angle</code>	Number	Sets the preferential ejection angle θ_{pref} for sputtering under normal incidence angles (see Figure 22 on page 179). It can be used only if you specify <code>sputter_gamma</code> .	0 Energy-dependent expression, evaluating to $[0, 90]$	degree
<code>sputter_type</code>	Character	Sets the sputter reemission type of the model. Options are: <ul style="list-style-type: none">• <code>diffuse</code>• <code>reflective</code> You can use this parameter only if you do not use <code>sputter_gamma</code> . If you do not use either <code>sputter_gamma</code> or <code>sputter_type</code> , then <code>sputter_type=diffuse</code> is used.	none	none

Chapter 6: Input Commands

add_reaction_properties

Crystal Parameters

Note:

A single `add_reaction_properties` command specifies the reaction probabilities for both amorphous and crystalline or polycrystalline surface materials. For each surface material, you must always specify all probabilities since the structure can contain the same material both in amorphous and in crystalline or polycrystalline form. For example:

```
add_reaction_properties reaction=R p=1 \
    diamond_p_100=1 diamond_p_110=0.7 diamond_p_111=0.1
```

You must specify a collection of parameters for reactions involving crystalline materials:

- For etching or sputtering reactions on a crystalline surface (see [Etching and Sputtering Reactions on page 85](#)):

```
<crystal_parameters> =
  (<crystal_type>_p_<hkl>=<n> | 
   <crystal_type>_probability_<hkl>=<c>)
  [<crystal_type>_activation_energy_<hkl>=<n> | 
   (<crystal_type>_energy_exponent_<hkl>=<n>
    <crystal_type>_energy_reference_<hkl>=<n>
    <crystal_type>_energy_threshold_<hkl>=<n>) ]
```

where:

- `<crystal_type>` stands for the crystal type of the surface (the only option is `<crystal_type>=diamond`)
- `<hkl>` stands for the Miller index ($h \ k \ l$) of the crystal plane on which this parameter is defined (for diamond, `<hkl>=100, 110, or 111`)

For example:

```
add_reaction_properties reaction=R p=1 \
    diamond_p_100=1 diamond_p_110=0.7 diamond_p_111=0.1
```

- For adsorption or deposition of material on a crystalline surface (see [Adsorption and Deposition Reactions on page 86](#)):

```
<crystal_parameters> =
  (<crystal_type>_p_<hkl>=<n> | 
   <crystal_type>_probability_<hkl>=<c>)
  [<crystal_type>_activation_energy_<hkl>=<n> | 
   (<crystal_type>_energy_exponent_<hkl>=<n>
    <crystal_type>_energy_reference_<hkl>=<n>
    <crystal_type>_energy_threshold_<hkl>=<n>) ]
  product_morphology=amorphous | same_as_surface_reactant
```

Chapter 6: Input Commands

add_reaction_properties

For example:

```
add_reaction_properties reaction=R p=1 \
    diamond_p_100=1 diamond_p_110=0.7 diamond_p_111=0.1 \
    product_morphology=amorphous
```

- For adsorption or deposition of crystalline material on an amorphous surface (see [Adsorption and Deposition Reactions on page 86](#)):

```
<crystal_parameters> =
    product_morphology=crystalline
    crystal_type=<c>
    p_island_growth=<n>
    p_nucleation=<n>
```

For example:

```
add_reaction_properties reaction=R p=1 \
    product_morphology=crystalline crystal_type=diamond \
    p_nucleation=1e-5 p_island_growth=1
```

Activation Energy

Note:

The following parameters are available only when the machine specified with the machine parameter is energy dependent:

```
<energy_parameters> =
    (activation_energy=<n> |
     energy_exponent=<n> energy_reference=<n> energy_threshold=<n>)
    (product_energy_factor=<n> |
     product_energy_max=<n> product_energy_min=<n>)
```

If you set the activation_energy parameter, the following energy-dependent reaction probability $p(E, \theta)$ is assumed for the reaction at hand:

$$p(E, \theta) = p(\theta) \exp\left(-\frac{E_a}{E}\right) \quad (75)$$

where:

- $p(\theta)$ denotes the probability specified with the p parameter or the angle-dependent probability function specified with the probability parameter.
- E_a denotes the value given to the activation_energy parameter. It is worth noting that, according to [Equation 75](#), the probability is not zero for energies smaller than the value of the activation_energy parameter, but it decays rapidly as the energy falls below such a value.

If the probability parameter specifies an energy-dependent probability, an error will be issued in this case.

Chapter 6: Input Commands

add_reaction_properties

When you specify the `energy_exponent`, `energy_reference`, and `energy_threshold` parameters, the following energy-dependent reaction probability $p(E, \theta)$ is assumed for the reaction at hand [2]:

$$p(E, \theta) = p(\theta) \cdot \begin{cases} 0 & E \leq E_{\text{th}} \\ \frac{E^m - E_{\text{th}}^m}{E_{\text{ref}}^m - E_{\text{th}}^m} & E_{\text{th}} < E < E_{\text{ref}} \\ 1 & E \geq E_{\text{ref}} \end{cases} \quad (76)$$

In Equation 76:

- $p(\theta)$ denotes the probability specified with the `p` parameter or the angle-dependent probability function specified with the `probability` parameter. It is the probability when the energy is greater than or equal to the value specified by the `energy_reference` parameter.
- m denotes the value given to the `energy_exponent` parameter.
- E_{ref} denotes the value (in electron volts) given to the `energy_reference` parameter.
- E_{th} denotes the value (in electron volts) given to the `energy_threshold` parameter.

If the `probability` parameter specifies an energy-dependent probability, then an error will be issued in this case.

The `product_energy_factor`, `product_energy_max`, and `product_energy_min` parameters can be specified only for reactions that have at least one product with the `<p>`, `<r>` species-type modifier.

When parameter `product_energy_factor` is specified, the energy of the product having the `<g>`, `<p>`, or `<r>` species-type modifier will be proportional to the energy of the reactant with the `<g>` species-type modifier, and the proportionality constant will be the value of the `product_energy_factor` parameter.

When parameters `product_energy_max` and `product_energy_min` are specified, the energy of the product having the `<g>`, `<p>`, or `<r>` species-type modifier will be uniformly distributed over the energy window defined by the values of these two parameters.

The main direction in which sputtered particles are emitted can be controlled with the parameter `sputter_gamma`. It is used to specify the value of the parameter γ in the function $f(\theta, \gamma)$. This function is used to determine the angle between the surface normal and the symmetry axis of the sputter emission distribution.

The symmetry axis of the sputter emission distribution lies in the plane defined by the direction of the incoming particles and the surface normal. The angle between the symmetry axis of the sputter emission distribution and the surface normal is considered to be negative

Chapter 6: Input Commands

add_reaction_properties

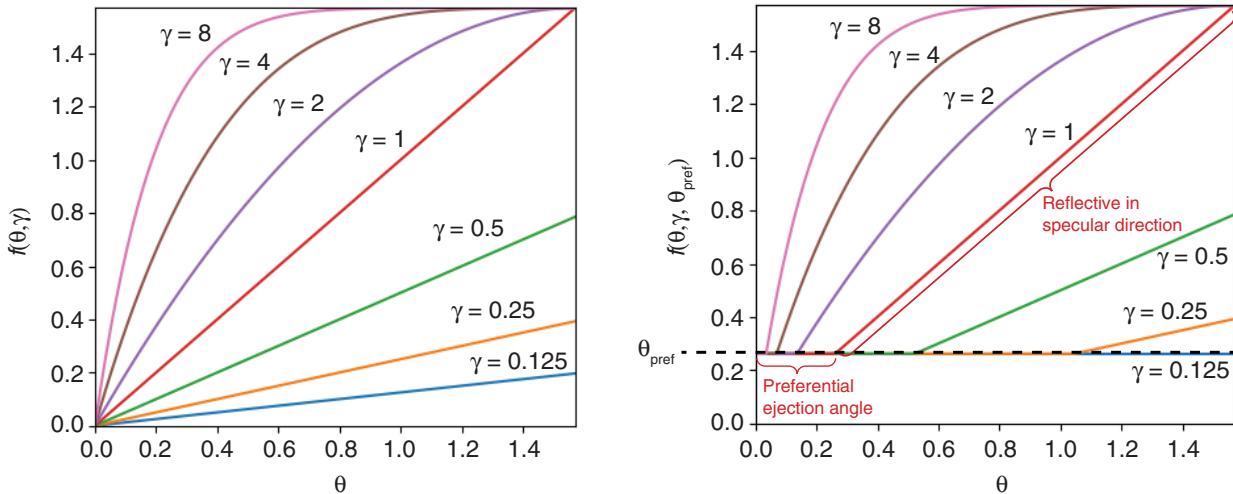
if the symmetry axis of the sputter emission distribution is on the same side of the surface normal as the direction of the incoming particles. Otherwise, it is considered to be positive.

$$f(\theta, \gamma) = \begin{cases} \gamma\theta & \text{for } 0 \leq |\gamma| \leq 1 \\ \text{sgn}(\gamma)\frac{\pi}{2}\left(1 - \left(1 - \frac{\theta}{\frac{\pi}{2}}\right)^{|\gamma|}\right) & \text{for } 1 \leq |\gamma| \end{cases} \quad (77)$$

$$\text{where } \text{sgn}(x) = \begin{cases} +1 & \text{for } x > 0 \\ 0 & \text{for } x = 0 \\ -1 & \text{for } x < 0 \end{cases}$$

Figure 21 (left) shows $f(\theta, \gamma)$ for different values of γ . The horizontal axis represents the angle θ between the surface normal and the incoming particle. The vertical axis represents the angle between the surface normal and the symmetry axis of the sputter emission distribution.

Figure 21 (Left) Function $f(\theta, \gamma)$ and (right) function $f(\theta, \gamma, \theta_{\text{pref}})$ for values of $\gamma = 0.125, 0.25, 0.5, 1, 2, 4$, and 8



For $\gamma = 0$, the symmetry axis of the sputter emission distribution is perpendicular to the surface. Therefore, `sputter_gamma=0` gives the same result as `sputter_type=diffuse`.

For $\gamma = 1$, the symmetry axis of the sputter emission distribution is the reflected direction of the incoming particle. Therefore, `sputter_gamma=1` gives the same result as `sputter_type=reflective`.

For $\gamma < 0$, the sputtered particles are not emitted forward as for positive values of γ , but back in the direction of the incoming particle.

Chapter 6: Input Commands

add_reaction_properties

You can also specify a preferential ejection angle for the emitted particle. This can be useful when modeling low-energetic sputtering processes, which are determined by under-cosine or heart-like sputter angle distributions. You define a preferential ejection angle for normally incident particles by using the `sputter_preferential_angle` parameter, which modifies the function $f(\theta, \gamma)$, as shown in [Figure 21 \(right\)](#).

For almost normally incident particles with small incidence angles, the sputtered particle is emitted with angle `sputter_preferential_angle` with a spread determined by `sputter_exponent`. For incident particles with oblique incidence angles, the outgoing particle is sputtered in the specular direction with a spread determined by `sputter_exponent`.

In addition, for normally incident particles, you can specify a ratio $p_0 \in [0, 1]$ of particles that will be sputtered isotropically (that is, with an angular spread $\cos\theta$) around the surface normal.

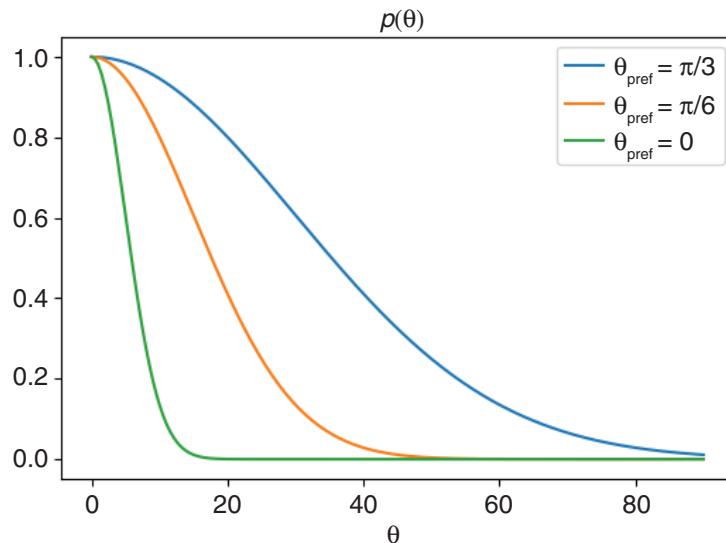
You can set this ratio by using the `sputter_isotropic_ratio` parameter. The ratio of isotropically sputtered particles depends on the incidence angle θ by the expression:

$$p(\theta) = p_0 \exp\left(-\frac{\theta^2}{2\sigma^2}\right) \quad (78)$$

where $\sigma = \frac{1}{2}\max(10^\circ, \theta_{\text{pref}})$ and θ_{pref} is the preferential ejection angle specified by the parameter `sputter_preferential_angle`.

For normal incidence angles ($\theta = 0$), the ratio equals `sputter_isotropic_ratio`. For oblique incidence angles ($\theta \rightarrow \pi/2$), the ratio decreases to zero.

Figure 22 Ratio of isotropically sputtered particles as a function of the incidence angle θ



Chapter 6: Input Commands

add_reaction_properties

Examples

```
# Reaction model definition
define_model name=m description=""

add_source_species model=m name=I
add_source_species model=m name=N

# Adsorption of species N on species Silicon
add_reaction model=m name=adsorption_reaction \
    expression="N<g> + Silicon<s> = SiliconN<s>"

# Reflection of species I by species Silicon
add_reaction model=m name=reflection_reaction \
    expression="I<g> + Photoresist<s> = I<r> + Photoresist<s>"

# Etching of species SiliconN by species I
add_reaction model=m name=ion_etch_reaction \
    expression="I<g> + SiliconN<s> = SiliconN<v>"

# Sputtering of species Silicon by species I
add_reaction model=m name=sputtering_reaction \
    expression="I<g> + Silicon<s> = Silicon<p>"
finalize_model model=m

# ...
# Define the yield function for sputtering of Silicon by species I
define_yield name=my_yield energy=0 species=I material=Silicon \
    theta_max=60 yield_max=1.4

# Define the probability function for reflection of species I
# from Photoresist
define_probability name=my_probability energy=0 mizuno_k=0.05

# Define a machine using the reaction model 'm'. Since the used
# model contains sputtering reactions, yield functions must be
# provided when defining the machine.
define_etch_machine model=m yield=my_yield ...

# Reactions 'adsorption_reaction' and 'ion_etch_reaction' are neither
# reflection nor sputtering reactions. Therefore, the parameter 'p'
# is used to set their angle-independent probabilities.
add_reaction_properties reaction=adsorption_reaction p=0.9
add_reaction_properties reaction=ion_etch_reaction p=0.7

# Reaction 'reflection_reaction' is a reflection reaction. Therefore,
# it is possible to set parameter 'reflection_exponent' for it.
# Moreover, the probability function defined with the
# 'define_probability' command will be used as the reaction probability.
add_reaction_properties reaction=reflection_reaction \
    reflection_exponent=1000 probability=my_probability
```

Chapter 6: Input Commands

add_source_species

```
# Reaction 'sputtering_reaction' is a sputtering reaction. Therefore,  
# it is possible to set parameter 'sputter_exponent' for it. The  
# default value of parameter 'sputter_type' is used here.  
add_reaction_properties reaction=sputtering_reaction p=0.8 \  
sputter_exponent=100
```

add_source_species

This command adds a new source species to the specified reaction model.

Syntax

```
add_source_species model=<c> name=<c>
```

Table 22 Parameters of add_source_species command

Parameter	Type	Description	Default	Unit
model	Character	<p>Sets the name of the model to which the source species is added.</p> <p>Note: This parameter must specify a reaction model, that is, the type parameter must <i>not</i> have been set in the define_model command for that model.</p>	none	none
name	Character	Sets the name of the source species to add.	none	none

Chapter 6: Input Commands

add_species

add_species

This command adds a new plasma species to a specified plasma model.

Note:

If you specify an `add_species` command with the name of a plasma species that already exists, then this command is ignored.

Syntax

```
add_species charge=<n> mass=<n> name=<c> plasma_model=<c>
```

Table 23 Parameters of add_species command

Parameter	Type	Description	Default [Range]	Unit
charge	Number	Sets the charge of the species in units of the unit charge e. Only single-charged species are allowed.	none {-1, 0, +1}	e
mass	Number	Sets the mass of the species.	none [0, +∞[amu
name	Character	Sets the name of the plasma species. This name is used to refer to the species in other commands (for example, in plasma bulk reaction expressions, or when defining species-dependent plasma parameters in the <code>define_reactor</code> command). See define_reactor on page 261 . Note: Plasma species names must not contain spaces and must not begin with a number. The name <code>e-</code> is predefined for electrons and cannot be used.	none	none
plasma_model	Character	Sets the name of the plasma model to which the species is added. This model must be already defined by using define_plasma_model on page 258 .	none	none

Examples

```
add_species name=Ar+ plasma_model=M mass=40<amu> charge=+1
```

Chapter 6: Input Commands

add_volumetric_source_species

add_volumetric_source_species

This command adds a new volumetric source species to a reaction model.

Syntax

```
add_volumetric_source_species model=<c> name=<c>
```

Table 24 Parameters of add_volumetric_source_species command

Parameter	Type	Description	Default	Unit
model	Character	<p>Sets the name of the model to which the volumetric source species is added.</p> <p>Note: This parameter must specify a reaction model, that is, the type parameter must not have been set in the define_model command for that model (see define_model on page 251).</p>	none	none
name	Character	<p>Sets the name of the volumetric source species to add.</p> <p>No add_volumetric_source_species commands must have been specified for the same model with the same name value.</p>	none	none

Chapter 6: Input Commands

define_boundary_conditions

define_boundary_conditions

This command defines the boundary conditions to apply when using a structure in a simulation.

Syntax

```
define_boundary_conditions [structure=<c>] [x=<c>] [y=<c>] \
    [xmin=<c>] [xmax=<c>] [ymin=<c>] [ymax=<c>]
```

Table 25 Parameters of *define_boundary_conditions* command

Parameter	Type	Description	Default	Unit
structure	Character	Sets the name of the structure for which boundary conditions must be defined.	default_structure	none
x	Character	Sets the boundary conditions for both planes $x = x_{\text{min}}$ and $x = x_{\text{max}}$, where x_{min} and x_{max} are the minimum and the maximum x -value in the simulation domain, respectively. Options are: <ul style="list-style-type: none">• none• periodic• reflective	none	none
xmax	Character	Sets the boundary conditions for the plane $x = x_{\text{max}}$, where x_{max} is the maximum x -value in the simulation domain. Options are: <ul style="list-style-type: none">• none• reflective	reflective	none
xmin	Character	Sets the boundary conditions for the plane $x = x_{\text{min}}$, where x_{min} is the minimum x -value in the simulation domain. Options are: <ul style="list-style-type: none">• none• reflective	reflective	none

Chapter 6: Input Commands

define_boundary_conditions

Table 25 Parameters of *define_boundary_conditions* command (Continued)

Parameter	Type	Description	Default	Unit
y	Character	Sets the boundary conditions for both planes $y = y_{\min}$ and $y = y_{\max}$, where y_{\min} and y_{\max} are the minimum and the maximum y -value in the simulation domain, respectively. Options are: <ul style="list-style-type: none">• none• periodic• reflective	none	none
y _{max}	Character	Sets the boundary conditions for the plane $y = y_{\max}$, where y_{\max} is the maximum y -value in the simulation domain. Options are: <ul style="list-style-type: none">• none• reflective	reflective	none
y _{min}	Character	Sets the boundary conditions for the plane $y = y_{\min}$, where y_{\min} is the minimum y -value in the simulation domain. Options are: <ul style="list-style-type: none">• none• reflective	reflective	none

Limitations

The following limitations or conditions apply:

- When using a level set-based model, the boundary conditions are set automatically to `none` in the planes in which a user-defined tilt breaks the symmetry. When using the PMC method, periodic boundary conditions are applied automatically to the planes in which a user-defined tilt breaks the symmetry (see [Structure Tilt on page 36](#) and [Boundary Conditions on page 37](#)).
- The structure referred to by the `structure` parameter must be already defined before the `define_boundary_conditions` command is called.
- The `define_boundary_conditions` command affects not only flux computation, but also surface evolution. Therefore, this command is also relevant for non-flux models.
- The `define_boundary_conditions` command has no effect on the built-in deposition model `crystal` and the built-in etch model `crystal`. Boundary conditions are always reflective when using those models.

Chapter 6: Input Commands

define_boundary_conditions

- The boundary conditions specified by the command `define_boundary_conditions` have no effect on the indirect flux computations for 2D structures.
- Only `reflective` boundary conditions are supported when using the `electrodeposition`, `spin_on`, or `wet` built-in deposition model.
- When using RFM models that use the `pad_pressure()` RFM function, boundary conditions specified by the command `define_boundary_conditions` are ignored and periodic boundary conditions are always applied.
- When using the PMC method, boundary conditions of type `none` are not available.
- You can specify periodic boundary conditions only for the PMC method.
- The `x` parameter cannot be simultaneously specified with either the `xmin` parameter or the `xmax` parameter. The `y` parameter cannot be simultaneously specified with either the `ymin` parameter or the `ymax` parameter.

Examples

This command sets the boundary conditions on the two x-bounding planes of the structure called `default_structure` and uses the default ones on the two y-bounding planes. The plane with the minimum `x` will be considered as a reflective boundary; while in the plane of maximum `x`, the boundary condition type `none` will be applied:

```
define_boundary_conditions xmin=reflective xmax=none
```

Chapter 6: Input Commands

define_bulk_solver

define_bulk_solver

This optional command defines the numeric parameters of the ordinary differential equation (ODE) solver used for a plasma bulk model.

Syntax

```
define_bulk_solver bulk_model_type=<c> name=<c> \
    [abs_error=<n>] [convergence_check_num_samples=<n>] \
    [convergence_check_step=<n>] [max_num_steps=<n>] \
    [max_time=<n>] [rel_error=<n>] \
    [stationary_state_tolerance=<n>] [step_size=<n>] \
    [time_average_solution=<b>]
```

Table 26 Parameters of define_bulk_solver command

Parameter	Type	Description	Default [Range]	Unit
abs_error	Number	Sets the absolute error used for adaptive step size control. Applies only to stepper_type=runge_kutta_dopri5.	10^{-8}] $0, \infty[$	none
bulk_model_type	Character	Sets the type of the bulk model to solve. The only option is global.	none	none
convergence_check_num_samples	Number	Sets the minimum number of samples to use when computing the convergence residual.	10^4] $1, \infty[$	none
convergence_check_step	Number	Sets the interval at which convergence to the stationary state is computed.	10] $1, \infty[$	none
max_num_steps	Number	Sets the maximum number of steps until convergence to the stationary state.	10^{10}] $1, \infty[$	none
max_time	Number	Sets the maximum time to which the model is integrated to reach the stationary state.	∞] $0, \infty[$	s
name	Character	Sets the name of the solver. This name is used to reference the solver in other commands, such as solve_reactor (see solve_reactor on page 487).	none	none

Chapter 6: Input Commands

define_bulk_solver

Table 26 Parameters of define_bulk_solver command (Continued)

Parameter	Type	Description	Default [Range]	Unit
rel_error	Number	Sets the relative error used for the adaptive step size control. Applies only to stepper_type=runge_kutta_dopri5.	10^{-8}] $0, \infty[$	none
stationary_state_tolerance	Number	Sets the threshold for convergence to the stationary state.	10^{-4}] $0, \infty[$	none
step_size	Number	Sets the initial step size, which might change due to the adaptive step size control.	10^{-6}] $0, \infty[$	s
time_average_solution	Boolean	When simulating pulsed power waveforms, this parameter specifies whether the bulk solution, consisting of periodic waveforms, should be averaged in time.	False	none

Note:

For the define_bulk_solver command:

- Convergence speed is mainly influenced by the parameters convergence_check_step and stationary_state_tolerance.
- If the iterative bulk solver exceeds the maximum number of steps (max_num_steps) or the maximum simulation time (max_time), the solver returns an error stating that the plasma bulk model did not succeed.
- A default bulk solver, default_global_bulk_solver, is implicitly defined and can be referenced by other commands, such as solve_reactor (see [solve_reactor on page 487](#)). The implicit definition of this solver is equivalent to:

```
define_bulk_solver name=default_global_bulk_solver \
bulk_model_type=global
```

Examples

```
define_bulk_solver name=s1 bulk_model_type=global \
abs_error=1e-8 rel_error=1e-8 \
max_num_steps=1000000 max_time=1000<s> step_size=1e-6<s> \
stationary_state_tolerance=1e-4
```

Chapter 6: Input Commands

define_charging_model

define_charging_model

This command defines a charging model for charge-up simulations with a PMC model. You can define the model parameters including the charge-up factor, the boundary conditions for the electric field, and the electron angle distribution.

Syntax

```
define_charging_model name=<c> \
  (charge_dose=<n> | charging_factor=<n>) \
  [bottom_bc=<c>] \
  [bottom_electric_field=<n> | bottom_potential=<n>] \
  [charge_scaling=<n> | num_samples_per_cell=<n>] \
  [electric_contacts=<l>] \
  [electron_exponents=<l> electron_fluxes=<l>] \
  [enforce_charge_neutrality=<b>] \
  [top_bc=<c>] [top_electric_field=<n> | top_potential=<n>]
```

Table 27 Parameters of *define_charging_model* command

Parameter	Type	Description	Default [Range]	Unit
bottom_bc	Character	Sets the boundary condition of the electric solver at the bottom of the simulation domain. Options are: <ul style="list-style-type: none">• dirichlet• floating• neumann	dirichlet	none
bottom_electric_field	Number	Sets the normal electric field at the lower Neumann boundary. You can specify this parameter only if <code>top_bc=neumann</code> .	0 $(-\infty, \infty)$	V/m
bottom_potential	Number	Sets the potential applied at the Dirichlet boundary at the bottom of the simulation domain. You can specify this parameter only if <code>bottom_bc=dirichlet</code> .	0 $(-\infty, \infty)$	V
charge_dose	Number	Sets the dose of charges sent to the structure, measured in the number of charges through a flat xy cross-section plane at a constant height z. You can specify either <code>charge_dose</code> or <code>charging_factor</code> .	none $[0, \infty)$	m^{-2}

Chapter 6: Input Commands

`define_charging_model`

Table 27 Parameters of `define_charging_model` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
<code>charge_scaling</code>	Number	<p>Sets a scaling factor used to reduce or enhance the charges deposited in the structure.</p> <p>This parameter controls the statistics and smoothness of the charge distribution on the surface during the charging phase. (It does not affect the physical total charge.)</p> <p>You can specify either <code>charge_scaling</code> or <code>num_samples_per_cell</code>.</p>	none [0, ∞)	none
<code>charging_factor</code>	Number	<p>Sets the factor to control the strength of the charging effect. The factor determines the charge dose sent to the structure during the charging simulation.</p> <p>If <code>charging_factor=1</code>, then the reference charge dose is $10^{17} \text{ m}^{-2} = 0.1 \text{ nm}^{-2}$.</p> <p>You can specify either <code>charge_dose</code> or <code>charging_factor</code>.</p>	none [0, ∞)	none
<code>electric_contacts</code>	List	Specifies a list of names referring to electric contacts defined in the <code>define_electric_contact</code> command.	none	none
<code>electron_exponents</code>	List	Specifies the exponent parameters $\{m_i\}$ of the angle distribution for the electrons. See parameter <code>electron_fluxes</code> for details.	none [1, ∞[for each entry in the list	none

Chapter 6: Input Commands

`define_charging_model`

Table 27 Parameters of `define_charging_model` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
<code>electron_fluxes</code>	List	<p>Sets the flux values $\{\Gamma_i\}$ of the angle distribution for the electrons, given by:</p> $f(\theta) = \frac{1}{2\pi} \sum_i \Gamma_i \cdot (m_i + 1) \cdot \cos^{m_i}(\theta)$ <p>where parameters m_i are specified by parameter <code>electron_exponents</code>. If not specified, then an isotropic distribution is used:</p> $f(\theta) = \frac{\Gamma_e}{\pi} \cos(\theta)$ <p>Note: The total electron flux $\Gamma_e = \sum_i \Gamma_i$ is rescaled to match the total ion flux to satisfy the charge neutrality condition. To switch off rescaling, set <code>enforce_charge_neutrality=false</code>.</p>	none [0, ∞[for each value in the list	mol/m ² /s
<code>enforce_charge_neutrality</code>	Boolean	Specifies whether to rescale the total electron flux such that it equals the total ion flux (to satisfy the charge neutrality condition).	true	none
<code>name</code>	Character	Sets the name of the charging model object. This name is referenced in the <code>define_etch_machine</code> command (see define_etch_machine on page 209).	none	none
<code>num_samples_per_cell</code>	Number	<p>Sets the number of charges per source area cell, sent to the structure during the charging simulation.</p> <p>This parameter controls the statistics and smoothness of the charge distribution on the surface during the charging phase. (It does not affect the physical total charge.) You can specify either <code>charge_scaling</code> or <code>num_samples_per_cell</code>.</p>	100 [0, ∞)	none

Chapter 6: Input Commands

define_charging_model

Table 27 Parameters of *define_charging_model* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
top_bc	Character	Sets the boundary condition of the electric solver at the top of the simulation domain. Options are: <ul style="list-style-type: none">• dirichlet• floating• neumann	neumann	none
top_electric_field	Number	Sets the normal electric field at the upper Neumann boundary. You can specify this parameter only if top_bc=neumann.	0 ($-\infty, \infty$)	V/m
top_potential	Number	Sets the potential applied at the Dirichlet boundary at the top of the simulation domain. You can specify this parameter only if top_bc=dirichlet.	0 ($-\infty, \infty$)	V

Examples

```
define_charging_model name=cm charging_factor=1 \
                      top_electric_field=-1e6
```

Chapter 6: Input Commands

define_damage

define_damage

This command defines the parameters of the damage model (see [Damage Modeling in PMC Simulations on page 42](#)).

Syntax

```
define_damage energy=<v> material=<c> name=<c> projected_range=<v> \
sigma=<v> sigma_lateral=<v> species=<c> total_damage=<n> type=<c> \
[max_depth=<n>] [max_lateral=<n>]
```

Table 28 Parameters of *define_damage* command

Parameter	Type	Description	Default	Unit
energy	Vector	Sets the energies corresponding to the energy-dependent parameters projected_range, sigma, and sigma_lateral. All four parameters are vectors of the same length.	none	eV
material	Character	Sets the material for which damage should accumulate. The <i>define_damage</i> command should be called multiple times for every combination of material and species that add to the accumulated damage.	none	none
max_depth	Number	For performance reasons, damage is computed only to <i>max_depth</i> from the point where the specified species impacts the surface.	$R_p + 3.0 \times \text{sigma}$	μm
max_lateral	Number	For performance reasons, damage is computed only to <i>max_lateral</i> from the point where the specified species impacts the surface.	$3.0 \times \text{sigma}_\text{lateral}$	μm
name	Character	Sets the name of the damage parameter set.	none	none
projected_range	Vector	Sets R_p , the peak of the profile.	none	μm
sigma	Vector	Sets σ , the spread of the profile in the primary direction.	none	μm
sigma_lateral	Vector	Sets σ_l , the spread of the profile in the lateral direction.	none	μm

Chapter 6: Input Commands

define_damage

Table 28 Parameters of define_damage command (Continued)

Parameter	Type	Description	Default	Unit
species	Character	Sets the species for which damage should accumulate. The <code>define_damage</code> command is called multiple times for each species and material specified.	none	none
total_damage	Number	Specifies the damage scale.	none	μm^{-2}
type	Character	Sets the type of damage calculation to use. Options are: <ul style="list-style-type: none"> • analytic_expression • hobler 	none	none

Description

The `projected_range`, `sigma`, and `sigma_lateral` parameters are specified as a piecewise linear function of energy, specifying a set of equal-length double arrays along with a double array of corresponding energies. For example:

```
define_damage <other required parameters> \
  energy = { 110.0<eV> 220.0<eV> 330.0<eV> } \
  projected_range = { 4.0<nm> 5.0<nm> 6.0<nm> } \
  sigma = { 0.70<nm> 0.80<nm> 0.90<nm> } \
  sigma_lateral = { 0.7<nm> 0.75<nm> 0.8<nm> }
```

where the projected range is 5 nm for energy 220 eV. Similarly, `sigma`=0.7 nm for 110 eV and 0.9 nm for 330 eV.

In addition, the parameters `max_depth` and `max_lateral` can be specified to balance accuracy with performance. Due to the large number of ion impacts in a typical simulation, the damage calculation can be computationally expensive. These parameters limit the domain considered for damage accumulation and are typically set to a small multiple (for example, 3) of `sigma` and `sigma_lateral`, respectively.

Examples

```
define_damage name=mydamage type=hobler species=N material=Nitride \
  total_damage=1.0 max_depth=5<nm> max_lateral=1.5<nm> \
  energy = { 110.0<eV> 220.0<eV> 330.0<eV> } \
  projected_range = { 2.0<nm> 2.25<nm> 2.5<nm> } \
  sigma = { 0.20<nm> 0.25<nm> 0.3<nm> } \
  sigma_lateral = { 0.2<nm> 0.25<nm> 0.3<nm> }
```

Chapter 6: Input Commands

define_deposit_machine

```
define_damage name=mydamage type=hobler species=N material=resist \
  total_damage=2.0 max_depth=4.5<nm> max_lateral=1.25<nm> \
  energy = { 100<eV> 200<eV> 300<eV> } \
  projected_range = { 1.75<nm> 2.0<nm> 2.25<nm> } \
  sigma = { 0.175<nm> 0.2<nm> 0.225<nm> } \
  sigma_lateral = { 0.175<nm> 0.2<nm> 0.225<nm> }
```

define_deposit_machine

This command defines a new machine for a deposition process. A deposition model can be specified with the parameter `model`.

Syntax

Atomic layer deposition:

```
define_deposit_machine anisotropy=<n> conformality=<n> \
  cycle_duration=<n> (exponent=<n> | iad=<c>) \
  growth_per_cycle=<n> material=<c> model=ald sticking=<n> \
  [nad=<c> | neutral_exponent=<n>] [name=<c>] [rotation=<n>] [tilt=<n>]
```

Crystal orientation-dependent deposition:

```
define_deposit_machine material=<c> model=crystal rate_100=<n> \
  rate_110=<n> rate_111=<n> \
  [name=<c>] [selective_materials=<l>]
```

Electrodeposition:

```
define_deposit_machine acc_adsorption_rate=<n> \
  acc_bulk_concentration=<n> bulk_distance=<n> \
  depo_bulk_concentration=<n> exchange_current_density=<n> \
  exchange_current_density_acc=<n> exchange_current_density_inh=<n> \
  inh_adsorption_rate=<n> inh_bulk_concentration=<n> \
  inh_diffusivity=<n> inh_displacement_rate=<n> \
  material=<c> model=electrodeposition overpotential=<n> \
  temperature=<n> \
  [acc_alpha=<n>] [acc_diffusivity=<n>] \
  [acc_saturation_surface_concentration=<n>] [alpha=<n>] \
  [curvature_scaling=<n>] [depo_diffusivity=<n>] \
  [depo_molar_volume=<n>] [depo_reference_concentration=<n>] \
  [duty_cycle=<n>] [inh_alpha=<n>] \
  [inh_saturation_surface_concentration=<n>] [name=<c>] \
  [overpotential_off=<n>] [z=<n>] [z_alpha=<n>]
```

Electroplating deposition:

```
define_deposit_machine delta=<n> material=<c> model=electroplating \
  rate=<n> [name=<c>]
```

Chapter 6: Input Commands

```
define_deposit_machine
```

High-density plasma deposition:

```
define_deposit_machine anisotropy=<n> (exponent=<n> | iad=<c>) \
material=<c> model=hdp rate=<n> redeposition=<n> \
s1=<n> s2=<n> sputter_rate=<n> sticking=<n> \
[nad=<c> | neutral_exponent=<n>] [name=<c>] [rotation=<n>] [tilt=<n>]
```

High-density plasma 2 deposition:

```
define_deposit_machine anisotropy=<n> (exponent=<n> | iad=<c>) \
material=<c> model=hdp2 \
rate=<n> reflection=<n> redeposition=<n> s1=<n> s2=<n> \
sputter_exponent=<n> sputter_rate=<n> sputter_type=<c> sticking=<n> \
[nad=<c> | neutral_exponent=<n>] [name=<c>] [rotation=<n>] [tilt=<n>]
```

Low-pressure chemical vapor deposition:

```
define_deposit_machine material=<c> model=lpcvd rate=<n> sticking=<n> \
[nad=<c> | neutral_exponent=<n>] [name=<c>] [rotation=<n>] [tilt=<n>]
```

Plasma-enhanced chemical vapor deposition:

```
define_deposit_machine anisotropy=<n> (exponent=<n> | iad=<c>) \
material=<c> model=pecvd rate=<n> sticking=<n> \
[nad=<c> | neutral_exponent=<n>] [name=<c>] [rotation=<n>] [tilt=<n>]
```

Physical vapor deposition:

```
define_deposit_machine (exponent=<n> | iad=<c>) material=<c> model=pvd \
rate=<n> [name=<c>] [rotation=<n>] [tilt=<n>]
```

Simple deposition:

```
define_deposit_machine anisotropy=<n> curvature=<n> material=<c> \
model=simple rate=<n> [name=<c>] [rotation=<n>] [tilt=<n>]
```

Spin-on-glass deposition:

```
define_deposit_machine angular_position=<n> angular_velocity=<n> \
density=<n> evaporation_rate=<n> initial_thickness=<n> \
material=<c> model=spin_on \
radial_distance=<n> surface_tension=<n> viscosity=<n> [name=<c>]
```

RFM model:

```
define_deposit_machine material=<c> model=<c> [iad=<c>] \
[maximum_error=<n>] [nad=<c>] [name=<c>] [reflection=<c>] \
[rotation=<n> | <c>] [tilt=<n>] [yield=<c>] ...
```

Chapter 6: Input Commands

define_deposit_machine

Table 29 Parameters of define_deposit_machine command

Parameter	Type	Description	Default [Range]	Unit
acc_adsorption_rate	Number	Sets the rate constant of the accelerator adsorption reaction.	none [0, ∞[μm/min
acc_alpha	Number	Sets the transfer coefficient for a surface fully covered by accelerators.	0.4 (from [3]) [0, 1]	none
acc_bulk_concentration	Number	Sets the bulk concentration of accelerators.	none [0, ∞[mol/cm ³
acc_diffusivity	Number	Sets the diffusivity of accelerators. If this parameter is omitted, the concentration of accelerators will be assumed to equal the value provided by the parameter acc_bulk_concentration everywhere.	none]0, ∞[cm ² /s
acc_saturation_surface_concentration	Number	Sets the accelerator surface concentration of a surface fully covered by accelerators.	8e-10 (from [3])]0, ∞[mol/cm ²
alpha	Number	Sets the transfer coefficient for a clean surface, that is, for a surface where no additives are adsorbed.	0.5 (from [3]) [0, 1]	none
angular_position	Number	Sets the angle between the plane that contains the yz plane of the wafer coordinate system and the plane that contains the z-axis of the wafer coordinate system and the center of the structure on which the process occurs.	none [-180, 180]	degree
angular_velocity	Number	Sets the angular velocity of the wafer.	none [0, ∞[rpm
anisotropy	Number	Sets the anisotropy coefficient.	none [0, 1]	none

Chapter 6: Input Commands

define_deposit_machine

Table 29 Parameters of *define_deposit_machine* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
bulk_distance	Number	Sets the distance from the topmost point of the exposed surface of the plane where the species concentrations are assumed to equal their bulk values.	none]0, ∞[µm
conformality	Number	Sets the conformality parameter for the atomic layer deposition (ALD) process.	none]0, ∞[none
curvature	Number	Sets the curvature coefficient.	none [0, 0.1]	µm
curvature_scaling	Number	Sets the factor by which the surface mean curvature is scaled to update the additive surface coverage.	1 [0, ∞[none
cycle_duration	Number	Sets the duration of one deposition cycle.	none]0, ∞[s
delta	Number	Sets the gradient of the accelerator surface concentration for the electrodeposition model.	none [0, ∞[µm ⁻¹
density	Number	Sets the density of the film to be deposited.	none]0, ∞[g/cm ³
depo_bulk_concentration	Number	Sets the bulk concentration of the deposited material.	none]0, ∞[mol/cm ³
depo_diffusivity	Number	Sets the diffusivity of the deposited material. If this parameter is omitted, the concentration of the material being deposited is assumed to equal its bulk value.	none]0, ∞[cm ² /s
depo_molar_volume	Number	Sets the molar volume of the deposited material.	7.1e-6 (from [4]) [0, ∞[m ³ /mol

Chapter 6: Input Commands

define_deposit_machine

Table 29 Parameters of *define_deposit_machine* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
depo_reference_concentration	Number	Sets the concentration at which exchange current densities are measured. If this parameter is omitted, the value of the parameter depo_bulk_concentration is assumed as the reference concentration.	none [0, ∞[mol/cm ³
duty_cycle	Number	Sets the duty cycle of the overpotential, that is, a fraction of a pulse period. The overpotential takes the value given by the overpotential parameter. In the remaining time of a pulse period, the overpotential is assumed to equal the value given by the overpotential_off parameter.	1 (0, 1]	none
evaporation_rate	Number	Sets the evaporation rate of the film to be deposited.	none [0, ∞[μm min ⁻¹
exchange_current_density	Number	Sets the exchange current density on a free surface, measured at the reference concentration set by the parameter depo_reference_concentration.	none [0, ∞[mA/cm ²
exchange_current_density_acc	Number	Sets the exchange current density on a surface fully covered by accelerators, measured at the reference concentration set by the parameter depo_reference_concentration.	none [0, ∞[mA/cm ²
exchange_current_density_inh	Number	Sets the exchange current density on a surface fully covered by inhibitors, measured at the reference concentration set by the parameter depo_reference_concentration.	none [0, ∞[mA/cm ²
exponent	Number	Sets the exponent used to characterize the ion angular distribution $\cos^m\theta$.	none [1, ∞[none

Chapter 6: Input Commands

define_deposit_machine

Table 29 Parameters of define_deposit_machine command (Continued)

Parameter	Type	Description	Default [Range]	Unit
growth_per_cycle	Number	Sets the growth per cycle.	none [0, ∞[μm
iad	Character	Sets the name of the IAD to be used.	none	none
inh_adsorption_rate	Number	Sets the rate constant of the inhibitor adsorption reaction.	none [0, ∞[μm/min
inh_alpha	Number	Sets the transfer coefficient for a surface fully covered by inhibitors.	0.5 (from [3]) [0, 1]	none
inh_bulk_concentration	Number	Sets the bulk concentration of inhibitors.	none [0, ∞[mol/cm ³
inh_diffusivity	Number	Sets the diffusivity of inhibitors.	none]0, ∞[cm ² /s
inh_displacement_rate	Number	Sets the rate constant of the reaction by which inhibitors are displaced by accelerators.	none [0, ∞[μm/min
inh_saturation_surface_concentration	Number	Sets the inhibitor surface concentration of a surface fully covered by inhibitors.	6e-11 (from [3])]0, ∞[mol/cm ²
initial_thickness	Number	Sets the thickness of the film over the substrate when the spin-on simulation starts.	none]0, ∞[μm
material	Character	Sets the material to be deposited.	none	none
maximum_error	Number	Sets the maximum tolerated equivalence error for all species of the model. This parameter can be used only if rotation=continuous. See Continuously Rotating and Tilted Structure Modeling on page 57 .	0.5 [0, 0.5]	none

Chapter 6: Input Commands

define_deposit_machine

Table 29 Parameters of define_deposit_machine command (Continued)

Parameter	Type	Description	Default [Range]	Unit
model	Character	Sets the model used to represent the physical machine.	none	none
nad	Character	Sets the name of the neutral angular distribution (NAD) to be used.	none	none
name	Character	Sets the name of the machine.	default_machine	none
neutral_exponent	Number	Sets the exponent used to characterize the NAD $\cos^m\theta$.	1 [1, ∞ [none
overpotential	Number	Sets the overpotential of the electroplating cell during the on-cycle of the overpotential pulse.	none]- ∞ , 0]	V
overpotential_off	Number	Sets the overpotential of the electroplating cell during the off-cycle of the overpotential pulse. This parameter can be specified only when duty_cycle is less than 1.	0]- ∞ , 0]	V
radial_distance	Number	Sets the distance of the center of the feature from the axis of rotation of the wafer.	none [0, ∞ [μm
rate	Number	Sets the deposition rate.	none	$\mu\text{m min}^{-1}$
rate_100	Number	Sets the deposition rate for the <100> direction.	none [0, ∞ [$\mu\text{m min}^{-1}$
rate_110	Number	Sets the deposition rate for the <110> direction.	none [0, ∞ [$\mu\text{m min}^{-1}$
rate_111	Number	Sets the deposition rate for the <111> direction.	none [0, ∞ [$\mu\text{m min}^{-1}$
redeposition	Number	Sets the ratio of the amount of material redeposited to the amount of material sputtered from the surface.	none [0, 1]	none

Chapter 6: Input Commands

define_deposit_machine

Table 29 Parameters of define_deposit_machine command (Continued)

Parameter	Type	Description	Default [Range]	Unit
reflection	Number/Character	For built-in models, it sets the reflection parameter used to evaluate the reflection probability (parameter k in Equation 12). For RFM models, it sets the name of the reflection function to use.	none [0, 1] for built-in models	none
rotation	Number/Character	Sets either the rotation angle in the wafer coordinate system or whether the wafer rotates continuously.	0 [-180, 180] / [continuous]	degree / none
s1	Number	Sets the first sputter coefficient.	none	none
s2	Number	Sets the second sputter coefficient.	none	none
selective_materials	List	Sets a list of materials on which the deposition can occur. If this list is empty, deposition occurs on all materials of the structure.	empty	none
sputter_exponent	Number	Sets the exponent used to characterize the angular distribution $\cos^m\theta$ of the sputtered material. The specified value must be an integer.	none [1, ∞[none
sputter_rate	Number	Sets the sputtering rate. The net deposition rate is equal to rate - sputter_rate.	none [0, 2]	μm min ⁻¹
sputter_type	Character	Sets the angular distribution type of the sputtered material. Options are: <ul style="list-style-type: none">• diffuse• reflective	none	none
sticking	Number	Sets the sticking probability of the deposition precursors.	none [0, 1]	none

Chapter 6: Input Commands

`define_deposit_machine`

Table 29 Parameters of `define_deposit_machine` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
surface_tension	Number	Sets the surface tension of the film to be deposited with respect to the surrounding fluid.	none [0, ∞[dyn/cm
temperature	Number	Sets the absolute temperature of the electroplating process.	none [0, ∞[K
tilt	Number	Sets the tilt angle.	0 [0, 90[degree
viscosity	Number	Sets the dynamic viscosity of the film to be deposited.	none]0, ∞[poise
yield	Character	Sets the name of the yield function to use for the deposition machine.	none	none
z	Number	Sets the number of electrons involved in deposition reactions.	2 [1, 2147483647]	none
z_alpha	Number	Sets the number of electrons transferred in the rate-determining elementary reaction.	1 [1, 2147483647]	none

Limitations

The following limitations or conditions apply:

- The `rotation` and `tilt` parameters do not support the radian measurement unit and are not supported when `model=crystal`, `model=electrodeposition`, or `model=spin_on`.
- There are only two possible numeric values of the parameter `rotation` when defining a machine to be used with a 2D structure. The numeric values are obtained by converting, if needed, the angles given by:

`-slice_angle [deg] + 90°`
`-slice_angle [deg] - 90°`

into the range [-180, 180] degrees. In these formulas, `slice_angle` denotes the slice angle of the structure, set by the `slice_angle` parameter of the `define_structure` command or stored in the loaded TDR file.

Chapter 6: Input Commands

define_deposit_machine

When the default value of the slice angle is used (-90), the possible values of rotation are 0 and 180 degrees, respectively, for a machine to be used with a 2D structure (see [Simulation Coordinate System on page 34](#) and [define_structure on page 309](#)).

- You can use rotation=continuous when defining a machine to be used with a 2D structure that uses a rate formula module (RFM) model supporting continuous rotation, independently of the value of the slice angle (see [Continuously Rotating and Tilted Structure Modeling on page 57](#), [Data Available for Rate Calculation on page 512](#), and [Chapter 11 on page 531](#)).

Examples

Define a machine named simpledepo for a silicon deposition process that uses the model simple, and set the anisotropy coefficient to 0.5, the curvature coefficient to 0.05 μm, and the deposition rate to 1 μm/minute:

```
define_deposit_machine anisotropy=0.5 curvature=0.05 \
    material=Silicon model=simple name=simpledepo rate=1
```

You can define the wafer plane tilt by using the tilt and rotation angles as described in [Structure Tilt on page 36](#).

The same example with a tilted structure is:

```
define_deposit_machine anisotropy=0.5 curvature=0.05 \
    material=Silicon model=simple name=simpledepo \
    rate=1 rotation=20 tilt=45
```

Chapter 6: Input Commands

define_electric_contact

define_electric_contact

This command defines an electric contact.

When simulating charge-up with a PMC model, you can use this command to assign a fixed potential to a conducting region in a structure.

Note:

If there is no conducting material at the specified position in the structure, then the command is ignored.

Syntax

```
define_electric_contact name=<c> position=<v> potential=<n>
```

Table 30 Parameters of define_electric_contact command

Parameter	Type	Description	Default [Range]	Unit
name	Character	Sets the name of the electric contact to be referenced in the command <code>define_electric_solver</code> (see define_electric_solver on page 206).	none	none
position	Vector	Sets the position of the point-like electric contact. The contact only takes effect if the position lies inside the conducting material. Conducting material can be defined in the command <code>define_species_properties</code> .	none $(-\infty, \infty) \times (-\infty, \infty)$ $\times (-\infty, \infty)$	μm
potential	Number	Sets the potential applied to the conducting material at the electric contact.	none $(-\infty, \infty)$	V

Chapter 6: Input Commands

define_electric_solver

define_electric_solver

This command defines an electric solver.

When simulating charge-up with a PMC model, you can use this command to configure the Poisson solver for computing the electric field.

Note:

This command is optional. If you do not define the electric solver explicitly, then the default values are used for the charge-up simulation.

Syntax

```
define_electric_solver name=<c> \
    [grc_grid_refinement_ratio=<n>] \
    [grc_max_num_iterations=<n>] \
    [grc_preconditioner=<c>] \
    [grc_solver=<c>] \
    [grc_solver_abs_error=<n>] \
    [grc_solver_optimization=<b>] \
    [grc_solver_rel_error=<n>]
```

Table 31 Parameters of *define_electric_solver* command

Parameter	Type	Description	Default [Range]	Unit
grc_grid_refinement_ratio	Number	Sets the mesh size of the Poisson solver grid in units of the PMC grid spacing.	1 [1, ∞)	none
grc_max_num_iterations	Number	Sets the maximum number of iterations for the Poisson solver to converge to the solution.	10000000 [1, ∞)	none
grc_preconditioner	Character	Selects the preconditioner used by the Poisson solver to prepare the linear matrix. Options are: <ul style="list-style-type: none">• spai0_preconditioner• ilu0_preconditioner• ilut_preconditioner• no_preconditioner	spai0_preconditioner	none

Chapter 6: Input Commands

`define_electric_solver`

Table 31 Parameters of `define_electric_solver` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
<code>grc_solver</code>	Character	Selects the linear solver used by Poisson solver to compute the electric potential. Options are: <ul style="list-style-type: none"> • <code>bicgstab_solver</code> • <code>cg_solver</code> • <code>cgs_solver</code> • <code>cgs2_solver</code> • <code>gmres_solver</code> 	<code>bicgstab_solver</code>	none
<code>grc_solver_abs_error</code>	Number	Sets the convergence threshold for the iterative linear solver for the electric potential, used by the Poisson solver. The absolute error is defined as the residuum of the linear equation system.	0 [0, ∞)	none
<code>grc_solver_optimization</code>	Boolean	Specifies whether to dynamically select the linear solver and preconditioner to optimize robustness and performance. (If switched on, then the values of <code>grc_solver</code> and <code>grc_preconditioner</code> are used as initial solvers or preconditioners, but they can change during the simulation.)	true	none
<code>grc_solver_rel_error</code>	Number	Sets the convergence threshold for the iterative linear solver for the electric potential, used by the Poisson solver. The relative error is defined as the relative residuum of the linear equation system (that is, the final residuum divided by the initial residuum).	1e-8 [0, ∞)	none
<code>name</code>	Character	Sets the name of the electric solver object. This name is referenced in the <code>etch</code> command.	none	none

Chapter 6: Input Commands

define_electric_solver

Examples

```
define_electric_solver name=cp \
    grc_grid_refinement_ratio=4 \
    grc_solver=cgs_solver \
    grc_preconditioner=ilu0_preconditioner \
    grc_solver_abs_error=1e-6 \
    grc_max_num_iterations=100000000
```

Chapter 6: Input Commands

define_etch_machine

define_etch_machine

This command defines a new machine for an etching model, or a simultaneous etching and deposition model, or a deposition reaction model. Parameters defined with the define_etch_machine command are material independent except when setting model=crystal.

Material-dependent parameters are defined with the add_material command. When setting model=crystal, the specified etching rates refer to the single etchable material. The add_material command cannot be used with a machine having model=crystal.

Syntax

Crystal orientation-dependent etching:

```
define_etch_machine etchable_material=<c> model=crystal \
    rate_100=<n> rate_110=<n> rate_111=<n> \
    [name=<c>] [pattern_density_model=<c>]
```

Dry etching:

```
define_etch_machine deposit_material=<c> model=dry \
    rate=<n> sticking=<n> \
    [nad=<c> | neutral_exponent=<n>] [name=<c>] \
    [pattern_density_model=<c>] [rotation=<n>] [tilt=<n>]
```

Simultaneous etching and deposition:

```
define_etch_machine (exponent=<n> | iad=<c>) deposit_material=<c> \
model=etchdepo rate=<n> sticking=<n> \
    [name=<c>] [pattern_density_model=<c>] [rotation=<n>] [tilt=<n>]
```

Simultaneous etching and deposition 2:

```
define_etch_machine (exponent=<n> | iad=<c>) deposit_material=<c> \
model=etchdepo2 rate=<n> sticking=<n> \
    [nad=<c> | (neutral_etch_exponent=<n> | neutral_exponent=<n>)] \
    [name=<c>] [pattern_density_model=<c>] [rotation=<n>] [tilt=<n>]
```

High-density plasma etching:

```
define_etch_machine (exponent=<n> | iad=<c>) model=hdp \
    [name=<c>] [pattern_density_model=<c>] [rotation=<n>] [tilt=<n>]
```

High-density plasma 2 etching:

```
define_etch_machine (exponent=<n> | iad=<c>) model=hdp2 \
    [nad=<c> | neutral_exponent=<n>] [name=<c>] \
    [pattern_density_model=<c>] [rotation=<n>] [tilt=<n>]
```

Chapter 6: Input Commands

```
define_etch_machine
```

Ion-enhanced etching:

```
define_etch_machine (exponent=<n> | iad=<c>) model=ion_enhanced \
[nad=<c> | neutral_exponent=<n>] [name=<c>] \
[pattern_density_model=<c>] [rotation=<n>] [tilt=<n>]
```

Ion-milling:

```
define_etch_machine model=ionmill [name=<c>] \
[pattern_density_model=<c>] [rotation=<n>] [tilt=<n>]
```

Reactive ion etching:

```
define_etch_machine (exponent=<n> | iad=<c>) model=rie \
[name=<c>] [pattern_density_model=<c>] [rotation=<n>] [tilt=<n>]
```

Reactive ion etching 2:

```
define_etch_machine (exponent=<n> | iad=<c>) model=rie2 \
[nad=<c> | neutral_exponent=<n>] [name=<c>] \
[pattern_density_model=<c>] [rotation=<n>] [tilt=<n>]
```

Simple etching:

```
define_etch_machine model=simple [name=<c>] \
[pattern_density_model=<c>] [rotation=<n>] [tilt=<n>]
```

Wet etching:

```
define_etch_machine diffusivity=<n> model=wet source_distance=<n> \
[name=<c>] [pattern_density_model=<c>] [source_concentration=<n>]
```

RFM model:

```
define_etch_machine model=<c> [applied_pressure=<n>] \
[deposit_material=<c>] [iad=<c>] [maximum_error=<n>] [nad=<c>] \
[name=<c>] [pad_poisson_ratio=<n>] [pad_roughness=<n>] \
[pad_young_modulus=<n>] [pattern_density_model=<c>] \
[reflection=<c>] [rotation=<n> | <c>] [tilt=<n>] [yield=<c>] ...
```

Reaction model:

```
define_etch_machine model=<c> species_distribution=<c> \
[charging_model=<c>] [damage=<c>] [name=<c>] \
[pattern_density_model=<c>] \
[rotation=<n> | <c>] [species_properties=<c>] \
[source_position=<n>] [tilt=<n>] \
[top_plane_distance=<n>] [volumetric_species_distribution=<c>] \
[yield=<c>]
```

Chapter 6: Input Commands

`define_etch_machine`

Table 32 Parameters of `define_etch_machine` command

Parameter	Type	Description	Default [Range]	Unit
applied_pressure	Number	Sets the external pressure applied to the pad. It is mandatory for RFM models using the RFM function <code>pad_pressure()</code> .	none [0, ∞[Pa
charging_model	Character	When simulating charge-up, this parameter sets the name of the charging model, defined by the <code>define_charging_model</code> command (see define_charging_model on page 189).	none	none
damage	Character	You can use this optional parameter with the reaction model that is available for PMC steps. It switches on the accumulation of damage by naming profiles created with the <code>define_damage</code> command.	none	none
deposit_material	Character	Sets the deposition material for simultaneous etching and deposition models.	none	none
diffusivity	Number	Sets the diffusivity of the etchant species.	none]0, ∞[cm ² /s
etchable_material	Character	Sets the material to be etched.	none	none
exponent	Number	Sets the exponent used to characterize the ion angular distribution $\cos^m\theta$.	none [1, ∞[none
iad	Character	Sets the name of the IAD to be used.	none	none

Chapter 6: Input Commands

define_etch_machine

Table 32 Parameters of *define_etch_machine* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
maximum_error	Number	Sets the maximum tolerated equivalence error for all species of the model. This parameter can be used only if rotation=continuous. See Continuously Rotating and Tilted Structure Modeling on page 57 .	0.5 [0, 0.5]	none
model	Character	Sets the model used to represent the physical machine.	none	none
nad	Character	Sets the name of the NAD to be used.	none	none
name	Character	Sets the name of the etch machine.	default_machine	none
neutral_etch_exponent	Number	Sets the exponent used to characterize the angular distribution $\cos^m\theta$ of the etching neutral flux ($\Gamma_{\text{neutral etch}}$ in Equation 40 of the model etchdepo2). Note: This parameter is available only when <code>model=etchdepo2</code> .	1 [1, ∞[none
neutral_exponent	Number	Sets the exponent used to characterize the neutral angular distribution $\cos^m\theta$. When <code>model=etchdepo2</code> , this parameter refers to the deposition neutral flux (Γ_{neutral} in Equation 40).	1 [1, ∞[none
pad_poisson_ratio	Number	Sets the Poisson ratio of the pad. It is mandatory for RFM models using the RFM function <code>pad_pressure()</code> .	none]-1, 0.5]	none

Chapter 6: Input Commands

define_etch_machine

Table 32 Parameters of define_etch_machine command (Continued)

Parameter	Type	Description	Default [Range]	Unit
pad_roughness	Number	This parameter is related to the standard deviation of the distribution of the pad asperities. It is mandatory for RFM models using the RFM function pad_pressure().	none]0, ∞[µm
pad_young_modulus	Number	Sets the Young's modulus of the pad. It is mandatory for RFM models using the RFM function pad_pressure().	none]0, ∞[Pa
pattern_density_model	Character	Sets the pattern density model defined in the define_pattern_density_model command (see define_pattern_density_model on page 256). This parameter can be used only for level set models.	none	none
rate	Number	Sets the rate of deposition for the material that was specified with the deposit_material parameter.	none]0, ∞[µm min ⁻¹
rate_100	Number	Sets the etching rate for the <100> direction.	none]0, ∞[µm min ⁻¹
rate_110	Number	Sets the etching rate for the <110> direction.	none]0, ∞[µm min ⁻¹
rate_111	Number	Sets the etching rate for the <111> direction.	none]0, ∞[µm min ⁻¹
reflection	Character	Sets the name of the reflection function to be used.	none	none
rotation	Number/ Character	Sets either the rotation angle in the wafer coordinate system or whether the wafer rotates continuously.	0 [-180, 180] / continuous	degree / none

Chapter 6: Input Commands

define_etch_machine

Table 32 Parameters of *define_etch_machine* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
source_concentration	Number	Sets the concentration of the etchant species on the source plane.	1]0, ∞[mol/cm ³
source_distance	Number	Sets the distance of the plane where the etchant concentration is assumed to be fixed from the topmost point of the exposed surface.	none]0, ∞[μm
source_position	Number	If specified, the z-coordinate of the particle source in the PMC simulator is fixed to this height. Otherwise, the position of the source is adjusted dynamically to keep a fixed distance to the structure top during the simulation. Note: The source position must lie inside the simulation bounding box. To extend the bounding box in the vertical direction, use the command fill material=Gas thickness=<n>, or use the top_gas_thickness parameter in the etch or filter_structure command.	none	μm

Chapter 6: Input Commands

define_etch_machine

Table 32 Parameters of define_etch_machine command (Continued)

Parameter	Type	Description	Default [Range]	Unit
species_distribution	Character	Sets the name of the species distribution to be used. Note: This parameter must be specified if the model set by the parameter <code>model</code> is a reaction model containing source species defined using the <code>add_source_species</code> command (see add_source_species on page 181).	none	none
species_properties	Character	Sets the name of the species properties to be used.	none	none
sticking	Number	Sets the sticking coefficient for the deposition material that was specified with the parameter <code>deposit_material</code> .	none [0, 1]	none
tilt	Number	Sets the tilt angle.	0 [0, 90[degree
top_plane_distance	Number	Sets the distance between the (dynamic) top plane and the exposed surface of the structure. This parameter can be used only for top plane reactions (see Example of Top Plane Reactions on page 167).	none]0, ∞]	µm

Chapter 6: Input Commands

define_etch_machine

Table 32 Parameters of define_etch_machine command (Continued)

Parameter	Type	Description	Default [Range]	Unit
volumetric_species_distribution	Character	Sets the name of the volumetric species distribution to be used. Note: This parameter must be specified if the model set by the parameter <code>model</code> is a reaction model containing volumetric source species defined using the command <code>add_volumetric_source_species</code> (see add_volumetric_source_species on page 183).	none	none
yield	Character	Sets the name of the yield function to be used.	none	none

Limitations

The following limitations or conditions apply:

- The `rotation` and `tilt` parameters do not support the radian measurement unit and are not supported when `model=crystal` or `model=wet`.
- There are only two possible numeric values of the parameter `rotation` when defining a machine to be used with a 2D structure. The numeric values are obtained by converting, if needed, the angles given by:

`-slice_angle [deg] + 90°`
`-slice_angle [deg] - 90°`

into the range [-180, 180] degrees. In these formulas, `slice_angle` denotes the slice angle of the structure, set by the `slice_angle` parameter of the `define_structure` command or stored in the loaded TDR file.

When the default value of the slice angle is used (-90), the possible values of `rotation` are 0 and 180 degrees for a machine to be used with a 2D structure (see [Simulation Coordinate System on page 34](#) and [define_structure on page 309](#)).

- You can use `rotation=continuous` when defining a machine to be used with a 2D structure that uses a rate formula module (RFM) model supporting continuous rotation, independently of the value of the slice angle (see [Continuously Rotating and Tilted](#)

Chapter 6: Input Commands

define_etch_machine

[Structure Modeling on page 57](#), [Data Available for Rate Calculation on page 512](#), and [Chapter 11 on page 531](#)).

- When defining a machine that uses a reaction model, you can use `rotation=continuous` only if no source species of the reaction model has an energy-dependent distribution computed by HPEM, the Benoit-Cattin plasma sheath model, or the Edelberg–Aydil plasma sheath model, or imported from an EAD file (see [define_species_distribution on page 290](#)).
- When defining an etch machine that uses a reaction model, only two cases are supported:
 - The species distributions specified for all the source species of the used reaction models are energy independent. In this case, also the yield functions required by the reaction model and its reaction probabilities must be energy independent.
 - The species distributions specified for all the source species of the used reaction models are energy dependent. In this case, the yield functions required by the reaction model and its reaction probabilities can be either energy independent or energy dependent.

In all other cases, an error will be issued.

Examples

Define a machine named `etchmachine` for an etching process that uses the model `simple`:

```
define_etch_machine name=etchmachine model=simple
```

You can define the wafer plane tilt by using the tilt and rotation angles as described in [Structure Tilt on page 36](#).

The same example with a tilted structure is:

```
define_etch_machine name=etchmachine model=simple rotation=45 tilt=30
```

Chapter 6: Input Commands

define_extraction

define_extraction

This command allows you to define extractions that can be used during a deposition step or an etching step or a plasma simulation.

Extractions for Deposition and Etching Simulations

This section describes the `define_extraction` command for deposition and etching simulations.

Syntax

Define the extraction of an interface between two different materials, or two different regions, or two different parts along a line in a structure:

```
define_extraction name=<c> type=interface <extraction_line> \
    <extraction_pair> tcl_output_variable=<c> \
    [description=<c>] [output=<c>]
```

Define the extraction of a property of a structure along a line:

```
define_extraction name=<c> type=probe property=length \
    <extraction_line> (materials=<l> | [probe_empty_space=<b>]) \
    [description=<c>] [file=<c>] [output=<c>] [tcl_output_variable=<c>]

define_extraction name=<c> type=probe property=<c> \
    <extraction_line> tcl_output_variable=<c> \
    [description=<c>] [output=<c>]
```

With the following command, you can define the extraction of several PMC quantities in a given bounding box:

- Number of executed reactions
- Volume of the specified species
- Volume deposited or sputtered according to the specified reactions
- Effective sputter yield of the specified reactions
- Volume density of the starting points of the volumetric source species
- When simulating charge-up, the charge density, the electric field, and the electric potential

```
define_extraction type=pmc_data name=<c> quantities=<l> \
    [axis=<c> position=<n> [crop_plane=<b>]] [csv_file=<c>] \
    [description=<c>] \
    [expression_pattern=<c> | reactions=<l>] \
    [file=<c>] [grid_output=<b>] [normalization=<c>] \
```

Chapter 6: Input Commands

define_extraction

```
[operator=<c>] [point_max=<v> point_min=<v>] \
[species=<l>] [tdr_dataset_grouping=<c>] \
[tcl_output_variable=<c>]
```

Note:

The `pmc_data` extraction type is available only for PMC simulations.

Define the extraction of the intersection of a 3D boundary or PMC structure with a plane:

```
define_extraction type=slice <extraction_plane> name=<c> \
[description=<c>] [file=<c>] [tcl_output_variable=<c>]
```

In this syntax:

- An extraction line specification `<extraction_line>`, as defined in the `extract` command, specifies the location at which the extraction is performed.
- An extraction pair specification `<extraction_pair>`, as defined in the `extract` command, specifies a pair of materials, or parts, or regions for which the extraction is performed.
- An extraction plane specification `<extraction_plane>`, as defined in the `extract` command, specifies the plane in which the extraction is performed.

Table 33 Parameters of *define_extraction* command

Parameter	Type	Description	Default	Unit
axis ^{1 2}	Character	Sets the direction of one of the coordinate axes. Options are: <ul style="list-style-type: none">• x• y• z	none	none
crop_plane	Boolean	Specifies whether or not to limit the 2D results to the portion of the specified plane that intersects the bounding box specified by <code>point_max</code> and <code>point_min</code> . If the specified plane does not intersect the given bounding box, then an error is issued.	false	none

Chapter 6: Input Commands

define_extraction

Table 33 Parameters of *define_extraction* command (Continued)

Parameter	Type	Description	Default	Unit
csv_file	Character	<p>Sets the name of the CSV file where to write the extracted scalar results in CSV format.</p> <p>If omitted, no CSV output is produced.</p> <p>If the specified file already exists when the first extraction is performed, then it is overwritten.</p> <p>Note: If multiple <code>define_extraction</code> commands with the same <code>name</code> value use the same value for the <code>csv_file</code> parameter, then an error is issued.</p>	none	none
description	Character	Describes the extraction.	none	none
direction ¹	Vector	Sets the direction of the extraction line.	none	none
expression_pattern	Character	<p>Sets the search pattern of the reaction expression that specifies for which reactions the results must be extracted. The search pattern must be given in the glob syntax.</p> <p>Results are extracted for all reactions with an expression that matches the specified search pattern.</p> <p>In any case, the set of matching reactions:</p> <ul style="list-style-type: none"> • Must not be empty • Must contain either only top plane reactions or no top plane reactions <p>You can specify this parameter when the parameter <code>quantities</code> contains the value <code>deposited_volume</code>, <code>reaction_executions</code>, <code>sputtered_volume</code>, or <code>sputter_yield</code>.</p> <p>When it denotes top plane reactions, the parameter <code>quantities</code> must contain only the value <code>reaction_executions</code>.</p>	none	none

Chapter 6: Input Commands

define_extraction

Table 33 Parameters of define_extraction command (Continued)

Parameter	Type	Description	Default	Unit
file	Character	Sets the file name or path to a file where the extraction results will be saved.	none	none
grid_output	Boolean	Specifies whether or not to produce 3D results.	false	none
material ³	Character	Sets the name of the first material of an extraction pair specification.	none	none
material2 ³	Character	Sets the name of the second material of an extraction pair specification.	none	none
materials	List	Sets the materials to take into account. Note: This parameter applies only if property=length.	none	none
name	Character	Sets the name of the extraction group to which the extraction is added.	none	none
normal ²	Vector	Sets the normal of the specified plane.	none	none
normalization	Character	Sets how to normalize the sum of the extracted quantities over the boundary box. Options are: <ul style="list-style-type: none">• none• box When normalization=box, the sum of the quantity values collected in the given bounding box, divided by the bounding box volume (area), is computed. When normalization=none, the sum of the quantity values collected in the given bounding box is computed.	none	none

Chapter 6: Input Commands

define_extraction

Table 33 Parameters of define_extraction command (Continued)

Parameter	Type	Description	Default	Unit
operator	Character	<p>Sets the mathematical operation to reduce the 3D volume data to a scalar value for the quantities specified by the parameter <code>quantities</code>.</p> <p>Reduction operators are available for the quantities <code>charge_density</code>, <code>electric_field</code>, and <code>electric_potential</code>:</p> <ul style="list-style-type: none"> • <code>mean</code> • <code>max</code> • <code>meanabs</code> • <code>min</code> • <code>stddev</code> • <code>sum</code> • <code>sumabs</code> 	mean	none
output	Character	<p>Sets a filtering option. Options are:</p> <ul style="list-style-type: none"> • <code>all</code> • <code>first</code> • <code>inside</code> • <code>last</code> • <code>outside</code> 	all	none
point_max	Vector	<p>Sets the requested maximum corner of the bounding box where the data must be extracted. It can be set only when <code>type=pmc_data</code>.</p> <p>The actual maximum corner of the bounding box where the data must be extracted is obtained by snapping the given <code>point_max</code> value to the closest grid point of the PMC structure on which the extraction is executed.</p>	If <code>reactions</code> or <code>expression_pattern</code> denotes top plane reactions: $\{\infty \infty\}$ Otherwise or when neither <code>reactions</code> nor <code>expression_pattern</code> is specified: $\{\infty \infty \infty\}$	μm

Chapter 6: Input Commands

define_extraction

Table 33 Parameters of define_extraction command (Continued)

Parameter	Type	Description	Default	Unit
point_min	Vector	Sets the requested minimum corner of the bounding box where the data must be extracted. It can be set only when type=pmc_data. The actual minimum corner of the bounding box, where the data must be extracted, is obtained by snapping the given point_min value to the closest grid point of the PMC structure on which the extraction is executed.	If reactions or expression_pattern denotes top plane reactions: { -∞ -∞ } Otherwise or when neither reactions nor expression_pattern is specified: { -∞ -∞ -∞ }	μm
point ^{1 2}	Vector	Sets a point on the extraction line or plane.	none	μm
point1 ^{1 2}	Vector	Sets the first point on the extraction line or plane.	none	μm
point2 ^{1 2}	Vector	Sets the second point on the extraction line or plane.	none	μm
point3 ²	Vector	Sets the third point on the extraction plane.	none	μm
position ²	Number	Sets the cutting plane distance from the origin measured in the direction defined by axis.	none	μm
probe_empty_space	Boolean	Specifies whether or not to probe empty space. Note: This parameter applies only if property=length.	false	none

Chapter 6: Input Commands

define_extraction

Table 33 Parameters of define_extraction command (Continued)

Parameter	Type	Description	Default	Unit
property	Character	<p>Sets the property to be associated with the returned segments. Options are:</p> <ul style="list-style-type: none"> • length • material • region • region_part <p>Note: You can use <code>property=region</code> and <code>property=region_part</code> only during a deposition step or an etching step using <code>method=levelset</code>.</p>	none	none
quantities	List	<p>Specifies the quantities to extract. Options are:</p> <ul style="list-style-type: none"> • charge_density • deposited_volume • electric_field • electric_potential • reaction_executions • species_volume • sputter_yield • sputtered_volume • volumetric_source_particles <p>Note: When you specify <code>volumetric_source_particles</code>, there must be a volumetric source species in the used model.</p>	none	none

Chapter 6: Input Commands

define_extraction

Table 33 Parameters of define_extraction command (Continued)

Parameter	Type	Description	Default	Unit
reactions	List	<p>Sets the names of the reactions for which results must be extracted. The following conditions apply:</p> <ul style="list-style-type: none"> • The list must not be empty. • The list must contain either only top plane reactions or no top plane reactions. • All listed reactions must exist in the model of the etch machine used by the <code>etch</code> command that uses this extraction. <p>You can specify this parameter when <code>quantities</code> contains the value <code>deposited_volume</code>, <code>reaction_executions</code>, <code>sputtered_volume</code>, or <code>sputter_yield</code>.</p> <p>When it denotes top plane reactions, the parameter <code>quantities</code> must contain only <code>reaction_executions</code>.</p>	none	none
region_part1 ³	List	Sets the names of the first region and part in an extraction pair specification.	none	none
region_part2 ³	List	Sets the names of the second region and part in an extraction pair specification.	none	none
region1 ³	Character	Sets the name of the first region in an extraction pair specification.	none	none
region2 ³	Character	Sets the name of the second region in an extraction pair specification.	none	none
species	List	<p>Sets the names of the species for which the result should be extracted. If not specified, then all species are extracted.</p> <p>This parameter can be specified only when extracting <code>charge_density</code>.</p>	All species extracted	none

Chapter 6: Input Commands

define_extraction

Table 33 Parameters of define_extraction command (Continued)

Parameter	Type	Description	Default	Unit
tcl_output_variable	Character	<p>Sets the name of the Tcl variable to which the extraction results will be written.</p> <p>If omitted, no Tcl output is produced.</p> <p>If the specified Tcl variable already exists when the first extraction is performed, then its value is overwritten.</p> <p>Note: If multiple define_extraction commands with the same name value use the same value for tcl_output_variable, then an error is issued.</p>	none	none
tdr_file_dataset_grouping	Character	<p>Specifies how to group the datasets in the TDR file for plotting in Sentaurus Visual. This parameter only affects the 1D function output of type pmc_data.</p> <p>Options are:</p> <ul style="list-style-type: none"> pmc_data: Datasets are all labeled with a pmc_data tag. quantity: Datasets are labeled with the respective quantity name. 	pmc_data	none
type	Character	<p>Sets the extraction type. Options are:</p> <ul style="list-style-type: none"> interface pmc_data probe slice <p>Note: The pmc_data extraction type is available only for PMC simulations.</p>	none	none

1. Parameter specifies an extraction line (see [Specifying Extraction Lines, Planes, and Pairs on page 376](#)).
2. Parameter specifies an extraction plane (see [Specifying Extraction Lines, Planes, and Pairs on page 376](#)).
3. Parameter specifies an extraction pair (see [Specifying Extraction Lines, Planes, and Pairs on page 376](#)).

Chapter 6: Input Commands

define_extraction

Description

When using the `define_extraction` command:

- Each extraction defined with the `define_extraction` command is added to the group of extractions specified with the parameter `name`.
- To use a group of extractions defined with the `define_extraction` command in the `deposit` or `etch` command, the name of the extraction group is specified with the `extraction` parameter, and the time interval at which the extractions are performed is specified with the `extraction_interval` parameter. See [deposit on page 322](#) and [etch on page 339](#).
- Except for those with `type=pmc_data`, extractions defined with the `define_extraction` command work in a similar way to corresponding extractions defined with the `extract` command.
- Each extraction defined with the `define_extraction` command can write its output to a Tcl variable, whose name is provided by the parameter `tcl_output_variable`.
- Except when `type=pmc_data`, the Tcl variable into which the extraction results are written is a list of values with the following format:

```
t_0 extraction_result_0 t_1 extraction_result_1 ...
```

where:

- `t_i` is the i -th extraction time in minutes.
- `extraction_result_i` is the result of an intermediate extraction.

When `quantities` is set to `deposited_volume`, `reaction_executions`, and `sputtered_volume`, it is a list of `{reaction_name extracted_value}` pairs.

In other cases, `extraction_result_i` is the result that the corresponding `extract` command would produce if run on the structure at time `t_i`.

- When `type=pmc_data`, the Tcl variable into which the extraction results are written is a list of values with the following format:

```
headers data
```

where:

- `headers` is a list of extracted value names, including the extraction time, which is the first element (`time`).
- `data` is a list of extracted values, including the extraction time, given in the order specified by `headers`.
- When specifying `type=probe` `property=length`, the extraction output can also be written to a TDR file such that it can be visualized in Sentaurus Visual.

Chapter 6: Input Commands

```
define_extraction
```

Note:

When using `model=pmc` in the `etch` command, if both the `extraction_interval` and `plot_interval` parameters are specified, their values must be the same.

Examples

```
define_extraction name=extr axis=z point={0.2 0.2 0} type=probe \
    property=length materials=Silicon file=output.tdr

define_extraction name=extr axis=z point={0.2 0.2 1} type=interface \
    tcl_output_variable=res

etch time=1 spacing=0.1 extraction_interval=0.1 extraction=extr

puts "res = $res"
```

In the first command, an extraction of type `probe` is added to the extraction group named `extr`. This will extract the vertical thickness of the material `Silicon` at the specified point and write the value to the specified TDR file.

In the second command, an extraction of type `interface` is added to the extraction group named `extr`. This will extract the position of an interface, and the result is assigned to the variable `res`.

The extraction group named `extr` is used every 0.1 minutes.

In the fourth line, the Tcl command `puts` is used to output the result of the interface extraction.

```
define_extraction name=extr type=pmc_data \
    quantities=deposited_volume \
    expression_pattern={R1 R2} tcl_output_variable=depo_vol_res \
    csv_file=depo_vol.csv

define_extraction name=extr type=pmc_data \
    quantities=reaction_executions \
    expression_pattern="*-*B<s>*" \
    tcl_output_variable=B_reactant_res file=B_reactant.tdr

etch method=pmc spacing=0.05 time=1 extraction_interval=0.2 \
    extraction=extr
```

In the first command, an extraction of quantity `deposited_volume` is added to the extraction group named `extr`. This extracts the volume deposited according to reactions `R1` and `R2`. The result will be assigned to the `depo_vol_res` variable, and it will be written to the CSV file `depo_vol.csv` and to the TDR file `<base_name>.tdr`.

In the second command, an extraction of quantity `reaction_executions` is added to the extraction group named `extr`. This extracts the number of executions of the reactions that

Chapter 6: Input Commands

define_extraction

have an expression matching the pattern `*=*B<s>*`, that is, those having `B<s>` as a reaction product. There must exist at least one reaction with a `B<s>` product in the model used by the `etch` command; otherwise, an error is issued. The result will be assigned to the variable `B_reactant_res` as well as written to the TDR file `B_reactant.tdr`.

The extraction group named `extr` is used every 0.2 minutes.

Extractions for Plasma Simulations

This section describes the `define_extraction` command for plasma simulations.

For plasma simulations, this command allows you to extract and monitor several plasma quantities, such as bulk densities, electron temperature, plasma reaction statistics, convergence residuals, electron energy distribution, and sheath characteristics (potential waveform, sheath currents, sheath capacitance, flux, electric field, and so on).

Syntax

```
define_extraction name=<c> type=plasma \
(bulk_model_type=<c> | sheath_model_type=<c>) \
[csv_file=<c>] \
[expression_pattern=<c> | reactions=<l>] \
[extraction_step=<n> | time_step=<n>] \
[file=<c>] [file_update_step=<n>] \
[log_file_update=<b>] [max_num_samples=<n>] [mode=<c>] \
[output_type=<l>] [pattern_type=<c>] \
[quantities=<l>] [regex_syntax=<c>] \
[species=<l> | species_pattern=<c>]
```

Table 34 Parameters of `define_extraction` command for `type=plasma`

Parameter	Type	Description	Default [Range]	Unit
<code>bulk_model_type</code>	Character	Sets the type of the bulk model. Depending on the bulk model type, different extraction quantities are available (see Table 35). The only option is <code>global</code> .	none	none
<code>csv_file</code>	Character	Sets the name of the CSV file. You can specify this parameter only for CSV-exportable quantities (see Table 35) and if <code>output_type=csv</code> .	<basename>_<solution_name>_<extraction_name>_<quantity>.csv	none

Chapter 6: Input Commands

define_extraction

Table 34 Parameters of define_extraction command for type=plasma (Continued)

Parameter	Type	Description	Default [Range]	Unit
expression_pattern	Character	<p>Sets the search pattern that specifies which reaction equations are considered for extraction. By default, the search pattern must be given in the glob syntax.</p> <p>To instead enter a regular expression, set pattern_type=regex. You can provide either a pattern or a list of reaction names (parameter reactions).</p> <p>You can specify this parameter only for reaction-dependent quantities (see Table 35).</p>	none	none
extraction_step	Number	<p>Sets the step interval at which plasma quantities are extracted.</p> <p>You can specify this parameter only for iteration-dependent quantities (see Table 35).</p> <p>You can specify either time_step or extraction_step.</p>	1	none
file	Character	<p>Sets the name of the TDR file.</p> <p>You can specify this parameter only for TDR-exportable quantities (see Table 35) and if output_type=tdr.</p>	<basename>.tdr	none
file_update_step	Number	<p>Sets the interval at which output files are written during the simulation. The interval is given in units of the specified time_step or extraction_step, respectively. For example, if file_update_step=1, then files are written at every extraction step.</p> <p>If not specified, then extraction files are written only at the end of the simulation.</p> <p>You can specify this parameter only for iteration-dependent or time-dependent quantities (see Table 35).</p>	none	none

Chapter 6: Input Commands

define_extraction

Table 34 Parameters of define_extraction command for type=plasma (Continued)

Parameter	Type	Description	Default [Range]	Unit
log_file_update	Boolean	Specifies whether to write a short notification in the log output whenever a dataset of the TDR file is updated.	true	none
max_num_samples	Number	Sets the maximum number of points used for the extraction curves.	10000 [2, ∞)	none
mode	Character	<p>Controls what happens when the number of extracted time points exceeds the maximum number of samples (specified by max_num_samples). Options are:</p> <ul style="list-style-type: none"> mode=all: Keep the full time range, but resample the datasets to reduce the number of samples. mode=loop: Keep only the latest time points up to the number specified by max_num_samples and remove all earlier time points. <p>You can specify this parameter only for time-dependent quantities (see Table 35).</p>	all	none
name	Character	Sets the name of the extraction. This name is used to reference the extraction in the solve_reactor command (see solve_reactor on page 487).	none	none
output_type	List	Defines the way the extraction data is written. Options are: <ul style="list-style-type: none"> csv: Datasets are written in a tabular format to a CSV file specified by csv_file. tdr: Datasets are written to a TDR file specified by tdr_file. 	tdr if the quantity is exportable to a TDR file (see Table 35), else: csv	none
pattern_type	Character	When using species_pattern or expression_pattern to select the plasma species or reactions, respectively, this parameter sets the type of the pattern. Options are glob and regex.	glob	none

Chapter 6: Input Commands

define_extraction

Table 34 Parameters of define_extraction command for type=plasma (Continued)

Parameter	Type	Description	Default [Range]	Unit
quantities	List	Specifies the quantities to be extracted. Table 35 lists the available quantities. If not specified, then all available quantities are extracted.	none	none
reactions	List	Specifies the names of the reactions to extract. If not specified, then all reactions are extracted. You can provide either a list of reaction names or a pattern for the reaction equation (parameter expression_pattern). You can specify this parameter only for reaction-dependent quantities (see Table 35).	none	none
regex_syntax	Character	When using expression_pattern with pattern_type=regex to select the plasma reactions, this parameter defines the syntax of the regular expression. Options are: <ul style="list-style-type: none"> • extended • awk • basic • ecma_script • egrep • grep 	extended	none
sheath_model_type	Character	Sets the type of the sheath model. Depending on the sheath model type, different extraction quantities are available. Options are analytic and circuit.	none	none
species	List	Specifies the names of the species to be considered for extraction. If not specified, then all species are extracted. You can specify this parameter only for species-dependent quantities (see Table 35).	none	none

Chapter 6: Input Commands

define_extraction

Table 34 Parameters of define_extraction command for type=plasma (Continued)

Parameter	Type	Description	Default [Range]	Unit
species_pattern	Character	Sets the search pattern that specifies which species names are considered for extraction. By default, the search pattern must be given in the glob syntax. To instead enter a regular expression, set pattern_type=regex. You can provide either a pattern or a list of species names (parameter species). You can specify this parameter only for species-dependent quantities (see Table 35).	none	none
time_step	Number	Sets the time interval at which plasma quantities are extracted. You can specify this parameter only for time-dependent quantities (see Table 35). You can specify either time_step or extraction_step.	none	s
type	Character	Sets the extraction type. The applicable option is plasma.	none	none

Description

For plasma simulations, you can extract different dataset types, for example, time-dependent, iteration-dependent, and energy-dependent curves. [Table 35](#) lists all available extraction quantities with their extraction properties:

- **Time-dependent** quantities can be written at regular time intervals if file_update_step is specified. In addition, you can define a time_step and the extraction mode.
- **Iteration-dependent** quantities can be written at regular iteration intervals if file_update_step is specified. In addition, you can define an extraction_step and the extraction mode.
- Quantities **exportable to TDR files** can be written as a TDR file when output_type=tdr.

Chapter 6: Input Commands

define_extraction

- Quantities **exportable to CSV files** can be written as a CSV file when `output_type=csv`.
- For **species-dependent** quantities, you can specify species name filters (see parameters `species` and `species_pattern`).
- For **reaction-dependent** quantities, you can specify the reaction names (parameter `reactions`) or a reaction expression pattern (parameter `expression_pattern`).
- When specifying patterns, you can further choose the `pattern_type` and, if applicable, the `regex_syntax`.

You can define more than one extraction, but each extraction must have a unique name. Finally, you must register a list of the extraction names in the `solve_reactor` command (see [solve_reactor on page 487](#)).

Table 35 Extraction quantities for `define_extraction type=plasma` command

Model	Quantity	Description	Time dependent	Iteration dependent	Exportable to TDR	Exportable to CSV	Species dependent	Reaction dependent
bulk (global)	electron_energy_distribution	Final energy distribution of the electrons in the bulk			x	x		
bulk (global)	reactions	Number of bulk reaction executions as a function of time	x	x	x	x		x
bulk (global)	residuals	Individual and total convergence residuals as a function of time	x	x	x	x	x	
bulk (global)	species_balance	Time-integrated rates of consumption or production by reactions for each species	x ¹	x		x	x	x
bulk (global)	state	Species densities and electron temperature as a function of time	x	x	x	x	x	
bulk (global)	waveforms	Final density and electron temperature waveforms (only available for pulsed-power models)			x	x	x	
sheath (circuit)	capacitive_displacement_current	Displacement current waveform through the capacitive sheath in the equivalent circuit model		x	x	x		

Chapter 6: Input Commands

define_extraction

Table 35 Extraction quantities for define_extraction type=plasma command (Continued)

Model	Quantity	Description	Time dependent	Iteration dependent	Exportable to TDR	Exportable to CSV	Species dependent	Reaction dependent
sheath (circuit)	electron_current	Electron current waveform through the sheath in the equivalent circuit model (corresponds to the I–V characteristics of a diode)		x	x	x		
sheath (circuit)	energy_distribution	Final energy distributions of the ions and neutrals after passing the sheath layer			x	x	x	
sheath (circuit)	flux	Flux waveform of the particles impinging the electrode			x	x	x	
sheath (circuit)	rf_bias_current	Current waveform applied to the electrode at the RF bias supply		x	x	x		
sheath (circuit)	rf_bias_current_iterations	Amplitudes per frequency of the bias current waveform applied to the electrode at the RF bias supply as a function of the iteration		x	x	x		
sheath (circuit)	rf_bias_power	RF bias power waveform absorbed into the sheath		x	x	x		
sheath (circuit)	rf_bias_power_iterations	The (time-averaged) RF bias power values per frequency as a function of the iteration		x	x	x		
sheath (circuit)	rf_bias_power_residuals	Convergence residual of the RF bias power toward the user-defined target value for each iteration		x	x	x		
sheath (circuit)	rf_bias_voltage	Electric potential waveform between the plasma bulk-sheath-edge and the electrode		x	x	x		

Chapter 6: Input Commands

define_extraction

Table 35 Extraction quantities for define_extraction type=plasma command (Continued)

Model	Quantity	Description	Time dependent	Iteration dependent	Exportable to TDR	Exportable to CSV	Species dependent	Reaction dependent
sheath (circuit)	rf_bias_voltage_iterations	Amplitudes per frequency of the wall potential waveform applied to the electrode at the RF bias supply as a function of the iteration		x	x	x		
sheath (circuit)	sheath_capacitance	Capacitance waveform of the capacitive sheath in the equivalent circuit model		x	x	x		
sheath (circuit)	sheath_width	The distance waveform between the plasma bulk-sheath-edge and the electrode		x	x	x		
sheath (circuit)	total_ion_current	Total ion current waveform through the sheath in the equivalent circuit model		x	x	x		
sheath (circuit)	wall_electric_field_z	Waveform of the vertical component of the electric field at the position of the electrode		x	x	x		
sheath (circuit)	wall_potential	Electric potential waveform between the plasma bulk-sheath-edge and the electrode		x	x	x		
sheath (circuit)	wall_potential_residuals	Convergence residual of the sheath potential drop waveform for each iteration		x	x	x		

- Even if the extracted data is not a function of time, the extraction itself collects time-dependent data. The final extraction data consists of the time-integrated values.

Chapter 6: Input Commands

define_iad

define_iad

This command defines a new ion angular distribution (IAD) that can be used in a flux model. IADs can be defined by either using a tabular format, reading them from a TDR file, or specifying the exponent of an analytic distribution ([Equation 6 on page 49](#)).

Syntax

For built-in models:

```
define_iad exponent=<n> name=<c>  
define_iad file=<c> name=<c> [tdr_geometry=<c>]  
define_iad name=<c> table=<v> [angle_unit=<c>]
```

For RFM models:

```
define_iad exponent=<n> name=<c> species=<c> [energy=<n>]  
define_iad file=<c> name=<c> species=<c> [energy=<n>] [tdr_geometry=<c>]  
define_iad name=<c> species=<c> table=<v> [angle_unit=<c>] [energy=<n>]
```

Then, the defined IAD can be referred to under the name set by the `name` parameter.

Table 36 *Parameters of define_iad command*

Parameter	Type	Description	Default [Range]	Unit
angle_unit	Character	Sets the unit of the angles listed in the <code>table</code> parameter. Options are: <ul style="list-style-type: none">• deg• rad	deg	none
energy	Number	Sets the ion energy for which the angular distribution is defined. Note: When <code>energy=0</code> , the defined distribution is energy independent.	0 [0 , ∞[eV
exponent	Number	Sets the exponent used to characterize the ion angular distribution $\cos^m\theta$.	none [1 , ∞[none
file	Character	Sets the name of the TDR file containing an IAD.	none	none

Chapter 6: Input Commands

define_iad

Table 36 Parameters of *define_iad* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
name	Character	Sets the name of the IAD. This parameter can be used to reference the IAD in either a <code>define_deposit_machine</code> or a <code>define_etch_machine</code> command.	none	none
species	Character	Sets the name of the flux species for which the angular distribution is defined.	none	none
table	Vector	Sets the tabular format of the IAD (see Tabular Format on page 238 and Examples on page 239).	none	none
tdr_geometry	Character	Sets the name of the geometry to load from a TDR file specified by <code>file</code> . If you omit <code>tdr_geometry</code> , then the geometry to read is specified by the <code>name</code> parameter.	none	none

Tabular Format

The format of a table for an IAD is as follows:

- Each line consists of a pair of numbers: The first number represents the angle and the second number represents the value of the flux distribution for this angle.
- The lines are sorted in increasing order by angle.
- The first entry is for angle 0°.
- The last entry is for angle 90° and the flux distribution value is zero.
- Angles must be unique.
- All flux distribution values must be nonnegative and finite.

Note:

A violation of any of these conditions will cause an error.

When using spherical coordinates, the ion flux on a flat unshadowed surface is defined as:

$$\Gamma = \int_0^{2\pi} d\phi \int_0^{\pi/2} \sin\theta \, d\theta \, f(\theta)$$

where $f(\theta)$ is the IAD. Therefore, when using a tabular IAD, the provided data must not contain the factor $\sin\theta$.

Chapter 6: Input Commands

define_iad

The definition of ion distributions is not part of any model. In this way, it is easy to reuse data that has been measured or simulated under certain conditions. In fact, data about many ion fluxes can be stored in a file, which is sourced into the command file when needed.

For example, the following command sets the distribution of the ion flux I according to [Equation 6 on page 49](#) with $m = 100$:

```
define_iad name=my_iad species=I exponent=100
```

The parameter `species` states the name of the flux to which the ion distribution refers.

Sentaurus Topography 3D supports collections of ion distributions, so that data can be reused easily. For example, a collection of ion distributions named `my_iad` is defined with the commands:

```
define_iad name=my_iad species=I1 table=$table1
define_iad name=my_iad species=I2 exponent=1000
define_iad name=my_iad species=I3 table=$table3
```

In this way, the entire collection can be referred to using one name.

Note:

There is no support for energy-dependent flux integration. Therefore, only energy-independent IADs can be used.

Examples

This command loads an IAD with the TDR geometry name `iad_1` from the file `iad.tdr`:

```
define_iad name=iad_1 file=iad.tdr
```

The following commands first define a Tcl variable called `table` that contains a very simple IAD consisting of eleven angles and the corresponding flux distribution values:

```
# Define an IAD table using a Tcl list. First column represents angles
# in degree and second column represents absolute flux values.
set table {
    0.0    1.0
    2.0    0.95
    4.0    0.8
    6.0    0.6
    8.0    0.4
   10.0    0.2
   15.0    0.1
   20.0    0.05
   30.0    0.02
   40.0    0.0
   90.0    0.0
}
```

Chapter 6: Input Commands

define_layout

```
# Use the Tcl variable in the definition of the IAD.  
define_iad name=my_iad table=$table
```

Note:

The user-defined IADs are normalized internally so that the direct flux on a flat unshadowed surface of the species they refer to equals one.

define_layout

This command reads a GDSII, an OASIS, or a TCAD layout file, which can then be used to create logical masks with the [define_mask](#) command (see [define_mask on page 243](#)).

Syntax

For GDSII and OASIS layout files:

```
define_layout cell=<c> domain_max=<v> domain_min=<v> \  
layer_names=<l> layer_numbers=<l> layout_file=<c> name=<c> \  
[domain=<c>] [scale=<n>] [shift_to_origin=<b>] \  
[tcad_layout_output_file=<c>]
```

For TCAD layout files:

```
define_layout name=<c> tcad_layout_file=<c> \  
[scale=<n>] [shift_to_origin=<b>] [verbose=<b>]
```

Table 37 Parameters of *define_layout* command

Parameter	Type	Description	Default [Range]	Unit
cell	Character	Sets the name of the layout cell from which a domain is extracted.	none	none
domain	Character	Sets the name given to the domain that is extracted from a GDSII or an OASIS layout file.	SIM3D	none
domain_max	Vector	Sets the 2D maximum point of the layout domain.	none	µm
domain_min	Vector	Sets the 2D minimum point of the layout domain.	none	µm

Chapter 6: Input Commands

define_layout

Table 37 Parameters of define_layout command (Continued)

Parameter	Type	Description	Default [Range]	Unit
layer_names	List	Sets a list of layer names that will be used instead of the layer numbers specified in layer_numbers. The number of entries in both lists must be the same.	none	none
layer_numbers	List	Sets a list of layer numbers that are read from a GDSII or an OASIS layout file.	none	none
layout_file	Character	Sets the name of a GDSII or an OASIS file from which the layout is read.	none	none
name	Character	Sets the name of the layout.	default_layout	none
scale	Number	Sets the scaling factor to apply to all layout coordinates.	1.0]0, ∞[none
shift_to_origin	Boolean	Specifies whether to shift the layout, so that the lower-left corner of all domains in the layout shifts to the origin of the simulation coordinate system.	true	none
tcad_layout_file	Character	Sets the name of the TCAD layout file from which the layout is read.	none	none
tcad_layout_output_file	Character	If specified, the layout read from a GDSII or an OASIS file is written to a TCAD layout file with the specified name.	none	none
verbose	Boolean	Specifies whether to output the names of 3D domains and the names of layers in the layout.	false	none

Examples

Read a layout from a GDSII file:

```
define_layout cell=A10 domain_max={6.5 10.0} domain_min={0 0} \
    layer_names={contact} layer_numbers={15:0} \
    layout_file=cont.gds name=1
```

Chapter 6: Input Commands

define_litho_machine

define_litho_machine

This command defines a new machine for a lithography process. A lithography machine and its properties are used in a lithography simulation when the name of the machine is specified with the `machine` parameter in the `litho` command.

Note:

A machine defined with the `define_litho_machine` command can be used only to process a 3D structure.

You cannot use this command when you start Sentaurus Topography 3D with a value greater than 1 for the command-line option `--processes` (see [From the Command Line on page 17](#)).

You add properties to the machine by using the `add_litho_command` command (see [add_litho_command on page 159](#)).

Syntax

```
define_litho_machine [name=<c>]
```

Table 38 Parameters of *define_litho_machine* command

Parameter	Type	Description	Default	Unit
name	Character	Sets the name of the litho machine that is defined.	default_machine	none

Chapter 6: Input Commands

define_mask

define_mask

This command defines a new logical mask.

In contrast to physical masks, logical masks do not have a polarity. They are simply a set of polygons that can be combined by different operations to form new logical masks.

The binary operations and the unary operation `complement`, which are used to form a new mask, can be interpreted as set operations (complement, difference, intersection, symmetric difference, or union) on the polygons in the masks. Except for the operation `difference`, these operations can also be interpreted as simple Boolean operations (*not*, *and*, *xor*, or *or*) where areas covered by polygons have the value *true* and areas that are not covered by polygons have the value *false*.

Note:

To use a logical mask for an etch or a patterning step, you must specify the reticle and resist type with which the logical mask is being used.

Syntax

```
define_mask domain=<c> layer=<c> layout=<c> [name=<c>] [shift=<v>]  
define_mask mask1=<c> mask2=<c> operation=<c> [name=<c>]  
define_mask mask1=<c> operation=complement [name=<c>]  
define_mask mask1=<c> operation=shift shift=<v> [name=<c>]  
define_mask polygon=<v> [domain_max=<v>] [domain_min=<v>] [name=<c>]
```

Table 39 Parameters of *define_mask* command

Parameter	Type	Description	Default	Unit
domain	Character	Sets the name of the domain for which a mask is created.	none	none
domain_max	Vector	Sets the maximum corner point of the mask domain. Sets the upper-left point of a rectangular 2D domain. This domain limits the extent of the inverted mask. It is relevant only if the inverse of the polygonal mask domain is required during an operation involving this mask.	{1e3 1e3}	µm

Chapter 6: Input Commands

define_mask

Table 39 Parameters of define_mask command (Continued)

Parameter	Type	Description	Default	Unit
domain_min	Vector	Sets the minimum corner point of the mask domain. Sets the lower-left point of a rectangular 2D domain. This domain limits the extent of the inverted mask. It is relevant only if the inverse of the polygonal mask domain is required during an operation involving this mask.	{-1e3 -1e3}	µm
layer	Character	Sets the name of the layer for which a mask is created.	none	none
layout	Character	Sets the name of the layout.	default_layout	none
mask1	Character	Sets the name of the input mask for a unary or binary mask operation.	none	none
mask2	Character	Sets the name of the input mask for a binary mask operation.	none	none
name	Character	Sets the name of the new logical mask.	default_mask	none
operation	Character	Sets the operation used to create a new mask from one or two existing masks. Options are: <ul style="list-style-type: none">• complement• difference• intersection• shift• symmetric_difference• union	none	none

Chapter 6: Input Commands

define_mask

Table 39 Parameters of define_mask command (Continued)

Parameter	Type	Description	Default	Unit
polygon	Vector	<p>Sets the vertices of the polygon defining the mask.</p> <p>The specified vector must be non-empty and contain an even number of elements N.</p> <p>The vector elements are interpreted as the sequence of (x, y) coordinates of the polygon vertices, listed in either a clockwise order or a counterclockwise order. Therefore, the $i \cdot N/2$-th element contains the x-coordinate of the i-th polygon vertex, and the $(i \cdot N/2 + 1)$-th element contains the y-coordinate of the i-th polygon vertex.</p>	none	μm
shift	Vector	Sets the amount by which the output mask is shifted in relation to the input mask. This parameter is required when operation=shift.	{ 0 0 }	μm

Limitations

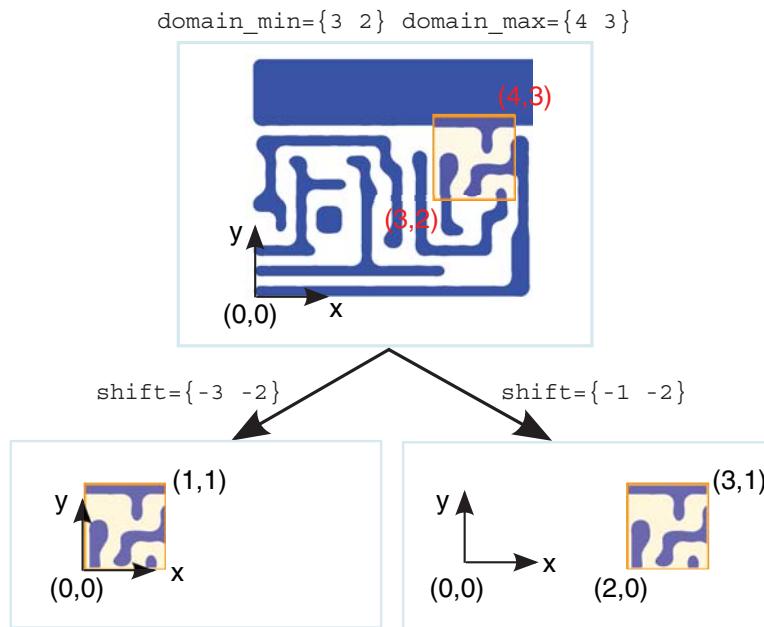
The following limitations apply:

- When performing an operation on two input masks to create a new mask, the domains of the input masks must match.
- Since a domain contains only a small part of the layout, shifting the layout might result in the mask only partially covering the simulation domain, causing unexpected results. Therefore, when analyzing the effects of misaligned layers, it is recommended to define a layout domain that is larger than the simulation domain to ensure that the simulation domain is always fully covered.

Chapter 6: Input Commands

define_mask

Figure 23 Illustration of the relationship of the domain_min, domain_max, and shift parameters



Examples

Define a square mask:

```
define_mask name=m polygon={0 0 1 0 1 1 0 1}
```

Chapter 6: Input Commands

define_material_replacement

define_material_replacement

When running PMC simulations, this command defines a material replacement mapping of a compound material such as SiGe to the constituent PMC species, for example, silicon and germanium.

Syntax

```
define_material_replacement material=<c> name=<c> species=<l> \
    volume_fractions=<l>
```

Mix constant and field-based mole fractions:

```
define_material_replacement constant_species={ cs1 cs2 ... } \
    field_names={ fn1 fn2 ... } field_species={ fs1 fs2 fs3 fs4 ... } \
    volume_fractions={ v1 v2 ... }
```

Table 40 Parameters of define_material_replacement command

Parameter	Type	Description	Default	Unit
constant_species	Character	Specifies a list of species that are constant in the compound material (material). This parameter is a synonym for the species parameter. It is clearer to specify constant_species and field_species, instead of species and field_species, but both pairs will work. The constant_species list corresponds to the volume_fractions list and must be the same size.	none	none
field_names	List	Specifies a list of fields in the TDR file. Each field sets the mole fraction of two species. The number of field_species should be twice the number of field_names. The values read from the TDR file are restricted to the range 0.0 to 1.0, inclusively. See the Description for details.	none (no limit)	μm
field_species	List	Specifies a list of PMC species corresponding to mole fractions as defined by the TDR fields named in the field_names list. See Description for details.	none (no limit)	μm
material	Character	Sets the compound material.	none	none

Chapter 6: Input Commands

define_material_replacement

Table 40 Parameters of define_material_replacement command (Continued)

Parameter	Type	Description	Default	Unit
name	Character	Sets the name of the material replacement map.	none	none
species	List	Specifies a list of species corresponding to the compound material defined in the material parameter.	none	none
volume_fractions	List	Specifies a list of fractions corresponding to the species defined in the species parameter. The sum of all fractions must be 1.0.	none	none

Description

The mapping defined by this command can be used in the filter_structure command or the etch command to replace or to *split* the compound material species with its constituents. After the material is split into separate species, you can use different reactions or reaction rates to obtain mole fraction-dependent PMC etching or deposition. Later, the same mapping can be used to restore or to *merge* the original compound material from PMC cells containing the constituent species. The merge operation can be specified with the filter_structure command or can be performed on the fly when saving a boundary file by using the save command.

You can mix constant and field-based mole fractions in the same command specification. The computation of the mole fractions is as follows. Let f_n be the n^{th} field and c_n be the n^{th} constant species given by the volume_fractions parameter. Then, the total of the constant fractions is $\sum_n c_n$.

Now, let R be the fraction available for field-based species, $R = 1 - \sum_n c_n$, and let n_f be the number of field_names.

For each pair p of field_species, f_{2p}, f_{2p+1} , where $0 \leq p < n_f$:

$$f_{2p} = \frac{R}{n_f} \times \text{field_name}[p] \quad (79)$$

$$f_{2p+1} = \frac{R}{n_f} \times (1 - \text{field_name}[p])$$

Chapter 6: Input Commands

define_material_replacement

Examples

Convert default_structure from a boundary representation (brep) structure to a PMC structure, and set regions of material SiGe to 40% silicon and 60% germanium:

```
define_material_replacement name=r1 material=SiGe \
    species={Silicon Germanium} volume_fractions={0.4 0.6}
    filter_structure type=convert_to_pmc split_replacements={r1}
```

The sum of the fractions specified in volume_fractions must be 1.0. If not, then fractions are normalized by 1.0/sum-of-all-fractions. If a fraction is modified by more than 1%, then the modified fractions are printed to screen.

Convert default_structure from a brep structure to a PMC structure, and set regions of material AlGaAs to 10% Al, 40% Ga, and 50% As, and proceed to etch the structure with PMC:

```
define_material_replacement name=r2 material=AlGaAs \
    species={ Aluminum Gallium Arsenic } \
    volume_fractions={ 0.1 0.4 0.5 }
    etch split_replacements={ r2 } method=pmc spacing=<n> time=<n>
```

Save structure from previous step, replacing cells of 10±5% Al, 40±5% Ga, and 50±5% As with material AlGaAs:

```
save file=merged.tdr merge_replacements={ r2 } merge_tolerance=0.05
```

Similarly, this command can be used to create a structure merged in memory, which would be available for further processing:

```
filter_structure name=merged merge_replacements={r2}
merge_tolerance=0.05
```

Mix constant and field-based mole fractions:

```
define_material_replacement name=sige_mf material=SiliconGermanium \
    field_names={XMoleFraction} field_species={Silicon Germanium}
```

where:

- Total constant species fraction = 0.0
- $R = 1$ and $n_f = 1$
- Germanium = XMoleFraction
- Silicon = $(1 - \text{XMoleFraction})$

```
define_material_replacement name=ingaaS_mf material=InGaAs \
    volume_fractions={0.5} constant_species={Arsenic} \
    field_names={XMoleFraction} field_species={Indium Gallium}
```

Chapter 6: Input Commands

define_material_replacement

where:

- Total constant species fraction = 0.5
- $R = 0.5$ and $n_f = 1$
- Arsenic = 0.5
- Indium = $0.5 \times (\text{XMoleFraction})$
- Gallium = $0.5 \times (1 - \text{XMoleFraction})$

```
define_material_replacement name=ingaasp_mf material=InGaAsP \
    field_names={ XMoleFraction YMoleFraction } \
    field_species={ Indium Gallium Arsenic Phosphorus }
```

where:

- Total constant species fraction = 0
- $R = 1$ and $n_f = 2$
- Indium = $1/2 \times (\text{XMoleFraction})$
- Gallium = $1/2 \times (1 - \text{XMoleFraction})$
- Arsenic = $1/2 \times (\text{YMoleFraction})$
- Phosphorus = $1/2 \times (1 - \text{YMoleFraction})$

Chapter 6: Input Commands

define_model

define_model

This command starts the definition of a new RFM model or a reaction model.

Syntax

```
define_model description=<c> name=<c> [type=<c>]
```

Table 41 Parameters of *define_model* command

Parameter	Type	Description	Default	Unit
description	Character	Describes the model.	none	none
name	Character	<p>Sets the unique name of the model. This name is used to reference the model in all commands for configuring a model and in the <code>define_deposit_machine</code> and <code>define_etch_machine</code> commands, for using the model in a machine.</p> <p>Note: The name must not contain slashes (/).</p>	none	none
type	Character	<p>Sets the model type. Options are:</p> <ul style="list-style-type: none">• deposit: Deposition• etch: Etching• etchdepo: Simultaneous etching and deposition <p>If you set this parameter, the new model is assumed to be specified using rate formulas (see add_flux_properties on page 146, add_formula on page 149, add_int_parameter on page 151, add_ion_flux on page 157, and add_neutral_flux on page 163).</p> <p>If you omit this parameter, the new model is assumed to be a reaction model (see add_reaction on page 164 and add_source_species on page 181).</p>	none	none

Chapter 6: Input Commands

define_nad

define_nad

This command defines a new neutral angular distribution (NAD) that can be used in a flux model. NADs can be defined by either using a tabular format, reading them from a TDR file, or specifying the exponent of an analytic distribution (see [Equation 6 on page 49](#)). Then, the defined NAD can be referred to under the name set by the `name` parameter.

Syntax

```
define_nad exponent=<n> name=<c> [energy=<n>] [species=<c>]  
define_nad file=<c> name=<c> [energy=<n>] [species=<c>] \  
[tdr_geometry=<c>]  
define_nad name=<c> table=<v> [angle_unit=<c>] [energy=<n>] \  
[species=<c>]
```

Table 42 Parameters of `define_nad` command

Parameter	Type	Description	Default [Range]	Unit
angle_unit	Character	Sets the unit of the angles listed in the <code>table</code> parameter. Options are: <ul style="list-style-type: none">• deg• rad	deg	none
energy	Number	Sets the neutral energy for which the angular distribution is defined. Note: When <code>energy=0</code> , the defined distribution is energy independent.	0 [0, ∞[eV
exponent	Number	Sets the exponent used to characterize the ion angular distribution $\cos^m\theta$.	none [1, ∞[none
file	Character	Sets the name of the TDR file containing a NAD.	none	none
name	Character	Sets the name of the NAD, which can be used to reference the NAD in either a <code>define_deposit_machine</code> or a <code>define_etch_machine</code> command.	none	none
species	Character	Sets the name of the flux species for which the angular distribution is defined.	neutral	none

Chapter 6: Input Commands

define_nad

Table 42 Parameters of define_nad command (Continued)

Parameter	Type	Description	Default [Range]	Unit
table	Vector	Sets the tabular format of the NAD (see Tabular Format on page 238 ; while the content in this section refers to IADs, it also applies to NADs).	none	none
tdr_geometry	Character	Sets the name of the geometry to load from the TDR file specified by <code>file</code> . If you omit <code>tdr_geometry</code> , then the geometry to read is specified by the <code>name</code> parameter.	none	none

Description

The definition of neutral distributions is not part of any model. In this way, it is easy to reuse data that has been measured or simulated under certain conditions. In fact, data about many neutral fluxes can be stored in a file, which is sourced into the command file when needed.

For example, the following command sets the distribution of the neutral flux `N` according to [Equation 6 on page 49](#) with `m=1.5`:

```
define_nad name=my_nad species=N exponent=1.5
```

The parameter `species` states the name of the flux to which the neutral distribution refers.

Sentaurus Topography 3D supports collections of neutral distributions, so that data can be reused easily. For example, a collection of neutral distributions named `my_nad` is defined with the commands:

```
define_nad name=my_iad species=N1 table=$table1
define_nad name=my_iad species=N2 exponent=1
define_nad name=my_iad species=N3 table=$table3
```

In this way, the entire collection can be referred to using one name.

Note:

A collection of neutral distributions to be used in a simultaneous etching and deposition machine having `model=etchdepo2` must contain:

- The distribution of the etching neutral flux ($\Gamma_{\text{neutral etch}}$) in [Equation 40 on page 78](#) set with `species=neutral_etch`
- The distribution of the deposition neutral flux ($\Gamma_{\text{neutral depo}}$ in [Equation 40 on page 78](#)) set with `species=neutral`

There is no support for energy-dependent flux integration. Therefore, only energy-independent NADs can be used.

Chapter 6: Input Commands

define_pattern_density

User-defined NADs are normalized internally, so that the direct flux on a flat unshadowed surface of the species they refer to equals one.

define_pattern_density

This command defines a new pattern density object or extends the definition of an existing pattern density object. Each `define_pattern_density` command adds an array of density values that correspond to the cells of a rectangular grid. The position of this grid is specified in coordinates of the simulation domain.

If a pattern density object with the same name already exists, then the new pattern density is added to the already defined pattern density. This behavior allows for the creation of complex pattern density maps by using multiple `define_pattern_density` commands. See [define_pattern_density_model on page 256](#).

Syntax

```
define_pattern_density name=<c> density=<l> point_max=<v> \
    point_min=<v> [n_x=<n>] [n_y=<n>] [type=<c>]
```

Table 43 Parameters of `define_pattern_density` command

Parameter	Type	Description	Default [Range]	Unit
density	List	Sets the local pattern densities of the $n_x \times n_y$ grid cells. The value is a list of elements sorted in row-major order from upper-left to bottom-right: $\{ v_{1,n_y} v_{2,n_y} v_{3,n_y} \dots v_{n_x,n_y} \dots v_{12} v_{22} v_{32} \dots v_{n_x,2} v_{11} v_{21} v_{31} \dots v_{n_x,1} \}$	none [0, 1]	none
n_x	Number	Sets the number of grid cells in the x-direction.	1 [1, 2147483647]	none
n_y	Number	Sets the number of grid cells in the y-direction.	1 [1, 2147483647]	none

Chapter 6: Input Commands

define_pattern_density

Table 43 Parameters of *define_pattern_density* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
name	Character	Sets the name of the local pattern density object. If a pattern density object with the same name exists already, then the new pattern density is added to the existing pattern density.	none	none
point_max	Vector	Sets the upper-right corner of the 2D cell grid in coordinates of the simulation domain.	none ($-\infty$, ∞)	μm
point_min	Vector	Sets the lower-left corner of the 2D cell grid in coordinates of the simulation domain.	none ($-\infty$, ∞)	μm
type	Character	Sets the input format of the pattern density. The only option is <code>grid</code> .	<code>grid</code>	none

Examples

```
define_pattern_density name=lpd \
    type      = grid \
    n_x       = 2 \
    n_y       = 2 \
    point_min = { 0<um> -1<um> } \
    point_max = { 1<um> 2<um> } \
    density   = { 0.9 0.6 0.8 0.7 }

define_pattern_density name=lpd \
    point_min = { 10 3 } \
    point_max = { 12 5 } \
    density   = 0.5
```

define_pattern_density_model

This command defines a new pattern density model that models the pattern density effect observed in semiconductor etching processes. The influence of the pattern density on the etching rate is modeled as:

$$R_{PD} = P(\rho(x, y), \alpha, \beta) \cdot R_{Local} \quad (80)$$

where R_{Local} represents the local etching rate without pattern density effects, and R_{PD} represents the corrected final etching rate. $P(\rho(x, y), \alpha, \beta)$ denotes the rate correction factor that depends on the effective pattern density $\rho(x, y)$ and on the calibration parameters α and β . Here, x and y are coordinates on the wafer surface.

The rate correction factor is given by the following equation, as proposed in [5][6][7][8]:

$$P(\rho(x, y), \alpha, \beta) = e^{-\alpha\rho(x, y)^{\beta}} \quad (81)$$

The effective pattern density $\rho(x, y)$ is constructed by convolution of the *local pattern density* $d(x, y)$, which is defined by the `define_pattern_density` command (see [define_pattern_density on page 254](#)):

$$\rho(x, y) = (d*f)(x, y) \quad (82)$$

Here, $f(x, y) = \frac{1}{2\pi w^2} \exp\left(-\frac{x^2 + y^2}{2w^2}\right)$ is a Gaussian kernel with range w .

Syntax

```
define_pattern_density_model name=<c> \
    alpha=<n> beta=<n> pattern_density=<c> range=<n> \
    [invert=<b>] [rate_correction_function=<c>] [response_function=<c>]
```

Table 44 Parameters of define_pattern_density_model command

Parameter	Type	Description	Default [Range]	Unit
alpha	Number	Sets the value of the primary calibration parameter that affects the correction to the pattern density-dependent etching rate. Negative values model a reverse pattern density effect. A value of zero deactivates pattern density-dependent effects.	none]-∞, ∞[none
beta	Number	Sets the value of the secondary calibration parameter that affects the correction to the pattern density-dependent etching rate.	none]0, ∞[none

Chapter 6: Input Commands

define_pattern_density_model

Table 44 Parameters of define_pattern_density_model command (Continued)

Parameter	Type	Description	Default [Range]	Unit
invert	Boolean	Specifies whether to invert the local pattern density function, that is, it uses $1 - d(x, y)$ instead of $d(x, y)$.	false	none
name	Character	Sets the name of the effective pattern density object.	none	none
pattern_density	Character	Sets the pattern density object (see define_pattern_density on page 254).	none	none
range	Number	Sets the range of influence of a local pattern density impulse.	none]0,∞[μm
rate_correction_function	Character	Sets the type of the function that corrects the rates as a function of the effective pattern density. The only option is exponential.	exponential	none
response_function	Character	Sets the type of the kernel function used in the convolution integral to compute the effective pattern density from the local pattern density. The only option is gaussian.	gaussian	none

Examples

```
define_pattern_density name=lpd \
    point_min = { 10 3 } \
    point_max = { 12 5 } \
    density    = 0.5

define_pattern_density_model name=pdm \
    pattern_density      = lpd \
    response_function    = gaussian \
    range                = 1<um> \
    rate_correction_function = exponential \
    alpha                = 1.1 \
    beta                 = 0.5 \
    invert               = true
```

Chapter 6: Input Commands

define_plasma_model

define_plasma_model

This command defines a new plasma model. A plasma model consists of a bulk model that describes the physics of the plasma bulk region, and a sheath model that describes the transport of the plasma particles from the plasma bulk region to the structure (see [Chapter 4 on page 92](#)).

In addition, you can define only a plasma bulk model or only a plasma sheath model.

Syntax

A plasma bulk model coupled to a plasma sheath model:

```
define_plasma_model bulk_model_type=<c> name=<c> sheath_model_type=<c>
```

A plasma bulk model only:

```
define_plasma_model bulk_model_type=<c> name=<c>
```

A plasma sheath model only:

```
define_plasma_model name=<c> sheath_model_type=<c>
```

Table 45 Parameters of *define_plasma_model* command

Parameter	Type	Description	Default	Unit
bulk_model_type	Character	Sets the bulk model to be used. The only option is global.	none	none
name	Character	Sets a unique name for the plasma model. It is used to reference the model in other plasma commands used to configure the model, such as define_reactor (see define_reactor on page 261).	none	none
sheath_model_type	Character	Sets the sheath model to be used. Options are: <ul style="list-style-type: none">• analytic• circuit	none	none

Examples

```
define_plasma_model name=M bulk_model_type=global
```

```
define_plasma_model name=M sheath_model_type=circuit
```

```
define_plasma_model name=M bulk_model_type=global \
    sheath_model_type=circuit
```

Chapter 6: Input Commands

define_probability

define_probability

This command defines a new energy- and angle-dependent probability (hereafter, referred to as the *probability function*).

Syntax

Specify the angle-dependent part of the probability function for a given energy level as a table:

```
define_probability energy=<n> name=<c> table=<v> [angle_unit<c>]
```

or according to the Mizuno model (see [Equation 12 on page 52](#)):

```
define_probability energy=<n> mizuno_k=<n> name=<c>
```

or with an analytic expression:

```
define_probability energy=<n> expression=<c> name=<c>
```

Specify an analytic energy-dependent probability function:

```
define_probability expression=<c> name=<c>
```

Table 46 Parameters of define_probability command

Parameter	Type	Description	Default [Range]	Unit
angle_unit	Character	Sets the unit of the angles listed in the table parameter. Options are: <ul style="list-style-type: none">• deg• rad	deg	none
energy	Number	Sets the energy for which the probability function is defined. Energy-independent probability functions can be specified by setting <code>energy=0</code> . In this case, only one specification for the given <code>name</code> is allowed.	none [0 , ∞[eV

Chapter 6: Input Commands

define_probability

Table 46 Parameters of define_probability command (Continued)

Parameter	Type	Description	Default [Range]	Unit
expression	Character	<p>Sets the formula used to calculate the probability function.</p> <p>When you specify expression, the probability function is defined from the user-specified expression, and the energy parameter is optional. For details about energy- and angle-dependent expressions, see Syntax for Expressions on page 131.</p> <p>There are the following cases:</p> <ul style="list-style-type: none"> • When energy is not specified, the specified expression defines an energy-dependent probability function that is valid for all energy levels. • When energy=0, the specified expression defines an energy-independent probability function. • When energy is set to a positive value, the specified expression defines the angle-dependent part of the probability function for the given energy level. 	none	none
mizuno_k	Number	Sets the species of the parameter k of Equation 12 .	none [0, 1]	none
name	Character	Sets the name used to refer to the energy- and angle-dependent probability in an add_reaction_properties command.	none	none
table	Vector	Sets the tabular format of the probability function. When using the parameter table, the same restrictions on the format of the table apply as described in define_iad on page 237 , except that the value specified for 90° does not have to be zero. In addition, all of the probability function values must be in the range [0, 1].	none	none

Chapter 6: Input Commands

define_reactor

define_reactor

This command defines the reactor parameters for a plasma model (see [Chapter 4 on page 92](#)).

Syntax

Define a new plasma reactor for a global bulk model (`bulk_model_type=global`):

```
define_reactor name=<c> plasma_model=<c> \
    gas_temperature=<n> height=<n> inlet_gas_flow=<l> \
    (power=<n> | \
     power_period=<n> power_waveform=<c> [num_waveform_samples=<n>]) \
    pressure=<n> radius=<n> type=<c> \
    ([density=<l>] [electron_temperature=<n>] | [solution=<c>] | \
     [solution_file=<c>]) \
    [min_density=<n>] [min_density_relaxation_time=<n>] \
    [min_electron_density=<n>] [min_electron_energy_density=<n>] \
    [outlet_gas_flow=<n>] [power_absorption_coefficient=<n>] \
    [pressure_relaxation_time=<n>]
```

Define a new plasma reactor for an analytic sheath model (`sheath_model_type=analytic`):

```
define_reactor name=<c> plasma_model=<c> \
    gas_temperature=<n> rf_bias_frequency=<n> type=<c> \
    (ac_voltage=<n> dc_voltage=<n> | rf_voltage=<n>) \
    ((density=<l> electron_temperature=<n>) | solution=<c> | \
     solution_file=<c>)
```

Define a new plasma reactor for a circuit sheath model (`sheath_model_type=circuit`):

```
define_reactor name=<c> plasma_model=<c> \
    gas_temperature=<n> height=<n> radius=<n> \
    (rf_bias_frequency=<l> \
     (rf_bias_current=<l> | rf_bias_voltage=<l> | \
      (rf_bias_power=<l> [bias_mode=<c>])) | \
     (bias_current_waveform=<c> bias_current_period=<n> \
      [num_waveform_samples=<n>]) | \
     (wall_potential_waveform=<c> wall_potential_period=<n> \
      [num_waveform_samples=<n>])) \
    ((density=<l> electron_temperature=<n>) | solution=<c> | \
     solution_file=<c>) \
    type=<c>
```

Chapter 6: Input Commands

`define_reactor`

If you couple a bulk model to a sheath model, then provide the union of the parameter sets of the individual models.

Table 47 Parameters of `define_reactor` command

Parameter	Type	Description	Default [Range]	Unit
<code>ac_voltage</code>	Number	Sets the AC sheath voltage of the reactor. Applies only to the analytic sheath model and if <code>rf_voltage</code> is not specified.	none] $0, \infty[$	V
<code>bias_current_period</code>	Number	If you specify <code>bias_current_waveform</code> , then you must also define the period of the bias current waveform by using this parameter.	none] $0, \infty[$	s
<code>bias_current_waveform</code>	Character	<p>Sets the time-dependent periodic RF current waveform applied to the electrode. This parameter describes the total electric current through the sheath.</p> <p>Note:</p> <p>When using waveform expressions, you must also specify the period of the waveform, using the parameter <code>bias_current_period</code>.</p> <p>You can specify either <code>rf_bias_current</code> or <code>rf_bias_power</code> or <code>bias_current_waveform</code> or <code>wall_potential_waveform</code>.</p> <p>This parameter applies only to the circuit sheath model.</p> <p>For information about the syntax for function expressions, see Syntax for Expressions on page 131.</p>	Functional expression, depending on time t and evaluating to] $0, \infty[$	A
<code>bias_mode</code>	Character	If you specify <code>rf_bias_power</code> , then you can also choose the driving mode of the sheath by using this parameter, that is, you can define how the power is fed into the system. Options are <code>current_driven</code> and <code>voltage_driven</code> .	<code>current_driven</code>	none
<code>dc_voltage</code>	Number	Sets the DC sheath voltage of the reactor. Applies only to the analytic sheath model and if <code>rf_voltage</code> is not specified.	none] $0, \infty[$	V

Chapter 6: Input Commands

define_reactor

Table 47 Parameters of define_reactor command (Continued)

Parameter	Type	Description	Default [Range]	Unit
density	List	Sets a list of initial densities for one or more plasma species. The list must be formatted as follows: {{species1 value1} {species2 value2} ...} You can specify either density and electron_temperature, or solution, or solution_file.	Computed automatically from pressure [0, +∞[m ⁻³
electron_temperature	Number	Sets the initial electron temperature. You can specify either density and electron_temperature, or solution, or solution_file.	2]0, ∞[eV
gas_temperature	Number	Sets the temperature of the neutral gas.	none]0, ∞[K
height	Number	Sets the effective height of the plasma bulk (see parameter L_{bulk} , Equation 45).	none]0, ∞[m
inlet_gas_flow	List	Sets a list of inlet gas flow rates for one or more plasma species. The list must be formatted as follows: {{species1 value1} {species2 value2} ...}	0 [0, +∞[sccm
min_density	Number	Sets the numeric minimal density for every species.	1 [0, +∞[m ⁻³
min_density_relaxation_time	Number	Sets the numeric relaxation time to the minimum density.	10 ⁻⁶]0, +∞[s
min_electron_density	Number	Sets the numeric minimal density of electrons.	10 ¹⁵ [0, +∞[m ⁻³
min_electron_energy_density	Number	Sets the numeric minimal energy density of electrons.	1 [0, +∞[eVm ⁻³
name	Character	Sets the name of the reactor. This name is used to refer to the reactor in other commands, such as solve_reactor (see solve_reactor on page 487).	none	none

Chapter 6: Input Commands

define_reactor

Table 47 Parameters of define_reactor command (Continued)

Parameter	Type	Description	Default [Range]	Unit
num_waveform_samples	Number	Sets the number of samples used to resolve the specified waveform expressions. You can specify this parameter only if bias_current_waveform, power_waveform, or wall_potential_waveform is specified.	100 [2, +∞[none
outlet_gas_flow	Number	Sets the initial outlet gas flow rate. The outlet gas flow changes during the simulation to reach the given pressure value (parameter pressure).	0 [0, +∞[sccm
plasma_model	Character	Sets the name of the plasma model, which must have been already defined by the define_plasma_model command (see define_plasma_model on page 258).	none	none
power	Number	Sets the RF power applied to the reactor. The power absorbed by the electrons and ions in the plasma bulk is the product of the parameters power and power_absorption_coefficient. You can specify either power or power_waveform.	none]0, ∞[W
power_absorption_coefficient	Number	Sets the fraction of the applied RF power absorbed by the electrons and ions in the plasma bulk.	1]0, 1]	none
power_period	Number	If you specify power_waveform, then you must also define the period of the power waveform by using this parameter.	none]0, ∞[s

Chapter 6: Input Commands

define_reactor

Table 47 Parameters of `define_reactor` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
power_waveform	Character	<p>Sets the time-dependent periodic RF power waveform applied to the reactor.</p> <p>Note: When using waveform expressions, you must also specify the period of the power waveform, by using the parameter <code>power_period</code>.</p> <p>You can specify either <code>power</code> or <code>power_waveform</code>. For information about the syntax for function expressions, see Syntax for Expressions on page 131.</p>	Functional expression, depending on time t and evaluating to $]0, \infty[$	W
pressure	Number	Sets the pressure inside the reactor. The pressure remains constant during the simulation by a pressure controller (see the <code>pressure_relaxation_time</code> parameter).	none $]0, \infty[$	Pa
pressure_relaxation_time	Number	Sets the characteristic relaxation time of the pressure controller. The pressure controller calibrates the outlet gas flow to match the target pressure given by the <code>pressure</code> parameter.	10^{-6} $]0, \infty[$	s
radius	Number	Sets the effective radius of the plasma bulk (see parameter R_{bulk} , Equation 45).	none $]0, \infty[$	m
reactor	Character	Sets the name of another plasma reactor that will be used as a template for creating the new reactor. All parameters of this reactor are initialized with the values of the template reactor. You can also change some values by explicitly specifying the corresponding parameters.	none	none
rf_bias_current	List	<p>Specifies the amplitudes $\{I_i\}$ of the sinusoidal current applied at the electrode:</p> $I(t) = \sum_i I_i \sin(2\pi f_i t)$ <p>This parameter applies only to the circuit sheath model.</p>	none Each bias current must be in the range $]0, \infty[$	A

Chapter 6: Input Commands

define_reactor

Table 47 Parameters of *define_reactor* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
rf_bias_frequency	List	<p>Specifies the bias frequencies $\{f_i\}$ of the sinusoidal current or voltage applied at the electrode (see parameters <code>rf_bias_current</code> and <code>rf_bias_voltage</code> for details).</p> <p>This parameter applies only to the circuit sheath model.</p>	<p>none Each bias frequency must be in the range $]0, \infty[$</p>	Hz
rf_bias_power	List	<p>Specifies the power $\{P_i\}$ absorbed into the sheath per frequency.</p> <p>For <code>bias_mode=current_driven</code>:</p> $P_i = -\frac{1}{\tau} \int_0^{\tau} I_i \sin(2\pi f_i t) \cdot V_W(t) dt$ <p>For <code>bias_mode=voltage_driven</code>:</p> $P_i = -\frac{1}{\tau} \int_0^{\tau} I(t) \cdot V_i \sin(2\pi f_i t) dt$ <p>This parameter applies only to the circuit sheath model.</p>	<p>none Each bias power must be in the range $]0, \infty[$</p>	W
rf_bias_voltage	List	<p>Specifies the amplitudes $\{V_i\}$ of the sinusoidal voltage applied at the electrode:</p> $V(t) = V_{DC} + \sum_i V_i \sin(2\pi f_i t)$ <p>Note that V_{DC} is computed automatically in a self-consistent way.</p> <p>This parameter applies only to the circuit sheath model.</p>	<p>none Each voltage must be in the range $]0, \infty[$</p>	V
rf_voltage	Number	Sets the amplitude of the RF voltage applied to the reactor. Applies only to the analytic sheath model.	<p>none $]0, \infty[$</p>	V

Chapter 6: Input Commands

define_reactor

Table 47 Parameters of define_reactor command (Continued)

Parameter	Type	Description	Default [Range]	Unit
solution	Character	Sets the name of the plasma solution, which is used to initialize the state of the reactor (= initial densities and electron temperature). The solution name is specified in the command solve_reactor (see solve_reactor on page 487). You can specify either density and electron_temperature, or solution, or solution_file.	none	none
solution_file	Character	Sets the name of the file containing the plasma solution, which is used to initialize the state of the reactor (= initial densities and electron temperature). You can specify either density and electron_temperature, or solution or solution_file.	none	none
type	Character	Sets the type of the plasma reactor. Options are: <ul style="list-style-type: none">• ccp• icp	none	none
wall_potential_period	Number	If you specify wall_potential_waveform, then you must also define the period of the wall potential waveform by using this parameter.	none]0, ∞[s

Chapter 6: Input Commands

define_reactor

Table 47 Parameters of *define_reactor* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
wall_potential_waveform	Character	<p>Sets the time-dependent periodic RF voltage waveform applied to the electrode. The voltage describes the potential drop across the sheath and is defined as the difference between the plasma potential and the electrode potential.</p> <p>Note:</p> <p>When using waveform expressions, you must also specify the period of the waveform, using the parameter <code>wall_potential_period</code>.</p> <p>You can specify either <code>rf_bias_current</code> or <code>rf_bias_power</code> or <code>bias_current_waveform</code> or <code>wall_potential_waveform</code>.</p> <p>This parameter applies only to the circuit sheath model.</p> <p>For information about the syntax for function expressions, see Syntax for Expressions on page 131.</p>	<p>Functional expression, depending on time t and evaluating to $]-\infty, +\infty[$</p> <p>The time-averaged wall potential (DC part) must be negative</p>	V

Note:

Gas flows are typically specified in units of standard cubic centimeters per minute (sccm). The model implicitly converts sccm to SI units, which is given by:

$$1 \text{ sccm} = \frac{p_0}{k_B T_0} \frac{\text{cm}^3}{\text{min}} = 4.4194 \cdot 10^{17} \frac{\text{particles}}{\text{s}}$$

for $p_0 = 10^5 \text{ Pa}$ and $T_0 = 273.15 \text{ K}$ (according to the IUPAC; see [STP on page 559](#)).

Examples

```
define_reactor name=R plasma_model=M type=icp radius=15<cm> \
height=7.5<cm> gas_temperature=305<K> pressure=10<mTorr> \
rf_bias_frequency=1<MHz> rf_bias_power=20<W> power=800<W> \
density={{Ar 1e16<m^-3>} {Ar* 1e14<m^-3>}} \
inlet_gas_flow={{Ar 20<sccm>} {Ar* 0.1<sccm>}} \
solve_reactor name=sol1 reactor=R
```

Chapter 6: Input Commands

define_reflection

define_reflection

This command specifies the properties of a new reflection function that depends on the ion species, the ion energy, and the surface material.

This reflection function can be used in RFM models that take reflection into account.

Syntax

```
define_reflection energy=<n> material=<c> name=<c> species=<c> \
    table=<v> [angle_unit=<c>]

define_reflection energy=<n> material=<c> name=<c> species=<c>
    reflection=<n>
```

Table 48 Parameters of *define_reflection* command

Parameter	Type	Description	Default [Range]	Unit
angle_unit	Character	Sets the unit of the angles listed in the table parameter. Options are: <ul style="list-style-type: none">• deg• rad	deg	none
energy	Number	Sets the energy for which the reflection function is defined. Energy-independent reflection functions can be specified by setting <code>energy=0</code> . In this case, only one specification for each combination of ion species and surface material is allowed.	none [0, ∞[eV
material	Character	Sets the name of the surface material for which the reflection function is defined.	none	none
name	Character	Sets the name used to reference the reflection function in a <code>define_deposit_machine</code> or <code>define_etch_machine</code> command.	none	none
reflection	Number	Sets the reflection parameter used to evaluate the reflection probability according to Equation 12 .	none [0, 1]	none
species	Character	Sets the name of the flux species for which the reflection function is defined.	none	none

Chapter 6: Input Commands

define_reflection

Table 48 Parameters of define_reflection command (Continued)

Parameter	Type	Description	Default [Range]	Unit
table	Vector	Sets the tabular format of the reflection function. When using the parameter table, the same restrictions on the format of the table apply as described in define_jad on page 237 , except that the value specified for 90° does not have to be zero. In addition, all reflection probability values must be in the range [0, 1].	none	none

Description

For energy-dependent reflection functions, for each combination of ion species and surface material, the properties of the reflection function for at least two different energy levels must be specified.

For energy values that have not been specified, the value of the reflection function is calculated by linearly interpolating the values of the parameter reflection for the analytic reflection function or the specified values for the tabular reflection functions.

Sentaurus Topography 3D supports collections of reflection functions, as shown by the following commands, which define a collection of energy-independent reflection functions called my_reflection for flux I and several target materials:

```
define_reflection name=my_reflection species=I energy=0 \
    material=Silicon reflection=0.4

define_reflection name=my_reflection species=I energy=0 \
    material=Oxide reflection=0.1

define_reflection name=my_reflection species=I energy=0 \
    material=Nitride table=$reflection_table

define_reflection name=my_reflection species=I energy=0
    material=Photoresist reflection=0.9
```

Note:

There is no support for energy-dependent flux integration. Therefore, only energy-independent reflection functions can be used.

Chapter 6: Input Commands

define_shape

define_shape

This command defines a new shape for a geometric etch or deposit step.

Syntax

Cone (3D):

```
define_shape type=cylinder name=<c> center1=<v> center2=<v> \
    radius1=<n> radius2=<n> \
    [refinement=<n>] [<transformation_options>]
```

Cone (3D) with circular cross-section:

```
define_shape type=cylinder name=<c> center1=<v> center2=<v> \
    radius1=<n> radius2=<n> \
    [refinement=<n>] [rotation=<n>] [<transformation_options>]
```

Cone (3D) with elliptic cross-section (see [Figure 24](#)):

```
define_shape type=cylinder name=<c> \
    center1=<v> center2=<v> \
    radius1_a=<n> radius1_b=<n> radius2_a=<n> [radius2_b=<n>] \
    [elliptic_exponent=<n> | (elliptic_exponent1=<n> \
    elliptic_exponent2=<n>)] \
    [refinement=<n>] \
    [rotation=<n> | (rotation1=<n> rotation2=<n>)] \
    [<transformation_options>]
```

Cylinder (3D) with circular cross-section:

```
define_shape type=cylinder name=<c> center1=<v> center2=<v> \
    radius=<n> \
    [refinement=<n>] [rotation=<n>] [<transformation_options>]
```

Cylinder (3D) with elliptic cross-section:

```
define_shape type=cylinder name=<c> center1=<v> center2=<v> \
    radius_a=<n> radius_b=<n> \
    [elliptic_exponent=<n>] [refinement=<n>] [rotation=<n>] \
    [<transformation_options>]
```

Cuboid (3D):

```
define_shape type=cube name=<c> point_max=<v> point_min=<v> \
    ([scale_bottom=<n>] | [scale_bottom_x=<n>] [scale_bottom_y=<n>]) \
    ([scale_top=<n>] | [scale_top_x=<n>] [scale_top_y=<n>]) \
    [<transformation_options>]
```

Chapter 6: Input Commands

define_shape

Ellipsoid (3D):

```
define_shape type=ellipsoid name=<c> center=<v> radius_a=<n> \
radius_b=<n> radius_c=<n> \
[refinement=<n>] [<transformation_options>]
```

Sphere (3D):

```
define_shape type=sphere name=<c> center=<v> radius=<n> \
[refinement=<n>] [<transformation_options>]
```

Extruded mask (3D):

```
define_shape type=mask mask=<c> profile=<v> name=<c> \
[invert_mask=<b>] [<transformation_options>]

define_shape type=mask mask=<c> taper_angles=<v> z_coordinates=<v> \
name=<c> [invert_mask=<b>] [<transformation_options>]
```

Circle (2D):

```
define_shape type=circle center=<v> radius=<n> [refinement=<n>]
```

Rectangle (2D):

```
define_shape type=rectangle point_max=<v> point_min=<v> \
[scale_bottom=<n>] [scale_top=<n>]
```

Asymmetric hole (3D):

```
define_shape type=asymmetric_hole name=<c> center=<v> \
profile1=<l> profile2=<l> \
[rotation=<n>] [refinement=<n>] [<transformation_options>]
```

An asymmetric hole is a generalized 3D vertical cylinder with an arbitrary vertical profile, specified by two vertical surface lines `profile1` and `profile2` (see [Figure 25 on page 284](#)). At each height z , the contour of the xy cross-section of the asymmetric hole is a circle.

Here, `center` is a 2D point defining the xy position of the vertical reference axis of the hole. The parameters `profile1` and `profile2` are the left and right surface lines of the hole, respectively, given as a list of lateral offsets as a function of z . By default, the left and right surface lines are located along the x -direction from the given reference axis. To measure the surface lines in another direction from the reference axis, you can use the `rotation` parameter, which defines the measurement direction of the offsets in the xy plane (`rotation=0`: along the x -axis (default), `rotation=90`: along the y -axis, `rotation=180`: along the $-x$ -axis, and so on).

Chapter 6: Input Commands

`define_shape`

Stacked cylinder (3D):

```
define_shape type=stacked_cylinder name=<c> center_list=<v> \
(radius_list=<v> | (radius_a_list=<v> radius_b_list=<v>)) \
[elliptic_exponent_list=<v>] [refinement=<n>] [rotation_list=<v>] \
[<transformation_options>]
```

A stacked cylinder consists of a sequence of 3D cylinders or cones, stacked on each other. Therefore, `type=stacked_cylinder` is a generalization of `type=cylinder` for multiple cylinders. The vertical stack of cylinders is specified by a sequence of 3D cylinder axis points, given by the `center_list` parameter. For each axis point, you can specify the radius of the cylinder, measured in the xy plane through that axis point. Use the parameter `radius_list` to define circular cross sections, or the parameters `radius_a_list`, `radius_b_list`, `rotation_list`, or `elliptic_exponent_list` to define elliptic cross sections.

Table 49 Parameters of `define_shape` command

Parameter	Type	Description	Default [Range]	Unit
center	Vector	Sets the center point of a sphere or circle. For an asymmetric hole, it sets the xy position of the vertical reference axis of the hole. The format is {x y}.	none]-∞, +∞[μm
center_list	List	Specifies a flat list of 3D cylinder axis points along the z-axis. The format is {x1 y1 z1 x2 y2 z2 ...}. Note: The z-values must be either increasing monotonically or decreasing monotonically.	none]-∞, +∞[μm
center1	Vector	Sets the first axis point of a cylinder.	none	μm
center2	Vector	Sets the second axis point of a cylinder.	none	μm
elliptic_exponent	Number	Sets the exponent p in the equation defining the (super-)ellipse curve: $\left(\frac{ x }{r_1}\right)^p + \left(\frac{ y }{r_2}\right)^p = 1$	2]0, ∞[none

Chapter 6: Input Commands

`define_shape`

Table 49 Parameters of `define_shape` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
<code>elliptic_exponent_list</code>	List	Specifies a list of elliptic exponents p of the elliptical cross-sections along the z-axis (one value for each axis point in <code>center_list</code>).	2 10, ∞	none
<code>elliptic_exponent1</code>	Number	Sets the exponent p for the ellipse with the center at <code>center1</code> .	2 10, ∞	none
<code>elliptic_exponent2</code>	Number	Sets the exponent p for the ellipse with the center at <code>center2</code> .	2 10, ∞	none
<code>invert_mask</code>	Boolean	Specifies whether to use the inverted mask instead of the regular mask.	false	none
<code>mask</code>	Character	Sets the name of the mask defined in a <code>define_mask</code> command.	none	none
<code>name</code>	Character	Sets the unique name of the shape.	none	none
<code>point_max</code>	Vector	Sets the maximum corner point of a cube or rectangle.	none	μm
<code>point_min</code>	Vector	Sets the minimum corner point of a cube or rectangle.	none	μm
<code>profile</code>	Vector	Specifies a list of pairs of lateral coordinates and z-coordinates, for example, {11 z1 12 z2}. The lateral coordinates are measured relative to the edge of the 2D mask. Positive lateral coordinates mean the mask is slanted outside, and negative values slant the mask inside. The coordinate pairs are ordered either in ascending or descending z order.	none	μm

Chapter 6: Input Commands

`define_shape`

Table 49 Parameters of `define_shape` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
<code>profile1</code>	List	Defines the (left) surface line of a hole. You must specify the horizontal distance of the surface to the reference axis of the hole as a function of z. The z-coordinates must be sorted in either increasing order or decreasing order. The first and last z-coordinates in <code>profile1</code> and <code>profile2</code> must coincide. The format is {11 z1 12 z2 ...}.	none]-∞, +∞[μm
<code>profile2</code>	List	Defines the (right) surface line of a hole, opposite to the surface line given in <code>profile1</code> . You must specify the horizontal distance of the surface to the reference axis of the hole as a function of z. The z-coordinates must be sorted in either increasing order or decreasing order. The first and last z-coordinates in <code>profile1</code> and <code>profile2</code> must coincide. The format is {11 z1 12 z2 ...}.	none]-∞, +∞[μm
<code>radius</code>	Number	Sets the radius of a circle, a sphere, or a cylinder base circle.	none [0, ∞[μm
<code>radius_a</code>	Number	Sets the main axis length in the x-direction for an elliptic cylinder or ellipsoid.	none [0, ∞[μm
<code>radius_a_list</code>	List	Specifies a list of main axis lengths in the x-direction of the elliptical cross-sections along the z-axis (one value for each axis point in <code>center_list</code>).	none [0, ∞[μm
<code>radius_b</code>	Number	Sets the main axis length in the y-direction for an elliptic cylinder or ellipsoid.	none [0, ∞[μm
<code>radius_b_list</code>	List	Specifies a list of main axis lengths in the x-direction of the elliptical cross-sections along the z-axis (one value for each axis point in <code>center_list</code>).	none [0, ∞[μm

Chapter 6: Input Commands

`define_shape`

Table 49 Parameters of `define_shape` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
<code>radius_c</code>	Number	Sets the main axis length in the z-direction for an ellipsoid.	none [0 , ∞[μm
<code>radius1</code>	Number	Sets the radius of the circle with the center at <code>center1</code> of a truncated cone.	none [0 , ∞[μm
<code>radius1_a</code>	Number	Sets the main axis length in the x-direction of the ellipse with the center at <code>center1</code> of an elliptic cylinder.	none [0 , ∞[μm
<code>radius1_b</code>	Number	Sets the main axis length in the y-direction of the ellipse with the center at <code>center1</code> of an elliptic cylinder.	none [0 , ∞[μm
<code>radius2</code>	Number	Sets the radius of the circle with the center at <code>center2</code> of a truncated cone.	none [0 , ∞[μm
<code>radius2_a</code>	Number	Sets the main axis length in the x-direction of the ellipse with the center at <code>center2</code> of an elliptic cylinder.	none [0 , ∞[μm
<code>radius2_b</code>	Number	Sets the main axis length in the y-direction of the ellipse with the center at <code>center2</code> of an elliptic cylinder.	none [0 , ∞[μm
<code>refinement</code>	Number	Sets the refinement as follows: <ul style="list-style-type: none"> If a circle, this is the number of circle vertices. If a cylinder, this is the number of base circle points of the cylinder. If a sphere, this is the number of icosahedron faces: $20 \cdot 4^{\text{refinement}}$. If an asymmetric hole, this is the number of circle points at each orthogonal cross-section to the hole axis. 	20 (circle) 20 (cylinder) 2 (sphere) 20 (asymmetric hole) [1 , ∞[none

Chapter 6: Input Commands

define_shape

Table 49 Parameters of define_shape command (Continued)

Parameter	Type	Description	Default [Range]	Unit
rotation	Number	Sets the angle by which a cone or cylinder is rotated about its axis. For an asymmetric hole, it defines the measurement direction of the distance between the surface line and the vertical reference axis. For example, if rotation=0, the lateral offsets given in profile1 and profile2 are measured in the x-direction. If rotation=90, the lateral offsets are measured in the y-direction, and so on.	0 [-180, 180]	degree
rotation_list	List	Specifies a list of rotation angles of the elliptical cross-sections along the z-axis (one value for each axis point in center_list).	0 [-180, 180]	degree
rotation1	Number	Sets the angle by which the ellipse with the center at center1 is rotated about the axis through center1 and center2.	0 [-180, 180]	degree
rotation2	Number	Sets the angle by which the ellipse with the center at center2 is rotated about the axis through center1 and center2.	0 [-180, 180]	degree
scale_bottom	Number	Sets the scale factor for the bottom face or edge of the initial cube or rectangle. A truncated pyramid or a symmetric trapezoid is created instead of a cube or a rectangle, respectively.	1 [0, ∞[none
scale_bottom_x	Number	Sets the scale factor for the two sides parallel to the x-axis of the bottom face of a cube.	1 [0, ∞[none
scale_bottom_y	Number	Sets the scale factor for the two sides parallel to the y-axis of the bottom face of a cube.	1 [0, ∞[none
scale_top	Number	Sets the scale factor for the top face or edge of the initial cube or rectangle. A truncated pyramid or a symmetric trapezoid is created instead of a cube or a rectangle, respectively.	1 [0, ∞[none

Chapter 6: Input Commands

define_shape

Table 49 Parameters of *define_shape* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
scale_top_x	Number	Sets the scale factor for the two sides parallel to the x-axis of the top face of a cube.	1 [0, ∞[none
scale_top_y	Number	Sets the scale factor for the two sides parallel to the y-axis of the top face of a cube.	1 [0, ∞[none
taper_angles	Vector	Specifies a list of taper angles to be specified in combination with z_coordinates. The number of taper angles must match the number of intervals. Therefore, this vector has one element less than z_coordinates.	none Each vector element must be in the range [-89, 89]	degree
<transformation_options>	Character	Specifies a set of parameters with which to transform the shape after creation (see <i>Shape Transformations: <transformation_options></i>).	none	none
type	Character	Sets the shape to be defined. Options are: <ul style="list-style-type: none"> • asymmetric_hole • circle • cube • cylinder • ellipsoid • mask • rectangle • sphere • stacked_cylinder 	none	none
z_coordinates	Vector	Specifies a list of z-coordinates to be specified in combination with taper_angles. The slant of the intervals is specified with taper_angles; therefore, z_coordinates requires one element more than taper_angles.	none	µm

Chapter 6: Input Commands

define_shape

Shape Transformations: <transformation_options>

Depending on the required transformation, you need to specify the following parameters:

- For an Euler rotation: alpha=<n> beta=<n> gamma=<n>
- For a rotation around a specified axis, one of the following:
 - axis=<c> axis_rotation=<n> point=<v>
 - axis_rotation=<n> direction=<v> point=<v>
 - axis_rotation=<n> point1=<v> point2=<v>
- For shearing or tilting: shear_fix_axis=<n> shear_fix_position=<n>
shear_rotation=<n> shear_tilt=<n>

[Table 50](#) lists the parameters specifically for transforming shapes.

Table 50 Parameters for shape transformations

Parameter	Type	Description	Default [Range]	Unit
alpha	Number	Sets the first Euler angle for rotating an ellipsoid.	0 [-180, 180]	degree
axis	Character	Sets the rotation direction to one of the coordinate axes. It depends on the parameter <code>point</code> . Options are: <ul style="list-style-type: none">• x• y• z	none	none
axis_rotation	Number	Sets the angle by which the shape is rotated around a specified axis.	0 [-180, 180]	degree
beta	Number	Sets the second Euler angle for rotating an ellipsoid.	0 [-180, 180]	degree
direction	Vector	Sets the direction of the rotation axis. It depends on the parameter <code>point</code> .	none	µm
gamma	Number	Sets the third Euler angle for rotating an ellipsoid.	0 [-180, 180]	degree
point	Vector	Sets a point along the rotation axis. It depends on the parameter <code>axis_rotation</code> .	none	µm

Chapter 6: Input Commands

define_shape

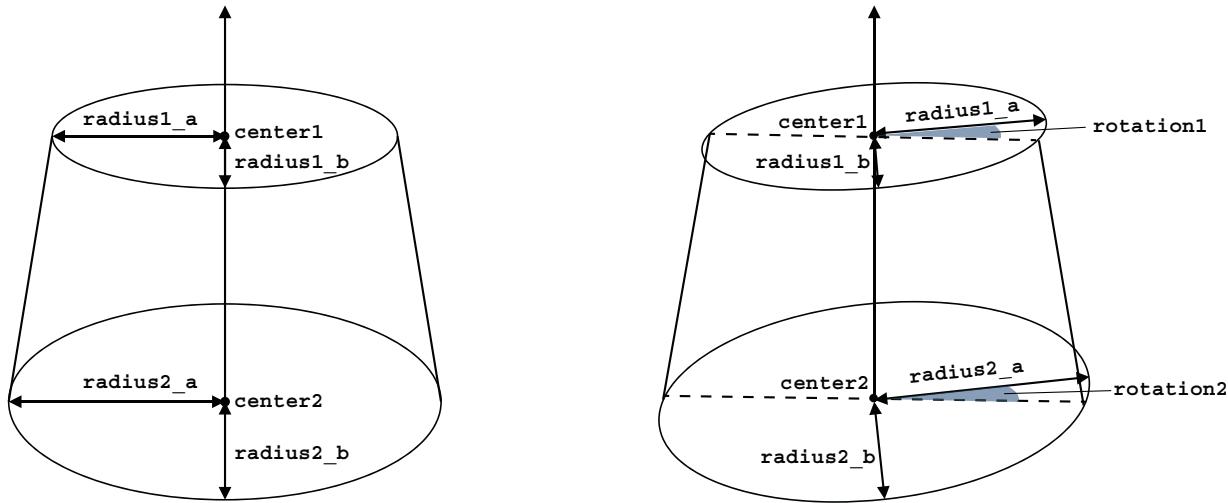
Table 50 Parameters for shape transformations

Parameter	Type	Description	Default [Range]	Unit
point1	Vector	Sets the first point of the rotation axis. It depends on the parameter axis_rotation.	none	µm
point2	Vector	Sets the second point of the rotation axis. It depends on the parameter point1.	none	µm
shear_rotation	Number	Sets the angle of the shear direction. By default, the shear direction is: <ul style="list-style-type: none"> • x if shear_fix_axis=z • y if shear_fix_axis=x • z if shear_fix_axis=y Then, the direction is rotated around the shear_fix_axis by the shear_rotation angle to calculate the shearing direction.	0 [-180, 180]	degree
shear_fix_position	Number	Sets the distance of the fixed plane to the coordinate root. The distance is measured in the direction given by shear_fix_axis. If this parameter is not set, then it is set internally to the highest point of the structure. This parameter depends on the shear_tilt parameter.	none	µm
shear_fix_axis	String	Sets the axis that defines the fixed plane for the shearing operation. Options are: <ul style="list-style-type: none"> • x • y • z This parameter depends on the shear_tilt parameter.	z	none
shear_tilt	Number	Sets the angle by which the shape is tilted.	0 [-89, 89]	degree

Chapter 6: Input Commands

```
define_shape
```

Figure 24 (Left) Three-dimensional cone with elliptic cross section and (right) same cone with rotation or twisting



Examples

Define a 2D circle with center {0.0 0.0}, radius 1.0 μm , and name `circle_1`:

```
define_shape type=circle name=circle_1 center={0.0 0.0} radius=1.0
```

Define a 2D rectangle with corners {0.0 0.0} {1.0 1.0} and the name `r_1`:

```
define_shape type=rectangle name=r_1 point_max={1.0 1.0} \
point_min={0.0 0.0}
```

Define a cuboid with bounding points {0.25 0.25 0.25}, {0.75 0.75 1.0} and with the name `cube_1` (the cuboid can be used in a subsequent etching or deposition step):

```
define_shape type=cube name=cube_1 point_max={0.75 0.75 1.0} \
point_min={0.25 0.25 0.25}
```

Define a cylinder with an axis of {0.0 0.0 0.0}, {0.0 0.0 1.0} parallel to the z-axis, with a radius of 0.25 μm and the name `cylinder_1` (the cylinder can be used in a subsequent etching or deposition step):

```
define_shape type=cylinder name=cylinder_1 \
center1={0.0 0.0 0.0} center2={0.0 0.0 1.0} radius=0.25
```

Define a sphere with a center point of {0.0 0.0 0.0} and a radius of 0.25 μm , with the name `sphere_1` (the sphere can be used in a subsequent etching or deposition step):

```
define_shape type=sphere name=sphere_1 center={0.0 0.0 0.0} radius=0.25
```

Chapter 6: Input Commands

define_shape

Define a pyramid with vertices at the points of coordinates {0.0 0.0 0.0}, {0.0 0.0 1.0}, {0.0 1.0 0.0}, {0.0 1.0 1.0}, and {0.5 0.5 1.0} (this pyramid can be used in a subsequent etching or deposition step):

```
define_shape type=cube name=pyramid point_min={0.0 0.0 0.0} \
    point_max={1.0 1.0 1.0} scale_top=0
```

Define a square truncated pyramid with vertices at the points of coordinates {0.0 0.0 0.0}, {0.0 0.0 1.0}, {0.0 1.0 0.0}, {0.0 1.0 1.0}, {0.25 0.25 1.0}, {0.75 0.25 1.0}, {0.25 0.75 1.0}, and {0.75 0.75 1.0} (this pyramid can be used in a subsequent etching or deposition step):

```
define_shape type=cube name=square_truncated_pyramid \
    point_min={0.0 0.0 0.0} \
    point_max={1.0 1.0 1.0} scale_top=0.5
```

Define a nonsquare truncated pyramid with vertices at the points of coordinates {0.0 0.0 0.0}, {0.0 0.0 1.0}, {0.0 1.0 0.0}, {0.0 1.0 1.0}, {0.25 0.375 1.0}, {0.75 0.375 1.0}, {0.25 0.625 1.0}, and {0.75 0.625 1.0} (this pyramid can be used in a subsequent etching or deposition step):

```
define_shape type=cube name=truncated_pyramid \
    point_min={0.0 0.0 0.0} point_max={1.0 1.0 1.0} \
    scale_top_x=0.5 scale_top_y=0.25
```

Define an ellipsoid centered at the point of coordinates {0.0 0.0 0.0} and with main axes of length 0.5 μm in the x-direction, 0.2 μm in the y-direction, and 1 μm in the z-direction (this ellipsoid can be used in a subsequent etching or deposition step):

```
define_shape type=ellipsoid name=e center={0.0 0.0 0.0} \
    radius_a=0.5 radius_b=0.2 radius_c=1
```

Define a 3D elliptic hole by three axis points, $\vec{c}_1 = (0,0,2)$, $\vec{c}_2 = (0.01,0,1)$, and $\vec{c}_3 = (-0.01,0,0)$. At each height z , the xy cross-section is an ellipse, defined by two radii, a rotation, and an elliptic exponent. For example, at \vec{c}_1 , the first radius of the ellipse is 1.0, the second radius is 1.5, the rotation with respect to the x-axis is 45°, and the elliptic exponent is 2:

```
define_shape type=stacked_cylinder name=cyl \
    center_list={ 0      0      2 \
                  0.01   0      1 \
                  -0.01  0      0 } \
    radius_a_list={1.0  0.9  1.1} \
    radius_b_list={1.5  1.4  1.5} \
    rotation_list={45  50  55} \
    elliptic_exponent_list={2  2  2} \
    refinement=360
```

Chapter 6: Input Commands

```
define_shape
```

Define a cube rotated by Euler angles `alpha`, `beta`, and `gamma` around the z-axis, x'-axis, and z"-axis:

```
define_shape name=cube type=cube point_min={ 0.2 0.2 0.0 } \
    point_max={0.6 0.6 1.0} alpha=90 beta=-20 gamma=-100
```

Define a cube rotated by 7° around an axis specified with a point and a direction:

```
define_shape name=cube type=cube point_min={0.2 0.2 0.0} \
    point_max={0.6 0.6 1.0} axis_rotation=7 \
    direction={0.3 0.2 0.5} point={1.0 2.4 1.8}
```

Etch an asymmetric hole, given by two opposite surface lines `profile1` and `profile2`. The x-coordinates in `profile1` and `profile2` are measured with respect to a reference axis at the origin, $(x,y)=(0,0)$, as specified by parameter `center`:

```
define_structure material=Oxide \
    point_min={ -0.4 -0.2 0.0 } \
    point_max={ 0.0 +0.2 1.0 }

define_shape name=s type=asymmetric_hole \
    profile1={ 0.00 1.0 \
        -0.10 0.9 \
        -0.18 0.8 \
        -0.20 0.7 \
        -0.20 0.5 \
        -0.19 0.4 \
        -0.20 0.2 \
        -0.25 0.1 } \
    profile2={ -0.30 0.1 \
        -0.30 0.4 \
        -0.28 0.5 \
        -0.30 0.7 \
        -0.35 0.75 \
        -0.40 1.0 } \
    center={0 0} refinement=90

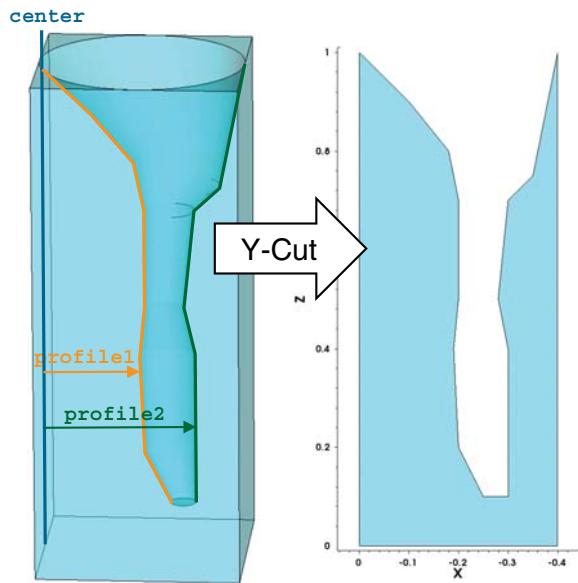
etch shape=s
```

[Figure 25 on page 284](#) shows the resulting shape.

Chapter 6: Input Commands

define_shape

Figure 25 Generation of an asymmetric hole



Chapter 6: Input Commands

define_sheath_solver

define_sheath_solver

This optional command defines the numeric parameters of the solver used for a plasma sheath model. Depending on the sheath model type, different solvers are available.

Note:

Two default sheath solvers are implicitly defined and can be referenced by other commands, such as `solve_reactor` (see [solve_reactor on page 487](#)). For example:

- `default_analytic_sheath_solver` (**for** `sheath_model_type=analytic`)
- `default_circuit_sheath_solver` (**for** `sheath_model_type=circuit`)

The implicit definitions of these default solvers are equivalent to:

```
define_sheath_solver name=default_analytic_sheath_solver \
    sheath_model_type=analytic
```

```
define_sheath_solver name=default_circuit_sheath_solver \
    sheath_model_type=circuit
```

Syntax

Analytic RF sheath model (see [Analytic RF Sheath Model on page 108](#)):

```
define_sheath_solver name=<c> sheath_model_type=analytic \
    [num_angle_samples=<n>] [num_energy_samples=<n>]
```

Self-consistent circuit RF sheath model (see [Self-Consistent Circuit RF Sheath Model on page 103](#)):

```
define_sheath_solver name=<c> sheath_model_type=circuit \
    [abs_error=<n>] [discard_slow_ions=<b>] \
    [frequency_snapping_tolerance=<n>] \
    ([ied_bin_size=<n>] | [num_ied_histogram_bins=<n>]) \
    [ied_solver=<c>] \
    [initial_rf_current=<n>] [initial_sheath_width=<n>] \
    [initial_wall_potential=<n>] \
    [max_num_waveform_periods=<n>] \
    [max_rf_current_step=<n>] \
    [max_rf_voltage_step=<n>] [max_sheath_iterations=<n>] \
    [max_sheath_width_integration_steps=<n>] \
    [max_sheath_width_iterations=<n>] \
    [max_wall_potential_iterations=<n>] \
    [min_sheath_width_integration_steps=<n>] \
    [monte_carlo_random_seed=<n>] \
    [num_angle_samples=<n>] [num_energy_samples=<n>] \
    [num_inv_cdf_histogram_bins=<n>] \
    [num_monte_carlo_samples=<n>] \
    [power_relaxation_constant=<n>] [power_tolerance=<n>] \
    [rel_error=<n>] [sheath_tolerance=<n>] [time_average_solution=<b>]
```

Chapter 6: Input Commands

`define_sheath_solver`

Table 51 Parameters of `define_sheath_solver` command

Parameter	Type	Description	Default [Range]	Unit
<code>abs_error</code>	Number	Sets the absolute error used for adaptive step size control during the trajectory integration in the Monte Carlo sheath solver. It applies only when <code>ied_solver=monte_carlo</code> .	10^{-6}] $0, \infty[$	none
<code>discard_slow_ions</code>	Boolean	Specifies whether to discard trajectories, which intersect the oscillating RF sheath boundary, in the Monte Carlo sheath solver.	true	none
<code>frequency_snapping_tolerance</code>	Number	When using time-dependent waveforms (from a bulk simulation with pulsed power) as input to the sheath model, this parameter sets the tolerance used when trying to find the least common multiple period between the input waveforms (densities and electron temperature) and the bias waveforms (bias current, voltage, or power).	0.01] $0, \infty[$	none
<code>ied_bin_size</code>	Number	Sets the energy resolution of the ion energy distribution (IED) histogram. You can specify either <code>ied_bin_size</code> or <code>num_ied_histogram_bins</code> .	none] $0, \infty[$	none
<code>ied_solver</code>	Character	Sets the method used to compute the IED of the ions crossing the sheath layer. Options are <code>analytic</code> and <code>monte_carlo</code> . This parameter is available only for <code>sheath_model_type=circuit</code> .	monte_carlo	none
<code>initial_rf_current</code>	Number	Sets the initial guess of the RF current.	3] $0, \infty[$	A
<code>initial_sheath_width</code>	Number	Sets the initial guess of the sheath width.	10^{-4}] $0, \infty[$	m
<code>initial_wall_potential</code>	Number	Sets the initial guess of the sheath wall potential.	-10] $0, \infty[$	V

Chapter 6: Input Commands

`define_sheath_solver`

Table 51 Parameters of `define_sheath_solver` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
<code>max_num_waveform_periods</code>	Number	When using time-dependent density and electron temperature waveforms from a bulk simulation with pulsed power as input to the sheath model, this parameter sets the maximum number of periods considered when trying to find the least common multiple period between the input waveforms (densities and electron temperature) and the bias waveforms (bias current, voltage, or power).	100 [1, ∞)	none
<code>max_rf_current_step</code>	Number	If you specify <code>rf_bias_power</code> with <code>bias_mode=current_driven</code> , then you can set the maximum-allowed relative change of the RF bias current during convergence toward the specified RF bias power by using this parameter.	0.05 [0, ∞)	none
<code>max_rf_voltage_step</code>	Number	If you specify <code>rf_bias_power</code> with <code>bias_mode=voltage_driven</code> , then you can set the maximum-allowed relative change of the RF bias voltage during convergence toward the specified RF bias power by using this parameter.	0.05 [0, ∞)	none
<code>max_sheath_iterations</code>	Number	Sets the maximum number of iterations to obtain a self-consistent overall sheath solution.	10^7 [1, ∞[none
<code>max_sheath_width_integration_steps</code>	Number	Sets the maximum number of integration steps to solve for the sheath width.	500 [1, ∞[none
<code>max_sheath_width_iterations</code>	Number	Sets the maximum number of iterations to solve for the sheath width.	8 [1, ∞]	none
<code>max_wall_potential_iterations</code>	Number	Sets the maximum number of iterations to solve for the wall potential.	8 [1, ∞]	none
<code>min_sheath_width_integration_steps</code>	Number	Sets the minimum number of integration steps to solve for the sheath width.	50 [1, ∞[none

Chapter 6: Input Commands

define_sheath_solver

Table 51 Parameters of define_sheath_solver command (Continued)

Parameter	Type	Description	Default [Range]	Unit
monte_carlo_random_seed	Number	Sets the random seed used to sample the IED by the Monte Carlo sheath solver. This parameter applies only when ied_solver=monte_carlo.	42 [0, ∞)	none
name	Character	Sets the name of the solver. This name is used to reference the solver in other commands, such as solve_reactor (see solve_reactor on page 487).	none	none
num_angle_samples	Number	Sets the number of angle samples used to resolve the energy-angular distribution.	180 [2, ∞[none
num_energy_samples	Number	Sets the number of energy samples used to resolve the energy-angular distribution.	1000 [2, ∞[none
num_ied_histogram_bins	Number	Sets the number of bins in the IED histogram. You can specify either ied_bin_size or num_ied_histogram_bins.	1000 [2, ∞[none
num_inv_cdf_histogram_bins	Number	Sets the number of bins of the inverse cumulative distribution function (CDF) histogram corresponding to the IED.	1000 [1, ∞]	none
num_monte_carlo_samples	Number	Sets the number of samples used to sample the ion energy distribution in the Monte Carlo sheath solver. You can specify this parameter only if ied_solver=monte_carlo.	10000 [2, ∞)	none
power_relaxation_constant	Number	If you specify rf_bias_power, then you can use this parameter to set the characteristic scale of the relaxation toward the specified RF bias power.	100]0, ∞)	none
power_tolerance	Number	Sets the tolerance for the convergence criterion of the RF bias power.	10^{-3}]0, ∞[none

Chapter 6: Input Commands

define_sheath_solver

Table 51 Parameters of define_sheath_solver command (Continued)

Parameter	Type	Description	Default [Range]	Unit
rel_error	Number	Sets the relative error used for adaptive step size control during the trajectory integration in the Monte Carlo sheath solver. You can specify this parameter only if ied_solver=monte_carlo.	10^{-6}] $0, \infty[$	none
sheath_model_type	Character	Sets the type of the sheath model to solve. Options are: <ul style="list-style-type: none">• analytic• circuit	none	none
sheath_tolerance	Number	Sets the tolerance for the convergence criterion of the sheath potential.	10^{-3}] $0, \infty[$	none
time_average_solution	Boolean	When using time-dependent density and electron temperature waveforms from a bulk simulation with pulsed power as input to the sheath model, this parameter specifies whether the sheath solution, consisting of time-dependent periodic waveforms, should be averaged in time.	true	none

Examples

```
define_sheath_solver name=s2 sheath_model_type=analytic \
    num_angle_samples=180 num_energy_samples=1000

define_sheath_solver name=s1 sheath_model_type=circuit \
    initial_sheath_width=1e-4<m> initial_wall_potential=-10<V> \
    initial_rf_current=3<A> max_sheath_iterations=1000 \
    max_sheath_width_iterations=8 \
    min_sheath_width_integration_steps=10 \
    max_sheath_width_integration_steps=100 \
    max_wall_potential_iterations=8 \
    num_angle_samples=180 \
    num_energy_samples=1000 num_ied_histogram_bins=200 \
    num_inv_cdf_histogram_bins=100 sheath_tolerance=1e-3 \
    power_tolerance=1e-2
```

Chapter 6: Input Commands

define_species_distribution

define_species_distribution

This command defines a new energy and angular distribution (EAD) that can be used in a reaction model. As a special case, it allows you to define angular distributions that are energy independent.

Syntax: Analytic and Tabular EADs

In the following syntax examples, a distribution is energy dependent only if you explicitly specify energy-related parameters:

```
<energy_parameters>=
  energy_center=<n> energy_center_weight=<n> energy_max=<c>
  energy_min=<c> energy_max_weight=<n> energy_min_weight=<n>
```

Otherwise, the distribution is assumed to be energy independent.

You can specify the following types of energy distribution:

- Monoenergetic: If you set the same value for `energy_min` and `energy_max`, then all particles will have the same energy.
- Uniform: You can define a uniform energy window by setting `energy_min` and `energy_max`.
- Bimodal: You can define a bimodal energy distribution by specifying the three characteristic points: the two peaks at `energy_min` and `energy_max`, and the minimum position at `energy_center`.

To specify the relative heights between the peaks and the minimum, use `energy_min_weight`, `energy_max_weight`, and `energy_center_weight`. (The weights do not define the absolute values of the final energy- and angle-dependent flux distribution (IEAD) used by the PMC method. This is because the actual amplitude of the IEAD is determined by the parameter flux, which is defined as the integral of the IEAD over all energies and angles.)

Define a distribution whose angular-dependent part is described by [Equation 6 on page 49](#):

```
define_species_distribution exponent=<c> flux=<c> name=<c> \
  species=<c> [<energy_parameters>] [orientation=<v>] \
  [sampling_time_step=<n>]
```

As an alternative to the exponent parameter, you can specify the characteristic width of the angular distribution in [Equation 6](#) by using the `angle_spread` parameter. This parameter is defined such that 68.3% of the particles have an angle smaller than `angle_spread` (in analogy to the first confidence interval of a Gaussian distribution):

```
define_species_distribution angle_spread=<n> flux=<c> name=<c> \
  species=<c> [<energy_parameters>] [orientation=<v>] \
  [sampling_time_step=<n>]
```

Chapter 6: Input Commands

define_species_distribution

Define a distribution that specifies particles traveling along the vertical direction:

```
define_species_distribution flux=<c> name=<c> species=<c> \
    unidirectional=<b> \
    [<energy_parameters>] [orientation=<v>] [sampling_time_step=<n>]
```

Define a distribution whose angular-dependent part is described by tabular data:

```
define_species_distribution flux=<c> name=<c> species=<c> table=<v> \
    [angle_unit=<c>] [<energy_parameters>] \
    [orientation=<v>] [sampling_time_step=<n>]
```

You can define an azimuth-dependent angular distribution with an anisotropic exponent. The analytic expression for anisotropic exponents is given by:

$$f(\theta, \phi) = A \cos^{m(\phi)}(\theta) \quad (83)$$

where the exponent depends on the azimuth ϕ as given by

$$m(\phi) = m_x \cos^2(\phi) + m_y \sin^2(\phi).$$

Therefore, the exponent takes the value m_x in the x-direction and m_y in the y-direction. An additional azimuthal rotation $\phi \rightarrow \phi - \phi_0$ allows you to orientate x- and y-anisotropy axes along an arbitrary direction (see [Figure 27 on page 302](#)).

Define an azimuth-dependent species distribution:

```
define_species_distribution type=azimuthal exponent_x=<c> \
    exponent_y=<c> flux=<c> name=<c> species=<c> \
    [<energy_parameters>] [orientation=<v>] [rotation=<n>]
```

Syntax: EADs From Plasma Models

An energy and angular distribution can also be computed by the Benoit-Cattin plasma sheath model for capacitively coupled plasma (CCP) reactors [\[9\]](#)[\[10\]](#)[\[11\]](#). This model depends on the parameters of the plasma reactor, such as the temperature T_e of the electrons, the ion bulk density n_i , mass m_i , and temperature T_i of the plasma ions, and the applied RF voltage $V(t) = V_{rf} \sin(2\pi f_{rf}t)$, where V_{rf} denotes the amplitude of the RF voltage and f_{rf} is the RF pulsing frequency. The distribution can be defined with the following command:

```
define_species_distribution type=ccp_sheath \
    electron_temperature=<n> \
    ion_bulk_density=<n> ion_mass=<n> ion_temperature=<n> name=<c> \
    rf_frequency=<n> rf_voltage=<n> species=<c> \
    [flux=<c>] [sampling_time_step=<n>]
```

As an alternative to the applied RF voltage V_{rf} , you can specify the sheath voltage $V_{rf}(t) = V_{DC} + V_{AC} \sin(2\pi f_{ft}t)$ by defining the AC sheath voltage V_{AC} and the DC sheath voltage V_{DC} :

```
define_species_distribution type=ccp_sheath ac_voltage=<n> \
    dc_voltage=<n> electron_temperature=<n> \
```

Chapter 6: Input Commands

define_species_distribution

```
ion_bulk_density=<n> ion_mass=<n> \
ion_temperature=<n> name=<c> rf_frequency=<n> species=<c> \
[flux=<c>] [sampling_time_step=<n>]
```

See [Analytic RF Sheath Model on page 108](#).

For inductively coupled plasma (ICP) reactors, the energy and angular distribution is computed by the Edelberg–Aydil plasma sheath model [10][12]. This model also depends on reactor parameters such as the electrode area A , the electron temperature T_e , the ion bulk density n_i , mass m_i , and temperature T_i of the plasma ions, and the applied RF current $I(t) = I_{rf} \sin(2\pi f_{rf} t)$, where I_{rf} denotes the amplitude of the RF current and f_{rf} is the RF pulsing frequency.

The distribution can be defined with the following command:

```
define_species_distribution type=icp_sheath electrode_area=<n> \
electron_temperature=<n> ion_bulk_density=<n> ion_mass=<n> \
ion_temperature=<n> name=<c> rf_current=<n> rf_frequency=<n> \
species=<c> \
[flux=<c>] [sampling_time_step=<n>]
```

Instead of specifying the applied RF current I_{rf} , you can specify the RF bias power P_{rf} :

```
define_species_distribution type=icp_sheath electrode_area=<n> \
electron_temperature=<n> ion_bulk_density=<n> ion_mass=<n> \
ion_temperature=<n> name=<c> rf_frequency=<n> rf_power=<n> \
species=<c> \
[flux=<c>] [sampling_time_step=<n>]
```

See [Self-Consistent Circuit RF Sheath Model on page 103](#).

Define a species distribution from the solution of a plasma model (see [Chapter 4 on page 92](#) and [solve_reactor on page 487](#)):

```
define_species_distribution type=plasma name=<c> species=<c> \
[flux=<c>] [solution=<c> | solution_file=<c>]
```

Syntax: EADs From Files

Import an energy and angular distribution computed by the Plasma Chemistry Monte Carlo (PCMC) module of the HPEM plasma simulator [13][14]:

```
define_species_distribution hpem_file=<c> name=<c> species=<c> \
[energy_max=<c> energy_min=<c> flux=<c> hpem_material_index=<n> \
hpem_species=<c>] [sampling_time_step=<n>]
```

Define an energy-independent angular distribution by integrating an energy and angular distribution computed by HPEM:

```
define_species_distribution hpem_file=<c> \
integration_energy_max=<n> \
integration_energy_min=<n> name=<c> species=<c> \
```

Chapter 6: Input Commands

define_species_distribution

```
[flux=<c> hpem_material_index=<n> hpem_species=<c>] \
[sampling_time_step=<n>]
```

Import an energy and angular distribution from an EAD file (see [Appendix B on page 554](#)):

```
define_species_distribution type=ead_file file=<c> name=<c> \
species=<c> \
[energy_min=<c> energy_max=<c>] [flux=<c>] \
[sampling_method=<c>] [sampling_time_step=<n>]
```

Syntax: Combining Distributions

Previously defined species distributions can be combined into a single species distribution by summing the flux distributions of individual distributions with the following command:

```
define_species_distribution type=sum distributions=<l> name=<c> \
species=<c> [flux=<c>]
```

Table 52 Parameters of *define_species_distribution* command

Parameter	Type	Description	Default [Range]	Unit
ac_voltage	Number	Sets the AC sheath voltage. This parameter applies only if <code>type=ccp_sheath</code> and if <code>rf_voltage</code> is not specified.	none]0, ∞[V
angle_spread	Number	Sets the characteristic width of the angular distribution $\cos^m(\theta)$. The angle spread is defined such that 68.3% of the particles have an angle smaller than <code>angle_spread</code> .	none]0, 55.716]	degree
angle_unit	Character	Sets the unit of the angles listed in the <code>table</code> parameter. Options are: <ul style="list-style-type: none">• deg• rad	deg	none
dc_voltage	Number	Sets the DC sheath voltage. This parameter applies only if <code>type=ccp_sheath</code> and if <code>rf_voltage</code> is not specified.	none]0, ∞[V

Chapter 6: Input Commands

define_species_distribution

Table 52 Parameters of define_species_distribution command (Continued)

Parameter	Type	Description	Default [Range]	Unit
distributions	List	Specifies a list of lists, containing the names and species of the flux distributions to be summed. Format: <code> {{distribution1 species1} {distribution2 species2} ...}</code> This parameter applies only if type=sum.	none	none
electrode_area	Number	Sets the area of the electrode. This parameter applies only if type=icp_sheath.	none]0, ∞[cm ²
electron_temperature	Number	Sets the electron temperature at the plasma sheath edge. This parameter applies if either type=ccp_sheath or type=icp_sheath.	none]0, ∞[eV
energy_center	Number	Sets the position of the energy minimum when defining a bimodal energy distribution.	none]0, ∞[eV
energy_center_weight	Number	Sets the height of a bimodal energy distribution at the energy specified by energy_center. This parameter can be specified only if energy_center is specified.	1]0, ∞[none
energy_max	Character	Sets the maximum energy of the energy distribution. When hpem_file is specified, energy_max sets the maximum energy to take into account when reading the specified HPEM output file. When type=ead_file, energy_max sets the upper energy limit of the EAD. Energy values above this limit are not considered during import.	∞ when type=ead_file or hpem_file is used; otherwise, none Time-dependent expression, evaluating to]0, ∞[and not smaller than the value of the time-dependent expression energy_min at the same time	eV

Chapter 6: Input Commands

define_species_distribution

Table 52 Parameters of define_species_distribution command (Continued)

Parameter	Type	Description	Default [Range]	Unit
energy_max_weight	Number	Sets the height of a bimodal energy distribution at the energy specified by energy_max. This parameter can be specified only if energy_max is specified.	1 [0, ∞[none
energy_min	Character	Sets the minimum energy of the energy distribution. When hpem_file is specified, energy_min sets the minimum energy to take into account when reading the specified HPEM output file. When type=ead_file, energy_min sets the lower energy limit of the EAD. Energy values below this limit are not considered during import.	0 when type=ead_file or hpem_file is used; otherwise, none Time-dependent expression, evaluating to [0, ∞[and not larger than the value of the time-dependent expression energy_max at the same time	eV
energy_min_weight	Number	Sets the height of a bimodal energy distribution at the energy specified by energy_min. This parameter can be specified only if energy_min is specified.	1 [0, ∞[none
exponent	Character	Sets the expression of the exponent of the angular distribution $\cos^m(\theta)$ of the species.	none [1, ∞[none
exponent_x	Character	Sets the exponent in the x-direction. Note: This parameter applies only if type=azimuthal.	none Time-dependent expression, evaluating to [1, ∞[none
exponent_y	Character	Sets the exponent in the y-direction. Note: This parameter applies only if type=azimuthal.	none Time-dependent expression, evaluating to [1, ∞[none

Chapter 6: Input Commands

define_species_distribution

Table 52 Parameters of define_species_distribution command (Continued)

Parameter	Type	Description	Default [Range]	Unit
file	Character	Sets the path to the EAD file to be imported. Note: This parameter applies only if type=ead_file.	none	none
flux	Character	Sets the expression of the number of molecules or atoms of the specified species entering the top face of the simulation domain per unit time and area. This parameter operates as follows: <ul style="list-style-type: none">• If specified when reading data from an HPEM file, flux overwrites the flux specified by the HPEM file.• If specified when defining the sum of previously defined distributions, flux overwrites the total flux.• If specified when using the Benoit-Cattin or Edelberg–Aydil plasma sheath model, flux overwrites the flux computed from the plasma sheath model.• If specified when importing an energy and angular distribution from an EAD file, flux overwrites the flux specified in the EAD file.• If specified when defining the distribution from a plasma model, flux overwrites the flux computed by the plasma model.	none Time-dependent expression, evaluating to [0, ∞[mol/s/m ²
hpem_file	Character	Sets the name of the file containing the output of the PCMC module of the HPEM simulator, that is, the energy and angular distributions.	none	none
hpem_material_index	Number	Sets the index of the HPEM material where the HPEM index species was measured.	none	none

Chapter 6: Input Commands

define_species_distribution

Table 52 Parameters of define_species_distribution command (Continued)

Parameter	Type	Description	Default [Range]	Unit
hpem_species	Character	Sets the name of the HPEM species that the angular distributions must read. If omitted, the HPEM species with the name provided by the species parameter will be used.	none	none
integration_energy_max	Number	Sets the maximum energy to take into account when integrating an energy and angular distribution.	none [0, ∞[eV
integration_energy_min	Number	Sets the minimum energy to take into account when integrating an energy and angular distribution.	none]0, ∞[eV
ion_bulk_density	Number	Sets the number of ions per volume in the plasma bulk. This parameter applies only if type=ccp_sheath or type=icp_sheath.	none]0, ∞[1/cm ³
ion_mass	Number	Sets the mass of the ion species in the plasma. This parameter can be used for either type=ccp_sheath or type=icp_sheath.	none]0, ∞[g
ion_temperature	Number	Sets the ion temperature at the sheath edge. This parameter applies only if type=ccp_sheath or type=icp_sheath.	none]0, ∞[eV
name	Character	Sets the name of the energy and angular distribution.	none	none
orientation	Vector	<p>Sets the orientation of the center point of the source distribution. The orientation is specified by a pair of angles (θ, ϕ), where:</p> <ul style="list-style-type: none"> $\theta \in [0, 90]$ denotes the polar angle ($\theta = 0$ corresponds to the negative z-axis). $\phi \in [-180, 180]$ denotes the azimuth angle ($\phi = 0$ corresponds to the x-axis of the simulation coordinate system). <p>See Figure 26 on page 301.</p>	(0, 0) Time-dependent expression, evaluating to [0, 90] × [-180, 180]	degree

Chapter 6: Input Commands

define_species_distribution

Table 52 Parameters of define_species_distribution command (Continued)

Parameter	Type	Description	Default [Range]	Unit
rf_current	Number	Sets the amplitude of the sinusoidal current applied at the discharge. This parameter can be used only for type=icp_sheath and if rf_power is not specified.	none]0, ∞[A
rf_frequency	Number	Sets the applied radio frequency of the plasma source. This parameter applies only if type=ccp_sheath or type=icp_sheath.	none]0, ∞[Hz
rf_power	Number	Sets the amplitude of the sinusoidal current applied at the discharge. This parameter applies only if type=icp_sheath and if rf_current is not specified.	none]0, ∞[W
rf_voltage	Number	Sets the total voltage across the discharge. Alternatively, use ac_voltage and dc_voltage to specify the sheath voltage. This parameter applies only if type=ccp_sheath.	none]0, ∞[V
rotation	Number	Sets an azimuthal rotation of the elliptical distribution with respect to the simulation coordinate system. A rotation of 0° corresponds to an ellipse with exponent_x aligned in the x-direction and exponent_y aligned in the y-direction.	0 Time-dependent expression, evaluating to [-180, 180]	degree
sampling_method	Character	Sets the method used to sample the species distribution. Options are: <ul style="list-style-type: none"> • inverse_cdf • rejection This parameter applies only if type=ead_file.	inverse_cdf	none

Chapter 6: Input Commands

`define_species_distribution`

Table 52 Parameters of `define_species_distribution` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
<code>sampling_time_step</code>	Number	<p>Sets the smallest time step used to sample the time-dependent expression of <code>flux</code>.</p> <p>Note: This parameter is mandatory when the value of the <code>flux</code> parameter is a time-dependent expression. An expression is considered time dependent if it contains the <code>t</code> string (see Syntax for Expressions on page 305). Otherwise, it is time independent.</p>	none	minute
<code>solution</code>	Character	<p>Sets the name of the solution of the plasma model, specified in the <code>solve_reactor</code> command. It can be specified only if <code>type=plasma</code>. You can specify either <code>solution</code> or <code>solution_file</code>.</p>	none	none
<code>solution_file</code>	Character	<p>Sets the name of the file containing the solution of the plasma model. It can be specified only if <code>type=plasma</code>. You can specify either <code>solution</code> or <code>solution_file</code>.</p>	none	none
<code>species</code>	Character	<p>Sets the name of the species for which the distribution is defined. If the distribution is to be defined from the solution of a plasma model (<code>type=plasma</code>), the name must already exist as defined in the <code>add_species</code> command of the plasma model.</p>	none	none
<code>table</code>	Vector	<p>Sets the tabular format of the distribution. The same format must be used as for the <code>table</code> parameter of the <code>define_iad</code> command (see define_iad on page 237).</p>	none	none

Chapter 6: Input Commands

define_species_distribution

Table 52 Parameters of define_species_distribution command (Continued)

Parameter	Type	Description	Default [Range]	Unit
type	Character	Options are: <ul style="list-style-type: none"> • azimuthal: Defines an azimuth-dependent distribution. See Figure 27 on page 302. • ccp_sheath: Specifies the use of the Benoit-Cattin plasma sheath model for defining the species distribution. • ead_file: Specifies the import of an EAD file. • icp_sheath: Specifies the use of the Edelberg-Aydil plasma sheath model for defining the species distribution. • plasma: Specifies that the distribution comes from the solution of a plasma model. • sum: Specifies that the listed species distributions are combined into a single species distribution. 	none	none
unidirectional	Boolean	Specifies whether all the molecules or atoms are distributed along the vertical direction.	false	none

Description

To ease the reuse of measured or simulated data, the definition of species distributions is not part of any model. Species distributions are bound to reaction models when a machine using a reaction model is defined (see [define_etch_machine on page 209](#)).

Examples: Analytic EADs

For example, the following command specifies that atoms of species Ar travel in the reactor along the vertical direction and are energy independent. The flux of Ar atoms entering the reactor is set to 10^{-4} mol/s/m²:

```
define_species_distribution flux=1e-4 name=my_dist species=Ar \
    unidirectional=true
```

Chapter 6: Input Commands

```
define_species_distribution
```

An energy-independent angular distribution according to [Equation 6 on page 49](#) with $m = 1000$ can be set to species Ar with the following command:

```
define_species_distribution exponent=1000 flux=1e-4 name=my_dist \
    species=Ar
```

To specify an energy and angular distribution describing molecules of species F2 moving isotropically in the reactor and with energy uniformly distributed between 10 eV and 20 eV, use the command:

```
define_species_distribution exponent=1 flux=1e-4 name=my_dist \
    species=F2 energy_min=10 energy_max=20
```

[Figure 26](#) illustrates the definition of `orientation={theta phi}`, where `theta` defines the polar angle, and `phi` defines the azimuth angle of the central axis of the ion angle distribution; `theta=0` corresponds to the z-axis and `phi=0` corresponds to the x-axis of the simulation coordinate system.

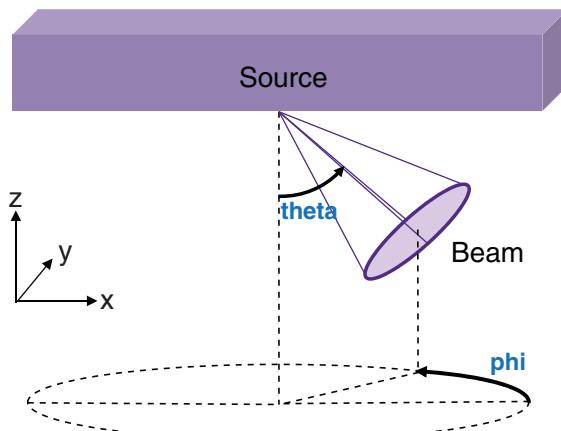
To specify an energy and angular distribution describing molecules of species Ar+ with a polar angle spread of 10° and a bimodal energy distribution (with peaks at 10 eV and 100 eV with heights 10 and 20, and with a minimum at 50 eV with height 1), use:

```
define_species_distribution angle_spread=10 flux=1e-4 \
    name=my_dist species=Ar+ \
    energy_min=10 energy_max=100 \
    energy_min_weight=10 energy_max_weight=20 \
    energy_center=50 energy_center_weight=1
```

Note:

The energy weights are not absolute flux values, since the distribution is normalized to the total flux, given by the parameter `flux`.

Figure 26 Definition of orientation={theta phi}

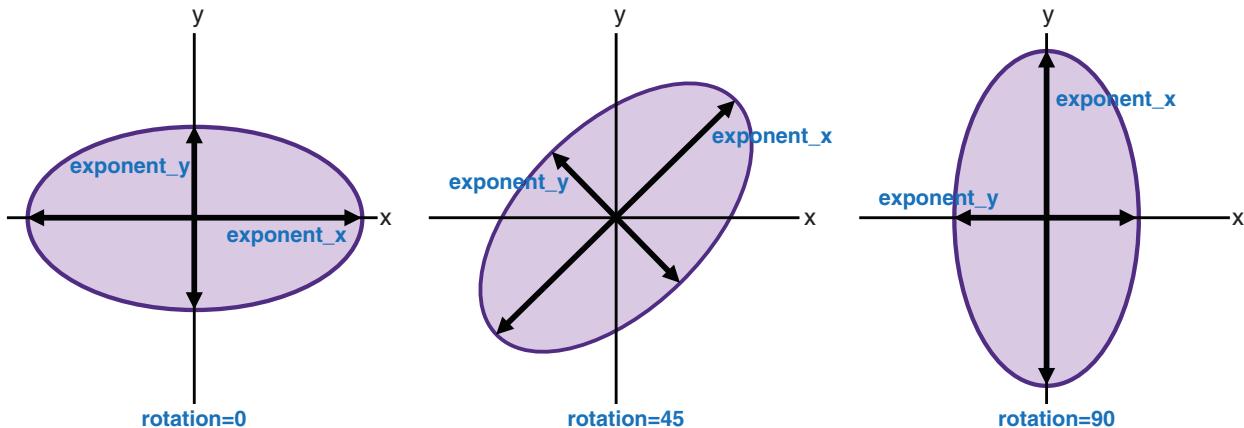


Chapter 6: Input Commands

```
define_species_distribution
```

Figure 27 illustrates the definition of `type=azimuthal`, the azimuth-dependent species distribution.

Figure 27 Definition of type=azimuthal



The flux distribution is shown at a fixed height z (top view), such that the isolines of the distribution take the shape of an ellipse. The `exponent_x` and `exponent_y` parameters define the exponents of the angle distribution along the two principal axes of the ellipse, respectively. The optional parameter `rotation` can be used to change the azimuthal orientation of the ellipse in the xy plane. If `rotation=0`, `exponent_x` corresponds to the x -axis and `exponent_y` corresponds to the y -axis of the simulation coordinate system.

Examples: HPEM Simulator

An energy and angular distribution computed by the PCMC module of the HPEM simulator for species `AR^` and stored in a file named `pcmc.prof` can be imported with the following command:

```
define_species_distribution name=my_dist species=AR^ \
    hpeem_file=pcmc.prof
```

Since the `flux` parameter is not specified, the flux of species `AR^` used in the simulation of a reaction model will be determined from the specified HPEM output.

When the `flux` parameter is specified, its value is used as the flux of the species distribution read from the HPEM output. For example, the following command defines an energy and angular distribution for species `AR^` as computed by the HPEM simulator, but with the flux equal to 10^{-3} mol/s/m²:

```
define_species_distribution name=my_dist species=AR^ flux=1e-3 \
    hpeem_file=pcmc.prof
```

Chapter 6: Input Commands

define_species_distribution

When parameters `energy_min` and `energy_max` are not specified, all the energy levels available in the HPEM output are imported. You can limit the imported energy levels to those included in the range specified by parameters `energy_min` and `energy_max`.

For example, the following command imports only the energy levels between 100 eV and 120 eV:

```
define_species_distribution name=my_dist species=AR^ \
    hpem_file=pcmc.prof energy_min=100 energy_max=120
```

Data computed by the HPEM simulator for one species can be used to define the distributions of multiple reaction model species. For example, the following commands define the energy and angular distributions of the reaction model species `AR_high_energy` and `AR_low_energy` using data computed by the HPEM simulator for the HPEM species named `AR^`:

```
define_species_distribution name=my_dist species=AR_high_energy \
    hpem_species=AR^ hpem_file=pcmc.prof energy_min=150 energy_max=200

define_species_distribution name=my_dist species=AR_low_energy \
    hpem_species=AR^ hpem_file=pcmc.prof energy_min=50 energy_max=100
```

You can also define an energy-independent angular distribution from an energy- and angular-dependent distribution computed by the HPEM simulator. For example:

```
define_species_distribution name=my_dist species=AR^ \
    hpem_file=pcmc.prof integration_energy_min=150 \
    integration_energy_max=200
```

This command defines an energy-independent angular distribution for species `AR^` obtained as the integral over the energy of the data specified for this species in the `pcmc.prof` file. The parameters `integration_energy_min` and `integration_energy_max` specify the integration range.

Examples: Benoit-Cattin Plasma Sheath Model

An energy and angular distribution computed by the Benoit-Cattin model for species Ar with ion mass $m_i = 6.63 \cdot 10^{-23}$ g, ion temperature $T_i = 0.33$ eV, and ion bulk density $n_i = 2.8 \cdot 10^9$ cm $^{-3}$, moving in a CCP discharge with electron temperature $T_e = 3.3$ eV, applied RF voltage $V_{rf} = 500$ V, and $f_{rf} = 80$ MHz can be specified with the following command:

```
define_species_distribution name=my_dist species=Ar type=ccp_sheath \
    electron_temperature=3.3 ion_bulk_density=2.8e9 ion_mass=6.63e-23 \
    ion_temperature=0.33 rf_frequency=80e6 rf_voltage=500
```

Chapter 6: Input Commands

```
define_species_distribution
```

Instead of specifying the applied RF voltage V_{rf} , you can define the sheath voltage $V_{rf}(t) = V_{DC} + V_{AC} \sin(2\pi f_{rf} t)$ by defining the AC sheath voltage V_{AC} and the DC sheath voltage V_{DC} . For example, for $V_{AC} = 415$ V and $V_{DC} = 207.5$ V, specify the following command:

```
define_species_distribution name=my_dist species=Ar type=ccp_sheath \
    ac_voltage=415 dc_voltage=207.5 electron_temperature=3.3 \
    ion_bulk_density=2.8e9 ion_mass=6.63e-23 ion_temperature=0.33 \
    rf_frequency=80e6
```

Examples: Edelberg–Aydil Plasma Sheath Model

An energy and angular distribution computed by the Edelberg–Aydil model for species Ar with ion mass $m_i = 6.63 \cdot 10^{-23}$ g, ion temperature $T_i = 0.33$ eV, and ion bulk density $n_i = 2.8 \cdot 10^9 \text{ cm}^{-3}$, moving in an ICP discharge with electrode area $A = 325 \text{ cm}^2$, electron temperature $T_e = 3.3$ eV, applied RF current $I_{rf} = 6.175$ A, and $f_{rf} = 80$ MHz can be specified with the following command:

```
define_species_distribution name=my_dist species=Ar type=icp_sheath \
    electrode_area=325 electron_temperature=3.3 \
    ion_bulk_density=2.8e9 ion_mass=6.63e-23 ion_temperature=0.33 \
    rf_current=6.175 rf_frequency=80e6
```

Instead of specifying the applied RF current I_{rf} , you can define the applied RF bias power P_{rf} . For example, for $P_{rf} = 80$ W, specify the following command:

```
define_species_distribution name=my_dist species=Ar type=icp_sheath \
    electrode_area=325 electron_temperature=3.3 \
    ion_bulk_density=2.8e9 ion_mass=6.63e-23 ion_temperature=0.33 \
    rf_frequency=80e6 rf_power=80
```

Examples: Importing an EAD File

To import a distribution function from an EAD file `my_sd.txt`, use the following command:

```
define_species_distribution type=ead_file name=sd species=Ar \
    file=my_sd.txt
```

Examples: Combining Species Distributions

To combine species distributions into a single species distribution:

```
define_species_distribution name=sd1 species=CF2 exponent=1 flux=0.001
define_species_distribution name=sd2 species=CF3 exponent=5 flux=0.05
define_species_distribution name=sd_sum species=CFx type=sum \
    distributions= {{sd1 CF2} {sd2 CF3}}
define_species_distribution name=sd_sum2 species=CFx type=sum \
    distributions= {{sd1 CF2} {sd2 CF3}} flux=1e-3
```

Chapter 6: Input Commands

define_species_distribution

Examples: Plasma Model Distributions

```
define_species_distribution name=sd species=Ar+ type=plasma \
    solution=soll
```

```
define_species_distribution name=sd species=Ar+ type=plasma \
    solution=soll flux=1e-3
```

Syntax for Expressions

For details about time-dependent expressions, see [Syntax for Expressions on page 131](#).

In addition, you can specify the unit of the numeric value to which an expression evaluates. If `expr` denotes an expression, then this can be accomplished using the following syntax:

```
(expr)<expression_unit>
```

If `expr` is a number, then you can omit the parentheses enclosing it. The possible values for `<expression_unit>` depend on the parameter to which the expression is assigned:

- When assigned to the `flux` parameter, `<expression_unit>` can be one of the following:
 - mol/m²/s (default)
 - mol/cm²/s
 - mol/m²/min
 - mol/cm²/min
- When assigned to the `energy_max` or `energy_min` parameter, `<expression_unit>` can be either eV (default) or J.

Examples

The following command defines a square pulse-shaped time-dependent flux with a period of 1 minute and a duty cycle of 0.8 for a distribution whose angular-dependent part is described by [Equation 6 on page 49](#). The smallest time step used to sample the defined time-dependent flux is set to 0.1 minutes:

```
define_species_distribution name=sd species=Ar exponent=1 \
    flux="square_pulse(t, 1e-3, 0.0, 1.0, 0.8)" sampling_time_step=0.1
```

The following command defines a square pulse-shaped time-dependent flux with a period of 1 s and a duty cycle of 0.8 for a distribution whose angular-dependent part is described by [Equation 6](#). The smallest time step used to sample the defined time-dependent flux is set to 0.1 minutes:

```
define_species_distribution name=sd species=Ar exponent=1 \
    flux="square_pulse(t<s>, 1e-3, 0.0, 1.0, 0.8)" \
    sampling_time_step=0.1
```

Chapter 6: Input Commands

define_species_properties

The following command defines a species distribution whose angular-dependent part is described by [Equation 6 on page 49](#), with:

- A flux of 1 mole per cm² per second
- A square pulse-shaped time-dependent exponent
- A uniform energy distribution between 10 eV and a time-dependent upper energy limit expressed in joules
- The smallest time step used to sample all time-dependent parameters of the distribution is set to 0.1 minutes

```
define_species_distribution name=sd species=Ar \
    exponent="square_pulse(t, 1000, 1, 2.0, 0.75)" \
    flux=1<mol/cm2/s> energy_min=10 \
    energy_max="(2e-19*square_pulse(t,50,30,1.0,0.5))<J>" \
    sampling_time_step=0.1
```

define_species_properties

This command defines the properties of a species used in a reaction model.

When not simulating charge-up, it is not mandatory to issue this command for all the species involved in a simulation. The species whose properties are not set with the `define_species_properties` command are reemitted when they interact with the structure and no reaction occurs.

Syntax

To specify the default event to be executed when a flux species hits a structure and no reaction occurs:

```
define_species_properties name=<c> species=<c> [default_event=<c>]
```

When simulating charge-up, you must specify the mass and charge of every flux species:

```
define_species_properties charge=<n> mass=<n> name=<c> species=<c> \
    [default_event=<c>]
```

Note:

The charge-up feature is switched on automatically when a charged flux species is defined by this command.

In addition, when simulating charge-up, the electric properties of each material in the structure must be specified:

```
define_species_properties name=<c> species=<c> \
    [conductivity=<n>] [permittivity=<n>]
```

Chapter 6: Input Commands

define_species_properties

Note:

Any material with a nonzero conductivity is treated as an ideal conductor, that is, the value of the conductivity is not relevant for the electric field computation. A material with zero conductivity is treated automatically as a dielectric with the specified permittivity.

Table 53 Parameters of *define_species_properties* command

Parameter	Type	Description	Default	Unit
charge	Number	Sets the charge of the species in units of the unit charge.	0 ($-\infty, \infty$)	1
conductivity	Number	Sets the electrical conductivity of the species. By default, the species is treated as a dielectric (<code>conductivity=0</code>).	0 [0, ∞)	$(\Omega m)^{-1}$
default_event	Character	Sets what happens to a molecule or an atom of the specified species when it interacts with the structure and no reaction occurs. Options are: <ul style="list-style-type: none">• discard• reemit	reemit	none
mass	Number	Sets the mass of the species.	none	amu (atomic mass units)
name	Character	Sets the name used to refer to the species properties in a <code>define_etch_machine</code> command.	none	none
permittivity	Number	Sets the relative permittivity (ϵ_r) of the medium. A species is treated as a dielectric only if it is nonconducting, that is, the value of this parameter is considered only if <code>conductivity=0</code> .	1 (0, ∞)	1
species	Character	Sets the name of the species for which the properties are defined.	none	none

Chapter 6: Input Commands

define_species_properties

Examples

Define positively charged argon ions with a mass of 40 amu:

```
define_species_properties name=sp species=I mass=40 charge=1
```

Define a dielectric material with a relative electric permittivity of 11.7:

```
define_species_properties name=sp species=Silicon permittivity=11.7
```

Define a conducting material:

```
define_species_properties name=sp species=Aluminum \
conductivity=3.5e7
```

Chapter 6: Input Commands

define_structure

define_structure

This command defines a new structure. The structure can be loaded from a TDR boundary file, or it can be a cuboid (3D) or a rectangle (2D). For a PMC simulation, the structure can also be loaded from a PMC file or from a TDR file containing a 3D GC structure.

Note:

Multiple structures can be defined in the same input file, even of different dimensions. This allows you to mix 2D and 3D simulations in the same input file (see [Simulating Process Steps on 2D and 3D Structures on page 31](#)).

Syntax

To load a structure from a TDR file containing a boundary:

```
define_structure file=<c> [conformalize=<b>] [name=<c>] \
    [read_fields=<b>] [slice_angle=<n>] [tdr_geometry=<c>]
```

To load a structure from a TDR file containing a 3D GC structure:

```
define_structure file=<c> [name=<c>] [read_fields=<b>] \
    [slice_angle=<n>] [tdr_geometry=<c>]
```

To create a 3D cuboid or 2D rectangular structure from the beginning:

```
define_structure material=<c> point_max=<v> point_min=<v> \
    [flat_orientation=<v>] [name=<c>] [read_fields=<b>] [region=<c>] \
    [slice_angle=<n>] [vertical_orientation=<v>]
```

To load a PMC structure from a PMC file:

```
define_structure pmc_file=<c> \
    [initial_structure_name=<c>] [name=<c>] [read_fields=<b>] \
    [slice_angle=<n>]
```

Table 54 Parameters of *define_structure* command

Parameter	Type	Description	Default [Range]	Unit
conformalize	Boolean	If true, an operation is performed to make input boundaries conformal.	false	none
file	Character	Sets the path of the TDR boundary file containing the structure.	none	none

Chapter 6: Input Commands

define_structure

Table 54 Parameters of define_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
flat_orientation	Vector	Sets the Miller index of the direction perpendicular to the wafer flat (y-axis of the wafer coordinate system) for the newly created region.	{1 1 0}	none
initial_structure_name	Character	Sets a new name for the initial structure loaded from a PMC file and that can be used as the body of Boolean operations.	none	none
material	Character	Sets the material of the initial cube.	none	none
name	Character	Sets the name of the structure.	default_structure	none
pmc_file	Character	Sets the file name or path to a PMC file from which to load the initial data structure.	none	none
point_max	Vector	Sets the maximum corner point of the initial cube.	none	µm
point_min	Vector	Sets the minimum corner point of the initial cube.	none	µm
read_fields	Boolean	If true, then all vertex scalar data is read from the mixed-element geometry of the TDR file and is available to use in the field_names parameter of the define_material_replacement command (see define_material_replacement on page 247).	false	none
region	Character	Sets the name of the created region if the structure is not loaded from a file.	none	none

Chapter 6: Input Commands

define_structure

Table 54 Parameters of define_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
slice_angle	Number	Sets the rotation of the x-axis of the simulation coordinate system with respect to the y-axis of the wafer coordinate system. Note: This parameter does not support the radian measurement unit.	-90 [-180, 180]	degree
tdr_geometry	Character	Sets the name of a TDR geometry in the TDR file. Only this specified geometry is loaded. The name must exist in the file specified by file.	none	none
vertical_orientation	Vector	Sets the Miller index of the direction perpendicular to the wafer surface (z-axis of the wafer coordinate system) for the newly created region.	{0 0 1}	none

Note:

To create a rectangle, the vector values of point_min and point_max must have two components. To create a cuboid, three components must be specified.

The Miller indices for a plane are usually written as (hkl), and the set of equivalent planes is written as {hkl}. A crystal direction is usually written as [hkl], and the set of equivalent directions is written as <hkl>. The value assigned to the parameters flat_orientation and vertical_orientation is a Tcl list of indices representing a crystal direction. The braces in the specified value are required to form a Tcl list and do not indicate a set of equivalent planes.

Examples

Example of conformal input:

```
define_structure file=input.tdr
```

This command imports a structure from the file input.tdr. The input is assumed to have conformal boundaries.

Chapter 6: Input Commands

define_structure

Note:

If multiple geometries are present and the `tdr_geometry` parameter is not specified, the last valid TDR boundary in the TDR file is selected automatically.

Example of loading a specific TDR geometry from a file:

```
define_structure file=input.tdr tdr_geometry="geometry_0"
```

This command imports the TDR geometry `geometry_0` from the file `input.tdr`, and it ignores all other geometries in the file. The input is assumed to have conformal boundaries.

Example of nonconformal input:

```
define_structure file=input.tdr conformalize=true \
    name=conformalized_input
```

This command imports a structure from the file `input.tdr`. An operation on the structure is performed to create conformal boundaries between regions. The name of the structure is `conformalized_input`.

Example of defining a 3D cuboid structure from the beginning:

```
define_structure material=Silicon point_max={1 1 1} \
    point_min={0 0 0} region="substrate"
```

This command creates an initial cuboid structure, with silicon as the material and `substrate` as the name of the region.

Example of defining a 2D rectangular structure from the beginning:

```
define_structure material=Silicon point_max={1 1} point_min={0 0}
```

This command creates an initial 2D rectangular structure, with silicon as the material.

Loading a PMC Structure

A PMC structure is loaded from a PMC file with the `define_structure` command when the parameter `pmc_file` is used.

When the PMC file contains an initial structure, it will be loaded. The parameter `initial_structure_name` can be used to rename the initial structure to the specified name. See [Saving PMC Structures on page 478](#).

Note:

A PMC file is not a TDR file and cannot be visualized with Sentaurus Visual. It can only be loaded in to Sentaurus Topography 3D.

Chapter 6: Input Commands

define_volumetric_species_distribution

define_volumetric_species_distribution

This command defines the distribution of a volumetric source.

Note:

The defined volumetric distribution is energy-dependent only if `energy_max` and `energy_min` are specified.

Syntax

```
define_volumetric_species_distribution name=<c> \
    generation_rate=<c> species=<c> \
    [energy_max=<c> energy_min=<c>] [point_max=<v>] [point_min=<v>] \
    [sampling_time_step=<n>]
```

Table 55 Parameters of `define_volumetric_species_distribution` command

Parameter	Type	Description	Default [Range]	Unit
<code>energy_max</code>	Character	Sets the expression of the maximum energy of the specified uniform energy distribution.	0 Time-dependent expression, evaluating to $[0, \infty[$ and not smaller than the value of the time-dependent expression <code>energy_min</code> at the same time	eV
<code>energy_min</code>	Character	Sets the expression of the minimum energy of the specified uniform energy distribution.	none Time-dependent expression, evaluating to $[0, \infty[$ and not larger than the value of the time-dependent expression <code>energy_max</code> at the same time	eV

Chapter 6: Input Commands

define_volumetric_species_distribution

Table 55 Parameters of define_volumetric_species_distribution command (Continued)

Parameter	Type	Description	Default [Range]	Unit
generation_rate	Character	Sets the time- and position-dependent expression for the number of molecules or atoms of the specified species generated per unit time and volume. For details about time- and position-dependent expressions, see Syntax for Expressions on page 131 .	none Time-dependent expression, evaluating to $[0, \infty[$	mol/m ³ /s
name	Character	Sets the name of the volumetric species distribution.	none	none
point_max	Vector	Sets the requested maximum corner of the cuboid domain where particles will be generated. The generation rate of the particles will be zero outside of this cuboid domain. The actual maximum corner of the cuboid domain is obtained by snapping the point_max value to a grid point of the PMC structure used during the simulation	{ $\infty \infty \infty$ }	μm
point_min	Vector	Sets the requested minimum corner of the cuboid domain where particles will be generated. The generation rate of the particles will be zero outside of this cuboid domain. The actual minimum corner of the cuboid domain is obtained by snapping the point_min value to a grid point of the PMC structure used during the simulation.	{ $-\infty -\infty -\infty$ }	μm

Chapter 6: Input Commands

define_volumetric_species_distribution

Table 55 Parameters of *define_volumetric_species_distribution* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
sampling_time_step	Number	<p>Sets the smallest time step used to sample the time-dependent expression of generation_rate, energy_min, and energy_max.</p> <p>Note: This parameter is mandatory when the value of generation_rate, energy_min, or energy_max is a time-dependent expression. An expression is considered time dependent if it contains the t string. Otherwise, it is time independent.</p>	none	minute
species	Character	Sets the name of the species for which the distribution is defined.	none	none

Chapter 6: Input Commands

define_yield

define_yield

This command specifies the properties of a new yield function that depends on the species of the incoming particles, its energy, and the target surface material. When defining a yield function for a machine using an RFM model, the value of the `species` parameter must be the name of an ion flux of that model.

As for ion distributions, Sentaurus Topography 3D supports collections of yield functions (see *Examples*).

This yield function can be used in RFM or reaction models that take sputtering into account.

Syntax

Define energy-dependent or energy-independent yield functions:

```
define_yield energy=<n> material=<c> name=<c> normalized=<b> \
    species=<c> table=<v> [angle_unit=<c>]

define_yield energy=<n> material=<c> name=<c> s1=<n> s2=<n> \
    species=<c> \
    [threshold_energy=<n>] [yield_at_zero=<n>]

define_yield energy=<n> material=<c> name=<c> species=<c> \
    sputtering=false

define_yield energy=<n> material=<c> name=<c> species=<c> \
    theta_max=<n> yield_max=<n> \
    [threshold_energy=<n>] [yield_at_zero=<n>]
```

Define two-part yield function:

```
define_yield (energy=<n> | threshold_energy=<n>) material=<c> \
    name=<c> sigma1=<n> sigma2=<n> species=<c> \
    theta_max=<n> yield_at_90=<n> \
    yield_at_zero=<n> yield_max=<n>
```

Define yield function using expression:

```
define_yield expression=<c> material=<c> name=<c> [energy=<n>]
```

Chapter 6: Input Commands

`define_yield`

Table 56 Parameters of `define_yield` command

Parameter	Type	Description	Default [Range]	Unit
angle_unit	Character	Sets the unit of the angles listed in the table parameter. Options are: <ul style="list-style-type: none"> • deg • rad 	deg	none
energy	Number	Sets the energy for which the yield function is defined. When <code>energy=0</code> , the yield function is valid at all energies, that is, it is energy independent.	none [0, ∞[eV
expression	Character	Sets the formula used to calculate the yield function. When you specify <code>expression</code> , the yield function is defined from the user-specified expression, and the <code>energy</code> parameter is optional. For details about energy- and angle-dependent expressions, see Syntax for Expressions on page 131 . There are the following cases: <ul style="list-style-type: none"> • When <code>energy</code> is not specified, the specified expression defines an energy-dependent yield function that is valid for all energy levels. • When <code>energy=0</code>, the specified expression defines an energy-independent yield function. • When <code>energy</code> is set to a positive value, the specified expression defines the angle-dependent part of the yield function for the given energy level. 	none	none
material	Character	Sets the name of the surface material for which the yield function is defined.	none	none
name	Character	Sets the name used to reference the yield function in a <code>define_deposit_machine</code> or <code>define_etch_machine</code> command.	none	none

Chapter 6: Input Commands

`define_yield`

Table 56 Parameters of `define_yield` command (Continued)

Parameter	Type	Description	Default [Range]	Unit
normalized	Boolean	Specifies whether the value of the yield function at 0° must be 1 (<code>normalized=true</code>) or whether it can be any nonnegative value (<code>normalized=false</code>).	none	none
s1	Number	Sets the first sputter coefficient.	none	none
s2	Number	Sets the second sputter coefficient.	none	none
sigma1	Number	Sets the angular spread for the lower part of a two-part yield model.	none [0, 90]	degree
sigma2	Number	Sets the angular spread for the upper part of a two-part yield model.	none [0, 90]	degree
species	Character	Sets the name of the flux species for which the yield function is defined.	none	none
sputtering	Boolean	When <code>sputtering=false</code> for a certain species and material pair, sputtering by particles of that species from that material is deactivated completely. The only option is <code>sputtering=false</code> .	none	none
table	Vector	Sets the tabular format of the yield function. Values of the yield function are specified in a tabular form similar to the ion angular distribution (IAD), where the first column represents angles and the second column is the value of the yield function. The restrictions on the format of the table are the same as for the <code>define_iad</code> command (see define_iad on page 237). In addition, if <code>normalized=true</code> , then the value of the yield function for 0° must be exactly 1.	none	none
theta_max	Number	Sets the angle at which the yield function has its maximum.	none [0, 90]	degree

Chapter 6: Input Commands

define_yield

Table 56 Parameters of define_yield command (Continued)

Parameter	Type	Description	Default [Range]	Unit
threshold_energy	Number	When this parameter is specified, the yield function becomes energy dependent (see Equation 84). This parameter cannot be used with the energy, expression, or table parameters.	0 [0, ∞[eV
yield_at_90	Number	Sets the value of the yield function at the grazing incidence.	0 [0, ∞[none
yield_at_zero	Number	Sets the value of the yield function at normal incidence.	1 [0, ∞[(when s1 and s2 are used) (0, ∞[(when theta_max and yield_max are used))	none
yield_max	Number	Sets the maximum value of the yield function.	none [1, ∞[none

Description

You can define yield functions distinctly in each combination of species and material. Yield functions are naturally angle dependent. There are different ways to handle energy dependence of yield:

- **Energy independent:** Set `energy=0` to specify that a yield function is valid at all energies.
- **Linear interpolation:** Specify a yield function for at least two different energy levels. The lowest energy specified is the threshold energy for the yield function. For ion energy values that have not been specified, the value of the yield function is calculated by linearly interpolating the parameter values for analytic yield functions or the specified values for tabular yield functions.
- **Energy dependent:** Specify the `threshold_energy` parameter, which cannot be used with the `energy` parameter, so that the built-in yield function becomes energy dependent as follows:

$$\gamma(\theta, E) = (\sqrt{E} - \sqrt{E_{\text{th}}})\gamma(\theta) \quad (84)$$

Chapter 6: Input Commands

define_yield

where E_{th} represents the value of the threshold_energy parameter.

- **Energy-dependent expression:** Specify expressions containing E to define a custom energy-dependent yield function.

Note:

Energy-dependent yield functions can be used only with reaction models, not with level set-based models.

When you specify the parameters s_1 and s_2 , the yield function is defined as:

$$\gamma(\theta) = \gamma_0(s_1 \cos \theta + s_2 \cos^2 \theta + (1 - s_1 - s_2) \cos^4 \theta) \quad (85)$$

where γ_0 denotes the value of the yield_at_zero parameter.

When you specify the parameters theta_max and yield_max, the yield function is defined as [15]:

$$\gamma(\theta) = \gamma_0 \cos^{-f} \theta \exp(-s(1/\cos \theta - 1)) \quad (86)$$

where $f = -\ln(\gamma_{\text{max}}/\gamma_0)/(\ln(\cos(\theta_{\text{max}})) + 1 - \cos \theta_{\text{max}})$ and $s = f \cos \theta_{\text{max}}$.

When you specify sigma1, sigma2, theta_max, yield_at_zero, yield_at_90, and yield_max, you obtain a two-part yield function, which is given by:

$$\gamma(\theta) = \begin{cases} \gamma_l(\theta), & \theta < \theta_{\text{max}} \\ \gamma_u(\theta), & \theta \geq \theta_{\text{max}} \end{cases} \quad (87)$$

where:

$$\gamma_l(\theta) = \gamma_0 + (\gamma_{\text{max}} - \gamma_0) \frac{\left\{ e^{-\frac{1}{2}\left(\frac{\theta_{\text{max}} - \theta}{\sigma_1}\right)^2} - e^{-\frac{1}{2}\left(\frac{\theta_{\text{max}}}{\sigma_1}\right)^2} \right\}}{\left\{ 1 - e^{-\frac{1}{2}\left(\frac{\theta_{\text{max}}}{\sigma_1}\right)^2} \right\}} \quad (88)$$

$$\gamma_u(\theta) = \gamma_{90} + (\gamma_{\text{max}} - \gamma_{90}) \frac{\left\{ e^{-\frac{1}{2}\left(\frac{\theta - \theta_{\text{max}}}{\sigma_2}\right)^2} - e^{-\frac{1}{2}\left(\frac{90 - \theta_{\text{max}}}{\sigma_2}\right)^2} \right\}}{\left\{ 1 - e^{-\frac{1}{2}\left(\frac{90 - \theta_{\text{max}}}{\sigma_2}\right)^2} \right\}} \quad (89)$$

Chapter 6: Input Commands

define_yield

and where:

- σ_1 , σ_2 represent the values of the `sigma1` parameter and `sigma2` parameter, respectively.
- θ_{\max} represents the value of the `theta_max` parameter.
- γ_{90} represents the value of the `yield_at_90` parameter.
- γ_0 represents the value of the `yield_at_zero` parameter.
- γ_{\max} represents the value of the `yield_max` parameter.
- The angles θ and θ_{\max} are measured in degrees.

Examples

Define an energy-independent yield function for Silicon sputtering from species Ar that is proportional to the cosine of the angle between the direction of the impinging particle and the surface normal at the collision point:

```
define_yield name=my_yield species=Ar material=Silicon energy=0 \
    expression="0.3*cos(theta)"
```

Define an energy-dependent yield function for Silicon sputtering from species Ar that is equal to zero when the energy of the impinging particle is smaller than 85 eV and that is proportional to the cosine of the angle between the direction of the impinging particle and the surface normal at the collision point, otherwise:

```
define_yield name=my_yield species=Ar material=Silicon \
    expression="E >= 85.0 ? 0.3*cos(theta) : 0."
```

Define a collection of energy-independent yield functions called `my_yield` for flux I and several target materials:

```
define_yield name=my_yield species=I energy=0 material=Silicon \
    theta_max=55 yield_max=1.5

define_yield name=my_yield species=I energy=0 material=Oxide s1=6.0 \
    s2=-5.5

define_yield name=my_yield species=I energy=0 material=Nitride \
    theta_max=60 yield_max=1.75

define_yield name=my_yield species=I energy=0 material=Photoresist \
    table=$yield_table
```

Chapter 6: Input Commands

deposit

deposit

This command starts a deposition simulation either by using a machine defined using the `define_deposit_machine` command or by geometrically depositing a shape defined by the `define_shape` command.

Syntax

Deposition simulation on the surface of a structure using a machine:

```
deposit (cycles=<n> | time=<n>) spacing=<v> \
[accuracy=<n>] [cfl=<n>] [comment=<c>] [decimate=<b>] \
[diffusion_flux_error_control_strategy=<c>] \
[diffusion_flux_integration_error=<n>] [engine=<c>] \
[exposed_only=<b>] \
[extraction=<c> (extraction_interval=<n> | \
extraction_times=<v> [extraction_times_unit=<c>]) \
[extraction_file_update_interval=<n>] \
[force_intermediate_extraction=<b>]] \
[file=<c>] [flat_orientation=<v>] \
[integration_error=<n> | integration_samples=<n>] \
[machine=<c>] [max_ion_reflections=<n>] [merge=<b>] \
[min_angle=<n>] [min_dihedral_angle=<n>] \
[min_ion_reflection_probability=<n>] \
[(plot_interval=<n> [compute_first_plot=<b>] \
[compute_last_plot=<b>] | plot_times=<v> [plot_times_unit=<c>]) \
[force_intermediate_plot=<b>] [plot_file_update_interval=<n>] \
[plot_type=<l>] [split_plots=<b>]] \
[random_seed=<n>] [region=<c>] [remove_spurious_parts=<b>] \
[ridge_angle=<n>] [shortest_edge=<n>] \
[stop_plane=<n> | stop_point=<w>] \
[structure=<c>] [surface_accuracy=<n>] [surface_decimate=<b>] \
[surface_min_angle=<n>] [surface_min_dihedral_angle=<n>] \
[surface_ridge_angle=<n>] [surface_shortest_edge=<n>] \
[update_scheme=<c>] \
[variance_reduction=<c>] [vertical_orientation=<v>]
```

Geometric deposition using a shape:

On a boundary structure:

```
deposit material=<c> shape=<c> \
[accuracy=<n>] [comment=<c>] [decimate=<b>] [flat_orientation=<v>] \
[merge=<c>] [min_angle=<n>] [min_dihedral_angle=<n>] [region=<c>] \
[ridge_angle=<n>] [shortest_edge=<n>] [structure=<c>] \
[vertical_orientation=<v>]
```

On a PMC structure:

```
deposit material=<c> shape=<c> [comment=<c>] [structure=<c>]
```

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command

Parameter	Type	Description	Default [Range]	Unit
accuracy	Number	<p>Sets the maximum deviation between the decimated surface of the boundary structure produced by Boolean operations and its original surface.</p> <p>When set to -1, the used value is half of the structure-dependent spacing epsilon¹.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <code>structure</code> parameter denotes a boundary structure. • You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_accuracy=<n></code> $\{-1\} \cup [0, \infty[$	µm
cfl	Number	Sets the Courant–Friedrichs–Lewy (CFL) number [16] for the integration of the surface evolution equation.	0.95 (when using a machine having <code>model=simple</code>), 0.5 (when using a machine not having <code>model=simple</code>) $]0, 1]$	none
comment	Character	Sets a comment for this process step that will be displayed or written to the log file.	empty string	none
compute_first_plot	Boolean	When <code>plot_interval</code> is set, the parameter <code>compute_first_plot</code> specifies whether to produce an intermediate plot at <code>time=0</code> .	true	none
compute_last_plot	Boolean	When <code>plot_interval</code> is set, the parameter <code>compute_last_plot</code> specifies whether to produce an intermediate plot at the end of the simulation.	true	none

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command (Continued)

Parameter	Type	Description	Default [Range]	Unit
cycles	Number	Sets the number of deposition cycles for a machine using the <code>ald</code> model.	none [0, 2147483647]	none
decimate	Boolean	Specifies whether to decimate the boundary structure produced by Boolean operations. It can be specified only if <code>structure</code> denotes a boundary structure.	The value set by <code>let decimate=</code>	none
diffusion_flux_error_control_strategy	Character	Sets which type of error the parameter <code>diffusion_flux_integration_error</code> calculates. Options are: <ul style="list-style-type: none"> • absolute: Absolute error. • relative: Relative error. 	relative	none
diffusion_flux_integration_error	Number	Sets the requested flux error of the diffusing species.	0.05 [0, 1]	none
engine	Character	Sets the engine to use for flux computation. This choice is available only for flux models. Options are: <ul style="list-style-type: none"> • monte_carlo • radiosity 	radiosity	none
exposed_only	Boolean	Specifies whether deposition is possible only on exposed portions of the surface when using the <code>simple</code> model.	false	none
extraction	Character	Sets the name of the group of extractions to run during this process step.	none	none

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command (Continued)

Parameter	Type	Description	Default [Range]	Unit
extraction_file_update_interval	Number	<p>Sets the time interval after which intermediate extraction results are written to file.</p> <p>When set to 0, intermediate extraction results are written to file immediately after they are created.</p> <p>Note: This parameter can be set only when at least one intermediate extraction produces file output.</p>	0 [0, ∞[minute
extraction_interval	Number	Sets the process time interval after which intermediate extractions must be executed.	none]0, ∞[minute
extraction_times	Vector	<p>Sets the times at which to perform intermediate extractions.</p> <p>The given vector must:</p> <ul style="list-style-type: none"> Contain only nonnegative values Be sorted in ascending order Contain no value larger than the value of parameter <code>time</code> <p>If any of these conditions is not met, then an error is issued.</p>	none	none
extraction_times_unit	Character	<p>Sets the unit of the <code>extraction_times</code> values.</p> <p>Options are:</p> <ul style="list-style-type: none"> <code>s</code> (seconds) <code>min</code> (minutes) 	min	none

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command (Continued)

Parameter	Type	Description	Default [Range]	Unit
file	Character	Sets the file name into which intermediate surfaces and surface data should be saved. Used in conjunction with plot_interval only. In the file name, <base_name> denotes the value of the base_name parameter of the let command.	<base_name>.tdr	none
flat_orientation	Vector	Sets the Miller index of the direction perpendicular to the wafer flat (y-axis of the wafer coordinate system) for the newly created region. It can be specified only if structure denotes a boundary structure.	{1 1 0}	none
force_intermediate_extraction	Boolean	Specifies whether to compute intermediate extractions and, if applicable, write them to files even if Sentaurus Topography 3D was started with the command-line option --no_intermediate_extraction.	false	none
force_intermediate_plot	Boolean	Specifies whether to compute intermediate plots and write them to files even if Sentaurus Topography 3D was started with the --no_intermediate_plot command-line option.	false	none
integration_error	Number	Sets the maximum relative discretization error in the direct flux calculation.	0.05]0, 1]	none

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command (Continued)

Parameter	Type	Description	Default [Range]	Unit
integration_samples	Number	<p>Sets the number of integration points in the numeric integration. The meaning of this parameter depends on the value of the engine parameter:</p> <ul style="list-style-type: none"> • When engine=radiosity, integration_samples describes the number of discretization points, per surface element, in the hemisphere where visibility is determined. • When engine=monte_carlo, integration_samples describes the number of particle paths, per surface element, used in the flux integration. <p>Note: This parameter is deprecated for engine=radiosity. Instead, use the parameter integration_error.</p>	1000 (engine=radiosity), 700 (engine=monte_carlo and variance_reduction=M-2016.12), 350 (engine=monte_carlo and variance_reduction=N-2017.09) [1, ∞[none
machine	Character	Sets the name of the machine that is used for the deposition process.	default_machine	none
material	Character	Sets the material to be deposited.	none	none
max_ion_reflections	Number	Sets the maximum number of ion reflections to compute during flux integration. The specified value must be an integer. Only applicable when using RFM models.	1 [1, ∞[none
merge	Boolean	If true, the deposited region merges with the already existing regions of the same material in the structure. It can be specified only if structure denotes a boundary structure.	false	none

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command (Continued)

Parameter	Type	Description	Default [Range]	Unit
min_angle	Number	<p>Sets the smallest angle in the elements of the decimated surface of the boundary structure produced by Boolean operations.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <code>structure</code> parameter denotes a boundary structure. • You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_min_angle=<n></code> <code>[0, 180]</code>	degree
min_dihedral_angle	Number	<p>Sets the smallest dihedral angle between two elements of the decimated surface of the boundary structure produced by Boolean operations that share an edge.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <code>structure</code> parameter denotes a boundary structure. • You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_min_dihedral_angle=<n></code> <code>[0, 180]</code>	degree
min_ion_reflection_probability	Number	Sets the threshold for the reflection probability below which multiple reflection is stopped.	0.001 <code>[0, 1]</code>	none

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command (Continued)

Parameter	Type	Description	Default [Range]	Unit
plot_file_update_interval	Number	Sets the time interval after which intermediate plots are written to file. When set to 0, intermediate plots are written to file immediately after they are created. Note: This parameter can be set only when <code>split_plots=false</code> .	0 [0, ∞[minute
plot_interval	Number	Sets the process time interval after which intermediate exposed surfaces and surface data should be plotted. See TDR Dataset Names for Fluxes on page 337 .	none [0, ∞[minute
plot_times	Vector	Sets the times at which to perform intermediate plots. The given vector must: <ul style="list-style-type: none">• Contain only nonnegative values• Be sorted in ascending order• Contain no value larger than the value of parameter <code>time</code> If any of these conditions is not met, then an error is issued.	none	none
plot_times_unit	Character	Sets the unit of the <code>plot_times</code> values. Options are: <ul style="list-style-type: none">• <code>s</code> (seconds)• <code>min</code> (minutes)	min	none
plot_type	List	Sets the data type to save at the time intervals specified by <code>plot_interval</code> . Options are: <ul style="list-style-type: none">• <code>structure</code>: Saves multiregion boundary.• <code>surface</code>: Saves exposed surface with datasets.	surface	none

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command (Continued)

Parameter	Type	Description	Default [Range]	Unit
random_seed	Number	Sets the seed of the pseudo-random number generator used by the Monte Carlo flux integrator. Note: This parameter can be specified only when engine=monte_carlo.	1 [1, 2147483647]	none
region	Character	Sets the name of the region that is deposited. It can be specified only if structure denotes a boundary structure.	none	none
remove_spurious_parts	Boolean	Specifies whether or not to remove spurious parts after running the process step.	false	none
ridge_angle	Number	Sets the angle used by the decimation algorithm to determine geometric features of the boundary structure produced by Boolean operations. This parameter can be specified only if all of the following conditions are met: <ul style="list-style-type: none"> • The structure parameter denotes a boundary structure. • You specify decimate=true or, if you do not specify decimate, you did not specify let decimate=false. 	The value set by let decimate_ ridge_angle=<n> [0, 180]	degree
shape	Character	Sets the name of the shape that is used for the deposition process.	none	none

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command (Continued)

Parameter	Type	Description	Default [Range]	Unit
shortest_edge	Number	<p>Sets the shortest edge of the decimated surface of the boundary structure produced by Boolean operations.</p> <p>When set to <code>-1</code>, the used value is half of the structure-dependent spacing epsilon¹.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <code>structure</code> parameter denotes a boundary structure. • You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_shortest_edge=<n></code> $\{-1\} \cup [0, \infty[$	μm
spacing	Vector	<p>Sets the cell size of the grid used to compute the surface evolution for each dimension. A vector with only one component can be used to set the grid cell size to the same value for all dimensions.</p> <p>For a machine using any model other than <code>spin_on</code>, for 2D and 3D structures, a vector with two and three components, respectively, can be used to specify the grid cell size separately for each dimension.</p> <p>For a machine using the <code>spin_on</code> model, for 3D structures, a vector with two components can be used to specify the grid cell size separately for each dimension.</p> <p>See Discretization Size and Accuracy on page 25.</p>	none	μm
split_plots	Boolean	Specifies whether to write intermediate plots each to a different file.	false	none

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command (Continued)

Parameter	Type	Description	Default [Range]	Unit
stop_plane	Number	Sets the vertical coordinate of a horizontal plane above the exposed surface. When the exposed surface reaches or crosses this plane, time-stepping stops. See Stopping Time-Stepping on page 336 .	none	μm
stop_point	Vector list	Specifies a list of coordinates of points above the exposed surface. When the exposed surface reaches or crosses any of these points, the simulation stops. See Stopping Time-Stepping on page 336 .	none	μm
structure	Character	Sets the name of the structure on which deposition will occur.	default_structure	none
surface_accuracy	Number	<p>Sets the maximum deviation between the decimated exposed surface and the original exposed surface.</p> <p>When set to -1, the used value is half of the structure-dependent spacing epsilon¹.</p> <p>It can be specified only if <code>surface_decimate=true</code>, or if you do not specify the <code>surface_decimate</code> parameter and specify <code>let surface_decimate=true</code>.</p>	<p>The value set by <code>let surface_decimate_accuracy=<n></code></p> <p>$\{-1\} \cup [0, \infty[$</p>	μm
surface_decimate	Boolean	Specifies whether to decimate the exposed surface used as an operand of Boolean operations.	The value set by <code>let surface_decimate=</code>	none

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command (Continued)

Parameter	Type	Description	Default [Range]	Unit
surface_min_angle	Number	Sets the smallest angle in the elements of the decimated exposed surface used as an operand of Boolean operations. It can be specified only if surface_decimate=true, or if you do not specify the surface_decimate parameter and specify let surface_decimate=true.	The value set by let surface_decimate_min_angle=<n> [0, 180]	degree
surface_min_dihedral_angle	Number	Sets the smallest dihedral angle between two elements, which share an edge, of the decimated exposed surface used as an operand of Boolean operations. It can be specified only if surface_decimate=true, or if you do not specify the surface_decimate parameter and specify let surface_decimate=true.	The value set by let surface_decimate_min_dihedral_angle=<n> [0, 180]	degree
surface_ridge_angle	Number	Sets the angle used by the decimation algorithm to determine geometric features of the exposed surface used as an operand of Boolean operations. It can be specified only if surface_decimate=true, or if you do not specify the surface_decimate parameter and specify let surface_decimate=true.	The value set by let surface_decimate_ridge_angle=<n> [0, 180]	degree

Chapter 6: Input Commands

deposit

Table 57 Parameters of deposit command (Continued)

Parameter	Type	Description	Default [Range]	Unit
surface_shortest_edge	Number	Sets the shortest edge of the decimated exposed surface used as an operand of Boolean operations. When set to -1, the used value is half of the structure dependent spacing epsilon ¹ . It can be specified only if surface_decimate=true, or if you do not specify the surface_decimate parameter and specify let surface_decimate=true.	The value set by let surface_decimate_shortest_edge=<n> {-1} ∪ [0, ∞[µm
time	Number	Sets the simulation time.	none	minute
update_scheme	Character	Sets the update scheme for the time integration of the level-set function. Options are: <ul style="list-style-type: none">• lax_friedrichs• upwind	upwind	none
variance_reduction	Character	Sets the algorithm to use for variance reduction by the Monte Carlo flux integrator. Options are: <ul style="list-style-type: none">• N-2017.09• M-2016.12	N-2017.09	none
vertical_orientation	Vector	Sets the Miller index of the direction perpendicular to the wafer surface (z-axis of the wafer coordinate system) for the newly created region. It can be specified only if structure denotes a boundary structure.	{0 0 1}	none

1. The structure-dependent spacing epsilon is logged out when defining a structure using the define_structure command.

Chapter 6: Input Commands

deposit

Note:

The Miller indices for a plane are usually written as (hkl), and the set of equivalent planes is written as {hkl}. A crystal direction is usually written as [hkl], and the set of equivalent directions is written as <hkl>. The value assigned to the parameters flat_orientation and vertical_orientation is a Tcl list of indices representing a crystal direction. The braces in the specified value are required to form a Tcl list and do not indicate a set of equivalent planes.

Limitations

The following limitations apply:

- The parameter structure cannot specify a PMC structure. To process such a structure, an explicit conversion of the structure is needed before calling the deposit command (see [Integration With Other Sentaurus Topography 3D Functionality on page 519](#)).
- When using a machine having model=crystal, the following parameters are not supported:

cfl, compute_first_plot, compute_last_plot, extraction_interval, extraction_times, extraction_times_unit, file, merge, plot_interval, plot_times, plot_times_unit, plot_type, stop_plane, stop_point, update_scheme

- The stop_plane, stop_point, and update_scheme parameters are not supported when using a machine having model=spin_on.
- For machines having model=spin_on, structures whose exposed surface cannot be described as a single-valued function of the x- and y-coordinates in three dimensions, or of the x-coordinate in two dimensions, are not supported. However, topologically connected exposed surfaces containing elements parallel to the vertical direction are supported.

Examples

Deposition process using a machine:

```
deposit machine=depomachine spacing={0.05 0.3 0.05} time=0.5
```

This command simulates a deposition process on the structure, using the machine depomachine with a grid spacing of dx=0.05 μm, dy=0.3 μm, dz=0.05 μm, and a simulation time of 0.5 minutes.

Note:

The resolution is adjusted internally to match the actual structure.

Chapter 6: Input Commands

deposit

Deposition process using a shape:

```
deposit shape=shape_1 material=Oxide comment="deposit shape"
```

This command deposits the shape `shape_1` onto the structure. The created region will consist of the material `Oxide`. The comment for this process step is displayed in the log file.

Note:

Parts of `shape_1` that overlap the existing structure are not added.

Deposition process with intermediate surface plotting:

```
deposit machine=depomachine spacing={0.1} time=0.5 \
    file=depo_surfaces.tdr plot_interval=0.1
```

This command deposits using the machine `depomachine`, with a grid spacing of $dx=dy=dz=0.1 \mu\text{m}$. Intermediate surfaces along with data such as surface velocities and fluxes are plotted into the file `depo_surfaces.tdr`. The surface of the first and last time steps of the simulation will always be plotted. During the simulation, whenever the simulation time surpasses an integral multiple of `plot_interval` (for example, 0.1, 0.2, 0.3...), a surface will be plotted to the file. This feature can check process parameters and model behavior.

Stopping Time-Stepping

The `stop_plane` and `stop_point` parameters of the `deposit` command allow you to stop time-stepping when a specified plane or point has been reached before the final process time.

These parameters can be used when processing both 2D and 3D structures.

The dimensions of the points specified with `stop_point` must match the dimensions of the structure for which it is used.

The values specified with `stop_plane` and `stop_point` must fulfill certain constraints with respect to the computational domain and the initial exposed surface.

The computational domain and the initial exposed surface are defined in [Boundary Types on page 30](#):

- When using `stop_plane`, the specified plane must intersect the computational domain, and the specified plane must be above the initial exposed surface.
- When using `stop_point`, the specified point must be inside the computational domain, and the specified point must lie above the initial exposed surface.

Due to the discrete time steps, the exposed surface can cross the specified plane or point in the last time step. The maximum distance of the final exposed surface beyond the specified

Chapter 6: Input Commands

deposit

plane or point is limited by the value of the parameter `spacing` that specifies the discretization of the level-set grid.

When using `stop_plane` or `stop_point`, the output written to the log file indicates why the time-stepping of the corresponding process step stopped.

If the specified plane or point was reached or crossed, the output indicates this. The output includes the specified plane or point and the process time at which this occurred. For example:

```
Exiting loop early: stop_plane= <> [um] has been reached at t=<>.
```

```
Exiting loop early: stop_point= {} [um] has been reached at t=<>.
```

If the specified plane or point is not reached or crossed, this is indicated in the output together with the specified plane or point and the specified process time. For example:

```
stop_plane= <> [um] has not been reached at any time step within  
time=<> [min].
```

```
stop_point= {} [um] has not been reached at any time step within  
time=<> [min].
```

Examples

Specify a stop plane for deposition on a 2D or 3D structure:

```
deposit spacing=0.1 time=1.0 stop_plane=1.0
```

Specify a stop point for deposition on a 3D structure:

```
deposit spacing=0.1 time=1.0 stop_point={0.0 0.5 0.3}
```

Specify a stop point for deposition on a 2D structure:

```
deposit spacing=0.1 time=1.0 stop_point={0.0 0.5}
```

TDR Dataset Names for Fluxes

For models that calculate ion or neutral fluxes, the values of the fluxes are written to TDR datasets when using the parameter `plot_interval`.

The name of a flux dataset starts with the name of the flux species. If the flux takes a yield function into account, it is followed by `yield`.

The type of flux is indicated by one of the following suffixes:

`direct` The direct flux that arrives from the plasma at a surface element.

`reactive` The incoming flux at a surface element that is available for reemission.

Chapter 6: Input Commands

deposit

- reflected The direct flux that has been reflected away from a surface element.
- reflection The incoming flux at a surface element that has been reflected from other surface elements.
- total Sum of the reactive flux and the incoming indirect flux caused by the reactive flux.

etch

This command starts an etching simulation or a simultaneous etching and deposition simulation, as well as a deposition simulation based on a reaction model, and can be used in one of the following ways:

- Using a machine defined by `define_etch_machine`
- Geometrically etching a shape defined with the `define_shape` command
- Geometrically etching a mask defined with the `define_mask` command

Syntax

Etching simulation on the surface of the structure using a machine with `method=levelset`:

```
etch method=levelset spacing=<v> time=<n> \
[accuracy=<n>] [cfl=<n>] [comment=<c>] [decimate=<b>] \
[diffusion_flux_error_control_strategy=<c>] \
[diffusion_flux_integration_error=<n>] \
[engine=<c>] [exposed_only=<b>] \
[extraction=<c> (extraction_interval=<n> | \
extraction_times=<v> [extraction_times_unit=<c>]) \
[extraction_file_update_interval=<n>] \
[force_intermediate_extraction=<b>]] \
[file=<c>] [fill_material=<c>] [fill_region=<c>] \
[flat_orientation=<v>] \
[integration_error=<n> | integration_samples=<n>] \
[machine=<c>] [max_ion_reflections=<n>] \
([abs_min_deposition_thickness=<n>] | min_deposition_thickness=<n>) \
[merge=<b>] [min_angle=<n>] [min_dihedral_angle=<n>] \
[min_ion_reflection_probability=<n>] \
[pad_pressure_acceleration=<b>] [pad_pressure_accuracy=<n>] \
[pad_pressure_max_iterations=<n>] \
[(plot_interval=<n> [compute_first_plot=<b>]
[compute_last_plot=<b>] | \
plot_times=<v> [plot_times_unit=<c>]) \
[force_intermediate_plot=<b>] [num_flux_orders=<n>] \
[plot_file_update_interval=<n>] [plot_type=<l>] [split_plots=<b>]] \
[random_seed=<n>] [region=<c>] [region_query_accuracy=<c>] \
[remove_spurious_parts=<b>] [ridge_angle=<n>] \
[selective_deposition=<b> [selective_deposition_threshold=<n>]] \
[shortest_edge=<n>] [stop_plane=<n> | stop_point=<w>] \
[structure=<c>] [surface_accuracy=<n>] [surface_decimate=<c>] \
[surface_min_angle=<n>] [surface_min_dihedral_angle=<n>] \
[surface_ridge_angle=<n>] [surface_shortest_edge=<n>] \
[update_scheme=<c>] [variance_reduction=<c>] \
[vertical_orientation=<v>]
```

Chapter 6: Input Commands

etch

Note:

The `fill_material` and `fill_region` parameters are not available for simultaneous etching and deposition models.

Etching using a machine with `method=levelset` is not supported when the structure specified with the parameter `structure` is a PMC structure. To process such a structure, an explicit conversion of the structure is needed before calling the `etch` command (see [Integration With Other Sentaurus Topography 3D Functionality on page 519](#)).

Etching simulation on the surface of the structure using a machine with `method=pmc`:

```
etch method=pmc spacing=<n> time=<n> \
[averaging_interval=<n> | \
averaging_times=<n> [averaging_times_unit=<c>]] \
[averaging_runs=<n>] \
[comment=<c>] [compute_process_graph=<b>] [deposition_method=<c>] \
[electric_field_update_accuracy=<n>] \
[electric_field_update_interval=<n>] \
[electric_solver=<c>] \
[extraction=<c> (extraction_interval=<n> | \
extraction_times=<n> [extraction_times_unit=<c>])) \
[extraction_file_update_interval=<n>] \
[force_intermediate_extraction=<b>]] \
[file=<c>] [force_html_report=<b>] [force_intermediate_plot=<b>] \
[html_report=<b>] \
[html_report_interval=<n> | (html_report_times=<n> \
[html_report_times_unit=<c>])] [html_report_type=<c>] \
[machine=<c>] [max_number_edges_process_graph=<n>] \
[max_number_reflections=<n>] \
[noise_reduction=<n>] \
[num_trajectories=<n> | trajectory_starting_positions=<l>] \
[(plot_interval<n> [compute_first_plot=<b> compute_last_plot=<b>] | \
plot_times<n> [plot_times_unit=<c>]) \
[plot_type=<l> [force_intermediate_plot=<b>] \
[gc_samples_per_cell=<n>] \
[num_flux_orders=<n>] [plot_quality=<n>] \
[point_max=<n> point_min=<n>]]] \
[save_averaging_samples=<b> [averaging_sample_base_name=<c>]] \
[selective_deposition=<b>] [split_replacements=<l>] \
[sputtering_method=<c>] \
[stop_material=<c> | stop_plane=<n> | stop_point=<w>] \
[structure=<c>] \
[tcl_equivalent_flux_variable=<c>] [top_gas_thickness=<n>]
```

Geometric etching using a shape:

On a boundary structure:

```
etch shape=<c> \
[accuracy=<n>] [comment=<c>] [decimate=<b>] [fill_material=<c>] \
[fill_region=<c>] [material=<c>] [min_angle=<n>] \
```

Chapter 6: Input Commands

etch

```
[min_dihedral_angle=<n>] \
[ridge_angle=<n>] [shortest_edge=<n>] [structure=<c>]
```

On a PMC structure:

```
etch shape=<c> \
[comment=<c>] [fill_material=<c>] [material=<c>] [structure=<c>]
```

Geometric etching using a mask:

On a boundary structure:

```
etch mask=<c> thickness=<n> type=<c> \
[accuracy=<n>] [comment=<c>] [decimate=<b>] [fill_material=<c>] \
[fill_region=<c>] [min_angle=<n>] [min_dihedral_angle=<n>] \
[ridge_angle=<n>] [shortest_edge=<n>] [structure=<c>]
```

On a PMC structure:

```
etch mask=<c> thickness=<n> type=<c> \
[comment=<c>] [fill_material=<c>] [structure=<c>]
```

Similar to the pattern command, geometric etching using a mask conveniently combines several processing steps. It can be interpreted as a shortcut for the following process steps:

- Deposition of a negative or positive resist
- Exposure of the resist with a dark-field or a light-field reticle
- Development of the resist
- Geometric etching of the structure
- Stripping of the resist

Note:

Geometric etching using a mask is not supported for 2D structures.

Geometric etching using a shape or using a mask is not supported when the structure specified with the parameter structure is a PMC structure. To process such a structure, an explicit conversion of the structure is needed before calling the etch command (see [Integration With Other Sentaurus Topography 3D Functionality on page 519](#)).

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command

Parameter	Type	Description	Default [Range]	Unit
abs_min_deposition_thickness	Number	Sets the absolute minimum thickness a deposited layer must grow during a time step to be created or added.	0 [0, ∞[µm
accuracy	Number	Sets the maximum deviation between the decimated surface of the boundary structure produced by Boolean operations and its original surface. When set to -1, the used value is half of the structure-dependent spacing epsilon ¹ . This parameter can be specified only if all of the following conditions are met: <ul style="list-style-type: none"> The structure parameter denotes a boundary structure. You specify decimate=true or, if you do not specify decimate, you did not specify let decimate=false. 	The value set by let decimate_accuracy=<n> {-1} ∪ [0, ∞[µm
averaging_interval	Number	Sets the time interval after which the different PMC runs are averaged. If the parameter is omitted, the value of the time parameter is used.	none]0, ∞[minute
averaging_runs	Number	Sets the number of PMC runs to be averaged.	1 [1, ∞[none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
averaging_sample_base_name	Character	<p>Sets the base name of the file names used to save the results of individual averaging runs in PMC format. The default is the value of the <code>base_name</code> parameter of the <code>let</code> command.</p> <p>Note: This parameter can be set only when <code>save_averaging_samples=true</code>.</p>	<base_name>	none
averaging_times	Vector	<p>Sets the times after which the different PMC runs are averaged. The given vector must:</p> <ul style="list-style-type: none"> Contain only nonnegative values Be sorted in ascending order Contain no value larger than the value of parameter <code>time</code> 	none	none
averaging_times_unit	Character	<p>Sets the unit of the <code>averaging_times</code> values. Options are:</p> <ul style="list-style-type: none"> <code>s</code> (seconds) <code>min</code> (minutes) 	min	none
cfl	Number	Sets the Courant–Friedrichs–Lewy (CFL) number [16] for the integration of the surface evolution equation.	0.95 (when using a machine having <code>model=simple</code>) 0.5 (when using a machine having not having <code>model=simple</code>) 10, 1]	none
comment	Character	Sets a comment for this process step that is displayed or written to the log file.	empty string	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
compute_first_plot	Boolean	When <code>plot_interval</code> is set, the parameter <code>compute_first_plot</code> specifies whether to produce an intermediate plot at <code>time=0</code> .	true	none
compute_last_plot	Boolean	When <code>plot_interval</code> is set, the parameter <code>compute_last_plot</code> specifies whether to produce an intermediate plot at the end of the simulation.	true	none
compute_process_graph	Boolean	Specifies whether to compute the process graph during a PMC simulation. Note: This parameter can be specified only if <code>noise_reduction=0</code> or if <code>noise_reduction</code> is not specified.	false	none
decimate	Boolean	Specifies whether to decimate the boundary structure produced by Boolean operations. It can be specified only if <code>structure</code> denotes a boundary structure.	The value set by <code>let decimate=</code>	none
deposition_method	Character	Sets the PMC deposition method. The methods differ with respect to the surface diffusion model, which affects the smoothness and the shape evolution of the deposited film. Options are: <ul style="list-style-type: none">• generic• layer• layer_minimum• minimum• patch	minimum	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
diffusion_flux_error_control_strategy	Character	Sets the value of the parameter diffusion_flux_integration_error. Options are: <ul style="list-style-type: none">• absolute: Absolute error.• relative: Relative error	relative	none
diffusion_flux_integration_error	Number	Sets the requested flux error of the diffusing species.	0.05 [0, 1]	none
electric_field_update_accuracy	Number	When simulating charge-up, this parameter sets a threshold used to determine whether to recompute the electric field. If the change of the local charges accumulated in the structure is greater than the given accuracy, then the electric field is updated. This check is performed at regular time intervals, specified by electric_field_update_interval.	0 (always update the electric field) [0, ∞)	none
electric_field_update_interval	Number	When simulating charge-up, you must specify the time interval after which the electric field is updated.	none (0, ∞)	minute
electric_solver	Character	When simulating charge-up, you can optionally set the name of an electric field solver object defined in the command define_electric_solver (see define_electric_solver on page 206). If not specified, then default parameters are used for the electric field solver.	none	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
engine		Sets the engine to use for flux computation. This parameter is available only for flux models. Options are: <ul style="list-style-type: none">• monte_carlo• radiosity	radiosity	none
exposed_only	Boolean	Specifies whether only exposed portions of the surface are etched when using the simple model.	false	none
extraction	Character	Sets the name of the group of extractions to run during this process step.	none	none
extraction_file_update_interval	Number	Sets the time interval after which intermediate extraction results are written to file. When set to 0, intermediate extraction results are written to file immediately after they are created. Note: This parameter can be set only when at least one intermediate extraction produces file output.	0 [0, ∞[minute
extraction_interval	Number	Sets the process time interval after which intermediate extractions must be executed.	none]0, ∞[minute

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
extraction_times	Vector	Sets the times at which to perform intermediate extractions. The given vector must: <ul style="list-style-type: none"> • Contain only nonnegative values • Be sorted in ascending order • Contain no value larger than the value of parameter <code>time</code> If any of these conditions is not met, then an error is issued.	none	none
extraction_times_unit	Character	Sets the unit of the <code>extraction_times</code> values. Options are: <ul style="list-style-type: none"> • <code>s</code> (seconds) • <code>min</code> (minutes) 	min	none
file	Character	Sets the file into which intermediate surfaces and surface data should be saved. Used in conjunction with <code>plot_interval</code> only. In the file name, <code><base_name></code> denotes the value of the <code>base_name</code> parameter of the <code>let</code> command.	<code><base_name>.tdr</code>	none
fill_material	Character	Sets the material to fill the etched volume. If this parameter is omitted, the etched volume is not filled.	none	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
fill_region	Character	Sets the name of the region filling the etched volume. The structure specified with the parameter <code>structure</code> must not contain regions with this name. If <code>fill_region</code> is omitted, the name of the region filling the etched volume is generated automatically. It can be specified only if <code>structure</code> denotes a boundary structure.	none	none
flat_orientation	Vector	Sets the Miller index of the direction perpendicular to the wafer flat (y-axis of the wafer coordinate system) for the newly created region for models that perform simultaneous etching and deposition.	{1 1 0}	none
force_html_report	Boolean	Specifies whether to produce a report about the performed PMC simulation even if Sentaurus Topography 3D was started with the <code>--no_html_report</code> command-line option.	false	none
force_intermediate_extraction	Boolean	Specifies whether to compute intermediate extractions and, if applicable, write them to files even if Sentaurus Topography 3D was started with the <code>--no_intermediate_extraction</code> command-line option.	false	none
force_intermediate_plot	Boolean	Specifies whether to compute intermediate plots and write them to files even if Sentaurus Topography 3D was started with the <code>--no_intermediate_plot</code> command-line option.	false	none
gc_samples_per_cell	Number	Sets the number of samples to take along the x- and y-directions per grid cell.	1 [1, 2147483647]	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
html_report	Boolean	Specifies whether to produce a report (HTML file) about the performed PMC simulation. If neither html_report_interval nor html_report_times is specified, then the report is produced at the end of the simulation.	true	none
html_report_interval	Number	Sets the process time interval after which HTML reports must be produced.	none]0, ∞[minute
html_report_times	Vector	Sets the times at which to create HTML reports. The given vector must: <ul style="list-style-type: none"> Contain only nonnegative values Be sorted in ascending order Contain no value larger than the value of parameter time If any of these conditions is not met, then an error is issued.	none	none
html_report_times_unit	Character	Sets the unit of the html_report_times values. Options are: <ul style="list-style-type: none"> min (minutes) s (seconds) 	min	none
html_report_type	Character	Sets the HTML report type to produce. Options are: <ul style="list-style-type: none"> cumulative (reported statistics are collected starting from the beginning of the simulation) differential (reported statistics are collected starting from when the previous HTML report was produced) 	cumulative	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
integration_error	Number	Sets the maximum relative discretization error in the direct flux calculation.	0.05 [0, 1]	none
integration_samples	Number	<p>Sets the number of integration points in the numeric integration. The meaning of this parameter depends on the value of the engine parameter:</p> <ul style="list-style-type: none"> When engine=radiosity, integration_samples describes the number of discretization points, per surface element, in the hemisphere where visibility is determined. When engine=monte_carlo, integration_samples describes the number of particle paths, per surface element, used in the flux integration. <p>Note: This parameter is deprecated for engine=radiosity. Instead, use the integration_error parameter.</p>	1000 (engine=radiosity), 700 (engine=monte_carlo and variance_reduction=M-2016.12), 350 (engine=monte_carlo and variance_reduction=N-2017.09) [1, ∞[none
machine	Character	Sets the name of the machine that is used for the etch process.	default_machine	none
mask	Character	Sets the name of the mask to be used for the etch process.	none	none
material	Character	Sets the material that can be etched. If omitted, all materials can be etched.	none	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
max_ion_reflections	Number	Sets the maximum number of ion reflections to compute during flux integration. The specified value must be an integer. Only applicable when using RFM models.	1 [1, ∞[none
max_number_edges_process_graph	Number	Sets the maximum number of edges of the process graph that can be visualized.	10000 [1, 2147483647]	none
max_number_reflections	Number	Sets the number of reflections a species can undergo before being discarded.	2000000 [0, 2147483647]	none
merge	Boolean	If true, the deposited region merges with the already existing regions of the same material in the structure.	false	none
method	Character	Sets the simulation method to use. Options are: • levelset • pmc	levelset	none
min_angle	Number	Sets the smallest angle in the elements of the decimated surface of the boundary structure produced by Boolean operations. This parameter can be specified only if all of the following conditions are met: <ul style="list-style-type: none">• The structure parameter denotes a boundary structure.• You specify decimate=true or, if you do not specify decimate, you did not specify let decimate=false.	The value set by let decimate_min_angle=<n> [0, 180]	degree

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
min_deposition_thickness	Number	Sets the minimum thickness a deposited layer must grow during a time step to be created or added as a fraction of the mean spacing.	0.01 [0, 1]	none
min_dihedral_angle	Number	<p>Sets the smallest dihedral angle between two elements of the decimated surface of the boundary structure produced by Boolean operations that share an edge.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> The <code>structure</code> parameter denotes a boundary structure. You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_min_dihedral_angle=<n></code> [0, 180]	degree
min_ion_reflection_probability	Number	Sets the threshold for the reflection probability below which multiple reflection is stopped.	0.001 [0, 1]	none
noise_reduction	Number	<p>Specifies the amount of noise reduction.</p> <p>Note: This parameter can be specified only if the used model includes sputtering reactions.</p>	0 [0, 1]	none
num_flux_orders	Number	Sets the number of flux orders to produce for each PMC intermediate surface plot. It can be specified only when <code>plot_type</code> contains the value <code>surface</code> .	1 [1, 2147483647]	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
num_trajectories	Number	When writing intermediate plots with <code>plot_type=trajectories</code> , this parameter sets the number of trajectories to be plotted. You can specify either <code>num_trajectories</code> or <code>trajectory_starting_positions</code> .	50 [1, ∞)	none
pad_pressure_acceleration	Boolean	Specifies whether to use an accelerated solver to evaluate the RFM function <code>pad_pressure()</code> . Note: You can set this parameter to false only when processing 2D structures.	true	none
pad_pressure_accuracy	Number	Sets the relative residual below which the solver stops when evaluating the RFM function <code>pad_pressure()</code> .	1e-3 [0, ∞ [none
pad_pressure_max_iterations	Number	Sets the maximum number of iterations that are allowed to evaluate the RFM function <code>pad_pressure()</code> .	100 [1, 2147483647]	none
plot_file_update_interval	Number	Sets the time interval after which intermediate plots are written to file. When set to 0, intermediate plots are written to file immediately after they are created. Note: This parameter can be set only when <code>split_plots=false</code> .	0 [0, ∞ [minute

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
plot_interval	Number	Sets the process time interval after which intermediate exposed surfaces and surface data must be plotted. See TDR Dataset Names for Fluxes on page 366 .	none [0, ∞[minute
plot_quality	Number	Sets the quality of a PMC intermediate surface plot. The larger its value, the better the plot quality. It can be specified only when <code>plot_type=surface</code> .	1]0, ∞[none
plot_times	Vector	Sets the times at which to create intermediate plots. The given vector must: <ul style="list-style-type: none"> Contain only nonnegative values Be sorted in ascending order Contain no value larger than the value of parameter <code>time</code> If any of these conditions is not met, then an error is issued.	none	none
plot_times_unit	Character	Sets the unit of the <code>plot_times</code> values. Options are: <ul style="list-style-type: none"> <code>s</code> (seconds) <code>min</code> (minutes) 	min	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
plot_type	List	<p>Sets the type of data to save at time intervals specified by plot_interval.</p> <p>When method=levelset, options are:</p> <ul style="list-style-type: none"> • structure • surface <p>When method=pmc, options are:</p> <ul style="list-style-type: none"> • charge_density • electric_field • electric_potential • gc • surface • vbe • volume_fractions 	surface	none
point_max	Vector	<p>Sets the requested maximum corner of the bounding box of the grid containing the data. It can be set only when plot_type=charge_density, electric_field, electric_potential, or volume_fractions.</p> <p>The actual maximum corner of the bounding box of the grid containing the data is obtained by snapping the given point_max value to a grid point of the PMC structure (specified with the parameter structure).</p>	automatic	µm

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
point_min	Vector	Sets the requested minimum corner of the bounding box of the grid containing the data. It can be set only when plot_type=charge_density, electric_field, electric_potential, or volume_fractions. The actual minimum corner of the bounding box of the grid containing the data is obtained by snapping the given point_min value to a grid point of the PMC structure (specified with the parameter structure).	automatic	µm
random_seed	Number	Sets the seed of the pseudo-random number generator used by the Monte Carlo flux integrator. Note: This parameter can be specified only when engine=monte_carlo.	1 [1, 2147483647]	none
region	Character	Sets the name of the region that can be redeposited in advanced etching models. Not valid for etching models that do not create redeposition phenomena.	none	none
region_query_accuracy	Character	Sets the accuracy used to determine the regions and materials associated with each element of the exposed surface. Options are: <ul style="list-style-type: none">• resolution• subresolution See Discretization Size and Accuracy on page 25 .	resolution	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
remove_spurious_parts	Boolean	Specifies whether to remove spurious parts after running the process step.	true	none
ridge_angle	Number	<p>Sets the angle used by the decimation algorithm to determine geometric features of the boundary structure produced by Boolean operations.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> The <code>structure</code> parameter denotes a boundary structure. You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_</code> <code>ridge_angle=<n></code> <code>[0, 180]</code>	degree
save_averaging_samples	Boolean	<p>Specifies whether to save the result of each averaging run in PMC format.</p> <p>Note: This parameter can be set only when the value of parameter <code>averaging_runs</code> is greater than 1.</p>	false	none
selective_deposition	Boolean	Specifies whether to suppress the surface spreading of deposited material across material interfaces, thereby avoiding lateral creeping of the deposited film.	false (for <code>method=levelset</code>) true (for <code>method=pmc</code>)	none
selective_deposition_threshold	Number	Sets the threshold for the thickness of a layer below which the deposited material is considered to be an artifact. The specified value is relative to the mean discretization size.	1.0 <code>[0.0, ∞[</code>	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
shape	Character	Sets the name of the shape that is used for the etch process.	none	none
shortest_edge	Number	<p>Sets the shortest edge of the decimated surface of the boundary structure produced by Boolean operations.</p> <p>When set to -1, the used value is half of the structure-dependent spacing epsilon¹.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> The structure parameter denotes a boundary structure. You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	<p>The value set by <code>let decimate_shortest_edge=<n></code></p> <p>$\{-1\} \cup [0, \infty[$</p>	μm
spacing	Vector	Sets the size of a grid cell for each direction. A vector with one component can be used to set the grid cell size to the same value for all dimensions. When <code>method=pmc</code> , this is the only supported option. Whereas, when <code>method=levelset</code> , nonuniform grid cell sizes are supported. When <code>method=levelset</code> , for 2D and 3D structures, a vector with two and three components, respectively, can be used to specify the grid cell size separately for each dimension. See Discretization Size and Accuracy on page 25 .	none	μm
split_plots	Boolean	Specifies whether to write intermediate plots each to a different file.	false	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
split_replacements	List	<p>Specifies a list of material replacement maps defined in the command <code>define_material_replacement</code>. The list consists of the names of material replacement maps. The material is replaced by a list of species defined by the material replacement map before the PMC step begins.</p> <p>Note: This parameter can be used only with <code>method=pmc</code>.</p>	none	none
sputtering_method	Character	<p>Sets the method to use when calculating from where to remove the material to be sputtered, with respect to the impact location of the incoming particle, which affects the smoothness and shape evolution of the structure.</p> <p>Options are:</p> <ul style="list-style-type: none"> • minimum • patch 	minimum	none
stop_material	Character	<p>Sets the bulk material where etching stops etching when the surface reaches the specified material.</p> <p>When <code>method=pmc</code>, the <code>stop_material</code> parameter can stop etching when the specified material is used for the first time as a bulk material in a reaction.</p>	none	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
stop_plane	Number	Sets the vertical coordinate of a horizontal plane below the exposed surface and not intersecting any void. When the exposed surface reaches or crosses this plane, time-stepping stops. See Stopping Time-Stepping on page 366 .	none	µm
stop_point	Vector list	Specifies a list of coordinates of points below the exposed surface and not inside any void. When the exposed surface reaches or crosses any of these points, the simulation stops. See Stopping Time-Stepping on page 366 .	none	µm
structure	Character	Sets the name of the structure to be etched.	default_structure	none
surface_accuracy	Number	Sets the maximum deviation between the decimated exposed surface and the original exposed surface. When set to -1, the used value is half of the structure-dependent spacing epsilon ¹ . It can be specified only if surface_decimate=true, or if you do not specify the surface_decimate parameter and specify let surface_decimate=true.	The value set by let decimate_accuracy=<n> {-1} ∪ [0, ∞[µm
surface_decimate	Boolean	Specifies whether to decimate the exposed surface used as an operand of Boolean operations.	The value set by let surface_decimate=	none

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
surface_min_angle	Number	Sets the smallest angle in the elements of the decimated exposed surface used as an operand of Boolean operations. It can be specified only if surface_decimate=true, or if you do not specify the surface_decimate parameter and specify let surface_decimate=true.	The value set by let surface_min_angle=<n> [0, 180]	degree
surface_min_dihedral_angle	Number	Sets the smallest dihedral angle between two elements, which share an edge, of the decimated exposed surface used as an operand of Boolean operations. It can be specified only if surface_decimate=true, or if you do not specify the surface_decimate parameter and specify let surface_decimate=true.	The value set by let surface_min_dihedral_angle=<n> [0, 180]	degree
surface_ridge_angle	Number	Sets the angle used by the decimation algorithm to determine geometric features of the exposed surface used as an operand of Boolean operations. It can be specified only if surface_decimate=true, or if you do not specify the surface_decimate parameter and specify let surface_decimate=true.	The value set by let surface_ridge_angle=<n> [0, 180]	degree

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
surface_shortest_edge	Number	Sets the shortest edge of the decimated exposed surface used as an operand of Boolean operations. When set to -1, the used value is half of the structure dependent spacing epsilon ¹ . It can be specified only if surface_decimate=true, or if you do not specify the surface_decimate parameter and specify let surface_decimate=true.	The value set by let surface_shortest_angle=<n> {-1} ∪ [0, ∞[µm
tcl_equivalent_flux_variable	Character	Sets the name of the Tcl variable to which the equivalent fluxes will be written. When the etch command is executed, this Tcl variable will contain a list with an entry for each volumetric source species. Each entry is, in turn, a list of two elements containing the volumetric source species name and its equivalent flux. If you omit this parameter, then no Tcl output is produced.	none	none
thickness	Number	Sets the etch depth when etching with a mask, measured relative to the exposed surface of the structure.	none [0, ∞[µm
time	Number	Sets the simulation time.	none	minute
top_gas_thickness	Number	Sets the minimum thickness of the gas region added on top of the structure.	0 [0, ∞[µm

Chapter 6: Input Commands

etch

Table 58 Parameters of etch command (Continued)

Parameter	Type	Description	Default [Range]	Unit
trajectory_starting_positions	List	When writing intermediate plots with <code>plot_type=trajectories</code> , this parameter sets the starting positions of the trajectories on the xy plane above the structure. The starting positions must be given as a list of xy pairs: {x1 y1 x2 y2 ...}. You can specify either <code>num_trajectories</code> or <code>trajectory_starting_positions</code> .	none	µm
type	Character	Sets the type of the reticle and the resist. Options are: <ul style="list-style-type: none"> • <code>dark_negative</code> • <code>dark_positive</code> • <code>light_negative</code> • <code>light_positive</code> 	none	none
update_scheme	Character	Sets the update scheme for the time integration of the level-set function. Options are: <ul style="list-style-type: none"> • <code>lax_friedrichs</code> • <code>upwind</code> 	upwind	none
variance_reduction	Character	Sets the algorithm to use for variance reduction by the Monte Carlo flux integrator. Options are: <ul style="list-style-type: none"> • <code>N-2017.09</code> • <code>M-2016.12</code> 	N-2017.09	none
vertical_orientation	Vector	Sets the Miller index of the direction perpendicular to the wafer surface (z-axis of the wafer coordinate system) for the newly created region for models that perform simultaneous etching and deposition.	{0 0 1}	none

Chapter 6: Input Commands

etch

1. *The structure-dependent spacing epsilon is logged out when defining a structure using the define_structure command.*

Note:

The Miller indices for a plane are usually written as (hkl), and the set of equivalent planes is written as {hkl}. A crystal direction is usually written as [hkl], and the set of equivalent directions is written as <hkl>. The value assigned to the parameters flat_orientation and vertical_orientation is a Tcl list of indices representing a crystal direction. The braces in the specified value are required to form a Tcl list and do not indicate a set of equivalent planes.

Limitations

The following limitations apply:

- When method=pmc, the grid spacing specified with the parameter spacing is ignored if the structure specified with the parameter structure stores the result of a previous simulation using the PMC method.
- When running a simulation with a machine using a reaction model that allows deposition, the vertical extent of the computational domain (see [Boundary Types on page 30](#)) must be large enough to contain the deposited material. The vertical extent of the computational domain can be increased by adding Gas to the initial structure using the fill command (see [fill on page 415](#)).
- For machines using RFM models with a rate formula involving the function pad_pressure(), 2D structures whose exposed surface cannot be described as a single-valued function of the x-coordinate are not supported.
- When using a machine with model=crystal, the following parameters are not supported:

```
abs_min_deposition_thickness, cfl, compute_first_plot,  
compute_last_plot, extraction_interval, extraction_times,  
extraction_times_unit, file, min_deposition_thickness, plot_interval,  
plot_times, plot_times_unit, plot_type, region_query_accuracy,  
stop_plane, stop_point, update_scheme
```

- When you start Sentaurus Topography 3D with the --processes command-line option with a value greater than 1 (see [Table 1 on page 17](#)), except when using method=pmc, the parameter structure must not denote a structure obtained from the etch command using method=pmc averaging_runs=1.

Chapter 6: Input Commands

etch

- When you start Sentaurus Topography 3D with the `--processes` command-line option with a value greater than 1 (see [Table 1 on page 17](#)) and use `method=pmc` `averaging_runs=1`:
 - The following parameters are not supported:
`extraction`, `plot_interval`, `plot_times`, `stop_material`, `stop_plane`, `stop_point`
 - The parameter `compute_process_graph=true` is not supported.
 - Machines using the parameter `damage` or volumetric source species are not supported.
 - Structures using periodic boundary conditions are not supported.

Examples

Etching process using an etch machine:

```
etch machine=etchmachine spacing={0.05 0.3 0.05} time=0.5
```

This command simulates an etching process on the structure, using the machine `etchmachine` with a grid spacing of $dx=0.05 \mu\text{m}$, $dy=0.3 \mu\text{m}$, $dz=0.05 \mu\text{m}$, and a simulation time of 0.5 minutes.

Note:

The resolution is adjusted internally to match the actual structure.

Etching process using a shape:

```
etch shape=shape_1 comment="etch shape"
```

This command etches the shape `shape_1` from the structure. The comment for this process step is displayed in the log file.

Etching process using a mask:

```
etch mask=mask_1 thickness=0.1 comment="etch mask"
```

This command uses the mask `mask_1` to etch $0.1 \mu\text{m}$, measured relative to the exposed surface of the structure. The comment for this process step is displayed in the log file.

Etching process with intermediate surface plotting:

```
etch machine=etchmachine spacing={0.1} time=0.5 \
    file=etch_surfaces.tdr plot_interval=0.1
```

This command etches using the machine `etchmachine`, with a grid spacing of $dx=dy=dz=0.1 \mu\text{m}$. Intermediate surfaces along with data such as surface velocities and fluxes are plotted into the file `etch_surfaces.tdr`. The surface of the first and last time steps of the simulation will always be plotted. During the simulation, whenever the

Chapter 6: Input Commands

etch

simulation time surpasses an integral multiple of `plot_interval` (for example, 0.1, 0.2, 0.3, ...), a surface will be plotted to file. This feature is useful to check process parameters and model behavior.

Use the fast implicit level-set engine (available only with the `simple` model, which uses a uniform level-set mesh, where the spacing is set by the minimum coordinate of the `spacing` parameter):

```
define_etch_machine name=etchmachine model=simple
add_material machine=etchmachine material=Silicon rate=1.0 \
    anisotropy=0.9 curvature=0.0
etch spacing={ 0.002 0.002 0.002 } time=1.0 levelset_engine=implicit
```

Stopping Time-Stepping

This section discusses how to stop time-stepping.

Level Set Models

The `stop_plane` and `stop_point` parameters of the `etch` command are used in the same way as for the `deposit` command (see [Stopping Time-Stepping on page 336](#)).

The only differences are the constraints on the values specified with `stop_plane` and `stop_point`:

- When using `stop_plane`, the specified plane must intersect the computational domain. In addition, the specified plane must be below the initial exposed surface and must not intersect any voids.
- When using `stop_point`, the specified points must lie in the computational domain. In addition, the specified points must be below the exposed surface and must not lie inside any void.

PMC Models

For `method=pmc`, in addition to `stop_plane` and `stop_point`, the `stop_material` parameter is available.

TDR Dataset Names for Fluxes

For models that calculate ion or neutral fluxes, the values of the fluxes are written to TDR datasets when using the parameter `plot_interval`.

Chapter 6: Input Commands

etch

The name of a flux dataset starts with the name of the flux species. If the flux takes a yield function into account, it is followed by `yield`.

The type of flux is indicated by one of the following suffixes:

<code>direct</code>	The direct flux that arrives from the plasma at a surface element.
<code>reactive</code>	The incoming flux at a surface element that is available for reemission.
<code>reflected</code>	The direct flux that has been reflected away from a surface element.
<code>reflection</code>	The incoming flux at a surface element that has been reflected from other surface elements.
<code>total</code>	Sum of the reactive flux and the incoming indirect flux caused by the reactive flux.

Variance Reduction in PMC Simulations

PMC simulations use pseudo-random numbers to compute the evolution of the structure under the specified process conditions. When the sequence of pseudo-random numbers changes (for example, due to a different scheduling of the threads by the operating system), the results might also change, to an extent that depends on the model used and on the simulation parameters. This variability can be reduced by running the same simulation multiple times independently and by merging the different results.

You can control the number of PMC simulations to be run and merged together using the `averaging_runs` parameter. In addition, you can average the results at the end of the simulation or after a certain amount of time, which can be specified by the `averaging_interval` parameter.

For example, the following command runs a PMC simulation of a 1 minute etching process 8 times and takes the average of the eight runs as the final result:

```
etch time=1 spacing=0.01 method=pmc averaging_runs=8
```

The following command splits the simulation time into five intervals, each of which is 0.2 minutes long:

```
etch time=1 spacing=0.01 method=pmc averaging_runs=8 \
averaging_interval=0.2
```

The evolution of the structure is simulated 8 times over each time interval, and the average of the obtained results is used as the initial structure for the simulation over the next interval.

When starting Sentaurus Topography 3D with a value of the `--processes` command-line option that is greater than 1, execution of the PMC-averaging runs is distributed over

different processes using MPI. The processes run on the hosts specified with the `--mpi-file` command-line option (see [From the Command Line on page 17](#)).

Sentaurus Topography 3D tries to distribute the averaging runs evenly across the available processes, in a round-robin manner. When all PMC simulations are finished, the main process collects the results and averages them.

In addition, you can activate shared-memory parallelization of each PMC-averaging simulation (see [Basic Shared-Memory Parallelization on page 40](#) and [Advanced Shared-Memory Parallelization Options on page 41](#)).

Note:

For best performance, consider the following recommendations:

- Use a number of processes and hosts equal to the value of the `averaging_runs` parameter.
- The number of threads used on each node must not exceed the number of CPU cores available on the node. Use the `--threads` and `--max_threads` command-line options, or the `num_threads` parameter of the `let` command, to set the number of threads used by each process. Use the host file specified with the `--mpi-file` command-line option to specify the number of processes to run on each host. See [From the Command Line on page 17](#) and [let on page 440](#).

Chapter 6: Input Commands

extend_structure

extend_structure

This command extends a structure by mirroring it, or by copying and shifting it.

The initial structure and the extended structure merge automatically into one structure.

Note:

When you extend a structure by setting `type=shift`, the resulting structure might contain a path that connects the top gas with the bottom plane of the simulation domain, without passing through any other material. These types of structure are invalid for machine-based etching and deposition processes. There is no such problem with structures extended by mirroring.

Syntax

Extend a boundary structure:

```
extend_structure plane=<c> type=mirror \
[accuracy=<n>] [decimate=<b>] [min_angle=<n>] \
[min_dihedral_angle=<n>] [ridge_angle=<n>] \
[shortest_edge=<n>] [structure=<c>]

extend_structure plane=<c> type=shift \
[accuracy=<n>] [decimate=<b>] [min_angle=<n>] \
[min_dihedral_angle=<n>] [ridge_angle=<n>] \
[shortest_edge=<n>] [structure=<c>] [times=<n>]
```

Extend a PMC structure:

```
extend_structure plane=<c> type=mirror [structure=<c>]

extend_structure plane=<c> type=shift [structure=<c>] [time=1]
```

Note:

When you start Sentaurus Topography 3D with the `--processes` command-line option with a value greater than 1 (see [Table 1 on page 17](#)), the parameter `structure` must not denote a structure obtained from the `etch` command using `method=pmc averaging_runs=1`.

Chapter 6: Input Commands

`extend_structure`

Table 59 Parameters of `extend_structure` command

Parameter	Type	Description	Default [Range]	Unit
accuracy	Number	<p>Sets the maximum deviation between the decimated surface of the boundary structure produced by Boolean operations and its original surface.</p> <p>When set to -1, the used value is half of the structure-dependent spacing <code>epsilon</code>¹.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> The <code>structure</code> parameter denotes a boundary structure. You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_accuracy=<n></code> $\{-1\} \cup [0, \infty[$	µm
decimate	Boolean	Specifies whether to decimate the boundary structure produced by Boolean operations. It can be specified only if <code>structure</code> denotes a boundary structure.	The value set by <code>let decimate=</code>	none
min_angle	Number	<p>Sets the smallest angle in the elements of the decimated surface of the boundary structure produced by Boolean operations.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> The <code>structure</code> parameter denotes a boundary structure. You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_min_angle=<n></code> $[0, 180]$	degree

Chapter 6: Input Commands

extend_structure

Table 59 Parameters of extend_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
min_dihedral_angle	Number	<p>Sets the smallest dihedral angle between two elements of the decimated surface of the boundary structure produced by Boolean operations that share an edge.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> The structure parameter denotes a boundary structure. You specify decimate=true or, if you do not specify decimate, you did not specify let decimate=false. 	The value set by let decimate_min_dihedral_angle=<n> [0, 180]	degree
plane	Character	<p>Sets the side where the initial structure is extended (see Figure 28 on page 373). Options are:</p> <ul style="list-style-type: none"> back front left right <p>Note: The values back and front apply only to 3D structures.</p>	none	none
ridge_angle	Number	<p>Sets the angle used by the decimation algorithm to determine geometric features of the boundary structure produced by Boolean operations.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> The structure parameter denotes a boundary structure. You specify decimate=true or, if you do not specify decimate, you did not specify let decimate=false. 	The value set by let decimate_ridge_angle=<n> [0, 180]	degree

Chapter 6: Input Commands

extend_structure

Table 59 Parameters of extend_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
shortest_edge	Number	<p>Sets the shortest edge of the decimated surface of the boundary structure produced by Boolean operations.</p> <p>When set to -1, the used value is half of the structure-dependent spacing epsilon¹.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> The structure parameter denotes a boundary structure. You specify decimate=true or, if you do not specify decimate, you did not specify let decimate=false. 	The value set by let decimate_shortest_edge=<n> {-1} ∪ [0, ∞[μm
structure	Character	<p>Sets the name of the structure that is extended.</p> <p>Note: This parameter cannot specify a PMC structure. To process such a structure, an explicit conversion of the structure is needed before calling the extend_structure command (see Integration With Other Sentaurus Topography 3D Functionality on page 519).</p>	default_structure	none
times	Number	Sets the number of times the initial structure is copied and shifted.	1 [1, ∞[none
type	Character	Sets the operation used to extend the initial structure. Options are: <ul style="list-style-type: none"> mirror: Mirror initial structure. shift: Copy and shift initial structure. 	none	none

1. The structure-dependent spacing epsilon is logged out when defining a structure using the define_structure command.

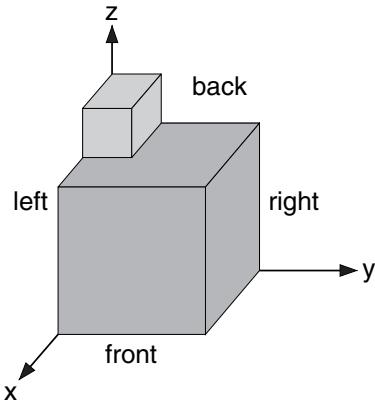
Chapter 6: Input Commands

extend_structure

Examples

[Figure 28](#) shows the initial structure used for these examples.

Figure 28 Initial structure and different values of the plane parameter

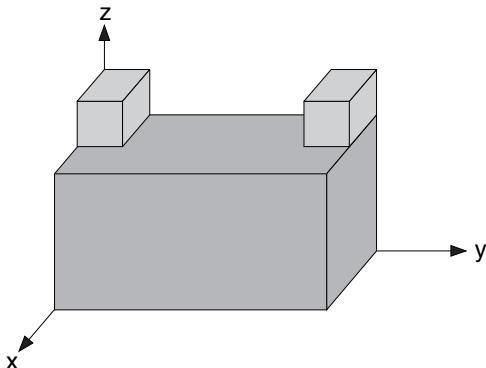


Extend the initial structure by mirroring at the xz plane at the maximum y -coordinate of the simulation domain:

```
extend_structure type=mirror plane=right
```

[Figure 29](#) shows the result of this command to the structure in [Figure 28](#).

Figure 29 Extending structure by mirroring



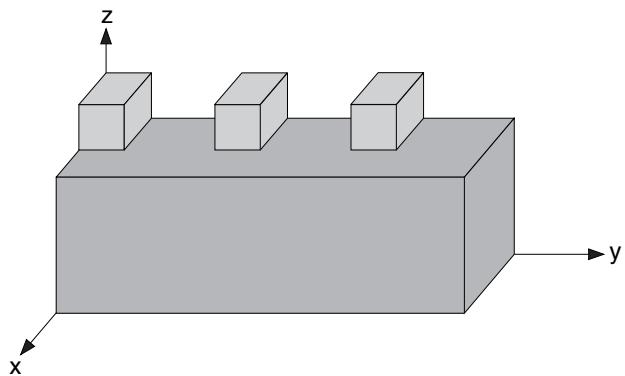
Extend the default structure by copying the default structure and shifting the copy to the xz plane at the maximum y -coordinate of the simulation domain. Then, a second copy is shifted to the maximum y -coordinate of the intermediate structure created by adding the first copy (see [Figure 30](#)):

```
extend_structure type=shift plane=right times=2
```

Chapter 6: Input Commands

extend_structure

Figure 30 Extending structure by copying and shifting twice



extract

This command extracts different properties of a structure and can be used more than once in the same command file.

The first time that you use the `extract` command with `type=1d_cut`, `type=2d_cut`, `type=interface`, `type=probe`, or `type=slice`, an extract file is created with the name specified with the parameter `file`. In subsequent uses of the command with the same value of the parameter `file`, the results are added to the previously computed and saved extractions. Each extraction has a name that is provided by the parameter `name`.

Note:

The `extract` command can extract properties from a structure only before or after a processing step. To extract properties during processing, see [define_extraction on page 218](#).

You can extract the following:

- [1D or 2D Cuts](#)
- [Properties of Exposed Surfaces](#)
- [Interface Area](#)
- [Areas and Volumes of Regions and Parts](#)
- [Region Names and Materials](#)
- [Region and Part Names](#)
- [Interface Position](#)
- [Intersections With a Line](#)
- [Intersections With a Plane](#)
- [Shortest Distance](#)
- [Shape Analysis](#)
- [Cylindrical Hole Profile](#)
- [Bounding Box of Materials and Regions](#)
- [Vertical Coordinates of the Top and Bottom of the Bounding Box of Materials and Regions](#)
- [Dimension of Structure](#)

Chapter 6: Input Commands

extract

- [Damage Integral](#)
- [Surface Coverage](#)

Specifying Extraction Lines, Planes, and Pairs

For some extractions, an extraction line, an extraction plane, or extraction pairs of materials or regions must be specified.

An extraction line can be specified in these different ways:

- Axis-aligned direction and one point:

```
axis=<c> point=<v>
```

- Arbitrary direction and one point:

```
direction=<v> point=<v>
```

- Two points:

```
point1=<v> point2=<v>
```

An extraction plane can be specified in these different ways:

- Axis-orthogonal plane at axis position:

```
axis=<c> position=<n>
```

- Arbitrary plane-normal direction and one point:

```
normal=<v> point=<v>
```

- Generic plane defined by three points:

```
point1=<v> point2=<v> point3=<v>
```

An extraction pair can be specified using pairs of identifiers:

- For two different materials:

```
material1=<c> material2=<c>
```

- For two different regions:

```
region1=<c> region2=<c>
```

- For two different parts:

```
region_part1=<l> region_part2=<l>
```

Chapter 6: Input Commands

extract

The names of regions can be extracted with `type=region_names` (see [Region Names and Materials on page 381](#)). Lists of region and part names can be extracted with `type=region_parts` (see [Region and Part Names on page 382](#)).

In the following syntax descriptions, the specifications for an extraction line, an extraction plane, and pairs of material or region are represented by `<extraction_line>`, `<extraction_plane>`, and `<extraction_pair>`, respectively.

Extraction Types

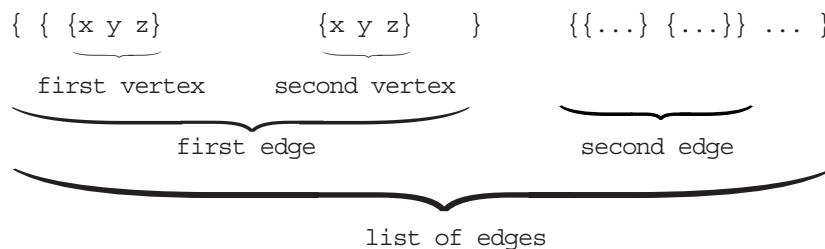
The extraction type is set with the parameter `type`.

1D or 2D Cuts

A `1d_cut` intersects the closed surface of a PMC structure or a boundary structure (defined in [Boundary Types on page 30](#)) with a line (the parameter `structure` must denote a boundary structure):

```
extract type=1d_cut <extraction_line> [file=<c>] [name=<c>] \
[silent=<b>] [structure=<c>]
```

The data returned by `type=1d_cut` is a list of edges represented in Tcl with the following format:



where:

- A vertex is a list of two or three floating-point numbers, depending on the dimension of the structure.
- An edge is a list of exactly two vertices.
- A 1D cut is a list of edges.

A `2d_cut` intersects the closed surface of a boundary structure with a plane (the parameter `structure` must denote a boundary structure):

```
extract type=2d_cut <extraction_plane> [file=<c>] [name=<c>] \
[silent=<b>] [structure=<c>]
```

Chapter 6: Input Commands

extract

Example: 1d_cut

```
extract axis=x point={-0.5 0.5 1.3} type=1d_cut
```

A line passing through the point (-0.5 0.5 1.3) and with a direction parallel to the x-axis is created and intersected with the closed surface. All the intersection points are written into a TDR file.

Extract the third coordinate of the second vertex of the second edge of a 1D cut:

```
set 1dcut = [extract axis=z point={0.5 0.5 0} type=1d_cut]
set edge1 [lindex $1dcut 1]
set vertex1 [lindex $edge1 1]
set coord2 [lindex $vertex1 2]
```

Example: 2d_cut

```
extract axis=z position=0.5 type=2d_cut
```

A plane with normal parallel to the z-axis and intersecting it at $z = 0.5$ is created and intersected with the closed surface. The intersection points are saved in a TDR boundary file. Every intersection between the plane and the closed surface has a name that is provided by the parameter `name`.

Note:

If not specified explicitly by setting the parameter `file`, 1D and 2D extractions are saved in the same TDR file.

Properties of Exposed Surfaces

To extract the bounding box of the exposed surface of a PMC structure or a boundary structure, use the command:

```
extract type=bounding_box_exposed [name=<c>] [silent=<b>] \
[structure=<c>]
```

It returns a list of the minimum and maximum vertices of the bounding box of the exposed surface.

To extract the minimum vertical coordinate of the exposed surface of a PMC structure or a boundary structure, use the command:

```
extract type=bottom_exposed [name=<c>] [silent=<b>] [structure=<c>]
```

To extract the maximum vertical coordinate of the exposed surface of a PMC structure or a boundary structure, use the command:

```
extract type=top_exposed [name=<c>] [silent=<b>] [structure=<c>]
```

Chapter 6: Input Commands

extract

Example: bottom_exposed

```
extract type=bottom_exposed
```

This command extracts and outputs the minimum vertical coordinate of the bounding box of the exposed surface.

Example: bounding_box_exposed

```
extract type=bounding_box_exposed
```

This command extracts and outputs a list containing the minimum and maximum vertices of the bounding box of the exposed surface.

Example: top_exposed

```
extract type=top_exposed
```

This command extracts and outputs the maximum vertical coordinate of the bounding box of the exposed surface.

Interface Area

To extract the area of the interface between two regions of a boundary structure (the parameter `structure` must denote a boundary structure), use the command:

```
extract region_1=<c> region_2=<c> type=interface_area \
[name=<c>] [silent=<b>] [structure=<c>]
```

It returns the total area of the interface between the two specified regions.

Example: interface_area

```
extract type=interface_area region_1=substrate \
region_2=source_contact
```

This command extracts the area formed by the interface between the regions named `substrate` and `source_contact`.

The following commands show how to extract the interface area and assign it to a Tcl variable. Here, the output generated by the `extract` command is suppressed by setting `silent=true`:

```
set contact_area [extract type=interface_area region_1=substrate \
region_2=source_contact silent=true]
puts "The source contact area is $contact_area"
```

Chapter 6: Input Commands

extract

Areas and Volumes of Regions and Parts

To extract geometric properties:

- Extraction of the surface area of a region of a boundary structure (the parameter structure must denote a boundary structure):

```
extract region=<c> type=area [name=<c>] [silent=<b>] \
[structure=<c>]
```

Returns the total surface area of the specified region.

- Extraction of the surface areas of the parts of a region of a boundary structure (the parameter structure must denote a boundary structure):

```
extract region=<c> type=part_area [name=<c>] [silent=<b>] \
[structure=<c>]
```

Returns the surface areas of all disconnected parts of the specified region as a Tcl list.

- Extraction of the volumes of the parts of a region of a boundary structure (the parameter structure must denote a boundary structure):

```
extract region=<c> type=part_volume [name=<c>] [silent=<b>] \
[structure=<c>]
```

Returns the volumes of all disconnected parts of the specified region as a Tcl list.

- Extraction of the volume of a region of a boundary structure (the parameter structure must denote a boundary structure):

```
extract region=<c> type=volume [name=<c>] [silent=<b>] \
[structure=<c>]
```

Returns the total volume of the specified region.

Example: area

```
extract type=area region=insulation
```

This command extracts the surface area of the region named insulation.

Example: part_area

```
extract type=part_area region=insulation
```

This command extracts the surface area of the individual parts of the region named insulation and creates the output separately.

Chapter 6: Input Commands

extract

Example: part_volume

```
extract type=part_volume region=insulation
```

This command extracts the volume of the individual parts of the region named `insulation` and creates the output separately.

Example: volume

```
extract type=volume region=bulk
```

This command extracts and outputs the volume of the `bulk` region of the processed structure.

Region Names and Materials

To extract region names and materials:

- Extraction of the material names of a boundary structure (the parameter `structure` must denote a boundary structure):

```
extract type=material_names [exposed_only=<b>] [name=<c>] \
[silent=<b>] [structure=<c>]
```

Returns the names of the materials of a structure as a Tcl list.

- Extraction of the material names of a PMC structure (the parameter `structure` must denote a PMC structure):

```
extract type=material_names exposed_only=false [name=<c>] \
[silent=<b>] [structure=<c>]
```

Returns the names of the materials of a structure as a Tcl list.

- Extraction of region materials of a boundary structure (the parameter `structure` must denote a boundary structure):

```
extract region=<c> type=material_name [exposed_only=<b>] \
[name=<c>] [silent=<b>] [structure=<c>]
```

Returns the name of the material of the specified region.

- Extraction of region names of a boundary structure (the parameter `structure` must denote a boundary structure):

```
extract type=region_names [exposed_only=<b>] [name=<c>] \
[silent=<b>] [structure=<c>]
```

Returns the names of the regions of a structure as a Tcl list.

Chapter 6: Input Commands

extract

Example: material_name

```
extract type=material_name region=insulation
```

This command extracts and outputs the name of the material of the insulation region.

Example: region_names

```
extract type=region_names
```

This command extracts and outputs the name of each region of the processed structure.

Region and Part Names

To extract the names of regions and the names of their parts:

- Extraction of all region and part names of a boundary structure (the parameter structure must denote a boundary structure):

```
extract type=region_parts [exposed_only=<b>] [name=<c>] \
[silent=<b>] [structure=<c>]
```

Returns a list of pairs. The first element of each pair is the name of a region of the structure and the second element is a list of the names of the parts contained in that region.

- Extraction of part names of a region of a boundary structure (the parameter structure must denote a boundary structure):

```
extract region=<c> type=region_parts [exposed_only=<b>] \
[name=<c>] [silent=<b>] [structure=<c>]
```

Returns a list containing the names of the parts in the given region.

Example: region_parts

```
extract type=region_parts
```

This command extracts and outputs the name of each part of each region of the processed structure.

The parts of a specific region can be extracted by specifying the region parameter:

```
extract type=region_parts region=mask
```

The name of each part of the region named mask is extracted and output separately.

Interface Position

To locate an interface between two different materials, or two different regions, or two different parts along a line in a structure, as well as to extract a vertex for the interface that

Chapter 6: Input Commands

extract

the line passes through or a segment of the interface that the line is tangent to, use the command:

```
extract type=interface <extraction_line> <extraction_pair> \
[name=<c>] [output=<c>] [silent=<b>] [structure=<c>]
```

If you specify the `material1` and `material2` parameters, the special name `Gas` can be used to represent not only explicitly available gaseous regions, but also the empty space at the exposed surface. The parameters `material1` and `material2` are supported when the `structure` parameter denotes a boundary structure and when it denotes a PMC structure.

If you want to find interfaces between two specific but different regions of a boundary structure, use the `region1` and `region2` parameters. In this case, the `structure` parameter denotes a boundary structure.

To find the interfaces between two specific but different parts of a boundary structure, use the `region_part1` and `region_part2` parameters. In this case, the `structure` parameter denotes a boundary structure.

You can use the `output` parameter to filter the computed interface positions. It can have the following values:

- `output=all`: Along the input line, returns all interfaces in the output format described below. This is the default.
- `output=first`: Returns only the first interface.
- `output=inside`: Returns only the vertices or segments that are fully contained within or exactly on the `point1` to `point2` input segment. Only available if the `extract` command is specified with `point1` and `point2`.
- `output=last`: Returns only the last interface.
- `output=outside`: Returns only the vertices or segments that do not overlap the `point1` to `point2` input segment. Only available if the `extract` command is specified with `point1` and `point2`.

The data returned by `type=interface` is structured as follows:

```
{ {x y z} {x y z} {x y z x y z} ... }
```

Each element of this list represents either a vertex or a segment.

The exposed surface is treated specially with `type=interface`. An interface is returned for the exposed surface even if the input structure does not contain a gas region that touches the exposed surface. The material name `Gas` can always be used for one of the two parameters `material1` or `material2`, irrespective of whether the structure actually contains a gas region touching the exposed surface. On the other hand, specifying an interface using either `region1` and `region2`, or `region_part1` and `region_part2`, with an assumed gas region is not allowed.

Chapter 6: Input Commands

extract

Example: interface

```
extract type=interface material1=Silicon material2=Oxide \
    point={0.5 0.5 0} direction={0 0 1}
```

This command extracts and outputs the interfaces between silicon and oxide in the processed structure along the specified line.

Intersections With a Line

Specifying `type=probe` intersects a structure with a line and returns a segment for each part of a region that the line passes through as well as a property associated with that segment, namely, the material name, or the region and part name:

```
extract type=probe <extraction_line> \
    (materials=<l> | [probe_empty_space=<b>]) \
    [name=<c>] [output=<c>] [property=<c>] [silent=<b>] [structure=<c>]
```

The `property` parameter specifies the data to be associated with each extracted segment in the returned value. This data refers to the part of a region where a segment comes from:

- `property=length`: The total length of the segments lying on the extraction line that go through the specified materials or through empty space. This property can be specified both when the `structure` parameter denotes a boundary structure and when it denotes a PMC structure.

If `probe_empty_space=false`, the total length of the segments lying on the extraction line that go through any material except `Gas` or empty space is returned.

If `probe_empty_space=true`, the total length of the segments lying on the extraction line that go through the material `Gas` or empty space is returned.

If the parameter `materials` is specified, the total length of the segments lying on the extraction line that go through the specified materials is returned.

- `property=material`: The material name of the underlying part of a region. This is the default and can be specified both when the `structure` parameter denotes a boundary structure and when it denotes a PMC structure.
- `property=region`: The region name of the underlying part. This property can be specified only when the `structure` parameter denotes a boundary structure.
- `property=region_part`: A list of two elements. The first element is the region name of the underlying part, and the second element is the name of the underlying part itself. The name of the underlying part is always a number. This property can be specified only when the `structure` parameter denotes a boundary structure.

Chapter 6: Input Commands

extract

The `output` parameter can have the following values:

- `output=all`: Along the input line, returns all segments within regions in the output format described below. This is the default.
- `output=first`: Returns only the first entry.
- `output=inside`: Returns only the segments that are fully contained within or exactly on the `point1` to `point2` input segment. Only available if the `extract` command is specified with `point1` and `point2`.
- `output=last`: Returns only the last entry.
- `output=outside`: Returns only the segments that do not overlap the `point1` to `point2` input segment. Only available if the `extract` command is specified with `point1` and `point2`.

For `property=length`, the data returned by `type=probe` is a number that represents the sum of the length of the segments that lie in the specified materials or in empty space.

For `property=material`, `property=region`, and `property=region_part`, the data returned by `type=probe` is structured as follows:

```
{}{{x y z} {x y z} property} {}{{x y z} {x y z} property} ...}
```

The returned `property` is enclosed in double quotation marks if it contains spaces.

Example: probe

```
extract type=probe point1={0 0 0} point2={0 0 1} property=material
```

The processed structure is intersected with a line passing through the points of the coordinates `{0 0 0}` and `{0 0 1}`.

Intersections With a Plane

Specifying `type=slice` intersects a 3D boundary or PMC structure with a plane. The obtained 2D cut represents a 2D boundary that contains information about materials. The 2D boundary is saved into a TDR file:

```
extract type=slice <extraction_plane> [file=<c>] [name=<c>] \
[silent=<b>] [structure=<c>]
```

The obtained 2D cut is saved into a TDR file specified using the `file` parameter. The `structure` parameter must denote either a 3D boundary or PMC structure.

The saved structure can be used as input for a 2D simulation if the exposed surface of the structure (excluding voids) does not touch the bottom of the simulation domain (see [Boundary Types on page 30](#)). In other words, if you imagine pouring a liquid onto the

Chapter 6: Input Commands

extract

structure, and the liquid does not reach the bottom of the rectangular domain, the structure is valid input for a simulation.

Example: slice

```
extract type=slice point={0 0 0} normal={0 1 0} file=slice.tdr
```

The processed structure is intersected with a plane having its normal direction parallel to the y-axis and passing through the point of the coordinates {0 0 0}. The 2D extracted boundary is saved in the TDR file named slice.tdr.

Shortest Distance

To extract the shortest distance between two different materials, or two different regions, or two different parts in a boundary structure (the `structure` parameter must denote a boundary structure), use the command:

```
extract type=shortest_distance <extraction_pair> \
[name=<c>] [silent=<b>] [structure=<c>]
```

The `material1` and `material2` parameters specify two different materials between which the shortest distance is sought.

The `region1` and `region2` parameters specify two different regions between which the shortest distance is sought. These two regions do not have to contain two different materials as in the case of the `material1` and `material2` parameters. All available parts of the two regions are considered in the search.

You can use `type=region_names` to extract a list of region names in a structure.

To find the shortest distance between two specific but different parts, use the `region_part1` and `region_part2` parameters.

You can use `type=region_parts` to extract a list of region and part names of a structure.

The data returned by `type=shortest_distance` is a list with the following two elements (an empty list is returned if one of the specified materials, regions, or parts does not exist in the structure, and a warning is issued):

```
{shortest_distance, segments}
```

The first element, `shortest_distance`, is the shortest distance measured in μm . The second element, `segments`, indicates where the shortest distance occurs. Since the shortest distance can occur at more than one location, `segments` is a list in the following format:

```
{segment1, ...} = { {vertex1, vertex2}, ... } = { { {x, y, z},  
{x, y, z} }, ... }
```

Chapter 6: Input Commands

extract

The list contains only one segment if the shortest distance occurs uniquely at one specific location in the structure. When the shortest distance occurs at several locations, the number of segments depends on the discretization of the structure.

Example: `shortest_distance`

```
extract type=shortest_distance region_part1={mask 0} \
        region_part2={mask 1}
```

This command computes and returns the shortest distance between the parts named 0 and 1 of the `mask` region along with the segments where the shortest distance occurs. The command returns a list whose first element is the requested shortest distance.

Example of the extraction of the shortest distance from the returned value:

```
set res [extract type=shortest_distance region_part1={mask 0} \
                  region_part2={mask 1}]

set shortest_distance [lindex $res 0]
```

Shape Analysis

When you specify `type=shape_analysis`, the shape of a structure is analyzed and compared to the given reference shape (set by `reference_shape`). The `structure` parameter must denote a structure resulting from a level set or PMC simulation, and you must define a suitable domain containing the entire substructure to be analyzed.

For `reference_shape=cylinder_hole`, the syntax would be:

```
extract type=shape_analysis reference_shape=cylinder_hole \
        max_radius=<n> point1=<v> point2=<v> \
        [bottom_shift=<n>] [csv_file=<c>] [doe_parameters=<l>] \
        [extract_surface=<b>] [interface=<l>] [name=<c>] [output_type=<c>] \
        [smoothing_order=<n>] [structure=<c>] \
        [tdr_file=<c>] [top_shift=<n>] [vertical_spacing=<n>] [z_planes=<l>]
```

For `reference_shape=trench`, the syntax would be:

```
extract type=shape_analysis reference_shape=trench \
        max_depth=<n> max_width=<n> point1=<v> point2=<v> \
        [bottom_shift=<n>] [csv_file=<c>] [doe_parameters=<l>] \
        [extract_surface=<b>] [interface=<l>] [name=<c>] [output_type=<c>] \
        [smoothing_order=<n>] [structure=<c>] \
        [tdr_file=<c>] [top_shift=<n>] [vertical_spacing=<n>] [z_planes=<l>]
```

Different reference shapes are available:

- Set `reference_shape=cylinder_hole` to analyze a cylindrical hole structure.

You must specify a suitable cylindrical extraction window (a bounding box) around the structure of interest. The cylindrical bounding box is specified by two points (`point1` and

Chapter 6: Input Commands

extract

`point2`) that define the bottom and top of the vertical cylinder axis, and by the radius (`max_radius`) of the cylinder (see [Figure 31 \(left\)](#)).

Note:

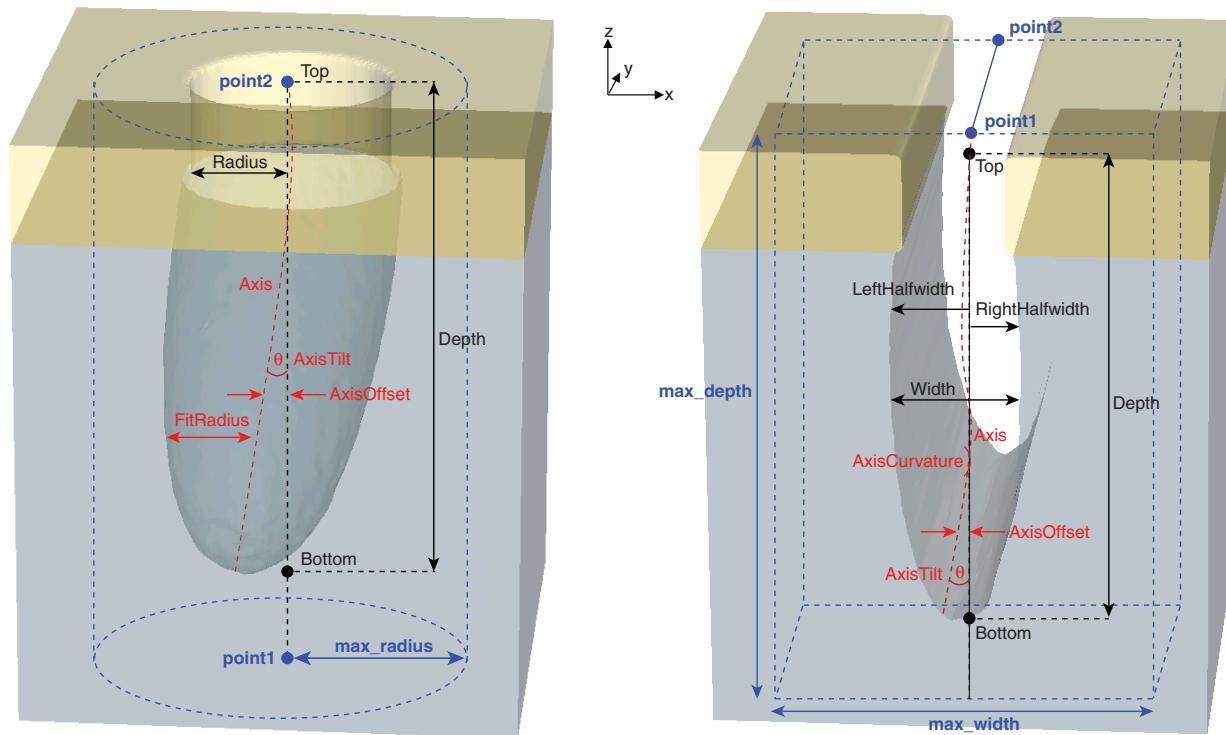
Only vertically aligned cylindrical bounding boxes are supported, that is, the x- and y-coordinates of `point1` and `point2` must coincide.

- Set `reference_shape=trench` to analyze a trench structure.

You must specify a suitable cuboid bounding box around the structure of interest. The orientation of the trench in the xy plane is specified by two points (`point1` and `point2`) that define the upper center line of the trench (see [Figure 31 \(right\)](#)). In addition, `point1` and `point2` must lie in the same xy plane (that is, their z-coordinates must coincide), but they can be arbitrarily oriented in the xy plane.

At the same time, `point1` and `point2` define the top and the orientation of the bounding box, which can have an arbitrary orientation in the xy plane, but must be vertically aligned with the z-axis. After you set `point1` and `point2`, set the width and depth of the bounding box by using the `max_width` and `max_depth` parameters.

Figure 31 Definition of extraction window for shape analysis when (left) reference_shape=cylinder_hole and (right) reference_shape=trench



Chapter 6: Input Commands

extract

Data can be output in different ways:

- *Scalar data*, such as the hole depth, average tilt angle, bottom radius, top radius, and average hole radius, is returned as *design-of-experiments (DOE) parameters* to Sentaurus Workbench. You activate this feature by setting `output_type=doe`.

Optionally, you can specify the DOE parameters of interest using the parameter `doe_parameters={parameter1 parameter2 ...}`. If multiple extractions are executed, the name of the extractor (specified by parameter `name`) is prepended to the DOE name to distinguish different extractions. [Table 60 on page 391](#) lists the available DOE parameters for different values of `reference_shape`.

- *One-dimensional datasets*, such as the hole radius as a function of the z-axis, are written to a TDR file if `output_type=tdr` is set. By default, the file name is the base name of the command file with the suffix `_extract.tdr`. Alternatively, you can specify a file name with the parameter `tdr_file`. The resulting file can be visualized in Sentaurus Visual. [Table 61 on page 396](#) lists the available datasets for different values of `reference_shape`.

You can store datasets in a comma-separated value (CSV) format by setting `output_type=csv` with the same naming convention as for TDR files, but with the suffix `_extract.csv`. A different file name can be specified with the parameter `csv_file`.

- *Surface points*: If you set `extract_surface=true`, then you can extract the surface points of a hole or trench in the xy plane at different heights z, specified by a list of z-coordinates in parameter `z_planes` (see [Figure 32 on page 390](#)). The surface line is returned as a list of surface points in Cartesian coordinates (datasets `x, y`). For holes, the surface points are also given in polar coordinates (datasets `Radius, Theta`), measured relative to the fitted hole axis at the respective height z.

In addition, the extractor returns the coordinates of an ellipse fitted to the hole cross-section (datasets `x_fit, y_fit, Radius_fit`). If `output_type=tdr`, then the surface points are written to a TDR file for plotting in Sentaurus Visual. If `output_type=csv`, then the datasets are written to a CSV file with the same naming convention as for TDR files, but with the suffix `_surface.csv`.

- Extracted data can be returned by the `extract` command as a Tcl list, which is formatted in the following way:

```
// DOE parameters (scalar quantities)
{
    { Depth value1 }
    { Tilt   value2 }
    ...
}
// Datasets (1D z-dependent quantities)
{
    { Z           { value1 value2 value3 ... } }
    { Radius_Avg { value1 value2 value3 ... } }
```

Chapter 6: Input Commands

extract

```
{ Radius_SD { value1 value2 value3 ... } }
...
}
// Surface points at given xy planes
{
{ Z { value1 value2 value3 ... } }
{ X { value1 value2 value3 ... } }
{ Y { value1 value2 value3 ... } }
...
}
```

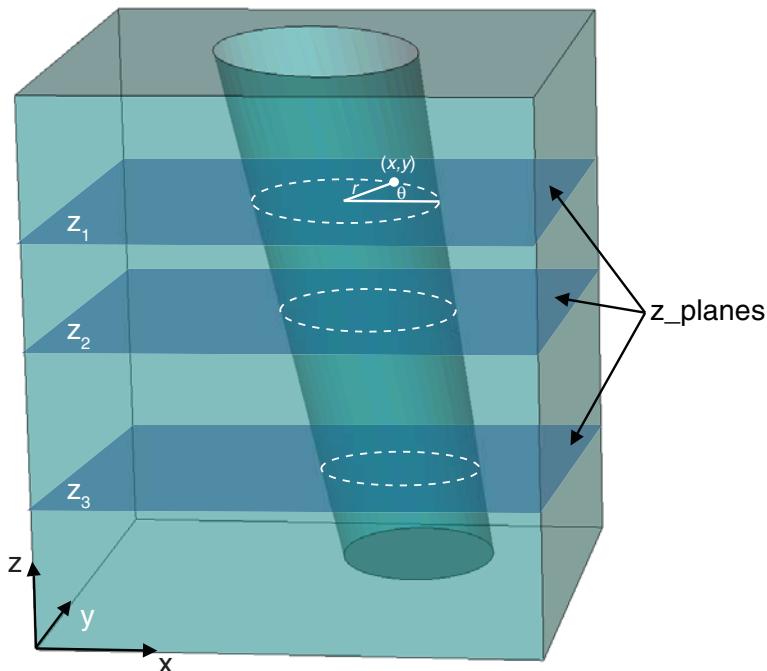
You can also write more than one output type simultaneously by setting `output_type` to a list of values. For example:

```
output_type={doe csv tdr}
```

If `output_type={}` is empty, no DOE or file output is generated; only the extracted data is returned as a Tcl list.

Depending on the size of the extraction window, more or fewer surface points of the structure are considered for the computation of the profile statistics. Therefore, by increasing the extraction window (for example, by increasing the distance between `point1` and `point2` for trenches in [Figure 31 on page 388](#)), statistical noise can be reduced, since more surface points are considered for the statistics.

Figure 32 Extracting surface points of a hole or trench in the xy plane at different heights z



Chapter 6: Input Commands

extract

Depending on the morphology of the structure, it might also be useful to apply a smoothing on the profile data by setting the `smoothing_order` parameter.

Table 60 *DOE parameters for shape analysis*

DOE parameter	Description
Parameters common to both <code>reference_shape=cylinder_hole</code> and <code>reference_shape=trench</code>	
<code>AxisCurvature_Avg</code>	Average axis curvature along the z-axis.
<code>AxisCurvature_Max</code>	Maximum axis curvature along the z-axis.
<code>AxisCurvature_Med</code>	Median of the axis curvature along the z-axis.
<code>AxisCurvature_Min</code>	Minimum axis curvature along the z-axis.
<code>AxisCurvature_SD</code>	Standard deviation of the axis curvature along the z-axis.
<code>AxisOffset_Avg</code>	Average axis offset along the z-axis.
<code>AxisOffset_Max</code>	Maximum axis offset along the z-axis.
<code>AxisOffset_Med</code>	Median of the axis offset along the z-axis.
<code>AxisOffset_Min</code>	Minimum axis offset along the z-axis.
<code>AxisOffset_SD</code>	Standard deviation of the axis offset along the z-axis.
<code>AxisTilt_Avg</code>	Average axis tilt angle along the z-axis.
<code>AxisTilt_Max</code>	Maximum axis tilt angle along the z-axis.
<code>AxisTilt_Med</code>	Median of the axis tilt angle along the z-axis.
<code>AxisTilt_Min</code>	Minimum axis tilt angle along the z-axis.
<code>AxisTilt_SD</code>	Standard deviation of the axis tilt angle along the z-axis.
<code>Depth</code>	Depth of the hole or trench.
<code>TiltTheta</code>	Vertical slope of the fitted tilt axis (given by the angle between the tilt axis and the z-axis).
<code>Z_AxisCurvature_Max</code>	The z-coordinate of the maximum axis curvature.
<code>Z_AxisCurvature_Min</code>	The z-coordinate of the minimum axis curvature.

Chapter 6: Input Commands

extract

Table 60 DOE parameters for shape analysis (Continued)

DOE parameter	Description
Z_AxisOffset_Max	The z-coordinate of the maximum axis offset along the z-axis.
Z_AxisOffset_Min	The z-coordinate of the minimum axis offset along the z-axis.
Z_AxisTilt_Max	The z-coordinate of the maximum tilt angle.
Z_AxisTilt_Min	The z-coordinate of the minimum tilt angle.
Z_Bottom	The z-coordinate at which the bottom of the hole or trench is measured. The measurement point can be shifted upwards by specifying <code>bottom_shift</code> in the <code>extract</code> command.
Z_Top	The z-coordinate at which the top of the hole or trench is measured. The measurement point can be shifted downwards by specifying <code>top_shift</code> in the <code>extract</code> command.
Parameters specific to reference_shape=cylinder_hole	
AxisOffsetPhi_Avg	Average direction of the axis offset along the z-axis (given by an angle ϕ relative to the x-axis).
AxisOffsetPhi_SD	Standard deviation of the axis offset direction along the z-axis.
Eccentricity_Avg	Average eccentricity of the elliptic hole along the z-axis.
Eccentricity_Max	Maximum eccentricity of the elliptic hole along the z-axis.
Eccentricity_Med	Median of the eccentricity of the elliptic hole along the z-axis.
Eccentricity_Min	Minimum eccentricity of the elliptic hole along the z-axis.
Eccentricity_SD	Standard deviation of the eccentricity of the elliptic hole along the z-axis.
EllipseRotation_Avg	Average angle between the major elliptic axis and the x-axis (the average is taken along the z-axis).
EllipseRotation_SD	Standard deviation of the angle between the major elliptic axis and the x-axis (measured along the z-axis).
FitRadius_Avg	Average fitted radius of the hole along the z-axis.
FitRadius_Bottom	Average fitted radius at $z=Z_{\text{Bottom}}$.

Chapter 6: Input Commands
extract

Table 60 DOE parameters for shape analysis (Continued)

DOE parameter	Description
FitRadius_Max	Maximum fitted hole radius along the z-axis.
FitRadius_Med	Median of the fitted hole radius along the z-axis.
FitRadius_Min	Minimum fitted hole radius along the z-axis.
FitRadius_SD	Standard deviation of the fitted hole radius along the z-axis.
FitRadius_Top	Average fitted hole radius at $z=Z_{\text{Top}}$.
MajorSemiaxis_Avg	Average major semi-axis of the elliptic hole along the z-axis.
MajorSemiaxis_Max	Maximum major semi-axis of the elliptic hole along the z-axis.
MajorSemiaxis_Med	Median of the major semi-axis of the elliptic hole along the z-axis.
MajorSemiaxis_Min	Minimum major semi-axis of the elliptic hole along the z-axis.
MajorSemiaxis_SD	Standard deviation of the major semi-axis of the elliptic hole along the z-axis.
MinorSemiaxis_Avg	Average minor semi-axis of the elliptic hole along the z-axis.
MinorSemiaxis_Max	Maximum minor semi-axis of the elliptic hole along the z-axis.
MinorSemiaxis_Med	Median of the minor semi-axis of the elliptic hole along the z-axis.
MinorSemiaxis_Min	Minimum minor semi-axis of the elliptic hole along the z-axis.
MinorSemiaxis_SD	Standard deviation of the minor semi-axis of the elliptic hole along the z-axis.
Phi_AxisOffset_Max	Direction of the maximum axis offset along the z-axis (given by an angle ϕ relative to the x-axis).
Phi_AxisOffset_Min	Direction of the minimum axis offset along the z-axis (given by an angle ϕ relative to the x-axis).
Radius_Avg	Average radius of the hole along the z-axis.
Radius_Max	Maximum hole radius along the z-axis.
Radius_Med	Median of the hole radius along the z-axis.

Chapter 6: Input Commands
extract

Table 60 DOE parameters for shape analysis (Continued)

DOE parameter	Description
Radius_Min	Minimum hole radius along the z-axis.
Radius_SD	Standard deviation of the hole radius along the z-axis.
TiltPhi	Orientation of the fitted tilt axis (given by an angle ϕ in the xy plane, measured relative to the x-axis).
Z_Eccentricity_Max	The z-coordinate of the maximum eccentricity of the elliptic hole.
Z_Eccentricity_Min	The z-coordinate of the minimum eccentricity of the elliptic hole.
Z_FitRadius_Max	The z-coordinate of the maximum fitted hole radius.
Z_FitRadius_Min	The z-coordinate of the minimum fitted hole radius.
Z_MajorSemiaxis_Max	The z-coordinate of the maximum major semi-axis of the elliptic hole.
Z_MajorSemiaxis_Min	The z-coordinate of the minimum major semi-axis of the elliptic hole.
Z_MinorSemiaxis_Max	The z-coordinate of the maximum minor semi-axis of the elliptic hole.
Z_MinorSemiaxis_Min	The z-coordinate of the minimum minor semi-axis of the elliptic hole.
Z_Radius_Max	The z-coordinate of the maximum hole radius.
Z_Radius_Min	The z-coordinate of the minimum hole radius.
Parameters specific to reference_shape=trench	
Halfwidth_Avg	Average distance between the user-defined trench center plane and the trench surface along the z-axis.
Halfwidth_Max	Maximum distance between the user-defined trench center plane and the trench surface along the z-axis.
Halfwidth_Med	Median distance between the user-defined trench center plane and the trench surface along the z-axis.
Halfwidth_Min	Minimum distance between the user-defined trench center plane and the trench surface along the z-axis.
Halfwidth_SD	Standard deviation of the distance between the user-defined trench center plane and the trench surface along the z-axis.

Chapter 6: Input Commands
extract

Table 60 DOE parameters for shape analysis (Continued)

DOE parameter	Description
LeftHalfwidth_Avg	Average distance between the user-defined trench center plane and the left trench surface along the z-axis.
LeftHalfwidth_Max	Maximum distance between the user-defined trench center plane and the left trench surface along the z-axis.
LeftHalfwidth_Med	Median distance between the user-defined trench center plane and the left trench surface along the z-axis.
LeftHalfwidth_Min	Minimum distance between the user-defined trench center plane and the left trench surface along the z-axis.
LeftHalfwidth_SD	Standard deviation of the distance between the user-defined trench center plane and the left trench surface along the z-axis.
RightHalfwidth_Avg	Average distance between the user-defined trench center plane and the right trench surface along the z-axis.
RightHalfwidth_Max	Maximum distance between the user-defined trench center plane and the right trench surface along the z-axis.
RightHalfwidth_Med	Median distance between the user-defined trench center plane and the right trench surface along the z-axis.
RightHalfwidth_Min	Minimum distance between the user-defined trench center plane and the right trench surface along the z-axis.
RightHalfwidth_SD	Standard deviation of the distance between the user-defined trench center plane and the right trench surface along the z-axis.
Width_Avg	Average width of the trench along the z-axis.
Width_Bottom	Average width of the trench at $z=Z_{Bottom}$.
Width_Max	Maximum width of the trench along the z-axis.
Width_Med	Median width of the trench along the z-axis.
Width_Min	Minimum width of the trench along the z-axis.
Width_SD	Standard deviation of the width of the trench along the z-axis.
Width_Top	Average width of the trench at $z=Z_{Top}$.
Z_Halfwidth_Max	The z-coordinate of the maximum trench half-width.

Chapter 6: Input Commands
extract

Table 60 DOE parameters for shape analysis (Continued)

DOE parameter	Description
Z_Halfwidth_Min	The z-coordinate of the minimum trench half-width.
Z_LeftHalfwidth_Max	The z-coordinate of the maximum left trench half-width.
Z_LeftHalfwidth_Min	The z-coordinate of the minimum left trench half-width.
Z_RightHalfwidth_Max	The z-coordinate of the maximum right trench half-width.
Z_RightHalfwidth_Min	The z-coordinate of the minimum right trench half-width.
Z_Width_Max	The z-coordinate of the maximum trench width.
Z_Width_Min	The z-coordinate of the minimum trench width.

Table 61 One-dimensional datasets for shape analysis

Parameter	Description
Parameters common to both reference_shape=cylinder_hole and reference_shape=trench	
AxisCurvature	Local curvature of the fitted center line.
AxisTilt	Local slope angle of the fitted center line.
SurfaceCoverage_<Material>_Avg	Surface coverage of material <Material> (averaged along the surface line at fixed height z). Note: This parameter can only be specified for PMC structures.
Thickness_<Material>_Avg	Thickness of the layer between the gas interface and the bulk interface of material <Material> (averaged along the surface line at fixed height z). Note: This parameter yields reasonable results only for closed material layers, such as thin deposited polymer layers covering the cylindrical hole surface. This parameter can only be specified for PMC structures.
Z	The z-coordinate.

Chapter 6: Input Commands
extract

Table 61 One-dimensional datasets for shape analysis (Continued)

Parameter	Description
Parameters specific to reference_shape=cylinder_hole	
AxisOffsetPhi	Direction of the shift of the fitted hole center from the cylinder axis as a function of z.
AxisOffsetR	Distance between the fitted hole center and the cylinder axis as a function of z.
AxisOffsetX	The x-shift of the fitted hole center from the cylinder axis as a function of z.
AxisOffsetY	The y-shift of the fitted hole center from the cylinder axis as a function of z.
Eccentricity	Eccentricity of the ellipse (fitted from the surface line at each height z). The eccentricity is defined by: $\text{Eccentricity} = \frac{\text{MajorSemiaxis} - \text{MinorSemiaxis}}{\text{MajorSemiaxis} + \text{MinorSemiaxis}}$
EllipseRotation	Angle between the major semi-axis of the fitted ellipse and the x-axis (fitted from the surface line at each height z).
FitRadius_Avg	Average distance between the fitted hole center and the hole surface (averaged along the surface line at fixed height z).
FitRadius_Max	Maximum distance between the fitted hole center and the hole surface (measured along the surface line at fixed height z).
FitRadius_Min	Minimum distance between the fitted hole center and the hole surface (measured along the surface line at fixed height z).
FitRadius_SD	Standard deviation of the distance between the fitted hole center and the hole surface (taken along the surface line at fixed height z).
MajorSemiaxis	Major semi-axis of the ellipse (fitted from the surface line at each height z).
MinorSemiaxis	Minor semi-axis of the ellipse (fitted from the surface line at each height z).

Chapter 6: Input Commands
extract

Table 61 One-dimensional datasets for shape analysis (Continued)

Parameter	Description
Radius_Avg	Average distance between the cylinder axis and the hole surface (averaged along the surface line at fixed height z).
Radius_Max	Maximum distance between the cylinder axis and the hole surface (measured along the surface line at fixed height z).
Radius_Min	Minimum distance between the cylinder axis and the hole surface (measured along the surface line at fixed height z).
Radius_SD	Standard deviation of the distance between the cylinder axis and the hole surface (taken along the surface line at fixed height z).
z	The z-coordinate.
Parameters specific to reference_shape=trench	
AxisOffset	Distance between the fitted trench center and the user-defined trench axis as function of z.
Halfwidth_Avg	Average half-width of the trench (averaged along the surface points at fixed height z).
Halfwidth_Max	Maximum half-width of the trench (averaged along the surface points at fixed height z).
Halfwidth_Min	Minimum half-width of the trench (averaged along the surface points at fixed height z).
Halfwidth_SD	Standard deviation of the half-width of the trench (averaged along the surface points at fixed height z).
LeftHalfwidth_Avg	Average left half-width of the trench (averaged along the surface points at fixed height z).
LeftHalfwidth_Max	Maximum left half-width of the trench (averaged along the surface points at fixed height z).
LeftHalfwidth_Min	Minimum left half-width of the trench (averaged along the surface points at fixed height z).
LeftHalfwidth_SD	Standard deviation of the left half-width of the trench (averaged along the surface points at fixed height z).

Table 61 One-dimensional datasets for shape analysis (Continued)

Parameter	Description
RightHalfwidth_Avg	Average right half-width of the trench (averaged along the surface points at fixed height z).
RightHalfwidth_Max	Maximum right half-width of the trench (averaged along the surface points at fixed height z).
RightHalfwidth_Min	Minimum right half-width of the trench (averaged along the surface points at fixed height z).
RightHalfwidth_SD	Standard deviation of the right half-width of the trench (averaged along the surface points at fixed height z).
Width_Avg	Average width of the trench (averaged along the surface points at fixed height z).
Width_Max	Maximum width of the trench (averaged along the surface points at fixed height z).
Width_Min	Minimum width of the trench (averaged along the surface points at fixed height z).
Width_SD	Standard deviation of the width of the trench (averaged along the surface points at fixed height z).

Cylindrical Hole Profile

```
extract type=cylinder_hole_profile center=<v> point_max=<v> \
point_min=<v> \
[file=<c>] [structure=<c>]
```

When specifying `type=cylinder_hole_profile`, the vertical profile of a cylindrical hole is extracted from a PMC structure. The `structure` parameter must denote a PMC structure. The values specified with the parameters `point_max` and `point_min` define a cuboid within which, for each vertical coordinate, the average, the maximum, and the minimum radius and the standard deviation of the radius are extracted.

The extracted values are written to a file in comma-separated value (CSV) format. By default, the file name is the base name of the command file and it uses the suffix `_cyl.csv`. Alternatively, a file name can be specified with the parameter `file`. The resulting file can be visualized in Sentaurus Visual or other visualization tools.

Bounding Box of Materials and Regions

To extract the bounding box of regions having any of the specified materials, use the command:

```
extract type=bounding_box \
[materials=<l>] [name=<c>] [silent=<b>] [structure=<c>]
```

This command returns the bounding box of regions of the given structure that have any of the specified materials. If no materials are specified, the bounding box of the entire structure is returned. If the list of materials is not empty but contains only materials that are not present in the specified structure, an error is issued.

Note:

When the structure specified with the `structure` parameter is a PMC structure, the list of materials must be empty.

To extract the bounding box of regions having any of the specified names, use the command:

```
extract type=bounding_box \
[name=<c>] [regions=<l>] [silent=<b>] [structure=<c>]
```

This command returns the bounding box of regions of the given structure that have any of the specified region names. If no regions are specified, the bounding box of the entire structure is returned. If the list of regions is not empty but contains only regions that are not present in the specified structure, an error is issued.

Note:

When the structure specified with the `structure` parameter is a PMC structure, the list of region names must be empty.

Vertical Coordinates of the Top and Bottom of the Bounding Box of Materials and Regions

To extract the bottom coordinate of the bounding box of the regions of a structure having any of the specified materials, use the command:

```
extract type=bounding_box_bottom \
[materials=<l>] [name=<c>] [silent=<b>] [structure=<c>]
```

This command returns the vertical coordinate of the bottom of the bounding box of regions of the given structure that have any of the specified materials. If no materials are specified, the vertical coordinate of the bottom of the bounding box of the entire structure is returned. If the list of materials is not empty but contains only materials that are not present in the specified structure, an error is issued.

Chapter 6: Input Commands

extract

Note:

When the structure specified with the `structure` parameter is a PMC structure, the list of materials must be empty.

To extract the top coordinate of the bounding box of the regions of a structure having any of the specified materials, use the command:

```
extract type=bounding_box_top \
[materials=<l>] [name=<c>] [silent=<b>] [structure=<c>]
```

This command returns the vertical coordinate of the top of the bounding box of regions of the given structure that have any of the specified materials. If no materials are specified, the vertical coordinate of the top of the bounding box of the entire structure is returned.

If the list of materials is not empty but contains only materials that are not present in the specified structure, an error is issued.

Note:

When the structure specified with the `structure` parameter is a PMC structure, the list of materials must be empty.

To extract the bottom coordinate of the bounding box of the regions of a boundary structure having any of the specified names, use the command:

```
extract type=bounding_box_bottom \
[name=<c>] [regions=<l>] [silent=<b>] [structure=<c>]
```

This command returns the vertical coordinate of the bottom of the bounding box of regions of the given structure that have any of the specified region names. The structure specified by the parameter `structure` must be a boundary structure. If no region names are specified, the vertical coordinate of the bottom of the bounding box of the entire structure is returned. If the list of region names is not empty but contains only regions that are not present in the specified structure, an error is issued.

To extract the top coordinate of the bounding box of the regions of a boundary structure having any of the specified names, use the command:

```
extract type=bounding_box_top \
[name=<c>] [regions=<l>] [silent=<b>] [structure=<c>]
```

This command returns the vertical coordinate of the top of the bounding box of regions of the given structure that have any of the specified region names. The structure specified by the parameter `structure` must be a boundary structure. If no region names are specified, the vertical coordinate of the top of the bounding box of the entire structure is returned. If the list of region names is not empty but contains only regions that are not present in the specified structure, an error is issued.

Dimension of Structure

To extract of the dimension of a structure, use the command:

```
extract type=dimension [name=<c>] [silent=<b>] [structure=<c>]
```

Note:

This extraction type supports both boundary structures and PMC structures.

Example: dimension

```
extract type=dimension
```

This command extracts and outputs the dimension of the processed structure.

Damage Integral

```
extract type=damage_integral [name=<c>] [point_max=<v>] \  
[point_min=<v>] [structure=<c>]
```

Damage is integrated over the entire structure or, if specified, a box defined by `point_min` and `point_max`. The damage value is multiplied by the cell volume for each cell, where the volume is in cm³.

Surface Coverage

```
extract type=surface_coverage [point_max=<v>] [point_min=<v>] \  
[silent=<b>] [species=<l>] [structure=<c>]
```

For PMC structures, the command extracts the surface coverage fraction from a list of species bounded by `point_min` and `point_max`, if given. If you do not specify a species list, then the command returns the surface coverage of all species. Similarly, if you do not specify `point_min` and `point_max`, then the command returns the surface coverage of the entire structure. The command returns a Tcl list of {<species> <fraction>} lists, where <fraction> is a number between 0 and 1. For example:

```
set coverages [extract type=surface_coverage species={Si Ge}]  
  
foreach spec_val $coverages {  
    puts "species: [lindex $spec_val 0], value: [lindex $spec_val 1]"  
}
```

Note:

When you start Sentaurus Topography 3D with the `--processes` command-line option with a value greater than 1 (see [Table 1 on page 17](#)), the parameter `structure` must not denote a structure obtained from the `etch` command using `method=pmc averaging_runs=1`.

Chapter 6: Input Commands

extract

Table 62 Parameters of extract command

Parameter	Type	Description	Default [Range]	Unit
axis	Character	Sets the direction of one of the coordinate axes. Options are: <ul style="list-style-type: none">• x• y• z	none	none
bottom_shift	Number	Sets the distance above the structure bottom, along the vertical z-axis, where the bottom width or radius is measured. Note: You can specify this parameter only if type=shape_analysis.	0 [0, ∞[µm
center	Vector	Sets the 2D position of the vertical extraction axis.	none	µm
csv_file	Character	Sets the name of a CSV file. Note: You can only specify this parameter if output_type=csv.	<basename>_extract.csv	none
direction	Vector	Sets the direction of the cutting line.	none	none
doe_parameters	List	Sets a list of DOE parameter names to be shown as output in Sentaurus Workbench. Note: You can specify this parameter only if output_type=doe and type=shape_analysis.	By default, all DOE parameters are shown; Table 60 lists possible DOE parameter names	none
exposed_only	Boolean	Specifies whether only the exposed materials, regions, or region parts must be extracted.	false	none

Chapter 6: Input Commands

extract

Table 62 Parameters of extract command (Continued)

Parameter	Type	Description	Default [Range]	Unit
extract_surface	Boolean	<p>Specifies whether to extract the surface line around the hole axis at given xy planes. The z-positions of the xy planes can be specified by the parameter z_planes.</p> <p>Note: You can specify this parameter only if type=shape_analysis.</p>	false	none
file	Character	Sets the name of the file in which the extracted information is saved.	<basename>.extract.tdr	none
interface	List	<p>Sets a pair of species names that specify the material interface at which the DOE parameters must be measured.</p> <p>Note: For this parameter:</p> <ul style="list-style-type: none"> Only layers containing the specified material interface are extracted. This might produce meaningless results for some DOE parameters such as Depth, Z_Bottom, and Z_Top. You can specify this parameter only for PMC structures and if type=shape_analysis. 	none	none
material1	Character	Sets the name of a material.	none	none
material2	Character	Sets the name of a material.	none	none

Chapter 6: Input Commands
extract

Table 62 Parameters of extract command (Continued)

Parameter	Type	Description	Default [Range]	Unit
materials	List	<p>Sets the list of materials to consider for the extraction when any of the following are specified:</p> <ul style="list-style-type: none"> • type=bounding_box • type=bounding_box_bottom • type=bounding_box_top • type=probe <p>When using type=probe, any edge found along the specified line that lies in a region having a material not included in the list will be ignored and will not be included in the result. If materials is omitted, any material will be included in the output.</p> <p>Note: When type=probe, you can specify materials only if property=length.</p>	none	none
max_depth	Number	<p>Sets the maximum depth of the cuboid bounding box used for extraction (see Figure 31 on page 388).</p> <p>Note: You can specify this parameter only if type=shape_analysis and reference_shape=trench.</p>	none]0, ∞[µm
max_radius	Number	<p>Sets the radius of the cylindrical bounding box used for extraction (see Figure 31).</p> <p>Note: You can specify this parameter only if type=shape_analysis and reference_shape=cylinder_hole.</p>	none]0, ∞[µm

Chapter 6: Input Commands

extract

Table 62 Parameters of extract command (Continued)

Parameter	Type	Description	Default [Range]	Unit
max_width	Number	<p>Sets the maximum width of the cuboid bounding box used for extraction (see Figure 31 on page 388).</p> <p>Note: You can specify this parameter only if <code>type=shape_analysis</code> and <code>reference_shape=trench</code>.</p>	none]0, ∞[µm
name	Character	<p>Sets the name of the extraction.</p> <p>Note: If <code>type=shape_analysis</code>, the name is prepended to the DOE parameter name to uniquely identify the extraction.</p>	Hole<counter> (if <code>reference_shape=cylinder_hole</code>) Trench<counter> (if <code>reference_shape=trench</code>) <counter> is an integer that uniquely identifies the extraction. If <code>counter=0</code> , the DOE prefix is omitted.	none
normal	Vector	Sets the normal of the specified plane.	none	none
output	Character	<p>Sets a filtering option. Options are:</p> <ul style="list-style-type: none"> • all • first • inside • last • outside <p>The values <code>inside</code> and <code>outside</code> only apply if a cutting line is specified with the <code>point1</code> and <code>point2</code> parameters.</p>	all	none

Chapter 6: Input Commands

extract

Table 62 Parameters of extract command (Continued)

Parameter	Type	Description	Default [Range]	Unit
output_type	Character	<p>Sets the output of extracted data. Options are:</p> <ul style="list-style-type: none"> • <empty>: If you specify <code>output_type={}</code>, no DOE or file output is generated; the extracted data is returned as a Tcl list. • <code>csv</code>: Writes datasets in a tabular format to a CSV file specified by the <code>csv_file</code> parameter. • <code>doe</code>: Returns scalar quantities to Sentaurus Workbench as DOE parameters. • <code>tdr</code>: Writes datasets to a TDR file specified by the <code>tdr_file</code> parameter. <p>Note: You can specify the parameter <code>output_type</code> only if <code>type=shape_analysis</code>.</p>	doe	none
point	Vector	<p>Sets a point on the cutting line or the cutting plane.</p> <p>Note: The dimension of this point must match the dimension of the structure for which it is used.</p>	none	µm
point_max	Vector	Sets the maximum corner point of the cuboid.	none	µm
point_min	Vector	Sets the minimum corner point of the cuboid.	none	µm

Chapter 6: Input Commands

extract

Table 62 Parameters of extract command (Continued)

Parameter	Type	Description	Default [Range]	Unit
point1	Vector	<p>If defining an extraction line, sets the first point on the cutting line or the cutting plane.</p> <p>Note: The dimension of this point must match the dimension of the structure for which it is used.</p> <p>When used for shape analysis (type=shape_analysis):</p> <ul style="list-style-type: none"> • If reference_shape=cylinder_hole, point1 specifies the first vertex of the axis of the cylindrical bounding box for extraction (see Figure 31 on page 388 (left)). • If reference_shape=trench, point1 is the first vertex of the trench orientation vector, defined at the top of the cuboid bounding box for extraction (see Figure 31 (right)). 	none	µm

Chapter 6: Input Commands

extract

Table 62 Parameters of extract command (Continued)

Parameter	Type	Description	Default [Range]	Unit
point2	Vector	<p>If defining an extraction line, sets the second point on the cutting line or the cutting plane.</p> <p>Note: The dimension of this point must match the dimension of the structure for which it is used.</p> <p>When used for shape analysis (type=shape_analysis):</p> <ul style="list-style-type: none"> If reference_shape=cylinder_hole, point2 specifies the second vertex of the axis of the cylindrical bounding box for extraction (see Figure 31 on page 388 (left)). If reference_shape=trench, point2 is the second vertex of the trench orientation vector, defined at the top of the cuboid bounding box for extraction (see Figure 31 (right)). 	none	µm
point3	Vector	Sets a third point on the cutting plane.	none	µm
position	Number	Sets the cutting plane distance from the origin measured in the direction defined by axis.	none	µm
probe_empty_space	Boolean	<p>Specifies whether to probe empty space.</p> <p>Note: You can specify this parameter only if property=length.</p>	false	none

Chapter 6: Input Commands

extract

Table 62 Parameters of extract command (Continued)

Parameter	Type	Description	Default [Range]	Unit
property	Character	Sets the property to be associated with segments returned by type=probe. Options are: <ul style="list-style-type: none">• length• material• region• region_part	material	none
reference_shape	Character	Sets the reference shape to be used for shape analysis. Options are: <ul style="list-style-type: none">• cylinder_hole• trench Note: You can specify this parameter only if type=shape_analysis.	none	none
region	Character	Sets the name of the region for which properties are extracted.	none	none
region_1	Character	Sets the name of the region on the first side of the interface for which the area is extracted.	none	none
region_2	Character	Sets the name of the region on the second side of the interface for which the area is extracted.	none	none
region_part1	List	Sets the name of a region and the name of a part of that region, given in the format: {region_name part_name}	none	none
region_part2	List	Sets the name of a region and the name of a part of that region, given in the format: {region_name part_name}	none	none
region1	Character	Sets the name of a region.	none	none

Chapter 6: Input Commands
extract

Table 62 Parameters of extract command (Continued)

Parameter	Type	Description	Default [Range]	Unit
region2	Character	Sets the name of a region.	none	none
regions	list	Sets the list of regions to consider when any of the following are specified: <ul style="list-style-type: none">• type=bounding_box• type=bounding_box_bottom• type=bounding_box_top	none	none
silent	Boolean	This parameter can be used with all extraction types to control whether or not the extracted values are written to the log file. When <code>silent=true</code> , the extracted values are not written to the log file, but they can be assigned to a Tcl variable.	false	none
smoothing_order	Number	Sets the order of the smoothing filter (Savitzky–Golay filter). If <code>smoothing_order=0</code> , smoothing is switched off. Note: You can specify this parameter only if <code>type=shape_analysis</code> .	0 [0, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25]	none
species	List	Specifies a list of species for which the surface coverage is extracted.	none	none
structure	Character	Sets the name of the structure whose properties must be extracted.	default_structure	none
tdr_file	Character	Sets the name of a TDR file. Note: You can specify this parameter only if <code>output_type=tdr</code> and <code>type=shape_analysis</code> .	<basename>_extract.tdr	none

Chapter 6: Input Commands
extract

Table 62 Parameters of extract command (Continued)

Parameter	Type	Description	Default [Range]	Unit
top_shift	Number	Sets the distance below the structure top, along the vertical z-axis, where the top width or radius is measured. Note: You can specify this parameter only if type=shape_analysis.	0 [0, ∞[µm
type	Character	Sets the extraction type. Table 63 lists the extraction types and summarizes which structure types are supported for the different extraction types. Note: You can use type=cylinder_hole_profile only for structures resulting from simulations that used the PMC method.	none	none
vertical_spacing	Number	Sets the vertical spacing to use when scanning the structure in the vertical direction during shape analysis. This parameter can only be specified for brep structures, and it is mandatory only if there is no preceding level-set etching or deposition step.	Vertical spacing of the last preceding etching or deposition step]0, ∞[µm
z_planes	List	Sets the z-values of the xy planes, on which the surface line of the hole should be extracted. Note: You can specify this parameter only if extract_surface=true.	none]-∞, ∞[µm

Chapter 6: Input Commands
extract

Table 63 Supported structure types for different extraction types

Extraction type	Boundary structure	PMC structure
1d_cut	Supported	Supported
2d_cut	Supported	Not supported
area	Supported	Not supported
bottom_exposed	Supported	Supported
bounding_box	Supported	Supported when parameters materials and regions are empty
bounding_box_bottom	Supported	Supported when parameters materials and regions are empty
bounding_box_exposed	Supported	Supported when parameters materials and regions are empty
bounding_box_top	Supported	Not supported
cylinder_hole_profile	Not supported	Supported
damage_integral	Not supported	Supported
dimension	Supported	Supported
interface	Supported	Supported only when using parameters material1 and material2
interface_area	Supported	Not supported
material_name	Supported	Not supported
material_names	Supported	Supported only when exposed_only=false
part_area	Supported	Not supported
part_volume	Supported	Not supported
probe	Supported	Supported only when property=length or property=material
region_names	Supported	Not supported
region_parts	Supported	Not supported

Chapter 6: Input Commands

extract

Table 63 Supported structure types for different extraction types (Continued)

Extraction type	Boundary structure	PMC structure
shape_analysis	Supported	Supported
shortest_distance	Supported	Not supported
slice	Supported	Supported
surface_coverage	Not supported	Supported
top_exposed	Supported	Supported
volume	Supported	Not supported

fill

This command performs one fill operation or multiple fill operations sequentially. A fill operation consists of filling a structure with a material up to a certain height, measured from the top of the structure. A fill operation is similar to a deposition followed by a planarization step.

Note:

When you start Sentaurus Topography 3D with the `--processes` command-line option with a value greater than 1 (see [Table 1 on page 17](#)), the parameter `structure` must not denote a structure obtained from the `etch` command using `method=pmc averaging_runs=1`.

Syntax

Fill up a boundary structure:

```
fill material=<l> thickness=<v> \
[accuracy=<n>] [comment=<c>] [decimate=<b>] [merge=<b>] \
[min_angle=<n>] [min_dihedral_angle=<n>] [region=<c>] \
[ridge_angle=<n>] [shortest_edge=<n>] [structure=<c>] [times=<n>]
```

Fill up a PMC structure:

```
fill material=<l> thickness=<v> [comment=<c>] [structure=<c>] \
[times=<n>]
```

Table 64 Parameters of fill command

Parameter	Type	Description	Default [Range]	Unit
accuracy	Number	<p>Sets the maximum deviation between the decimated surface of the boundary structure produced by Boolean operations and its original surface.</p> <p>When set to -1, the used value is half of the structure-dependent spacing <code>epsilon</code>¹.</p> <p>It can be specified only if both of the following conditions are met:</p> <ul style="list-style-type: none">• <code>structure</code> denotes a boundary structure.• <code>decimate=true</code>, or you do not specify the <code>decimate</code> parameter and specify <code>let decimate=true</code>.	The value set by <code>let decimate_accuracy=<n></code> $\{-1\} \cup [0, \infty[$	μm

Chapter 6: Input Commands

fill

Table 64 Parameters of fill command (Continued)

Parameter	Type	Description	Default [Range]	Unit
comment	Character	Sets a comment for this process step that will be displayed or written to the log file.	empty string	none
decimate	Boolean	Specifies whether to decimate the boundary structure produced by Boolean operations. It can be specified only if <code>structure</code> denotes a boundary structure.	The value set by <code>let decimate=</code>	none
material	List	Sets the materials with which to fill up a structure. The number of elements of parameter <code>material</code> must equal the number of elements of parameter <code>thickness</code> . The structure is filled up with the <i>i</i> -th material of <code>material</code> up to the height specified by the <i>i</i> -th element of <code>thickness</code> . The <i>i</i> -th element of <code>thickness</code> is measured from the top of the structure before the structure is filled with the current material. If more than one material is specified, then none of them can be gas.	none	none
merge	Boolean	If set to <code>true</code> , the deposited region merges with the already existing regions of the same material in the structure. It can be specified only if <code>structure</code> denotes a boundary structure.	false	none

Chapter 6: Input Commands

fill

Table 64 Parameters of fill command (Continued)

Parameter	Type	Description	Default [Range]	Unit
min_angle	Number	<p>Sets the smallest angle in the elements of the decimated surface of the boundary structure produced by Boolean operations.</p> <p>It can be specified only if both of the following conditions are met:</p> <ul style="list-style-type: none"> • structure denotes a boundary structure. • decimate=true, or you do not specify the decimate parameter and specify let decimate=true. 	The value set by let decimate_min_angle=<n> [0, 180]	degree
min_dihedral_angle	Number	<p>Sets the smallest dihedral angle between two elements of the decimated surface of the boundary structure produced by Boolean operations that share an edge.</p> <p>It can be specified only if both of the following conditions are met:</p> <ul style="list-style-type: none"> • structure denotes a boundary structure. • decimate=true, or you do not specify the decimate parameter and specify let decimate=true. 	The value set by let decimate_min_dihedral_angle=<n> [0, 180]	degree
region	Character	<p>Sets the name of the first region that is deposited.</p> <p>It can be specified only if structure denotes a boundary structure.</p>	none	none
ridge_angle	Number	<p>Sets the angle used by the decimation algorithm to determine geometric features of the boundary structure produced by Boolean operations.</p> <p>It can be specified only if both of the following conditions are met:</p> <ul style="list-style-type: none"> • structure denotes a boundary structure. • decimate=true, or you do not specify the decimate parameter and specify let decimate=true. 	The value set by let decimate_ridge_angle=<n> [0, 180]	degree

Chapter 6: Input Commands

fill

Table 64 Parameters of fill command (Continued)

Parameter	Type	Description	Default [Range]	Unit
shortest_edge	Number	<p>Sets the shortest edge of the decimated surface of the boundary structure produced by Boolean operations.</p> <p>When set to <code>-1</code>, the used value is half of the structure-dependent spacing <code>epsilon</code>¹.</p> <p>It can be specified only if both of the following conditions are met:</p> <ul style="list-style-type: none"> <code>structure</code> denotes a boundary structure. <code>decimate=true</code>, or you do not specify the <code>decimate</code> parameter and specify <code>let decimate=true</code>. 	The value set by <code>let decimate_shortest_edge=<n></code> $\{-1\} \cup [0, \infty[$	μm
structure	Character	<p>Sets the name of the structure that must be filled.</p> <p>Note: This parameter cannot specify a PMC structure. To process such a structure, an explicit conversion of the structure is needed before calling the <code>fill</code> command (see Integration With Other Sentaurus Topography 3D Functionality on page 519).</p>	default_structure	none
thickness	Vector	<p>Sets the thicknesses of the materials with which to fill up a structure.</p> <p>The number of elements of parameter <code>thickness</code> must equal the number of elements of parameter <code>material</code>. The structure is filled up with the <i>i</i>-th material of <code>material</code> up to the height specified by the <i>i</i>-th element of <code>thickness</code>.</p> <p>The <i>i</i>-th element of <code>thickness</code> is measured from the top of the structure before the structure is filled with the current material.</p>	<p>none</p> <p>Each element of the vector must be in the range $[0, \infty[$</p>	μm

Chapter 6: Input Commands

fill

Table 64 Parameters of *fill* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
times	Number	Sets how many times the structure must be filled with the specified materials.	1 [1, 2147483647]	none

1. The structure-dependent spacing epsilon is logged out when defining a structure using the `define_structure` command.

Examples

```
fill material=Oxide thickness=0.5 comment="oxide fill" \
      region="filled_oxide"
```

This command deposits a region consisting of material Oxide, with a thickness of 0.5 μm. The thickness is measured from the top of the existing structure. The comment for this process step is displayed in the log file, and the region name is set to filled_oxide.

```
define_structure material=Silicon point_min={0 0 0} \
                  point_max={1 1 1}
fill material={Oxide Nitride} thickness={0.1 0.2} times=100
```

This command deposits a stack of 200 layers on a silicon substrate. The materials of the layers are Oxide and Nitride, alternating. The thickness of all Oxide layers is 0.1 μm; the thickness of all Nitride layers is 0.2 μm.

Chapter 6: Input Commands

filter_structure

filter_structure

This command performs different operations.

Syntax

Execute Boolean operations:

```
filter_structure type=boolean \
[body=<c>] [boolean_accuracy=<n>] [decimate=<b>] \
[deposit_material=<c>] [min_angle=<n>] \
[min_dihedral_angle=<n>] [name=<c>] [ridge_angle=<n>] \
[shortest_edge=<n>] [structure=<c>] [surface_accuracy=<n>] \
[surface_decimate=<c>] [surface_min_angle=<n>] \
[surface_min_dihedral_angle=<n>] [surface_ridge_angle=<n>] \
[surface_shortest_edge=<n>]
```

Convert a boundary structure to a PMC structure. If you specify `split_replacements`, then split the specified compound materials into constituent species according to the material replacement map or maps specified in `split_replacements` and defined in the `define_material_replacement` command:

```
filter_structure type=convert_to_pmc spacing=<n> \
[name=<c>] [split_replacements=<l>] [structure=<c>] \
[top_gas_thickness=<n>]
```

Note:

This conversion is not available for 2D structures.

Create a copy of an existing structure:

```
filter_structure type=copy [name=<c>] [structure=<c>]
```

Convert a PMC structure to a boundary structure by using the dual-contouring method:

```
filter_structure type=dc \
[dc_min_angle=<n>] [dc_min_dihedral_angle=<n>] \
[dc_reference_volume_scaling=<n>] [dc_snap_to_box=<b>] \
[dc_version=<n>] [include_grain_index=<b>] \
[keep_top_gas=<b>] [mean_spacing=<n>]
```

Decimate the number of surface elements of a boundary structure:

```
filter_structure type=decimate tolerance=<n> \
[mean_spacing=<n>] [min_angle=<n>] [min_dihedral_angle=<n>] \
[name=<c>] [ridge_angle=<n>] \
[shortest_edge=<n>] [structure=<c>]
```

Chapter 6: Input Commands

filter_structure

Merge several regions of a boundary structure that have the same material into one region:

```
filter_structure type=merge_regions new_region_name=<c> \
merged_regions=<l> \
[name=<c>] [structure=<c>]
```

Create a boundary structure with a different surface discretization (not available for 2D structures):

```
filter_structure type=rediscretize_boundary \
[dc_min_angle=<n>] [dc_min_dihedral_angle=<n>] \
[decimate=<b>] [accuracy=<n>] [method=<c>] [min_angle=<n>] \
[min_dihedral_angle=<n>] [ridge_angle=<n>] [shortest_edge=<n>]] \
[material_priority=<l>] [material_selection=<c>] \
[mean_spacing=<n>] [structure=<c>]
```

Remove accumulated damage from a PMC structure:

```
filter_structure type=remove_damage [name=<c>] [structure=<c>]
```

Remove those parts from a boundary structure that are completely surrounded by gas and, therefore, are disconnected topologically from the bulk (see [Figure 33 on page 438](#)):

```
filter_structure type=remove_disconnected_top_parts \
[name=<c>] [structure=<c>]
```

Remove those parts from a boundary structure that fit the specified criteria of material, maximum area or maximum volume, materials in contact, and bounding box:

```
filter_structure type=remove_parts [exposed_only=<b>] \
[materials=<l>] [materials_in_contact=<l>] \
[maximum_area=<n>] [maximum_volume=<n>] \
[name=<c>] [point_max=<v>] [point_min=<v>] [silent=<b>] \
[structure=<c>]
```

Remove those parts from a PMC structure that fit the specified criterion of maximum volume:

```
filter_structure type=remove_parts maximum_volume=<n> \
[name=<c>] [silent=<b>] [structure=<c>]
```

Remove a region from a boundary structure:

```
filter_structure type=remove_region region=<c> \
[name=<c>] [structure=<c>]
```

Rename a region of a boundary structure:

```
filter_structure type=rename_region new_region_name=<c> \
region=<c> [name=<c>] [structure=<c>]
```

Chapter 6: Input Commands

filter_structure

Replace the material of a region in a boundary structure:

```
filter_structure type=replace_material region=<c> \
    new_material=<c> [name=<c>] [structure=<c>]
```

Replace a material in all matching regions of a boundary structure:

```
filter_structure type=replace_material \
    material=<c> new_material=<c> \
    [name=<c>] [structure=<c>]
```

Split specified compound materials in a PMC structure into constituent species according to a material replacement map or maps specified with `split_replacements` and defined in the `define_material_replacement` command:

```
filter_structure type=replace_material split_replacements=<l> \
    [name=<c>] [structure=<c>]
```

Convert materials in a PMC structure by mapping mixed PMC cells to compound materials (the map or maps specified in `merge_replacements` are created in the `define_material_replacement` command):

```
filter_structure type=replace_material merge_replacements=<l> \
    merge_tolerance=<n> [name=<c>] [structure=<c>]
```

Smooth a boundary structure:

```
filter_structure type=smooth \
    [factor=<n>] [mean_spacing=<n>] [name=<c>] [structure=<c>]
```

Note:

The `filter_structure` command with `type=smooth` is not available for 2D structures.

Create a boundary structure from a PMC structure (see [Saving Boundaries for PMC Structures on page 479](#)):

```
filter_structure type=vbe [decimate=<b> [accuracy=<n>]] \
    [grain_regions=<b>] [include_grain_index=<b>] \
    [material_priority=<l>] [material_selection=<c>] [structure=<c>]
```

Note:

When you start Sentaurus Topography 3D with the `--processes` command-line option with a value greater than 1 (see [Table 1 on page 17](#)), except when `type=copy`, the parameter `structure` must not denote a structure obtained from the `etch` command using `method=pmc averaging_runs=1`.

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command

Parameter	Type	Description	Default [Range]	Unit
accuracy	Number	<p>Sets the maximum deformation during decimation of the boundary structure as follows:</p> <ul style="list-style-type: none"> When <code>type=rediscretize_boundary</code> and <code>method</code> is not set to <code>dc</code>, the <code>accuracy</code> parameter specifies the maximum deformation during the decimation of the rediscretized boundary relative to the value of the <code>mean_spacing</code> parameter. When <code>type=vbe</code>, the <code>accuracy</code> parameter specifies the maximum deformation during the decimation of the boundary extracted with the volume boundary extraction (VBE) method relative to the mean PMC spacing. <p>It can be specified only if <code>decimate=true</code>.</p>	1e-3 [0, ∞[none
body	Character	Sets the name of the structure to use as the body of the Boolean operation. If omitted, the structure specified with the parameter <code>structure</code> will be used as the body.	none	none
boolean_accuracy	Number	<p>Sets the maximum deviation between the decimated surface of the boundary structure produced by Boolean operations and its original surface.</p> <p>When set to <code>-1</code>, the used value is half of the structure-dependent spacing epsilon¹.</p> <p>It can be specified only if <code>decimate=true</code>.</p>	-1 $\{-1\} \cup [0, \infty[$	µm

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
dc_min_angle	Number	When type=dc or type=rediscretize_boundary and method=dc, this parameter sets the minimum angle of the surface mesh.	The value of dc_min_angle parameter of let command [0, 180]	degree
dc_min_dihedral_angle	Number	When type=dc or type=rediscretize_boundary and method=dc, this parameter sets the minimum dihedral angle of the surface mesh.	The value of dc_min_dihedral_angle parameter of let command [0, 180]	degree
dc_reference_volume_scaling	Number	When type=dc, this parameter allows you to adjust the surface location when converting a PMC structure to a boundary. Larger values expand the structure into the gas.	none [0, 2[unitless
dc_snap_to_box	Boolean	When type=dc and dc_snap_to_box=true, this parameter snaps the outer boundary points to the bounding box. This can help tools that read the boundary to create a bulk mesh.	true	none
dc_version	Number	Selects the version of the dual-contouring algorithm. Options are: <ul style="list-style-type: none"> • 1: Selects the algorithm available in releases prior to S-2021.06. • 2: Selects the algorithm available starting from S-2021.06. 	1	none

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
decimate	Boolean	<p>Specifies whether to decimate the boundary structure as follows:</p> <ul style="list-style-type: none"> When <code>type=boolean</code>, decimation is produced by Boolean operations. When <code>type=rediscretize_boundary</code> and <code>method</code> is not set to <code>dc</code>, decimation is specified by the <code>structure</code> parameter. When <code>type=vbe</code>, decimation uses the VBE method. 	true	none
deposit_material	Character	<p>Sets the material to assign to all the deposited regions of a structure obtained from a PMC simulation when executing Boolean operations as follows:</p> <ul style="list-style-type: none"> When processing the result of a PMC simulation where a single material was deposited and <code>deposit_material</code> was not specified, that material is used as the deposited material for Boolean operations. If multiple materials were deposited in a PMC simulation, then <code>Anymaterial</code> is used as the deposited material for Boolean operations if <code>deposit_material</code> was not specified. If <code>deposit_material</code> is specified, that material is used as the deposited material for Boolean operations. 	See Description	none
exposed_only	Boolean	Specifies whether only exposed parts should be removed. This parameter applies only to boundary structures.	true	none

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
factor	Number	Sets how strong the smoothing is relative to the mean spacing of the last preceding etching or deposition step. Higher values cause stronger smoothing.	1 [0.2, 5]	none
grain_regions	Boolean	<p>Specifies whether the boundary structure produced by the VBE method should contain different regions for portions of the structure with the same material but with different material properties.</p> <p>Note: The VBE method can create boundary structures with no more than 240 regions.</p>	false	none
include_grain_index	Boolean	<p>Specifies whether to store the material property (grain index in polycrystalline materials) as a field when converting PMC to brep structures using the VBE or DC method.</p> <p>If include_grain_index=true, then when converting from PMC to boundary format, the command writes the grain index as a field into the structure for the filter_structure command, or directly to the file for the save command.</p> <p>Best results are obtained using the filter_structure command with decimate=false to create a boundary with a fine mesh for viewing the field.</p>	false	none
keep_top_gas	Boolean	Specifies whether to keep the top gas. This parameter applies only to the dual-contouring method (type=dc).	false	none

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
material	Character	Sets the material to replace. At least one region with the specified material must exist.	none	none
material_priority	List	Sets a material priority list for materials at interfaces when using the VBE method.	none	none
material_selection	Character	Sets the algorithm used to select the materials of a PMC cell when using the VBE method (see Saving Boundaries for PMC Structures on page 479). Options are: <ul style="list-style-type: none"> • maximum • mixed • proportional 	mixed	none
materials	List	Sets the materials of the parts to remove. If omitted, a part will be removed if it matches the other specified criteria, independently of its material.	none	none
materials_in_contact	List	A part can be removed if all materials with which it is in contact are included in the list of materials specified by this parameter. If you omit this parameter, a part will be removed if it matches the other specified criteria. This parameter applies only to boundary structures.	none	none
maximum_area	Number	Sets the maximum area of the parts to remove. If omitted, a part will be removed if it matches the other specified criteria, independently of its area. This parameter applies only to boundary structures.	[0, ∞[µm ²

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
maximum_volume	Number	Sets the maximum volume of the parts to remove. If omitted, a part will be removed if it matches the other specified criteria, independently of its volume. This parameter is mandatory when processing PMC structures.	[0, ∞[µm ³
mean_spacing	Number	Sets the mean spacing relative to which the decimation or smoothing is specified.	Mean spacing of the last preceding etching or deposition step [0, ∞[µm
merge_replacements	List	Specifies a list of material replacement maps defined in the define_material_replacement command. The list consists of the names of material replacement maps. PMC cells that contain the species matching the material replacement species and volume_fractions (with tolerance specified by merge_tolerance) are converted to the material specified by the material replacement map. Note: This parameter can be used only with PMC structures.	none	none
merge_tolerance	Number	Sets the allowed deviation from volume_fractions of the material replacement map, so as to perform the merge.	0.1	none
merged_regions	List	Sets the names of regions to be merged.	none	none

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
method	Character	When <code>type=rediscretize_boundary</code> , this parameter sets the rediscretization method. Options are: <ul style="list-style-type: none"> • <code>dc</code>: Dual-contouring method • <code>vbe</code>: VBE method 	none	none
min_angle	Number	Sets the smallest angle in the elements of the surface of the boundary structure obtained by decimating as follows: <ul style="list-style-type: none"> • When <code>type=boolean</code>, the boundary structure is produced by Boolean operations. • When <code>type=decimate</code>, the structure parameter specifies the boundary structure. • When <code>type=rediscretize_boundary</code> and <code>method</code> is not set to <code>dc</code>, the structure is a rediscretized boundary structure. • When <code>type=vbe</code>, the boundary structure is extracted using the VBE method. When <code>type=rediscretize_boundary</code> and when <code>type=vbe</code> , you can specify the <code>min_angle</code> parameter only if <code>decimate=true</code> .	0 [0, 180]	degree

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
min_dihedral_angle	Number	<p>Sets the smallest dihedral angle between two elements, which share an edge, in the surface of the boundary structure obtained by decimating as follows:</p> <ul style="list-style-type: none"> • When <code>type=boolean</code>, the boundary structure is produced by Boolean operations. • When <code>type=decimate</code>, the structure parameter specifies the boundary structure. • When <code>type=rediscretize_boundary</code> and <code>method</code> is not set to <code>dc</code>, the structure is a rediscretized boundary structure. • When <code>type=vbe</code>, the boundary structure is extracted using the VBE method. <p>When <code>type=rediscretize_boundary</code> and when <code>type=vbe</code>, you can specify the parameter <code>min_dihedral_angle</code> only if <code>decimate=true</code>.</p>	0 [0, 180]	degree
name	Character	Sets the name of the result structure. If omitted, the input structure will be replaced.	none	none
new_material	Character	Sets the name of the replacement material.	none	none
new_region_name	Character	When renaming regions, this parameter sets the new name for a region. No region with this name must exist before the renaming. When merging regions, this parameter specifies the name of the region created by merging. No region with this name must exist before the merge.	none	none

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of *filter_structure* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
point_max	Vector	Sets the maximum bounding vertex of the cuboid to which the parts to remove belong. If omitted, a part will be removed if it matches the other specified criteria, independently of its position. This parameter applies only to boundary structures.	none	µm
point_min	Vector	Sets the minimum bounding vertex of the cuboid to which the parts to remove belong. If omitted, a part will be removed if it matches the other specified criteria, independently of its position. This parameter applies only to boundary structures.	none	µm
region	Character	Sets the name of a region to remove or for which region the material will be replaced, or the name of a region to be renamed. The specified region must exist and, when removing a region, this region must not be the only region of the structure.	none	none

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of *filter_structure* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
ridge_angle	Number	<p>Sets the angle used to determine geometric features of the boundary structure obtained by decimating as follows:</p> <ul style="list-style-type: none">• When <code>type=boolean</code>, the boundary structure is produced by Boolean operations.• When <code>type=decimate</code>, the structure parameter specifies the boundary structure.• When <code>type=rediscretize_boundary</code> and <code>method</code> is not set to <code>dc</code>, the structure is a rediscretized boundary structure.• When <code>type=vbe</code>, the boundary structure is extracted using the VBE method. <p>When <code>type=rediscretize_boundary</code> and when <code>type=vbe</code>, you can specify the <code>ridge_angle</code> parameter only if <code>decimate=true</code>.</p>	179 [0, 180]	degree

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
shortest_edge	Number	<p>Sets the shortest edge of the surface of the boundary structure obtained by decimation as follows:</p> <ul style="list-style-type: none"> • When type=boolean, decimates the boundary structure produced by Boolean operations. • When type=decimate, decimates the structure specified by the structure parameter. • When type=rediscretize_boundary and method is not set to dc, decimates the rediscretized boundary structure. • When type=vbe, decimates the boundary structure extracted using the VBE method. <p>When type=rediscretize_boundary and when type=vbe, you can specify the shortest_edge parameter only if decimate=true.</p>	boolean_accuracy (boolean when type=boolean) mean_spacing * tolerance (boolean when type=decimate) mean_spacing * tolerance (boolean when type=rediscretize_boundary and type=vbe) [0, ∞[µm
silent	Boolean	Specifies whether to write information about the parts being removed to the log file.	false	none
spacing	Number	Sets the spacing to use to create a PMC structure. This parameter is mandatory if there is no preceding etching or deposition step specifying a spacing.	Minimum spacing of the last preceding etching or deposition step]0, ∞[µm

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
split_replacements	List	<p>Specifies a list of material replacement maps defined in the <code>define_material_replacement</code> command. The list consists of the names of material replacement maps. The material is replaced by a list of species defined by the material replacement map before the PMC step begins.</p> <p>Note: This parameter can be used only with <code>method=pmc</code>.</p>	none	none
structure	Character	Sets the name of the structure that must be used for the Boolean operations (when <code>type=boolean</code>), or copied (when <code>type=copy</code>), or decimated (when <code>type=decimate</code>), or smoothed (when <code>type=smooth</code>), or from which the disconnected parts must be removed (when <code>type=remove_disconnected_top_parts</code>).	default_structure	none
surface_accuracy	Number	<p>Sets the maximum deviation between the decimated exposed surface used as an operand of the Boolean operations and the original exposed surface.</p> <p>When set to <code>-1</code>, the used value is half of the structure-dependent spacing epsilon¹.</p> <p>It can be specified only if <code>surface_decimate=true</code>.</p>	-1 $\{-1\} \cup [0, \infty[$	μm
surface_decimate	Boolean	Specifies whether to decimate the exposed surface used as an operand of the Boolean operations.	false	none

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
surface_min_angle	Number	Sets the smallest angle in the elements of the decimated exposed surface used as an operand of the Boolean operations. It can be specified only if surface_decimate=true.	0 [0, 180]	degree
surface_min_dihedral_angle	Number	Sets the smallest dihedral angle between two elements, which share an edge, of the decimated exposed surface used as an operand of the Boolean operations. It can be specified only if surface_decimate=true.	0 [0, 180]	degree
surface_ridge_angle	Number	Sets the angle used by the decimation algorithm to determine geometric features of the exposed surface used as an operand of the Boolean operations. It can be specified only if surface_decimate=true.	179 [0, 180]	degree
surface_shortest_edge	Number	Sets the shortest edge of the decimated exposed surface used as an operand of the Boolean operations. When set to -1, the used value is half of the structure-dependent spacing epsilon ¹ . It can be specified only if surface_decimate=true.	-1 {-1} ∪ [0, ∞[µm

Chapter 6: Input Commands

filter_structure

Table 65 Parameters of filter_structure command (Continued)

Parameter	Type	Description	Default [Range]	Unit
surface_simplify	Boolean	Specifies whether the exposed surface of a structure obtained from a PMC simulation should be decimated before executing Boolean operations. Note: This parameter is deprecated. Use <code>surface_decimate</code> instead.	false	none
tolerance	Number	Sets the tolerance relative to the mean spacing for decimating the number of surface elements. Higher values cause stronger decimation.	none [0, 1]	none
top_gas_thickness	Number	Sets the minimum thickness of the gas region added on top of the created PMC structure.	0 [0, ∞[µm
type	Character	Sets the type of operation. Table 66 lists the operation types and summarizes, for each value of type, which structure type it can operate on (the parameter structure) and the type of the result structure (the parameter name).	none	none

1. The structure-dependent spacing `epsilon` is logged out when defining a structure using the `define_structure` command.

Table 66 Supported input structures for different values of type parameter

Type	Input structure		Output structure
	Boundary	PMC	
boolean	Not supported	Supported	Boundary
convert_to_pmc	Supported	Not supported	PMC
copy	Supported	Supported	Same type as input

Chapter 6: Input Commands

filter_structure

Table 66 Supported input structures for different values of type parameter (Continued)

Type	Input structure		Output structure
	Boundary	PMC	
dc	Not supported	Supported	Boundary
decimate	Supported	Not supported	Boundary
merge_regions	Supported	Not supported	Boundary
rediscretize_boundary	Supported	Not supported	Boundary
remove_damage	Not supported	Supported	PMC
remove_disconnected_top_parts	Supported	Not supported	Boundary
remove_parts	Supported	Supported	Same type as input
remove_region	Supported	Not supported	Boundary
rename_region	Supported	Not supported	Boundary
replace_material	Supported	Not supported	Boundary
smooth	Supported	Not supported	Boundary
vbe	Not supported	Supported	Boundary

Description

The default value of the mean spacing is calculated from the last preceding `etch` or `deposit` command. If there is no such command, the mean spacing must be specified with the parameter `mean_spacing`.

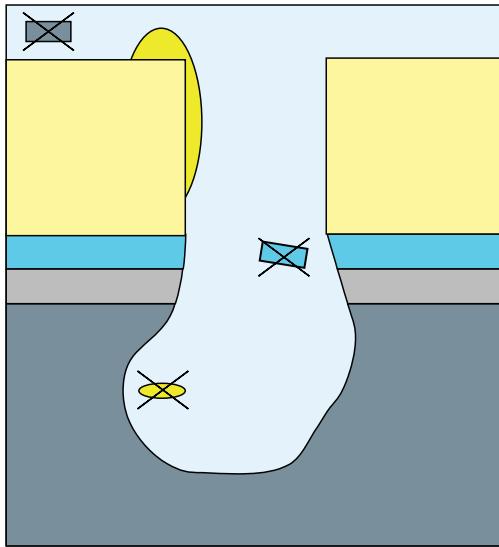
The default value of the `spacing` parameter is calculated from the last preceding `etch` or `deposit` command specifying a spacing. If there is no such command, the spacing must be specified with the parameter `spacing`.

Creating a boundary structure from a PMC structure is very similar to saving a boundary structure from a PMC structure except that a new structure is created instead of saving it to a TDR file. For more information about the VBE method, see [Saving Boundaries for PMC Structures on page 479](#).

Chapter 6: Input Commands

finalize_model

Figure 33 Using the filter_structure type=remove_disconnected_top_parts command removes the three cross-out parts



finalize_model

This command is used to indicate that the definition of the specified model is completed.

Following this command, the specified model cannot be modified using the command add_float_parameter, add_formula, add_int_parameter, add_reaction, or add_source_species.

Before using the command add_flux_properties, add_reaction_properties, define_deposit_machine, or define_etch_machine for the specified model, you must specify the finalize_model command.

Syntax

```
finalize_model model=<c>
```

Table 67 Parameters of finalize_model command

Parameter	Type	Description	Default	Unit
model	Character	Sets the name of the model that is finalized.	none	none

layout

This command queries the properties of a layout that has been created with the `define_layout` command (see [define_layout on page 240](#)).

Syntax

Query the bounding box of the specified 3D domain:

```
layout domain=<c> name=<c> type=bounding_box [silent=<b>]
```

Query the names of 3D domains defined in a layout:

```
layout name=<c> type=domains [silent=<b>]
```

Query the names of layers defined in a layout:

```
layout name=<c> type=layers [silent=<b>]
```

Table 68 Parameters of layout command

Parameter	Type	Description	Default	Unit
domain	Character	Sets the name of the domain for which the bounding box is queried.	none	none
name	Character	Sets the name of the layout.	default_layout	none
silent	Boolean	Specifies how the query results are returned. Options are: <ul style="list-style-type: none">• When <code>silent=false</code>, the results are returned as a Tcl list and are written to the log file.• When <code>silent=true</code>, the results are returned as a Tcl list but are not written to the log file.	false	none
type	Character	Sets the type of the query. Options are: <ul style="list-style-type: none">• <code>bounding_box</code>• <code>domains</code>• <code>layers</code>	none	none

let

This command sets the global properties of Sentaurus Topography 3D, one parameter at a time.

Syntax

```
let <parameter>=<type>
```

Table 69 Parameters of let command

Parameter	Type	Description	Default [Range]	Unit
base_name	Character	<p>Sets the base name of output files (see Output Files on page 20).</p> <p>Note: When Sentaurus Topography 3D is given the command file <code>commandfile.extension</code>, the command <code>let base_name=commandfile</code> is executed before starting to process the given command file.</p>	none	none
compatibility_version	Character	<p>Sets the version of Sentaurus Topography 3D with which the results must be compatible. Options are:</p> <ul style="list-style-type: none">• T-2022.03• S-2021.06• R-2020.09• Q-2019.12• P-2019.03• O-2018.06• N-2017.09	T-2022.03	none
dc_keep_top_gas	Boolean	<p>Specifies whether to keep the top gas when the dual-contouring method is used to convert a PMC structure to a boundary structure because <code>pmc_to_brep_conversion_method=dc</code> is specified.</p>	false	None

Chapter 6: Input Commands
let

Table 69 Parameters of let command (Continued)

Parameter	Type	Description	Default [Range]	Unit
dc_min_angle	Number	<p>Sets the minimum angle of the surface mesh for the dual-contouring method (see Dual-Contouring Method on page 480) in the following cases:</p> <ul style="list-style-type: none"> The dual-contouring method is used to convert a PMC structure to a boundary structure because <code>dc_version_for_pmc_to_brep_conversion=dc</code> is specified In the command <code>filter_structure type=dc</code> when its <code>dc_min_angle</code> parameter is not specified In the command <code>save type=dc</code> when its <code>dc_min_angle</code> parameter is not specified 	1e-6 [0, 180]	degree
dc_min_dihedral_angle	Number	<p>Sets the minimum dihedral angle of the surface mesh for the dual-contouring method (see Dual-Contouring Method on page 480) in the following cases:</p> <ul style="list-style-type: none"> The dual-contouring method is used to convert a PMC structure to a boundary structure because <code>dc_version_for_pmc_to_brep_conversion=dc</code> is specified In the command <code>filter_structure type=dc</code> when its parameter <code>dc_min_dihedral_angle</code> is not specified In the command <code>save type=dc</code> when its parameter <code>dc_min_dihedral_angle</code> is not specified 	5e-6 [0, 180]	degree

Chapter 6: Input Commands
let

Table 69 Parameters of let command (Continued)

Parameter	Type	Description	Default [Range]	Unit
dc_snap_to_box	Boolean	Specifies whether to snap the outer boundary points to the bounding box when the dual-contouring method is used to convert a PMC structure to a boundary structure because dc_version_for_pmc_to_brep_conversion=dc is specified. This can help tools that read the boundary to create a bulk mesh.	true	None
dc_version_for_pmc_slicing	Number	Selects the version of the dual-contouring algorithm for extract slicing during PMC steps. Options are: <ul style="list-style-type: none">• 1: Selects the algorithm available in releases prior to S-2021.06.• 2: Selects the algorithm available starting from S-2021.06.	1	none
dc_version_for_pmc_to_brep_conversion	Number	Selects the version of the dual-contouring algorithm. Options are: <ul style="list-style-type: none">• 1: Selects the algorithm available in releases prior to S-2021.06.• 2: Selects the algorithm available starting from S-2021.06.	1	None
decimate	Boolean	Specifies whether to decimate boundary structures.	true	none
decimate_accuracy	Number	Sets the maximum deviation between the decimated surface of a boundary structure and its original surface. When set to -1, the used value is half of the structure-dependent spacing epsilon ¹ .	-1 {-1} \cup [0, ∞ [μm

Chapter 6: Input Commands

let

Table 69 Parameters of let command (Continued)

Parameter	Type	Description	Default [Range]	Unit
decimate_min_angle	Number	Sets the smallest angle in the elements of the surface of the boundary structure produced by the decimation algorithm.	0 [0, 180]	degree
decimate_min_dihedral_angle	Number	Sets the smallest dihedral angle between two elements, which share an edge, of the surface of the boundary structure produced by the decimation algorithm.	0 [0, 180]	degree
decimate_ridge_angle	Number	Sets the angle used by the decimation algorithm to determine the geometric features of a boundary structure.	179 [0, 180]	degree
decimate_shortest_edge	Number	Sets the shortest edge of the surface of the boundary structure produced by the decimation algorithm. When set to -1, the used value is half of the structure-dependent spacing epsilon ¹ .	-1 {-1} ∪ [0, ∞[µm
keep_parallel_licenses	Boolean	Specifies whether parallel licenses must remain checked out after a command using them has been executed. Note: This parameter has no effect if you specify the command-line option --keep_parallel_licenses when Sentaurus Topography 3D is started (see Table 1 on page 17).	false	none
log_file	Character	Sets the name of the log file. In the file name, <base_name> denotes the value of the base_name parameter of the let command.	<base_name>.log	none

Chapter 6: Input Commands

let

Table 69 Parameters of let command (Continued)

Parameter	Type	Description	Default [Range]	Unit
log_level	Character	Sets the verbosity level. Options are: <ul style="list-style-type: none">• all: Saves all output to the log file.• error: Saves only error messages to the log file.• info: Saves minimal output to the log file.	info	none
log_target	Character	Sets the log output. Options are: <ul style="list-style-type: none">• both: Directs output to the log file and screen.• console: Directs output only to the screen.• file: Directs output only to the log file.	both	none

Chapter 6: Input Commands

let

Table 69 Parameters of let command (Continued)

Parameter	Type	Description	Default [Range]	Unit
num_threads	Number	<p>Sets the number of threads per process to be used to execute the commands that support shared-memory parallelization. Only valid if <code>let parallel=true</code>. If no value is specified, an implementation-defined optimum number of threads is used. See Advanced Shared-Memory Parallelization Options on page 41.</p> <p>Note:</p> <p>With regard to this parameter:</p> <ul style="list-style-type: none"> • This parameter has no effect if you specify the command-line option <code>--threads</code> when Sentaurus Topography 3D is started (see Table 1 on page 17). • When running Sentaurus Topography 3D from Sentaurus Workbench using auto-detection of threads, an error is issued if no value is specified. • If the value for <code>num_threads</code> is larger than the one for the <code>--max_threads</code> command-line option, Sentaurus Topography 3D uses the value of <code>--max_threads</code>. 	Automatic [1...256]	none

Chapter 6: Input Commands

let

Table 69 Parameters of let command (Continued)

Parameter	Type	Description	Default [Range]	Unit
parallel	Boolean	<p>Specifies whether to activate parallel computation for models that support parallel computation (see Basic Shared-Memory Parallelization on page 40).</p> <p>Note: This parameter has no effect if you specify the command-line option <code>--threads</code> when Sentaurus Topography 3D is started (see Table 1 on page 17).</p>	false	none
parallel_license	Character	<p>Sets the behavior of the simulator if the number of requested threads cannot be used due to a lack of the respective number of parallel licenses (see Advanced Shared-Memory Parallelization Options on page 41). Options are:</p> <ul style="list-style-type: none"> • abort: Terminates the simulation. • available: Checks out the maximum number of available licenses, and uses the maximum number of threads available. If commands are parallelized using MPI, this option terminates the simulation if the maximum number of available threads is less than the number of requested processes. • serial: If commands are not parallelized using MPI, this option continues the simulation in serial mode. If commands are parallelized using MPI, this option terminates the simulation. • wait: Waits until enough parallel licenses become available. 	serial	none

Chapter 6: Input Commands

let

Table 69 Parameters of let command (Continued)

Parameter	Type	Description	Default [Range]	Unit
pmc_to_brep_conversion_method	Character	<p>Sets the method to use to convert a PMC structure to a boundary structure whenever a boundary structure is expected, but a PMC structure is given. Options are:</p> <ul style="list-style-type: none"> • dc • error • vbe <p>Note: When <code>error</code> is specified, an error message is issued whenever a boundary structure is expected, but a PMC structure is given.</p>	vbe	None
snmesh_repair	Boolean	<p>Specifies whether to repair the structure after Boolean operations. Activating repair reduces very thin regions and facilitates meshing of the structure.</p> <p>Note: The repair functionality sometimes uses excessive CPU time and memory.</p>	false	none
surface_decimate	Boolean	Specifies whether to decimate the exposed surface.	true	none
surface_decimate_accuracy	Number	<p>Sets the maximum deviation between the decimated exposed surface and the original exposed surface.</p> <p>When set to <code>-1</code>, the used value is half of the structure-dependent spacing epsilon¹.</p>	-1 $\{-1\} \cup [0, \infty[$	μm
surface_decimate_min_angle	Number	Sets the smallest angle in the elements of the exposed surface produced by the decimation algorithm.	0 $[0, 180]$	degree

Chapter 6: Input Commands

let

Table 69 Parameters of let command (Continued)

Parameter	Type	Description	Default [Range]	Unit
surface_decimate_min_dihedral_angle	Number	Sets the smallest dihedral angle between two elements, which share an edge, of the exposed surface produced by the decimation algorithm.	0 [0, 180]	degree
surface_decimate_ridge_angle	Number	Sets the angle used by the decimation algorithm to determine geometric features of the exposed surface.	179 [0, 180]	degree
surface_decimate_shortest_edge	Number	Sets the shortest edge of the decimated exposed surface. When set to -1, the used value is half of the structure-dependent spacing epsilon ¹ .	-1 {-1} ∪ [0, ∞[µm
vbe_material_priority	List	Sets a material priority list for materials at interfaces when the VBE method is used to convert a PMC structure to a boundary structure because <code>pmc_to_brep_conversion_method=vbe</code> is specified.	none	none
vbe_material_selection	Character	Sets the algorithm used to select the materials mixed in a PMC cell when the VBE method is used to convert a PMC structure to a boundary structure because <code>pmc_to_brep_conversion_method=vbe</code> is specified (see Volume Boundary Extraction Method on page 480). Options are: <ul style="list-style-type: none"> • maximum • mixed • proportional 	mixed	none

1. The structure-dependent spacing epsilon is logged out when defining a structure using the `define_structure` command.

Chapter 6: Input Commands

let

Examples

Set the verbosity level to error messages only, direct the output to both the screen and log file, and set a name for the log file:

```
let log_level=error  
let log_target=both  
let log_file=out.log
```

Activate parallel computation and set the number of threads:

```
let parallel=true  
let num_threads=8
```

Chapter 6: Input Commands

litho

litho

This command performs a lithography simulation and adds a resist region.

Note:

The `litho` command is not available for 2D structures.

When you start Sentaurus Topography 3D with the `--processes` command-line option with a value greater than 1 (see [Table 1 on page 17](#)), the parameter `structure` must not denote a structure obtained from the `etch` command using `method=pmc averaging_runs=1`.

Syntax

```
litho inputfile=<c> \
[accuracy=<n>] [comment=<c>] [decimate=<b>] [machine=<c>] \
[mask=<c>] [material=<c>] [min_angle=<n>] \
[min_dihedral_angle=<n>] \
[merge=<b>] [options=<c>] [outputfile=<c>] [region=<c>] \
[ridge_angle=<n>] [shortest_edge=<n>] [structure=<c>]
```

Table 70 Parameters of `litho` command

Parameter	Type	Description	Default [Range]	Unit
accuracy	Number	Sets the maximum deviation between the decimated surface of the boundary structure produced by Boolean operations and its original surface. When set to -1, the used value is half of the structure-dependent spacing <code>epsilon</code> ¹ . It can be specified only if <code>decimate=true</code> , or if you do not specify the <code>decimate</code> parameter and specify <code>let decimate=true</code> .	The value set by <code>let decimate_accuracy=<n></code> $\{-1\} \cup [0, \infty[$	μm
comment	Character	Sets a comment for this process step that will be displayed or written to the log file.	empty string	none
decimate	Boolean	Specifies whether to decimate the boundary structure produced by Boolean operations.	The value set by <code>let decimate=</code>	none
inputfile	Character	Sets the name of the input SLO file.	none	none

Chapter 6: Input Commands

litho

Table 70 Parameters of litho command (Continued)

Parameter	Type	Description	Default [Range]	Unit
machine	Character	Sets the name of the lithography machine to be used. This must be predefined with the define_litho_machine command.	default_machine	none
mask	Character	Sets the name of the mask file to be used. Note: If this parameter is not specified, the mask specified in the SLO file is used for the lithography simulation.	none	none
material	Character	Sets the name of the material of the created region.	Photoresist	none
merge	Boolean	Specifies whether the deposited region merges with already existing regions of the same material in the structure.	false	none
min_angle	Number	Sets the smallest angle in the elements of the decimated surface of the boundary structure produced by Boolean operations. It can be specified only if decimate=true, or if you do not specify the decimate parameter and specify let decimate=true.	The value set by let decimate_min_angle=<n> [0, 180]	degree
min_dihedral_angle	Number	Sets the smallest dihedral angle between two elements of the decimated surface of the boundary structure produced by Boolean operations that share an edge. It can be specified only if decimate=true, or if you do not specify the decimate parameter and specify let decimate=true.	The value set by let decimate_min_dihedral_angle=<n> [0, 180]	degree

Chapter 6: Input Commands

litho

Table 70 Parameters of litho command (Continued)

Parameter	Type	Description	Default [Range]	Unit
options	Character	Sets additional command-line options for Sentaurus Lithography. Note: If you specify more than one option, they must be enclosed in double quotation marks.	none	none
outputfile	Character	Sets the name of the output SLO file.	none	none
region	Character	Sets the name of the resist region that is created.	none	none
ridge_angle	Number	Sets the angle used by the decimation algorithm to determine geometric features of the boundary structure produced by Boolean operations. It can be specified only if decimate=true, or if you do not specify the decimate parameter and specify let decimate=true.	The value set by let decimate_ridge_angle=<n> [0, 180]	degree
shortest_edge	Number	Sets the shortest edge of the decimated surface of the boundary structure produced by Boolean operations. When set to -1, the used value is half of the structure-dependent spacing epsilon ¹ . It can be specified only if decimate=true, or if you do not specify the decimate parameter and specify let decimate=true.	The value set by let decimate_shortest_edge=<n> {-1} ∪ [0, ∞[µm

Chapter 6: Input Commands

litho

Table 70 Parameters of *litho* command (Continued)

Parameter	Type	Description	Default [Range]	Unit
structure	Character	Sets the name of the structure on which a lithography simulation must be run. Note: This parameter cannot specify a PMC structure. To process such a structure, an explicit conversion of the structure is needed before calling the <code>litho</code> command (see Integration With Other Sentaurus Topography 3D Functionality on page 519).	default_structure	none

1. *The structure-dependent spacing epsilon is logged out when defining a structure using the `define_structure` command.*

pattern

This command uses logical masks defined with the `define_mask` command to create patterned profiles on top of an existing structure (see [define_mask on page 243](#)).

Note:

The `pattern` command is not available for 2D structures.

When you start Sentaurus Topography 3D with the `--processes` command-line option with a value greater than 1 (see [Table 1 on page 17](#)), the parameter `structure` must not denote a structure obtained from the `etch` command using `method=pmc averaging_runs=1`.

Syntax

Create a patterned profile on top of a boundary structure:

```
pattern material=<c> thickness=<n> type=<c> \
[accuracy=<n>] [comment=<c>] [decimate=<b>] [mask=<c>] \
[merge=<b>] [min_angle=<n>] [min_dihedral_angle=<n>] [region=<c>] \
[ridge_angle=<n>] [shortest_edge=<n>] [structure=<c>]
```

Create a patterned profile on top of a PMC structure:

```
pattern material=<c> thickness=<c> type=<c> \
[comment=<c>] [mask=<c>] [structure=<c>]
```

Chapter 6: Input Commands

pattern

Table 71 Parameters of pattern command

Parameter	Type	Description	Default [Range]	Unit
accuracy	Number	<p>Sets the maximum deviation between the decimated surface of the boundary structure produced by Boolean operations and its original surface.</p> <p>When set to -1, the used value is half of the structure-dependent spacing <i>epsilon</i>¹.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <i>structure</i> parameter denotes a boundary structure. • You specify <i>decimate=true</i> or, if you do not specify <i>decimate</i>, you did not specify <i>let decimate=false</i>. 	The value set by let decimate_accuracy=<n> {-1} ∪ [0, ∞[µm
comment	Character	Sets a comment for this process step that will be displayed or written to the log file.	empty string	none
decimate	Boolean	Specifies whether to decimate the boundary structure produced by Boolean operations. It can be specified only if <i>structure</i> denotes a boundary structure.	The value set by let decimate=	none
mask	Character	Sets the name of the mask defined in a <i>define_mask</i> command.	default_mask	none
material	Character	Sets the material of the pattern that is deposited on the structure.	none	none
merge	Boolean	Specifies whether the deposited region merges with already existing regions of the same material in the structure. It can be specified only if <i>structure</i> denotes a boundary structure.	false	none

Chapter 6: Input Commands
pattern

Table 71 Parameters of pattern command (Continued)

Parameter	Type	Description	Default [Range]	Unit
min_angle	Number	<p>Sets the smallest angle in the elements of the decimated surface of the boundary structure produced by Boolean operations.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <code>structure</code> parameter denotes a boundary structure. • You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_min_angle=<n></code> <code>[0, 180]</code>	degree
min_dihedral_angle	Number	<p>Sets the smallest dihedral angle between two elements of the decimated surface of the boundary structure produced by Boolean operations that share an edge.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <code>structure</code> parameter denotes a boundary structure. • You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_min_dihedral_angle=<n></code> <code>[0, 180]</code>	degree
region	Character	Sets the region name of the pattern that is created. It can be specified only if <code>structure</code> denotes a boundary structure.	none	none

Chapter 6: Input Commands
pattern

Table 71 Parameters of pattern command (Continued)

Parameter	Type	Description	Default [Range]	Unit
ridge_angle	Number	<p>Sets the angle used by the decimation algorithm to determine geometric features of the boundary structure produced by Boolean operations.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <code>structure</code> parameter denotes a boundary structure. • You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_ridge_angle= <n> [0, 180]</code>	degree
shortest_edge	Number	<p>Sets the shortest edge of the decimated surface of the boundary structure produced by Boolean operations.</p> <p>When set to -1, the used value is half of the structure-dependent spacing <code>epsilon</code>¹.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <code>structure</code> parameter denotes a boundary structure. • You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_shortest_edge=<n> {-1} ∪ [0, ∞[</code>	µm

Table 71 Parameters of pattern command (Continued)

Parameter	Type	Description	Default [Range]	Unit
structure	Character	Sets the name of the structure that must be patterned. Note: This parameter cannot specify a PMC structure. To process such a structure, an explicit conversion of the structure is needed before calling the <code>pattern</code> command (see Integration With Other Sentaurus Topography 3D Functionality on page 519).	default_structure	none
thickness	Number	Sets the thickness of the pattern, measured from the top of the structure.	none [0, ∞[µm
type	Character	Sets the type of the reticle and the resist. Options are: <ul style="list-style-type: none">• dark_negative• dark_positive• light_negative• light_positive	none	none

1. The structure-dependent spacing *epsilon* is logged out when defining a structure using the `define_structure` command.

Description

The `pattern` command provides a convenient way to deposit a new region that depends on a logical mask by combining several process steps.

Depending on the type of the deposited material, there are different ways to interpret the `pattern` command:

- When creating a resist region, the `pattern` command can be interpreted as a shortcut for the following process steps:
 - Deposition of a negative or positive resist
 - Exposure of the resist with a dark-field or a light-field reticle
 - Development of the resist

Chapter 6: Input Commands

pattern

- When creating a region with a different material, the `pattern` command can be interpreted as a shortcut for the following process steps:
 - Deposition of the material
 - Deposition of a negative or positive resist
 - Exposure of the resist with a dark-field or a light-field reticle
 - Development of the resist
 - Geometric etching of the deposited material
 - Stripping of the resist

A reticle is a physical representation of a logical mask that has been created from a layout layer or by operations on one or more logical masks. A reticle has one of two types depending on whether the mask areas covered by polygons transmit or block light:

- When exposing a reticle of type *dark field*, light is transmitted in areas that lie inside of the layout polygons, and light is blocked in areas that lie outside of the layout polygons.
- When exposing a reticle of type *light field*, light is transmitted in areas that lie outside of the layout polygons, and light is blocked in areas that lie inside of the layout polygons.

Similar to a reticle, there are different types of resist:

- When exposed and developed, a negative resist is removed from areas that were not exposed to light, and it remains in areas that were exposed to light.
- When exposed and developed, a positive resist is removed from areas that were exposed to light, and it remains in areas that were not exposed to light.

There are different combinations of reticle and resist types, but there are only two possibilities for how material is deposited, as follows:

- Material is deposited in areas where the polygons are located in the logical mask (`dark_negative` and `light_positive`).
- Material is deposited in areas that are not covered by polygons in the logical mask (`dark_positive` and `light_negative`).

When only looking at the resulting region, it is redundant to provide all four combinations. However, providing all four combinations makes it possible to specify the intended type of processing explicitly.

Examples

This command uses `default_mask` to pattern a film consisting of silicon material, with a thickness of 0.1 µm, measured from the top of the current structure:

```
pattern material=Silicon thickness=0.1
```

Chapter 6: Input Commands

remove_material

Example using a mask and a region name:

```
pattern material=Silicon thickness=0.1 mask=mask_1 \
comment="pattern mask 1" region=mask_1_pattern
```

This command uses the mask `mask_1` to pattern a film consisting of silicon material, with a thickness of 0.1 μm, measured from the top of the device. The patterned region is given the name `mask_1_pattern`.

The comment for the process step is “pattern mask 1” and is displayed in the log file.

remove_material

This command removes material from a boundary structure or PMC structure.

Syntax

```
remove_material material=<c> [comment=<c>] [exposed_only=<b>] \
[structure=<c>]
```

Table 72 Parameters of remove_material command

Parameter	Type	Description	Default	Unit
comment	Character	Sets a comment for this process step that will be displayed or written to the log file.	empty string	none
exposed_only	Boolean	Specifies whether only the exposed material must be removed or all materials.	true	none
material	Character	Sets the material to be removed from the structure.	none	none
structure	Character	Sets the name of the structure from which a material must be removed.	default_structure	none

Examples

Remove all of the exposed silicon material from the structure (silicon that is completely covered by other material is not removed):

```
remove_material material=Silicon comment="strip silicon"
```

Chapter 6: Input Commands

save

save

This command saves or extracts different structures as well as pattern density-related functions, NADs, IADs, species distributions, probabilities, reflection, and yield functions.

Syntax

Save a boundary structure to a TDR file:

```
save type=final_structure [add_interfaces=<b>] [file=<c>] \
[force_save=<b>] [structure=<c>] [tdr_geometry=<c>]
```

Extract the exposed surface of a boundary structure and save it to a TDR file:

```
save type=exposed_surface [file=<c>] [force_save=<b>] \
[include_disconnected_parts=<b>] [include_voids=<b>] \
[structure=<c>] [tdr_geometry=<c>]
```

Extract the closed exposed surface of a boundary structure and save it to a TDR file:

```
save type=closed_surface [file=<c>] [force_save=<b>] \
[structure=<c>] [tdr_geometry=<c>]
```

Save a PMC structure converted to a boundary structure using the dual-contouring method to a TDR file (see [Dual-Contouring Method on page 480](#)):

```
save type=dc [dc_min_angle=<n>] [dc_min_dihedral_angle=<n>] \
[dc_reference_volume_scaling=<n>] [dc_snap_to_box=<b>] \
[dc_version=<n>] [file=<c>] [include_grain_index=<b>] \
[keep_top_gas=<b>] [mean_scaling=<n>] [structure=<c>]
```

Save a PMC structure converted to a GC structure to a TDR file:

```
save [type=gc] [force_save=<b>] [gc_samples_per_cell=<n>] \
[structure=<c>] [tdr_geometry=<c>]
```

Extract the boundary of a PMC structure and save it to a TDR file:

```
save type=vbe [add_interfaces=<b>] [decimate=<b> [accuracy=<n>]] \
[file=<c>] [force_save=<b>] [grain_regions=<b>] \
[include_grain_index=<b>] [material_priority=<l>] \
[material_selection=<c>] [min_angle=<n>] [min_dihedral_angle=<n>] \
[ridge_angle=<n>] [shortest_edge=<n>] [structure=<c>] \
[tdr_geometry=<c>]
```

Save a PMC structure to a PMC file:

```
save type=pmc [file=<c>] [force_save=<b>] [initial_structure=<c>] \
[structure=<c>]
```

Chapter 6: Input Commands

save

Save the local pattern density, effective pattern density, and rate correction factor as a function of x and y in a TDR file:

```
save pattern_density_model=<c> [file=<c>] [point_max=<v>] \
[point_min=<v>]
```

Note:

If the plot domain is not specified explicitly by `point_min` and `point_max`, then it is automatically made large enough to contain all local pattern density regions, plus sufficient space to capture the decay of the effective pattern density function. Since saving large and complex pattern density maps can lead to excessive calculation times and output file size, you should always limit the output domain by setting `point_min` and `point_max` explicitly.

Save PMC damage in a tensor structure in a TDR file:

```
save type=transfer [file=<c>] [point_min=<n>] [point_max=<n>] \
[structure=<c>]
```

Save a PMC structure to a boundary structure in a TDR file by mapping mixed PMC cells to compound materials (the map or maps specified in `merge_replacements` are created in the `define_material_replacement` command):

```
save merge_replacements=<l> [file=<c>] [merge_tolerance=<n>] \
[structure=<c>]
```

Save a grid containing the volume fractions of the species in a PMC structure to a TDR file:

```
save type=volume_fractions [file=<c>] [force_save=<b>] \
[point_max=<v>] [point_min=<v>] [structure=<c>] [tdr_geometry=<c>]
```

When simulating charge-up, save a grid containing the charge density, electric potential, or electric field to a TDR file:

```
save type=charge_density \
[file=<c>] [force_save=<b>] [point_min=<v>] [point_max=<v>] \
[structure=<c>] [tdr_geometry=<c>]
```

```
save type=electric_field \
[file=<c>] [force_save=<b>] [point_min=<v>] [point_max=<v>] \
[structure=<c>] [tdr_geometry=<c>]
```

```
save type=electric_potential \
[file=<c>] [force_save=<b>] [point_min=<v>] [point_max=<v>] \
[structure=<c>] [tdr_geometry=<c>]
```

Save tabular NADs, tabular IADs, probability, reflection, and yield functions to a TDR file:

```
save nad=<c> [file=<c>] [force_save=<b>] [tdr_geometry=<c>]
```

```
save iad=<c> [file=<c>] [force_save=<b>] [tdr_geometry=<c>]
```

Chapter 6: Input Commands

save

```
save probability=<c> [file=<c>] [force_save=<b>] [<sampling_options>]
save reflection=<c> [file=<c>] [force_save=<b>] [<sampling_options>]
save yield=<c> [file=<c>] [force_save=<b>] [<sampling_options>]
```

Save the angular distributions or energy angular distributions (EADs) of species distributions to a TDR file:

```
save species_distribution=<c> [azimuth=<n>] \
[file=<c>] [force_save=<b>] \
[output_type=distribution] [<sampling_options>] [t=<n>]
```

Save the numeric parameters used to create the species distributions as a function of time to a TDR file:

```
save species_distribution=<c> output_type=parameters time=<n> \
[file=<c>] [force_save=<b>]
```

Save species distributions in EAD file format (see [Appendix B on page 554](#)):

```
save species_distribution=<c> file_type=text species=<c> \
[azimuth=<n>] [conversion_factor=<n>] \
[distribution_format=<c>] \
[file=<c>] [file_format=<c>] [force_save=<b>] \
[<sampling_options>] [t=<n>] [version_number=<n>]
```

In the syntax, <sampling_options> denotes a set of optional sampling parameters for the energy- and angle-dependent functions:

```
<sampling_options>=
angle_max=<n> angle_min=<n> angle_samples=<n>
energy_max=<n> energy_min=<n> energy_samples=<n>
```

If you do not specify the energy bounds of the species distribution (energy_min, energy_max), then the bounds are computed automatically within a reasonable range for the given species distribution. You can adjust the precision of the EAD to be saved by increasing the number of angle samples and energy samples.

Note:

When saving an azimuth-dependent species distribution to a TDR file or an EAD text file, a 2D cross section of the distribution at constant azimuth (specified by parameter azimuth) is saved.

When saving time-dependent species distributions, you can use the parameter t to specify the time point at which to save the distribution.

When you start Sentaurus Topography 3D with the --processes command-line option with a value greater than 1 (see [Table 1 on page 17](#)), except when type=gc, the parameter structure must not denote a structure obtained from the etch command using method=pmc averaging_runs=1.

Chapter 6: Input Commands

save

Save the reaction rate coefficients, defined in your plasma model by the command `add_bulk_reaction` (see [add_bulk_reaction](#)), as a function of the electron temperature to a TDR file:

```
save plasma_model=<c> quantity=rate_coefficient \
[file=<c>] [force_save=<b>] \
[expression_pattern=<c> | reactions=<l>] \
[num_samples=<n>] [pattern_type=<c>] [regex_syntax=<c>] \
[temperature_max=<n>] [temperature_min=<n>]
```

Save the solution of a plasma simulation to a file:

```
save type=plasma solution=<c> [file=<c>] [force_save=<b>]
```

Table 73 *Parameters of save command*

Parameter	Type	Description	Default [Range]	Unit
accuracy	Number	Sets the maximum deformation during decimation of the boundary extracted with the VBE method relative to the mean PMC spacing. It can be specified only if <code>decimate=true</code> .	1e-3 [0, ∞[none
add_interfaces	Boolean	If set to <code>true</code> , interface regions for the interface of bulk regions that touch each other are added.	false	none
angle_max	Number	Sets the upper end of the angle window to save.	90 [0, 90]	degree
angle_min	Number	Sets the lower end of the angle window to save.	0 [0, 90]	degree

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
angle_samples	Number	Sets the number of angular samples to save.	90 if saving probability, reflection, or yield functions If saving species distributions, the default value is deduced automatically from the distribution [2, 2147483647]	none
azimuth	Number	Sets the azimuth angle ϕ at which the species distribution is saved ($\phi = 0$ corresponds to the x-axis of the simulation coordinate system).	0 [-180, 180]	degree
conversion_factor	Number	Sets a global conversion factor that scales the EAD.	1]0, ∞[none
dc_min_dihedral_angle	Number	Sets the minimum dihedral angle of the surface mesh (see Dual-Contouring Method).	The value of dc_min_dihedral_angle parameter of let command [0, 180]	degree
dc_min_angle	Number	Sets the minimum angle of the surface mesh (see Dual-Contouring Method on page 480).	The value of dc_min_angle parameter of let command [0, 180]	degree
dc_reference_volume_scaling	Number	When type=dc, this parameter allows you to adjust the surface location when converting a PMC structure to a boundary. Larger values expand the structure into the gas.	none]0, 2[unitless

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
dc_snap_to_box	Boolean	When type=dc and dc_snap_to_box=true, this parameter snaps the outer boundary points to the bounding box. This can help tools that read the boundary to create a bulk mesh.	true	none
dc_version	Number	Selects the version of the dual-contouring algorithm. Options are: <ul style="list-style-type: none"> • 1: Selects the algorithm available in releases prior to S-2021.06. • 2: Selects the algorithm available starting from S-2021.06. 	1	none
decimate	Boolean	Specifies whether to decimate the boundary extracted with the VBE method.	true	none
distribution_format	Character	Sets the mathematical format of the distribution function. Options are: <ul style="list-style-type: none"> • cos_convention • standard 	standard	none
energy_max	Number	Sets the upper end of the energy window to save.	0 if saving probability, reflection, or yield functions If saving species distributions, the default value is deduced automatically from the distribution [0 , ∞[eV

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
energy_min	Number	Sets the lower end of the energy window to save.	0 if saving probability, reflection, or yield functions If saving species distributions, the default value is deduced automatically from the distribution [0 , ∞[eV
energy_samples	Number	Sets the number of energy samples to save.	100 if saving probability, reflection, or yield functions If saving species distributions, the default value is deduced automatically from the distribution [2 , 2147483647]	none
expression_pattern	Character	Sets the search pattern that specifies which reaction equations are considered for plotting the rate coefficients. By default, the search pattern must be given in the glob syntax. To enter a regular expression instead, set pattern_type=regex. You can provide either a pattern or a list of reaction names (parameter reactions). You can specify this parameter only for quantity=rate_coefficient.	none	none

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
file	Character	Sets the path of the file where the data is saved. In the file name, <base_name> denotes the value of the base_name parameter of the let command.	<base_name>.tdr <base_name>.pmc (if type=pmc) <base_name>.txt (if file_type=text) <base_name>.plasma (if type=plasma)	none
file_format	Character	Sets the format of the EAD file.	ead_table	none
file_type	Character	Sets the file type to use for saving the species distribution. Options are: <ul style="list-style-type: none">• tdr• text	tdr	none
force_save	Boolean	Specifies whether to produce output files even if Sentaurus Topography 3D was started with the --no_save command-line option.	false	none
gc_samples_per_cell	Number	Sets the number of samples to take along the x- and y-directions per grid cell.	1 [1, 2147483647]	none
iad	Character	Sets the name of the tabular IAD to be saved.	none	none
include_disconnected_parts	Boolean	Specifies whether to save the parts of the exposed surface lying above the topmost connected component of the exposed surface that divides the simulation domain into two half-spaces.	true	none

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
grain_regions	Boolean	<p>Specifies whether the boundary structure produced by the VBE method should contain different regions for portions of the structure with the same material but with different material properties.</p> <p>Note: The VBE method can create boundary structures with no more than 240 regions.</p>	false	none
include_grain_index	Boolean	<p>Specifies whether to save the material property (grain index in polycrystalline materials) as a field when saving PMC to brep structures using the VBE or DC method.</p> <p>If <code>include_grain_index=true</code>, then when saving from PMC to boundary format, the command writes the grain index as a field into the structure for the <code>filter_structure</code> command, or directly to the file for the <code>save</code> command.</p> <p>Best results are obtained using the <code>filter_structure</code> command with <code>decimate=false</code> to create a boundary with a fine mesh for viewing the field.</p>	false	none
include_voids	Boolean	Specifies whether to save the parts of the exposed surface lying below the topmost connected component of the exposed surface that divides the simulation domain into two half-spaces.	true	none

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
initial_structure	Character	Sets the name of the initial structure that is saved in the PMC file and can be used as the body for Boolean operations.	none	none
keep_top_gas	Boolean	Specifies whether to keep the top gas. This parameter applies only to the dual-contouring method (<code>type=dc</code>). See Dual-Contouring Method on page 480 .	false	none
material_priority	List	Sets a material priority list for materials at interfaces when using the VBE method.	none	none
material_selection	Character	Sets the algorithm used to select the materials of a PMC cell when using the VBE method (see Volume Boundary Extraction Method on page 480). Options are: <ul style="list-style-type: none"> • maximum • mixed • proportional 	mixed	none
mean_spacing	Number	When <code>type=dc</code> , this parameter sets the resolution of the dual-contouring boundary conversion method (see Dual-Contouring Method on page 480).	Taken from most recent PMC step	none

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
merge_replacements	List	<p>Specifies a list of material replacement maps defined in the command <code>define_material_replacement</code>. The list consists of the names of material replacement maps. PMC cells that contain the species matching the material replacement species and <code>volume_fractions</code> (with tolerance specified by <code>merge_tolerance</code>) are converted to the material specified by the material replacement map before the structure is saved.</p> <p>Note: This parameter can be used only with PMC structures.</p>	none	none
merge_tolerance	Number	Sets the allowed deviation from <code>volume_fractions</code> of the material replacement map, so as to perform the merge.	0.1	none
min_angle	Number	<p>Sets the smallest angle in the elements of the decimated surface of the extracted boundary structure.</p> <p>It can be specified only if <code>decimate=true</code>.</p>	0 [0, 180]	degree
min_dihedral_angle	Number	<p>Sets the smallest dihedral angle between two elements, which share an edge, of the decimated surface of the extracted boundary structure.</p> <p>It can be specified only if <code>decimate=true</code>.</p>	0 [0, 180]	degree
nad	Character	Sets the name of the tabular NAD to be saved.	none	none

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
num_samples	Number	When saving the rate coefficient, this parameter sets the number of points used to resolve the rate coefficient curve. You can specify this parameter only for quantity=rate_coefficient.	1000 [2, ∞)	none
output_type	Character	Sets the type of output to produce when saving species distributions. Options are: <ul style="list-style-type: none">• distribution• parameters	distribution	none
pattern_density_model	Character	Sets the pattern density model defined in the define_pattern_density_model command (see define_pattern_density_model on page 256).	none	none
pattern_type	Character	When using expression_pattern to select the plasma reactions, this parameter sets the type of the pattern. Options are glob and regex. You can specify this parameter only for quantity=rate_coefficient.	glob	none
plasma_model	Character	Sets the name of the plasma model containing the reactions.	none	none

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
point_max	Vector	<p>When saving the 2D pattern density, this parameter sets the upper-right corner coordinates {x y} of the domain.</p> <p>Otherwise, it sets the requested maximum corner of the bounding box of the grid containing the data. It can be set only when type=volume_fractions, charge_density, electric_field, or electric_potential.</p> <p>The actual maximum corner of the bounding box of the grid containing the data is obtained by snapping the given point_max value to a grid point of the PMC structure (specified with the parameter structure).</p>	automatic	µm
point_min	Vector	<p>When saving the 2D pattern density, this parameter sets the lower-left corner coordinates {x y} of the domain.</p> <p>Otherwise, it sets the requested minimum corner of the bounding box of the grid containing the data. It can be set only when type=volume_fractions, charge_density, electric_field, or electric_potential.</p> <p>The actual minimum corner of the bounding box of the grid containing the data is obtained by snapping the given point_min value to a grid point of the PMC structure (specified with the parameter structure).</p>	automatic	µm
probability	Character	Sets the name of the probability function to save.	none	none

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
quantity	Character	Sets the quantity to save from the specified plasma model. You can specify this parameter only when <code>plasma_model</code> is specified. The only value is <code>rate_coefficient</code> .	none	none
reactions	List	Specifies the names of the reactions whose rate coefficients should be plotted. If not specified, then all reactions are extracted. You can provide either a list of reaction names or a pattern for the reaction equation (parameter <code>expression_pattern</code>). You can specify this parameter only for <code>quantity=rate_coefficient</code> .	none	none
reflection	Character	Sets the name of the reflection function to save.	none	none
regex_syntax	Character	When using <code>expression_pattern</code> with <code>pattern_type=regex</code> to select the plasma reactions, this parameter defines the syntax of the regular expression. Options are: <ul style="list-style-type: none">• extended• awk• basic• ecma_script• egrep• grep You can specify this parameter only for <code>quantity=rate_coefficient</code> .	extended	none

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
ridge_angle	Number	Sets the angle used by the decimation algorithm to determine geometric features of the extracted boundary structure. It can be specified only if decimate=true.	179 [0, 180]	degree
shortest_edge	Number	Sets the shortest edge of the decimated surface of the extracted boundary structure, relative to the mean PMC spacing. It can be specified only if decimate=true.	The value set by the accuracy parameter [0, ∞[none
solution	Character	Sets the name of the plasma solution, which is saved to a file. The solution name is specified in the command solve_reactor.	none	none
species	Character	Sets the name of the species to save.	none	none
species_distribution	Character	Sets the name of the species distribution to save.	none	none
structure	Character	Sets the name of the structure to save.	default_structure	none
t	Number	Sets the time at which to save the species distributions.	0 [0, ∞[minute
tdr_geometry	Character	Sets the name of the TDR geometry to save.	none	none
temperature_max	Number	Sets the upper limit of the plotting range when plotting the rate coefficient as a function of the electron temperature. You can specify this parameter only for quantity=rate_coefficient.	100 [0, ∞[eV

Chapter 6: Input Commands

save

Table 73 Parameters of save command (Continued)

Parameter	Type	Description	Default [Range]	Unit
temperature_min	Number	Sets the lower limit of the plotting range when plotting the rate coefficient as a function of the electron temperature. You can specify this parameter only for quantity=rate_coefficient.	0 [0, ∞[eV
time	Number	Sets the time until which to save the numeric parameters used to create the species distributions.	none [0, ∞[minute
type	Character	Sets the type of data to save. Options are: <ul style="list-style-type: none">• charge_density• closed_surface• dc• electric_field• electric_potential• exposed_surface• final_structure• gc• plasma• pmc• transfer• vbe• volume_fractions	gc if the parameter structure denotes a PMC structure; final_structure otherwise	none
version_number	Number	Sets the version number of the EAD file.	0.0 [0.0, ∞[none
yield	Character	Sets the name of the yield function to save.	none	none

Chapter 6: Input Commands

save

Note:

When `type=closed_surface`, or `type=exposed_surface`, or `type=final_structure`, the parameter `structure` cannot specify a PMC structure. To process such a structure, an explicit conversion of the structure is needed before calling the `save` command (see [Integration With Other Sentaurus Topography 3D Functionality on page 519](#)).

When `type=gc`, `type=pmc`, `type=transfer`, `type=volume_fractions`, or `type=vbe`, the parameter `structure` must specify a PMC structure.

Saving Structures

As described in [Boundary Types on page 30](#), the following boundary types can be saved:

- Closed surface (`type=closed_surface`)
- Exposed surface (`type=exposed_surface`)
- Final structure (`type=final_structure`)

By default, the `save` command saves the final structure.

Note:

See [Integration With Other Sentaurus Topography 3D Functionality on page 519](#) for the limitations when saving a PMC structure.

The command `save` can be used more than once in the same command file. The first time that a file name (default name or user-defined name using the parameter `file`) is used, a new TDR file with that name is created or overwritten in the case where the file already exists from a previous simulation.

In subsequent uses of the `save` command that refer to the same file name, the results are appended to the previously saved structures in that file independently of the saved type or of the simulation step that preceded the `save` command. In other words, a file that is created within a simulation is never overwritten, and data is always appended to it.

You can save the resulting structure into a different file by creating a new file with the parameter `file`.

Examples

```
save
```

```
save type=closed_surface
```

The first `save` command saves the final structure into the default TDR file. The second `save` command adds the corresponding closed surface to the same file.

```
save file="mosfet.tdr" tdr_geometry="pmos"
```

Chapter 6: Input Commands

save

This command saves the current structure to the TDR file `mosfet.tdr` and gives the created TDR geometry the name `pmos`. The name can be reused to load this specific geometry in the `define_structure` command, for example.

Saving Ion Angular Distributions

The `save` command can save ion angular distributions (IADs) to TDR files.

Examples

Save the IAD named `iad_1` to the default TDR file:

```
save iad=iad_1
```

Save the IAD `iad_1` to the file `iad1.tdr`, and name the IAD in the file `ion_iad`:

```
save iad=iad_1 file=iad1.tdr tdr_geometry=ion_iad
```

Saving PMC Structures

A PMC structure is stored in a PMC file with the `save` command when `type=pmc`.

The advantage of using PMC files to create splits in a process flow, compared to TDR files containing boundaries, is that no conversion to a boundary and back to the PMC representation is required. Therefore, loading and saving is more efficient and does not introduce artifacts.

To perform Boolean operations and to create a boundary from a PMC structure that has been loaded from a PMC file, the initial structure from which the PMC structure has been created is required. The parameter `initial_structure` is used to specify the initial structure that is stored in the PMC file.

Note:

A PMC file is not a TDR file and cannot be visualized with Sentaurus Visual. It can only be loaded in to Sentaurus Topography 3D.

Examples

```
etch time=0.5 spacing=0.05 method=pmc machine=m_100  
save file=pmc_only.pmc type=pmc
```

The first command etches a structure using the PMC method. The second command saves the PMC structure to a PMC file without saving the initial structure to the PMC file. Therefore, this PMC file can be used only as a starting point for other PMC simulations. It is not possible to perform Boolean operations or to convert the PMC structure to a boundary.

```
define_structure pmc_file=pmc_only.pmc  
etch time=0.5 spacing=0.05 method=pmc machine=m_100
```

Chapter 6: Input Commands

save

Here, the previously saved PMC structure is loaded from a PMC file and used for a PMC simulation.

```
define_structure material=Silicon point_min={0 0 0} point_max={1 1 1}
filter_structure type=copy name=sc
etch time=0.5 spacing=0.05 method=pmc machine=m_100
save file=pmc_and_initial.pmc type=pmc initial_structure=sc
```

In this example, an initial structure is created and copied to a structure named `sc`. After etching the initial structure, it is saved to a PMC file together with the boundary of the copy of the initial structure.

```
define_structure pmc_file=pmc_and_initial.pmc
etch time=0.5 spacing=0.05 method=pmc machine=m_100
filter_structure type=boolean body=sc
save file=f.tdr
```

In the first command, the PMC structure and the initial structure are loaded from a PMC file. After etching the PMC structure, Boolean operations are performed with the initial structure named `sc` loaded from the PMC file. Finally, the structure is saved as a boundary to a TDR file.

If the name of the initial structure stored in the PMC file is unknown, a new name can be specified with the parameter `initial_structure_name` as demonstrated in the following example:

```
define_structure pmc_file=pmc_and_initial.pmc \
    initial_structure_name=xyz
etch time=0.5 spacing=0.05 method=pmc machine=m_100
filter_structure type=boolean body=xyz
save file=f.tdr
```

Saving Boundaries for PMC Structures

To save a boundary structure for a PMC structure that has been created from a boundary structure using a PMC etch or deposition model, you can create a new boundary structure using `filter_structure type=boolean` and the initial boundary structure. The advantage of this method is that artifacts are limited to the parts of the structure that were modified by process steps. However, this method is computationally expensive.

There are alternative methods for saving a boundary structure that extract a boundary structure from a PMC structure without using the initial boundary structure:

- Dual-contouring method [17][18][19]
- Volume boundary extraction (VBE) method

The advantage of these methods is that they are computationally more efficient than the previously described method. However, when using either alternative method, conversion

Chapter 6: Input Commands

save

artifacts might be created throughout the entire structure, not only in the parts modified by process steps.

Dual-Contouring Method

Converting a PMC structure to a boundary structure using the dual-contouring method generally works well, but the conversion time and shape of the final boundaries can be fine-tuned with the parameters of the `save` command. To use the default parameter values, you only need to specify:

```
save type=dc
```

The main parameter that controls boundary fidelity is `mean_spacing`. The default of this parameter is the spacing used for the most recent PMC step. Specifying a smaller mean spacing leads to a more accurate boundary conversion; whereas, a larger mean spacing reduces runtime.

Note:

Do not specify `mean_spacing` less than approximately $\frac{1}{4}$ of the spacing used in the most recent PMC step because smaller spacing is not likely to improve the boundary fidelity further, but it is more likely to result in large runtimes.

The parameters `dc_min_angle` and `dc_min_dihedral_angle` can improve volume meshes in TCAD tools reading the boundary file. In most cases, the dual-contouring algorithm satisfies those criteria without compromising the boundary fidelity or introducing artifacts. However, a very high minimum dihedral angle threshold ($> 30^\circ$) might not be satisfied and can result in distortion or a slightly higher vertex count.

An improved version of the dual-contouring algorithm is available, which generates boundaries of a higher fidelity on interfaces and triple lines. It is also more effective at fitting the structure to the bounding box while satisfying the fidelity and quality shape criteria. These improvements might require additional computation time, which is usually less than 30%. The default version of the dual-contouring algorithm is the same as in releases prior to S-2021.06. To specify the improved version of the dual-contouring algorithm, specify `dc_version=2` either in the `filter_structure` command or in the `save` command.

Volume Boundary Extraction Method

The VBE method can also be used to save boundary structures for structures that were initially created as PMC structures.

Converting a PMC structure to a boundary structure using the VBE method might introduce artifacts at material interfaces due to the different discretization methods. The parameters `material_priority` and `material_selection` provide some control over the selection of materials at the interface of different materials.

Chapter 6: Input Commands

save

When using the material selection algorithm:

- If `material_selection=maximum`, the entire volume of a PMC cell is considered to be of the material that has the maximum partial volume.
- If `material_selection=proportional`, each material in a PMC cell is initially represented proportionally to its partial volume. Because the final result also depends on the materials in neighboring cells, materials for which the partial volume is less than 50% might be replaced by a different material.
- If `material_selection=mixed`, the effect is similar to `proportional`, but if the partial volume of the material with the highest partial volume in a cell is less than 50%, the partial volume is increased to 50% to ensure that this material is not replaced by a different material in the final result.

When two materials have the same partial volume in a cell, preference is given to the material with the higher priority. The parameter `material_priority` can specify materials according to their priority. The material listed first has the highest priority. Materials that are not in the list have lower priority than any of the listed materials.

Saving Species Distributions

The `save` command can save species distributions (EADs and angular distributions) to TDR files or EAD files.

The following command saves the EAD of a species distribution named `sd` in the energy range from 100 eV to 200 eV to the default TDR file:

```
save species_distribution=sd energy_min=100 energy_max=200
```

To save the angular distribution of a species distribution named `sd` at a certain energy level (for example, 100 eV) to the default TDR file, you must set `energy_min` and `energy_max` to the same value:

```
save species_distribution=sd energy_min=100 energy_max=100
```

You can change the default resolution of the saved EAD by specifying the number of energy and angle samples:

```
save species_distribution=sd angle_samples=180 energy_min=100 \
      energy_max=200 energy_samples=200
```

Accordingly, to change the default resolution of the angular distribution at energy 100 eV, specify:

```
save species_distribution=sd angle_samples=180 energy_min=100 \
      energy_max=100
```

Chapter 6: Input Commands

set_material_properties

The following command saves the species distribution to an EAD file:

```
save species_distribution=sd species=Ar file_type=text
```

You can change the default EAD file format, the format of the distribution, the conversion factor, the version number, the energy bounds, and the number of energy samples and angle samples:

```
save species_distribution=sd species=Ar file_type=text \
angle_samples=180 conversion_factor=10 version_number=1.1 \
distribution_format=cos_convention energy_min=10 \
energy_max=200 energy_samples=500 file_format=ead_table \
```

set_material_properties

This command sets the crystal material properties for a material or a region in a structure.

Note:

For this command:

- The material of structures defined using `define_structure material=<c> point_min=<v> point_max=<v> [structure=<c>] ...` is amorphous.
- The materials added using the commands `deposit shape=<c> ..., fill, and pattern` are amorphous.

Syntax

For 3D boundary and GC structures:

```
set_material_properties material=<c> type=<c> <material_properties> \
[structure=<c>]

set_material_properties region=<c> type=<c> <material_properties> \
[structure=<c>]
```

For PMC structures:

```
set_material_properties material=<c> type=<c> <material_properties> \
[structure=<c>]
```

Chapter 6: Input Commands

set_material_properties

Table 74 Parameters of set_material_properties command

Parameter	Type	Description	Default	Unit
anisotropy_xz, anisotropy_yz	Number	Specify the ratio of the average grain size in x, with respect to z. The average grain size in x is <code>anistropy_xz*average_grain_size</code> and, similarly, for y <code>anistropy_yz*average_grain_size</code> .	1]0, ∞[none
average_grain_size	Number	Sets the average size of the polycrystalline material grains.	none]0, ∞[µm
crystal_type	Character	Sets the crystal type of crystalline or polycrystalline materials. The only supported option is diamond.	none	none
flat_orientation	Vector	Sets the Miller index of the direction perpendicular to the wafer flat (y-axis of the wafer coordinate system) for the material or region. Note: The direction specified by this parameter must be orthogonal to the direction specified by the parameter <code>vertical_orientation</code> .	none	none
lattice_constant	Number	Sets the lattice constant of the crystal for crystalline or polycrystalline materials.	0.5431020511e-3]0, ∞[µm
material	Character	Sets the name of the material for which the properties are set.	none	none
<material_properties>	Character	Denotes a set of parameters dependent on the value of parameter <code>type</code> , as specified in Table 75 .	none	none

Chapter 6: Input Commands

set_material_properties

Table 74 Parameters of set_material_properties command (Continued)

Parameter	Type	Description	Default	Unit
preferred_vertical_orientation	Vector	Sets the Miller index of the grain-preferred direction perpendicular to the wafer surface (z-axis of the wafer coordinate system) for the material or region. If omitted, then grain orientations are distributed uniformly.	none	none
region	Character	Sets the name of the region for which the properties are set.	none	none
structure	Character	Sets the name of the structure for which the properties are set.	default_structure	none
type	Character	Sets the type of the material. Options are: <ul style="list-style-type: none">• amorphous• crystalline• polycrystalline	none	none
vertical_orientation	Vector	Sets the Miller index of the direction perpendicular to the wafer surface (z-axis of the wafer coordinate system) for the material or region. Note: The direction specified by this parameter must be orthogonal to the direction specified by flat_orientation.	none	none
vertical_orientation_spread	Number	Sets the angle of allowed deviation from the preferred orientation for polycrystalline materials.	10 [1, 90]	degree

Chapter 6: Input Commands

set_material_properties

Table 75 Material properties: type dependent

type	<material_properties>
amorphous	No parameters
crystalline	crystal_type=<c> flat_orientation=<v> vertical_orientation=<v> [lattice_constant=<n>]
polycrystalline	crystal_type=<c> average_grain_size=<n> [anisotropy_xz=<n>] [anisotropy_yz=<n>] [lattice_constant=<n>] [preferred_vertical_orientation=<v>] [vertical_orientation_spread=<n>]

Examples

Make material Oxide of structure default_structure amorphous:

```
set_material_properties material=Oxide type=amorphous
```

Make material Silicon of structure default_structure crystalline with a diamond crystal structure and crystal orientation such that:

- When cutting the material with a plane parallel perpendicular to the y-axis of the wafer coordinate system, a (1 1 0) plane is exposed
- When cutting the material with a plane parallel perpendicular to the z-axis of the wafer coordinate system, a (0 0 1) plane is exposed

```
set_material_properties material=Silicon type=crystalline \  
crystal_type=diamond flat_orientation={1 1 0} \  
vertical_orientation={0 0 1}
```

Make material Silicon of structure default_structure polycrystalline with grains of an average size of 0.1 mm and (0 0 1) as the preferred vertical orientation with an angular spread of 15°:

```
set_material_properties material=Silicon type=polycrystalline \  
crystal_type=diamond \  
preferred_vertical_orientation={0 0 1} \  
vertical_orientation_spread=15
```

Chapter 6: Input Commands

set_orientation

set_orientation

This command is used to set or change the crystal orientation of a region.

Syntax

```
set_orientation flat_orientation=<v> region=<c> \
    vertical_orientation=<v> [structure=<c>]
```

Table 76 Parameters of set_orientation command

Parameter	Type	Description	Default	Unit
flat_orientation	Vector	Sets the Miller index of the direction perpendicular to the wafer flat (y-axis of the wafer coordinate system) for this region.	none	none
region	Character	Sets the name of the region for which the crystal orientation is set.	none	none
structure	Character	Sets the name of the structure whose crystal orientation must be set.	default_structure	none
vertical_orientation	Vector	Sets the Miller index of the direction perpendicular to the wafer surface (z-axis of the wafer coordinate system) for this region.	none	none

Chapter 6: Input Commands

solve_reactor

solve_reactor

This command executes the plasma model solvers, that is, it solves the plasma bulk and plasma sheath models defined in the plasma model.

Syntax

```
solve_reactor name=<c> reactor=<c> \
    [bulk_solver=<c>] [extractions=<l>] [sheath_solver=<c>]
```

Table 77 Parameters of solve_reactor command

Parameter	Type	Description	Default	Unit
bulk_solver	Character	Sets the name of the plasma bulk solver to be used for solving the plasma model. It must have been already defined in the define_bulk_solver command (see define_bulk_solver on page 187).	default_global_bulk_solver	none
extractions	List	Specifies a list of the names of extractions to be used when solving the plasma model. Each name in the list must have been already defined in the define_extraction command (see define_extraction on page 218).	none	none
name	Character	Sets the name of the plasma solution. The name is used for later reference, for example, in the define_reactor or define_species_distribution command (see define_reactor on page 261 and define_species_distribution on page 290).	none	none

Chapter 6: Input Commands

solve_reactor

Table 77 Parameters of solve_reactor command (Continued)

Parameter	Type	Description	Default	Unit
reactor	Character	Sets the name of the plasma reactor (as defined in the define_reactor command), which contains the equipment parameters and the plasma model to be solved (see define_reactor on page 261).	none	none
sheath_solver	Character	Sets the name of the plasma sheath solver to be used for solving the plasma model, as defined in the command define_sheath_solver (see define_sheath_solver on page 285).	Depends on the sheath model type defined in the plasma model: default_analytic_sheath_solver or default_circuit_sheath_solver	none

Examples

```
solve_reactor name=sol1 reactor=R  
  
solve_reactor name=sol1 reactor=R \  
    bulk_solver=default_global_bulk_solver  
  
solve_reactor name=sol1 reactor=R bulk_solver=s1 sheath_solver=s2
```

Note:

The solution computed by the solve_reactor command is stored internally and can be referenced by the name parameter in other commands, such as save and define_species_distribution. Some scalar quantities of the solution are also returned as a Tcl list, for example, the bulk densities, the electron temperature, the species fluxes, and so on. For example:

```
set res [solve_reactor name=sol1 reactor=R]  
puts $res  
>>> {{Ar+_density 2.85574e+17} {Ar+_energy_max 45.4943}  
{Ar+_energy_min 13.6602} {Ar+_flux 0.000779226}  
{e-_temperature 3.00445} ...}
```

Chapter 6: Input Commands

transform_structure

transform_structure

This command performs an axis-mapping transformation on structures, that is, it remaps one axis onto another, changing the coordinate system of the structure.

If you set only one target axis, then the other axis is set automatically to the other axis with the same sign. For example, if you set only `target_y_axis= -x`, then `target_x_axis= -y` is set automatically.

Note:

The `axis_mapping` transformation works only for 2D brep structures.

Syntax

```
transform_structure type=<c> [name=<c>] [structure=<c>] \
[target_x_axis=<c>] [target_y_axis=<c>]
```

Table 78 Parameters of `transform_structure` command

Parameter	Type	Description	Default	Unit
name	Character	Sets the name of the transformed structure. If not set, then the input structure is replaced with the transformed structure.	none	none
structure	Character	Sets the name of the input structure.	default_structure	none
target_x_axis	Character	Sets the target x-axis. Options are: <ul style="list-style-type: none">• x• -x• y• -y The parameters <code>target_x_axis</code> and <code>target_y_axis</code> cannot reference the same axis.	none	none
target_y_axis	Character	Sets the target y-axis. Options are: <ul style="list-style-type: none">• x• -x• y• -y The parameters <code>target_x_axis</code> and <code>target_y_axis</code> cannot reference the same axis.	none	none

Chapter 6: Input Commands

transform_structure

Table 78 Parameters of transform_structure command (Continued)

Parameter	Type	Description	Default	Unit
type	Character	Sets the type of transformation to be performed. The only option is axis_mapping.	none	none

Examples

```
transform_structure type=axis_mapping target_x_axis= -y
```

Chapter 6: Input Commands

truncate

truncate

This command truncates a structure, removing everything that lies outside of a cuboid bounding box.

Note:

When you start Sentaurus Topography 3D with the `--processes` command-line option with a value greater than 1 (see [Table 1 on page 17](#)), the parameter structure must not denote a structure obtained from the `etch` command using `method=pmc averaging_runs=1`.

Syntax

Truncate a boundary structure:

```
truncate point_min=<v> point_max=<v> \
[accuracy=<n>] [comment=<c>] [decimate=<b>] [min_angle=<n>] \
[min_dihedral_angle=<n>] [ridge_angle=<n>] [shortest_edge=<n>] \
[structure=<c>]
```

Truncate a PMC structure:

```
truncate point_min=<v> point_max=<v> [comment=<c>] [structure=<c>]
```

Table 79 Parameters of *truncate* command

Parameter	Type	Description	Default [Range]	Unit
accuracy	Number	<p>Sets the maximum deviation between the decimated surface of the boundary structure produced by Boolean operations and its original surface.</p> <p>When set to -1, the used value is half of the structure-dependent spacing <code>epsilon</code>¹.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none">• The <code>structure</code> parameter denotes a boundary structure.• You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>.	The value set by <code>let decimate_accuracy=<n></code> $\{-1\} \cup [0, \infty[$	µm

Chapter 6: Input Commands

truncate

Table 79 Parameters of truncate command (Continued)

Parameter	Type	Description	Default [Range]	Unit
comment	Character	Sets a comment for this process step that will be displayed or written to the log file.	empty string	none
decimate	Boolean	Specifies whether to decimate the boundary structure produced by Boolean operations. It can be specified only if <code>structure</code> denotes a boundary structure.	The value set by <code>let decimate=</code>	none
min_angle	Number	<p>Sets the smallest angle in the elements of the decimated surface of the boundary structure produced by Boolean operations.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <code>structure</code> parameter denotes a boundary structure. • You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false.</code> 	The value set by <code>let decimate_min_angle=<n></code> [0, 180]	degree
min_dihedral_angle	Number	<p>Sets the smallest dihedral angle between two elements of the decimated surface of the boundary structure produced by Boolean operations that share an edge.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <code>structure</code> parameter denotes a boundary structure. • You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false.</code> 	The value set by <code>let decimate_min_dihedral_angle=<n></code> [0, 180]	degree

Chapter 6: Input Commands

truncate

Table 79 Parameters of truncate command (Continued)

Parameter	Type	Description	Default [Range]	Unit
point_max	Vector	<p>Sets the maximum corner point of the bounding cube.</p> <p>When truncating a PMC structure, the actual maximum corner of the truncated PMC structure is obtained by snapping the <code>point_max</code> value to a grid point of the input PMC structure.</p>	none	µm
point_min	Vector	<p>Sets the minimum corner point of the bounding cube.</p> <p>When truncating a PMC structure, the actual minimum corner of the truncated PMC structure is obtained by snapping the <code>point_min</code> value to a grid point of the input PMC structure.</p>	none	µm
ridge_angle	Number	<p>Sets the angle used by the decimation algorithm to determine geometric features of the boundary structure produced by Boolean operations.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> • The <code>structure</code> parameter denotes a boundary structure. • You specify <code>decimate=true</code> or, if you do not specify <code>decimate</code>, you did not specify <code>let decimate=false</code>. 	The value set by <code>let decimate_</code> <code>ridge_angle=<n></code> <code>[0, 180]</code>	degree

Chapter 6: Input Commands

truncate

Table 79 Parameters of truncate command (Continued)

Parameter	Type	Description	Default [Range]	Unit
shortest_edge	Number	<p>Sets the shortest edge of the decimated surface of the boundary structure produced by Boolean operations.</p> <p>When set to -1, the used value is half of the structure-dependent spacing <i>epsilon</i>¹.</p> <p>This parameter can be specified only if all of the following conditions are met:</p> <ul style="list-style-type: none"> The <i>structure</i> parameter denotes a boundary structure. You specify <i>decimate=true</i> or, if you do not specify <i>decimate</i>, you did not specify <i>let decimate=false</i>. 	The value set by let decimate_shortest_edge=<n> {-1} ∪ [0, ∞[μm
structure	Character	<p>Sets the name of the structure that must be truncated.</p> <p>Note: This parameter cannot specify a PMC structure. To process such a structure, an explicit conversion of the structure is needed before calling the <i>truncate</i> command (see Integration With Other Sentaurus Topography 3D Functionality on page 519).</p>	default_structure	none

1. The structure-dependent spacing *epsilon* is logged out when defining a structure using the *define_structure* command.

Examples

Remove all parts of a structure that are outside a bounding box with the specified corners {0 0 0} {1 1 1}:

```
truncate point_min={0 0 0} point_max={1 1 1} comment="truncation"
```

The comment for this process step is displayed in the log file.

References

- [1] T. Mizuno *et al.*, “Analytical Model for Oblique Ion Reflection at the Si Surface,” *IEEE Transactions on Electron Devices*, vol. 35, no. 12, pp. 2323–2327, 1988.
- [2] D. Zhang and M. J. Kushner, “Surface kinetics and plasma equipment model for Si etching by fluorocarbon plasmas,” *Journal of Applied Physics*, vol. 87, no. 3, pp. 1060–1069, 2000.
- [3] R. Akolkar and U. Landau, “Mechanistic Analysis of the “Bottom-Up” Fill in Copper Interconnect Metallization,” *Journal of the Electrochemical Society*, vol. 156, no. 9, pp. D351–D359, 2009.
- [4] J. A. Sethian and Y. Shan, “Solving partial differential equations on irregular domains with moving interfaces, with applications to superconformal electrodeposition in semiconductor manufacturing,” *Journal of Computational Physics*, vol. 227, no. 13, pp. 6411–6447, 2008.
- [5] K. O. Abrokwah, *Characterization and Modeling of Plasma Etch Pattern Dependencies in Integrated Circuits*, Master’s thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA, February 2006.
- [6] K. O. Abrokwah, P. R. Chidambaram, and D. S. Boning, “Pattern Based Prediction for Plasma Etch,” *IEEE Transactions on Semiconductor Manufacturing*, vol. 20, no. 2, pp. 77–86, 2007.
- [7] T. F. Hill *et al.*, “Pattern Density Based Prediction for Deep Reactive Ion Etch (DRIE),” in *Technical Digest of Solid-State Sensor, Actuator and Microsystems Workshop*, Hilton Head Island, SC, USA, June 2004.
- [8] D. Okumu Ouma *et al.*, “Characterization and Modeling of Oxide Chemical–Mechanical Polishing Using Planarization Length and Pattern Density Concepts,” *IEEE Transactions on Semiconductor Manufacturing*, vol. 15, no. 2, pp. 232–244, 2002.
- [9] P. Benoit-Cattin and L.-C. Bernhard, “Anomalies of the Energy of Positive Ions Extracted from High-Frequency Ion Sources. A Theoretical Study,” *Journal of Applied Physics*, vol. 39, no. 12, p. 5723–5726, 1968.
- [10] M. A. Lieberman and A. J. Lichtenberg, *Principles of Plasma Discharges and Materials Processing*, Hoboken, New Jersey: John Wiley & Sons, 2nd ed., 2005.
- [11] Y. Wang *et al.*, “Verification of collisionless sheath model of capacitive rf discharges by particle-in-cell simulations,” *Journal of Applied Physics*, vol. 110, no. 3, p. 033307, 2011.
- [12] E. A. Edelberg and E. S. Aydil, “Modeling of the sheath and the energy distribution of ions bombarding rf-biased substrates in high density plasma reactors and comparison to experimental measurements,” *Journal of Applied Physics*, vol. 86, no. 9, pp. 4799–4812, 1999.

Chapter 6: Input Commands

References

- [13] M. J. Kushner, "Hybrid modelling of low temperature plasmas for fundamental investigations and equipment design," *Journal of Physics D: Applied Physics*, vol. 42, no. 19, p. 194013, 2009.
- [14] HPEM is a plasma simulation tool developed by Prof. Kushner and distributed by Quantemol Ltd (<http://www.quantemol.com>). For further information, contact info@quantemol.com.
- [15] T. K. Chini *et al.*, "The angular dependence of sputtering yields of Ge and Ag," *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, vol. 72, no. 3–4, pp. 355–358, 1992.
- [16] R. Courant, K. Friedrichs, and H. Lewy, "On the Partial Difference Equations of Mathematical Physics," *IBM Journal*, vol. 11, no. 2, pp. 215–234, 1967.
- [17] T. Ju *et al.*, "Dual Contouring of Hermite Data," *ACM Transactions on Graphics (TOG)*, vol. 21, no. 3, pp. 339–346, 2002.
- [18] S. Schaefer and J. Warren, *Dual Contouring: "The Secret Sauce"*, Technical Report, Department of Computer Science, Rice University, Houston, TX, USA, March 2003.
- [19] Y. Zhang and J. Qian, "Dual Contouring for domains with topology ambiguity," *Computer Methods in Applied Mechanics and Engineering*, vol. 217–220, pp. 34–45, April 2012.

7

Integration With Sentaurus Process and Sentaurus Interconnect

This chapter describes the integration of Sentaurus Topography 3D functionality into Sentaurus Process and Sentaurus Interconnect.

Introduction

The Sentaurus Process and Sentaurus Interconnect executables support a subset of Sentaurus Topography 3D commands for advanced etching and deposition modeling. This integration allows the users of Sentaurus Process and Sentaurus Interconnect to easily incorporate one or more etching and deposition process steps into their simulation flow without having to create a separate command file and simulation node for Sentaurus Topography 3D.

Not all commands of Sentaurus Topography 3D are supported in Sentaurus Process and Sentaurus Interconnect, however.

Note:

For brevity, in this chapter, whenever *Sentaurus Process* is mentioned, it implicitly means *Sentaurus Process and Sentaurus Interconnect*.

Supported Commands and Syntax

Sentaurus Topography 3D functionality is accessible using the `topo` command in Sentaurus Process. The `topo` command acts as a prefix and branches into a subset of Sentaurus Topography 3D commands. In general, the `topo` command is followed by the Sentaurus Topography 3D subcommand.

For example, the following command calls the Sentaurus Topography 3D `etch` command from Sentaurus Process (the parameters are the same as for the original `etch` command):

```
topo etch spacing= {0.1} time=1.0
```

Chapter 7: Integration With Sentaurus Process and Sentaurus Interconnect

Introduction

The syntax of the Sentaurus Topography 3D subcommands is the same as in Sentaurus Topography 3D (see [Chapter 6 on page 129](#)).

Table 80 Sentaurus Topography 3D commands supported from within Sentaurus Process

Sentaurus Topography 3D command	Sentaurus Process command
add_bulk_reaction	topo add_bulk_reaction
add_float_parameter	topo add_float_parameter
add_flux_properties	topo add_flux_properties
add_formula	topo add_formula
add_int_parameter	topo add_int_parameter
add_ion_flux	topo add_ion_flux
add_material	topo add_material
add_neutral_flux	topo add_neutral_flux
add_reaction	topo add_reaction
add_reaction_properties	topo add_reaction_properties
add_source_species	topo add_source_species
add_species	topo add_species
define_bulk_solver	topo define_bulk_solver
define_deposit_machine	topo define_deposit_machine
define_etch_machine	topo define_etch_machine
define_iad	topo define_iad
define_model	topo define_model
define_plasma_model	topo define_plasma_model
define_probability	topo define_probability
define_reactor	topo define_reactor
define_reflection	topo define_reflection

Table 80 Sentaurus Topography 3D commands supported from within Sentaurus Process

Sentaurus Topography 3D command	Sentaurus Process command
define_sheath_solver	topo define_sheath_solver
define_species_distribution	topo define_species_distribution
define_species_properties	topo define_species_properties
define_yield	topo define_yield
deposit	topo deposit
etch	topo etch
finalize_model	topo finalize_model
solve_reactor	topo solve_reactor

Additional Supported Parameters

The `topo` command and Sentaurus Topography 3D subcommands in Sentaurus Process support some parameters that are exclusive to Sentaurus Process.

The info Parameter

This parameter sets the verbosity of the output of a command. See the *Sentaurus™ Process User Guide* for details.

Example

Call the `etch` command from Sentaurus Process with a verbosity level of 3:

```
topo etch spacing= {0.1} time=1.0 info=3
```

The parameters Parameter

This parameter prints a list of all the supported subcommands and their syntax.

Example

Print the syntax of all the supported Sentaurus Topography 3D subcommands:

```
topo parameters
```

The repair Parameter

The `topo etch` and `topo deposit` commands support the `repair` parameter, which switches on boundary repair for etching and deposition steps. By default, boundary repair is switched on. See the *Sentaurus™ Process User Guide* for details.

Example

Call the `topo deposit` command with boundary repair deactivated:

```
topo deposit spacing= {0.1} time=1.0 !repair
```

Different Default Behavior

To ensure consistent behavior of the Sentaurus Topography 3D commands when they are called from Sentaurus Process, some commands have different default behavior. These commands behave slightly differently when called from Sentaurus Topography 3D or Sentaurus Process.

Boundary Repair

The boundary repair behavior differs as follows:

- Sentaurus Topography 3D default behavior: Boundary repair is deactivated because it can use excessive CPU time and memory. You can activate boundary repair using the command:

```
let snmesh_repair=true
```

- Sentaurus Process default behavior: Boundary repair for `topo etch` and `topo deposit` is activated (see [The repair Parameter](#)).

Region Merging

Both when calling the `topo` command from Sentaurus Process and when using Sentaurus Topography 3D directly, regions with the same materials are not merged by default.

This behavior is different from deposition using the MGOALS module in Sentaurus Process, where regions merge by default.

To activate region merging in the `topo` command, use the `merge` parameter in the `deposit` and `etch` commands (see [deposit on page 322](#) and [etch on page 339](#)).

Parallelization

Parallelization differs as follows:

- In standalone Sentaurus Topography 3D, the `let num_threads` and `let parallel` commands as well as the command-line options `--max_threads` and `--threads` are used to activate shared-memory parallelization and to control the number of threads (see [From the Command Line on page 17](#) and [let on page 440](#)).
- In Sentaurus Process, the `math` command controls shared-memory parallelization. The `topo` command uses the number of threads defined with the `math` command. See the *Sentaurus™ Process User Guide* for details about parallelization.

Limitations and Known Issues

There are limitations when using Sentaurus Topography 3D from Sentaurus Process.

Boundary Conditions

In Sentaurus Topography 3D, boundary conditions are set using the command `define_boundary_conditions`, which is not a supported subcommand of the `topo` command. Therefore, only the default boundary conditions can be used from within Sentaurus Process.

Meshing Thin Layers

In standalone mode, Sentaurus Topography 3D creates boundary files as a result of process steps. Sentaurus Process, however, needs 3D meshes for most of its functionality. Boundaries are transferred from the `topo` command to Sentaurus Process automatically, and they are meshed when needed using Sentaurus Process commands such as `implant`.

Meshing of very thin material layers can be difficult. Meshing might either take a very long time or terminate if Sentaurus Mesh cannot mesh the structure.

Sentaurus Topography 3D often generates thin layers, especially in models where redeposition effects are modeled (for example, when using the `etchdepo` model).

8

Rate Formula Module

This chapter describes the rate formula module.

Introduction to the Rate Formula Module

The rate formula module (RFM) allows you to define deposition and etching models in Sentaurus Topography 3D. In contrast to the physical model interface (PMI), which requires C++ programming, the RFM uses only built-in commands. Therefore, the RFM simplifies the development of new models and provides greater flexibility in modeling.

You can specify the deposition and etching rate by writing a mathematical formula. This formula can involve fluxes, geometric quantities of the surface, user-defined parameters, and mathematical functions. It is used in each time step as described in [Sentaurus Topography 3D Computational Model on page 22](#).

There are three distinct stages when using an RFM model, as illustrated in [Figure 34 on page 503](#):

- *Model definition*

First, you define the model by specifying the fluxes used by the RFM model, the global and material-specific parameters, and the formulas for the etching or deposition rate. See [Model Definition on page 503](#).

- *Machine configuration*

Second, you define a machine that uses the newly defined model. See [Machine Configuration on page 507](#).

- *Machine use*

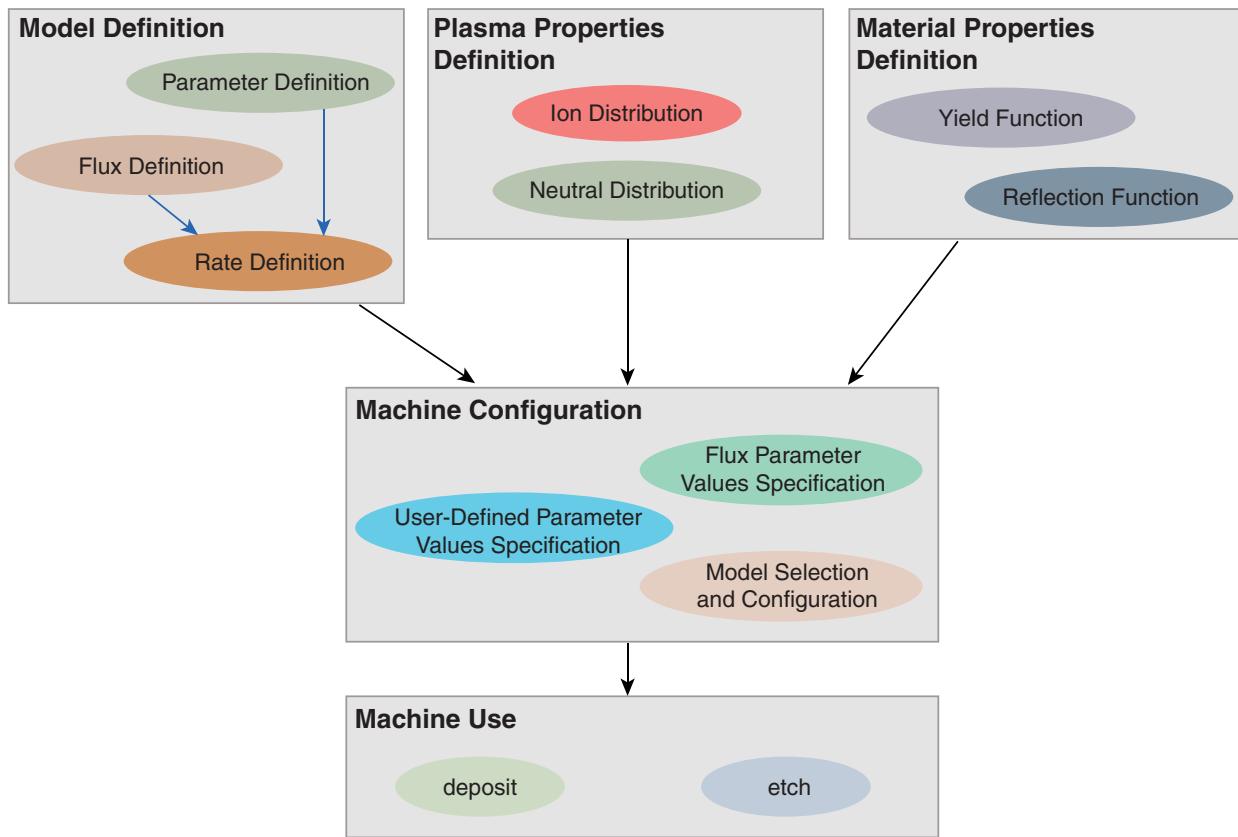
Finally, you use the machine with an `etch` or a `deposit` command. This is performed in exactly the same way as with machines that use built-in or PMI models (see [deposit on page 322](#) and [etch on page 339](#)).

Chapter 8: Rate Formula Module

Model Definition

Refer to the examples of how to create models using the RFM in the Sentaurus Topography 3D module of the TCAD Sentaurus Tutorial (see [TCAD Sentaurus Tutorial: Simulation Projects on page 21](#)).

Figure 34 RFM modeling flow



Model Definition

You define the model by specifying the fluxes used by the RFM model, the global and material-specific parameters, and the formulas for the etching or deposition rate. This stage is analogous to the creation of a PMI model where the new model is defined in the form of C++ code. Since there is only a small number of commands for defining a new model, it is much easier to learn how to create an RFM model than a PMI model.

You can add an arbitrary number of ion and neutral fluxes to an RFM model. The ion fluxes can be configured individually to take the physical effects of reflection, sputtering, and reemission into account. [Modeling Fluxes and Related Physical Effects on page 44](#) describes these physical effects and how they are calculated.

Chapter 8: Rate Formula Module

Model Definition

In general, deposition and etching models have parameters that are used to adjust the deposition and etching rates to specific process conditions. An arbitrary number of material-specific and global parameters can be added to an RFM model. During the machine configuration, the values of these parameters can be set in the command file in the same way as for built-in and PMI models.

At each time step, Sentaurus Topography 3D calculates the rate of each surface element. For an RFM model, the rate is calculated from a user-defined mathematical expression. This expression can contain the values of the previously defined fluxes, the values of user-defined parameters, mathematical functions, and arithmetic operators.

Defining a Model

The definition of a model consists of five steps, two of which are optional:

1. You start with the `define_model` command, which provides the name and the type of the model (`deposit`, `etch`, or `etchdepo`), as well as a short description of the model.
2. (Optional) For models that depend on neutral and ion fluxes, you add these with the `add_neutral_flux` and `add_ion_flux` commands, respectively. See [Defining Fluxes](#).
3. (Optional) You add the floating-point and integer parameters used in the rate formula with the `add_float_parameter` and `add_int_parameter` commands, respectively. See [Specifying User-Defined Parameters on page 505](#).
4. You must specify a rate formula as a mathematical expression with the `add_formula` command. See [Defining a Rate Formula on page 506](#).
5. The model definition is marked as complete by specifying the `finalize_model` command.

Defining Fluxes

This step introduces the ion and neutral fluxes involved in the model and states which physical effects are taken into account for ion fluxes. For neutral fluxes, reemission is always activated, as discussed in [Modeling Fluxes and Related Physical Effects on page 44](#).

Neutral and ion fluxes are introduced by the `add_neutral_flux` and `add_ion_flux` commands, respectively, and the definition of each flux requires a unique identifier for the flux using the parameter `name` as well as the model to which the flux refers.

Accordingly, a neutral flux called `N` to be used in the model `my_model` is introduced with the statement:

```
add_neutral_flux model=my_model name=N
```

Chapter 8: Rate Formula Module

Model Definition

Typically, you set the value of the sticking coefficient of a neutral flux using the `add_flux_properties` command in the machine configuration. The value also can be specified with the `sticking` parameter in the `add_neutral_flux` command.

It is important to note that, if the sticking coefficient of a neutral flux is set in the `add_neutral_flux` command, it is material independent and cannot be changed later. This is useful, for example, if you want to completely deactivate reemission for a neutral flux when defining the model, which can be achieved with a command such as:

```
add_neutral_flux model=my_model name=N sticking=1
```

Note:

The definition of a neutral flux does not include the specification of its angular distribution. The distribution of a neutral flux is isotropic by default, and you can set it by using the `define_nad` command (see [define_nad on page 252](#)).

The command `add_ion_flux` allows you to select the physical effects to take into account. For example, the following command defines an energy-independent ion flux called `I` for the model `my_model` with sputtering and reflection activated, and sputter deposition deactivated:

```
add_ion_flux model=my_model name=I energy=independent \
    reflection=true sputter_deposition=false sputtering=true
```

Note:

The definition of an ion flux does not include the specification of its angular distribution, which is provided using the `define_iad` command.

Specifying User-Defined Parameters

In this step, you specify user-defined parameters. Here, their attributes (for example, name, valid range, and default value) are set without assigning the actual values of the parameters.

User-defined parameters can be floating-point or integer values, and they are introduced using the `add_float_parameter` and `add_int_parameter` commands, respectively.

In addition, user-defined parameters can be either material dependent or global (material independent). If a parameter is material dependent, it can be assigned a different value for each material. Since deposition uses only one material, material-dependent parameters can be used only in the rate formulas of the `etch` and `etchdepo` models. A default value must be specified for material-dependent parameters. This value is used for materials that are present in the structure but have not been configured in the machine using the `add_material` command.

Chapter 8: Rate Formula Module

Model Definition

For example, a material-dependent floating-point parameter named `p`, with default value equal to 5, valid range from 3 to 7, and representing a velocity, can be introduced using the command:

```
add_float_parameter name=p model=my_model min=3 max=7 default=5 \
    quantity=velocity scope=material_dependent
```

As previously mentioned, the actual values of the parameters are not set here, but when you configure a machine using the model to which the parameters belong. The values of the global parameters are set with the `define_etch_machine` or `define_deposit_machine` command; whereas, the values of the material-dependent parameters are set with the `add_material` command.

Defining a Rate Formula

The rate formula is defined with the `expression` parameter of the `add_formula` command.

Such an expression might involve any user-defined parameter, and the direct and indirect fluxes of any neutral and ion fluxes, as well as built-in functions to access properties of the exposed surface and mathematical functions.

User-defined parameters can be used in the expression by using their names followed by parentheses `()`.

The values of the direct and indirect fluxes can be obtained in the rate formula by the RFM built-in functions listed in [Data Available for Rate Calculation on page 512](#).

For example, if a neutral flux `N`, an ion flux `I`, and the parameters `p1`, `p2`, `p3`, and `p4` have been introduced in the model `my_model`, the following statement is a rate formula definition:

```
add_formula model=my_model expression="p1() * total_flux(N) + \
    p2() * cos(theta()) - p3() * direct_flux(I) - p4() * sputtered_flux(I)"
```

Note:

In the rate formula, positive rates correspond to deposition, and negative rates correspond to etching.

Sentaurus Topography 3D also supports material-dependent rate formulas. This means that the mathematical expression used to compute the rate can be different for different materials. A rate formula for a specific material can be specified with the `material` parameter.

For example, the following commands define one rate formula specific for silicon, one specific for photoresist, and one used for all the other materials:

```
add_formula model=my_model material=Silicon \
    expression="-p3() * total_flux(I)"
```

Chapter 8: Rate Formula Module

Machine Configuration

```
add_formula model=my_model material=Photoresist \
    expression="-p2() * cos(theta())"

add_formula model=my_model expression="-p1() * total_flux(N)"
```

Machine Configuration

After the definition of a new model, you set up a machine that uses the newly defined model. This is very similar to setting up a machine using a built-in or a PMI model. However, the specification of model parameters related to ion or neutral fluxes must be performed with separate commands because an RFM model can use an arbitrary number of fluxes, and the reflection and yield functions must be configured separately.

Configuring a Machine

The configuration of a machine based on an RFM model consists of the following steps:

1. [Defining a Machine and Specifying User-Defined Global Parameters](#)
2. [Specifying Flux Parameter Values on page 508](#)
3. [Specifying User-Defined Material-Dependent Parameter Values on page 509](#)

Defining a Machine and Specifying User-Defined Global Parameters

Use the `define_etch_machine` command to configure a machine for models defined with `type=etch` or `type=etchdepo`. Use the `define_deposit_machine` command to configure a machine for models defined with `type=deposit`.

The configuration of a machine with an RFM model requires the following additional information:

- The name of the RFM model must be specified with the `model` parameter.
- If the model is a `deposit` model, then the material to deposit must be specified with the `material` parameter.
- If the model is an `etchdepo` model, then the material to deposit must be specified with the `deposit_material` parameter.
- If neutral fluxes are involved in the model, then the name of a collection of neutral distributions, containing the distributions of all the neutral fluxes of the model, can be specified with the `nad` parameter. Otherwise, all neutral fluxes will be assumed to have an isotropic distribution.

Chapter 8: Rate Formula Module

Machine Configuration

- If ion fluxes are involved in the model, then the name of a collection of ion distributions containing the distributions of all the ion fluxes of the model must be specified with the `iad` parameter.
- If sputtering is activated for at least one ion flux of the model, then the name of a collection of yield functions must be specified with the `yield` parameter. This collection must contain a yield function for all combinations of ion fluxes for which sputtering is activated and all materials that appear on the surface during at least one of the time steps.
- If reflection is activated for at least one ion flux of the model, then the name of a collection of reflection functions must be specified with the `reflection` parameter. This collection must contain a reflection function for all combinations of ion fluxes for which reflection is activated and all materials that appear on the surface during at least one of the time steps.
- The values of all the global parameters that are not optional, that is, those defined with `optional=false`.

Note:

The specified collection of reflection or yield functions does not have to contain functions for materials that are present in the structure but do not appear at the surface in any of the time steps.

For example, if `my_model` is an etch RFM model with ion fluxes having sputtering activated and it involves a mandatory global parameter `p`, a machine using such a model can be defined with the command:

```
define_etch_machine model=my_model iad=my_iad yield=my_yield p=0.5
```

If the parameter `p` is defined with `optional=true`, this parameter can be omitted from the `define_etch_machine` command.

Specifying Flux Parameter Values

The values of the parameters required to compute the indirect fluxes involved in the model are set using the `add_flux_properties` command.

For neutral fluxes, the sticking coefficient can be set with this command. For ion fluxes with sputter deposition activated, the sticking coefficient, the sputter type, and the sputter exponent can be set with this command.

For `etch` and `etchdepo` models, the flux parameters must be material dependent; whereas, for `deposit` models, they must be material independent.

Note:

The set of flux parameters is not defined by users, but it is determined by the fluxes involved in the model and by the physical effects activated for them. For example, the following command sets the sticking coefficient for the neutral flux `N` of a `deposit` model to 0.4:

```
add_flux_properties flux=N sticking=0.4
```

Specifying User-Defined Material-Dependent Parameter Values

The values of global parameters are set by the `define_etch_machine` or `define_deposit_machine` command; whereas, the values of material-dependent parameters are set by the `add_material` command. Each `add_material` command sets the values of the material-dependent parameters for a certain material.

Only the values of the material-dependent parameters defined with `optional=false` must be set in the `add_material` command. If a parameter is defined with `optional=true`, the specification of its value is optional in any `add_material` command, and its default value is used for all materials for which it has been omitted.

Therefore, the default value of a material-dependent parameter is used in two different cases:

- When no `add_material` command is issued for a certain material
- When a material-dependent parameter is defined as optional (`optional=true`) and it is omitted in an `add_material` command

Note:

It is not necessary to specify material properties with the command `add_material` for all materials, even if none of the material-dependent parameters is optional. All materials for which no properties have been specified explicitly with the command `add_material` are treated equally by using the default values of the material-dependent parameters.

For example, assume that two material-dependent parameters, `p1` and `p2`, are defined as follows:

```
add_float_parameter name=p1 scope=material_dependent default=0.1 \
    optional=true ...

add_float_parameter name=p2 scope=material_dependent default=0.5 \
    optional=false ...
```

Chapter 8: Rate Formula Module

RFM Commands

In addition, suppose that `p1` and `p2` are the only material-dependent parameters involved in the model for which the machine is being configured. Accordingly, the following statements are valid:

```
add_material material=Silicon p1=0.3 p2=0.9
```

```
add_material material=Oxide p2=0.7
```

The first statement sets the values of all the material-dependent parameters. In the second statement, the specification of the value of the optional parameter is omitted and its default value 0.1 is used. For any other material, the default values 0.1 and 0.5 for the parameters `p1` and `p2`, respectively, will be used.

RFM Commands

The commands for the creation and use of RFM models are provided in [Chapter 6 on page 129](#). The commands for creating RFM models are listed approximately in the order in which they are used:

- `define_model`
- `add_neutral_flux`
- `add_ion_flux`
- `add_float_parameter`
- `add_int_parameter`
- `add_formula`
- `finalize_model`

The commands for using RFM models are:

- `define_deposit_machine`
- `define_etch_machine`
- `add_flux_properties`

Rate Calculation

The rate formulas defined with the `add_formula` command are used to calculate the deposition or etching rate for each surface element (see [add_formula on page 149](#)):

- For deposition, the rate must be greater than or equal to zero.
 - For etching, the rate must be less than or equal to zero.
 - For simultaneous etching and deposition, there is no restriction on the values of the rate; deposition is represented by positive values and etching by negative values.
-

Syntax for Formulas and Subexpressions

The syntax for formulas and subexpressions consists of the following components:

- Numeric constants
- Arithmetic operators: +, -, *, /
- Mathematical constants: `e()`, `pi()`
- Mathematical functions: `abs(x)`, `sin(x)`, `cos(x)`, `tan(x)`, `asin(x)`, `acos(x)`, `atan(x)`, `atan2(x, y)`, `sinh(x)`, `cosh(x)`, `tanh(x)`, `exp(x)`, `log(x)`, `log10(x)`, `pow(x, y)`, `sqrt(x)`, `ceil(x)`, `floor(x)`
- Conditional and relational functions: `checked_div(x, y)`, `checked_div(x, y, z)`, `high_pass(x, y, z)`, `higher_pass(x, y, z)`, `low_pass(x, y, z)`, `lower_pass(x, y, z)`, `max(x, y)`, `min(x, y)`
- Parentheses
- Data access functions described in [Data Available for Rate Calculation on page 512](#)
- Subexpressions previously defined with the `subexpression` parameter of the `add_formula` command

The arguments `x`, `y`, and `z` to these functions are themselves expressions.

Conditional and Relational Functions

The following functions are available to implement conditions and control flow:

- `checked_div(<numerator>, <denominator>)`
Returns `numerator / denominator` if `denominator != 0`.
Otherwise, returns `numerator`, that is, it is equivalent to
`checked_div(<numerator>, <denominator>, <numerator>).`

Chapter 8: Rate Formula Module

Rate Calculation

- `checked_div(<numerator>, <denominator>, <replacement>)`
Returns numerator / denominator if denominator != 0.
Otherwise, returns replacement.
- `high_pass(<expression>, <threshold>, <attenuation>)`
Returns expression if expression >= threshold. Otherwise, returns attenuation.
- `higher_pass(<expression>, <threshold>, <attenuation>)`
Returns expression if expression > threshold. Otherwise, returns attenuation.
- `low_pass(<expression>, <threshold>, <attenuation>)`
Returns expression if expression <= threshold. Otherwise, returns attenuation.
- `lower_pass(<expression>, <threshold>, <attenuation>)`
Returns expression if expression < threshold. Otherwise, returns attenuation.
- `min(<left>, <right>)`
Returns left if left <= right. Otherwise, returns right.
- `max(<left>, <right>)`
Returns left if left >= right. Otherwise, returns right.

Data Available for Rate Calculation

The following functions are available to access data related to a surface element, which is available for use in the formula for a deposition or an etching rate:

- `defined_yield(<flux>)`: Returns the value of the yield function for the specified ion flux and the material of the current surface element, and the angle between the vertical and the surface normal of the current element. This function is available for all ion fluxes and the `default_species`.

Note:

Models using this function in any of their rate formulas are not supported by a machine having `rotation=continuous`. In models to be used with machines having `rotation=continuous`, the sputtered flux can be computed using the function `sputtered_flux(<flux>)`.

- `defined_yield(<flux>, <material>)`: Returns the value of the yield function for the specified ion flux and the specified material, and the angle between the vertical and the surface normal of the current element. This function is available for all ion fluxes and the `default_species`.

Chapter 8: Rate Formula Module

Rate Calculation

Note:

Models using this function in any of their rate formulas are not supported by a machine having `rotation=continuous`. In models to be used with machines having `rotation=continuous`, the sputtered flux can be computed using the function `sputtered_flux(<flux>)`.

- `direct_flux(<flux>)`: Returns the value of the direct flux arriving at the surface element for the specified flux (all neutral and ion fluxes).
- `directional_value(<value_100>, <value_110>, <value_111>)`: The first argument specifies the value for the `<100>` direction; the second argument, the `<110>` direction; and the third argument, the `<111>` direction. Depending on the direction of the surface normal of an element, an interpolated value is calculated and returned. This function can be used to implement orientation-dependent etching or deposition models.
- `is_void()`: Returns 1 if the surface element belongs to a void boundary; 0 otherwise.
- `pad_pressure()`: Returns the pressure produced by the pad at the surface element in Pa.

Note:

The following limitations apply:

- Only models of `type` equal to `etch` or `etchdepo` support this function in their rate formula expressions.
- Models using this function cannot use these RFM functions:
`direct_flux(<flux>)`,
`sputter_depo_flux(<flux>)`,
`sputtered_flux(<flux>)`,
`total_flux(<flux>)`.
- Periodic boundary conditions are used to evaluate the RFM function `pad_pressure()` independently of what has been specified with the command `define_boundary_conditions`.
- `<parameter>()`: Returns the value of the specified global or material-dependent user-defined parameter.
- `sputter_depo_flux(<flux>)`: Returns the value of the flux of the redeposited sputtered material associated with the specified ion flux arriving at the surface element (ion flux with sputter deposition activated).
- `sputtered_flux(<flux>)`: Returns the value of the sputter flux emitted from the surface element for the specified ion flux (ion flux with sputtering activated).
- `sticking(<flux>)`: Returns the value of the sticking parameter that was specified with the `add_flux_properties` command for the given flux (neutral flux or ion flux with sputter deposition activated).
- `<subexpression>()`: Returns the value of the specified subexpression. If the current expression is global, only other global subexpressions can be accessed. If the current

Chapter 8: Rate Formula Module

Rate Calculation

expression is material dependent, other subexpressions for the same material or global subexpressions can be accessed.

- `theta()`: Returns the value of the angle between the surface normal and the vertical. The value is in the range $[0, \pi]$.

Note:

Models using this function in any of their rate formulas are not supported by a machine having `rotation=continuous`.

- `total_flux(<flux>)`: Returns the value of the total flux arriving at the surface element for the specified flux (neutral flux or ion flux with reflection activated).
- The total integrated ion flux is defined as $\Gamma_{\text{total}}^{\text{ion}} = \Gamma_{\text{direct}}^{\text{ion}} - \Gamma_{\text{direct reflected}}^{\text{ion}} + \Gamma_{\text{reflection}}^{\text{ion}}$ where:
 - $\Gamma_{\text{direct reflected}}^{\text{ion}}$ is the portion of the direct flux that has been reflected away from a surface element.
 - $\Gamma_{\text{reflection}}^{\text{ion}}$ is the sum of all fluxes incoming at a surface element that were reflected from other surface elements.
- `visible()`: Returns if the surface element is visible from vertically above. Returns 1 if the surface element is visible or 0 otherwise.

Models using the `visible()` function in any of their rate formulas are not supported by a machine having `rotation=continuous`.

Note:

A syntax error occurs if functions taking a flux name as argument are used with a flux that is of an incompatible type.

The identifier `default_species` can be used as a placeholder parameter for the `defined_yield` functions in models that do not specify any ion fluxes to access yield functions. The yield functions must be defined with `default_species` as the species.

The integrated direct reflected flux and the reflection flux are not accessible as RFM functions. However, they are available as datasets created when using the `plot_interval` parameter of the `deposit` and `etch` commands.

Examples

The `direct_flux()` function is available to a model with only a neutral flux `n`. However, the `sputtered_flux()` function is not available:

```
define_model name=N type=deposit description=""  
add_neutral_flux model=N name=n  
# Next command causes a syntax error because n is not a valid argument
```

Chapter 8: Rate Formula Module

Example: Reimplementing and Using ionmill Etching Model

```
# for sputtered_flux().
add_formula model=N expression={direct_flux(n) + sputtered_flux(n)}

finalize_model model=N
```

Similarly, if ion fluxes are defined for a model but the necessary effects are deactivated, a syntax error occurs:

```
define_model name=I type=deposit description=""

add_ion_flux model=I name=i energy=independent reflection=false \
    sputtering=true sputter_deposition=false

# Next command causes a syntax error because i is not a valid argument
# for sputter_depo_flux().
add_formula model=I expression={direct_flux(i) + sputter_depo_flux(i)}

finalize_model model=I
```

Example: Reimplementing and Using ionmill Etching Model

The following example demonstrates how the `ionmill` etching model can be reimplemented and used as an RFM model:

```
define_model name=mck_ionmill type=etch \
    description="user-defined ionmill etching"

add_float_parameter model=mck_ionmill name=rate min=0 default=0 \
    scope=material_dependent quantity=velocity

add_float_parameter model=mck_ionmill name=s1 default=1 \
    scope=material_dependent quantity=dimensionless

add_float_parameter model=mck_ionmill name=s2 default=0 \
    scope=material_dependent quantity=dimensionless

add_formula model=mck_ionmill \
    expression={ -rate() * visible() * (s1() * cos(theta()) \
        + s2() * pow(cos(theta()), 2) \
        + (1 - s1() - s2()) * pow(cos(theta()), 4)) }

finalize_model model=mck_ionmill

define_etch_machine model=mck_ionmill

add_material material=Silicon rate=0.9 s1=5.5 s2=-6

add_material material=Oxide rate=0.7 s1=5 s2=-7

etch ...
```

References

- [1] T. K. Chini *et al.*, “The angular dependence of sputtering yields of Ge and Ag,” *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, vol. 72, no. 3–4, pp. 355–358, 1992.

9

Working With Reaction Models

This chapter explains how to use reaction models based on the particle Monte Carlo (PMC) method in Sentaurus Topography 3D. It also discusses how to integrate PMC-based simulations with other Sentaurus Topography 3D functionality.

Setting Up a Simulation Using a Reaction Model

This section provides a simple example to start using reaction models. A model for a simple process will be set up and a simulation using that model will be run.

In the process you want to model, fluorine gas is fed to the reactor, where a silicon wafer is being processed. When fluorine reaches silicon, they react with a given probability, and silicon is etched.

The first step is to define the model. A reaction model consists of:

- The source species, that is, the species coming from the reactor in a gaseous state
- The reactions relevant for the modeled process

A reaction model for the above-described process can be set up as follows:

```
# Begin the definition of a model named 'm1'.
define_model name=m1 description="Simple reaction model"

# Add a species named "F" coming from the reactor to model 'm1'.
add_source_species model=m1 name=F

# Add a reaction named 'etch_reaction' to model 'm1'.
# According to this reaction, when species 'F' and 'Silicon' react, a
# silicon atom is removed and a 'SiF' molecule goes into the reactor.
# The net effect is to remove a silicon atom from the structure.
add_reaction model=m1 name=etch_reaction \
    expression="F<g> + Silicon<s> = SiF<g>"

# End the definition of model 'm1'.
finalize_model model=m1
```

Chapter 9: Working With Reaction Models

Setting Up a Simulation Using a Reaction Model

As for RFM models, the definition of a new reaction model starts with the `define_model` command (see [define_model on page 251](#)) and ends with the `finalize_model` command (see [finalize_model on page 438](#)).

Since fluorine gas molecules are electrically neutral, their angular distribution is assumed to be isotropic. An angular distribution described by [Equation 6 on page 49](#) with m equal to 1 describes an isotropic distribution. Therefore, an isotropic angular distribution can be set up for species F with the following command (see [define_species_distribution on page 290](#)):

```
define_species_distribution name=my_distribution species=F \
    exponent=1 flux=1e-3
```

After the model and the distribution of its source species have been specified, a machine can be defined using the `define_etch_machine` command (see [Machines on page 28](#) and [define_etch_machine on page 209](#)):

```
define_etch_machine model=m1 name=m1_machine \
    species_distribution=my_distribution
```

In Sentaurus Topography 3D, machines are used to select the model to use for the simulation and to assign the parameter values required by the selected model. Reaction models require probabilities to be set for each reaction. The reaction probabilities specify the probability that a reaction will occur when its reactants become available.

The following `add_reaction_properties` command (see [add_reaction_properties on page 168](#)) sets the reaction probability to 0.7 for the reaction named `etch_reaction` of the model used by the machine named `m1_machine`:

```
add_reaction_properties machine=m1_machine reaction=etch_reaction p=0.7
```

The machine is now fully set up and can be used to process a structure. As explained in [Simulating Process Steps on 2D and 3D Structures on page 31](#), the initial structure can be either loaded from a TDR boundary file or defined using geometric etching and deposition steps. For simplicity, assume that the initial structure is stored in a TDR file called `init.tdr`. Then, the following two commands define the initial structure and start a simulation using the machine defined above, respectively:

```
define_structure file=init.tdr
etch spacing=0.1 time=0.5 method=pmc machine=m1_machine
```

The results of a simulation that used the PMC method can be saved in TDR file format with the command (see [save on page 461](#)):

```
save type=gc
```

Chapter 9: Working With Reaction Models

Integration With Other Sentaurus Topography 3D Functionality

Integration With Other Sentaurus Topography 3D Functionality

The `etch` command runs a simulation using the selected machine. The name of the resulting structure is specified by the parameter `structure` of the `etch` command.

The data structure used to store internally the result depends on the simulation method used (see [Sentaurus Topography 3D Computational Model on page 22](#) and parameter `method` of `etch` on page 339):

- When `method=levelset`, when etching with a mask or etching a geometric shape, the simulation result is stored internally in a boundary data structure (see [Boundary Types on page 30](#)).
- When `method=pmc`, the simulation result is stored internally in a volumetric data structure.

The commands in [Table 81](#) require input structures to be stored in a boundary data structure.

Table 81 Commands requiring input structures to be stored internally in a boundary data structure

Command	Comment
<code>deposit</code>	
<code>etch</code>	When using a level set-based model or when etching a geometric shape or a mask.
<code>extend_structure</code>	
<code>extract</code>	See Table 62 on page 403 .
<code>fill</code>	
<code>filter_structure</code>	See Table 65 on page 423 .
<code>litho</code>	
<code>pattern</code>	
<code>save</code>	When <code>type=closed_surface</code> , or <code>type=exposed_surface</code> , or <code>type=final_structure</code> . Note: When <code>type=gc</code> , <code>type=pmc</code> , <code>type=transfer</code> , <code>type=vbe</code> , or <code>type=volume_fractions</code> is specified, the <code>save</code> command requires the input structure to be stored in a <i>volumetric</i> data structure.

Table 81 Commands requiring input structures to be stored internally in a boundary data structure (Continued)

Command	Comment
truncate	

Converting a Result Structure to TDR Boundary File Format

Whenever any of the commands in [Table 81 on page 519](#) must be executed on the result of a simulation that used the PMC method, an explicit conversion is needed.

You can use the `filter_structure` command to convert the result of a PMC-based simulation to a structure stored in TDR boundary file format (see [filter_structure on page 420](#)), as demonstrated in the following example. In particular, the `filter_structure` command is used twice:

- Before starting the simulation using the PMC method, it creates a copy of the initial structure (`filter_structure type=copy`).
- When the simulation is completed, it runs Boolean operations (`filter_structure type=boolean`) and produces a structure stored in TDR boundary file format.

Example

```
# The 'define_structure' command creates a structure that is internally
# stored in TDR boundary file format. The newly created structure is
# named 'default_structure'.
define_structure material=Silicon point_min={0 0 0} point_max={1 1 1}

# The 'fill' command operates on a structure stored in TDR boundary file
# format. The result of the command 'fill' overwrites the structure named
# 'default_structure' and is still stored internally in TDR boundary file
# format.
fill material=Oxide thickness=0.5

# A shape named 's' is defined to create an opening in the oxide layer
# added with the command 'fill'.
define_shape type=cube name=s point_min={0.2 0.2 1} point_max={0.8 0.8 2}

# The shape 's' is etched away from the structure named
# 'default_structure'. The result of the command 'etch' overwrites the
# structure 'default_structure' and is still stored internally
# in TDR boundary file format.
etch shape=s

# A copy of structure 'default_structure' is created. The copy is named
# 'initial_structure' and is stored in TDR boundary file format as well.
```

Chapter 9: Working With Reaction Models

Integration With Other Sentaurus Topography 3D Functionality

```
filter_structure type=copy name=initial_structure

# A reaction model named 'reaction_model' is defined.
define_model name=reaction_model description=""
...
finalize_model model=reaction_model
...

# A machine using model 'reaction_model' is defined.
define_etch_machine model=reaction_model ...
...
# The structure named 'default_structure' is etched using the reaction
# model named 'reaction_model'. The result structure overwrites the
# structure named 'default_structure'.
# Since the PMC method was used, the result of this
# command is stored internally with a volumetric data structure.
etch spacing=0.1 time=1 method=pmc

# The structure 'default_structure' stored in a volumetric data
# structure is saved in the appropriate TDR file format (see Table 81).
save type=gc

# A new structure named 'boundary_structure' is created using a Boolean
# operation. The operands of the Boolean operation are specified with the
# parameters 'body' and 'structure'. It is worth noting that the
# structure given with the parameter 'body' is stored internally in TDR
# boundary file format; whereas, the structure given with parameter
# 'structure' is stored internally in a volumetric data structure in
# TDR file format.
filter_structure type=boolean body=initial_structure \
    structure=default_structure name=boundary_structure

# Since the structure named 'boundary_structure' is stored in TDR
# boundary file format, the 'extract' command can be used to return
# information on it (see Table 81).
extract type=1d_cut structure=boundary_structure ...
...
# Since the structure named 'boundary_structure' is stored in TDR
# boundary file format, it can be saved with type=final_structure
# (see Table 81).
save structure=boundary_structure type=final_structure
```

Boolean operations are not run automatically after each PMC-based simulation to convert the resulting structure into a boundary data structure because they are computationally expensive and are not always needed. Boolean operations are necessary only when one of the commands in [Table 81 on page 519](#) must be run, but they are not required to save the resulting structure in a TDR file format and to visualize it. In addition, PMC-based simulations can be run on the result of a previous PMC-based simulation without executing any Boolean operations.

10

Physical Model Interface for Etching and Deposition

This chapter describes the physical model interface for etching and deposition.

Overview

The physical model interface (PMI) in Sentaurus Topography 3D gives you the possibility to extend the modeling capabilities of Sentaurus Topography 3D by implementing a new model in the form of a C++ class.

Sentaurus Topography 3D uses the level-set method to move the exposed surface during a time step. However, direct use of the level-set data structures requires detailed knowledge of the particular implementation and, therefore, direct use is complicated and error prone. An explicit surface representation is used for the PMI in Sentaurus Topography 3D.

The PMI in Sentaurus Topography 3D supports three different modes: deposition, etching, and simultaneous etching and deposition.

The following sections describe the command file interface for using a PMI-based model, the C++ interface for creating a new model, and the input and output parameters.

Command File Interface

The `add_material`, `define_deposit_machine`, and `define_etch_machine` commands can handle PMI-based models.

The name with which a PMI-based model is specified, with the `model` parameter of the `define_deposit_machine` and `define_etch_machine` commands, is defined in the source code of the model and can have an arbitrary value. The only restriction is that it must not be identical to the name of a built-in model.

Note:

Model names beginning with the prefix `pmi` are guaranteed not to conflict with built-in models.

In the command file, the parameters of a PMI-based model are specified in the same way as for built-in models. Therefore, for deposition models, all model parameters are specified using the `define_deposit_machine` command. For etching models, the material-independent parameters of the model are specified with the `define_etch_machine` and the material-dependent parameters with the `add_material` command. For simultaneous etching and deposition, the deposition-specific parameters are specified with the `define_etch_machine` command.

A PMI-based model can have an arbitrary number of model-specific command file parameters. These parameters can have arbitrary names. The only restriction on the parameter names is that they must not be identical to the names of built-in parameters of a PMI-based model.

Table 82 lists the built-in parameters of the PMI-related commands. The command file parameters have four possible types: Boolean, floating point, integer, and string.

Table 82 Built-in parameters of commands for using PMI-based models

Command	Built-in parameters
<code>add_material</code>	<code>machine, material</code>
<code>define_deposit_machine</code>	<code>material, model, name, rotation, tilt</code>
<code>define_etch_machine</code>	<code>deposit_material, model, name, rotation, tilt</code>

A default value must be specified for each user-defined parameter, and you can specify whether a user-defined parameter is optional or mandatory. When specifying the parameters for a material with the `add_material` command, a parameter can be omitted if it has been declared as optional. In this case, the default value will be used.

Depending on the type of a parameter, the default value can be queried directly with the `bool_parameter_default()`, `float_parameter_default()`, `int_parameter_default()`, or `string_parameter_default()` function.

C++ Interface

At the C++ level, the interface consists of several abstract base classes:

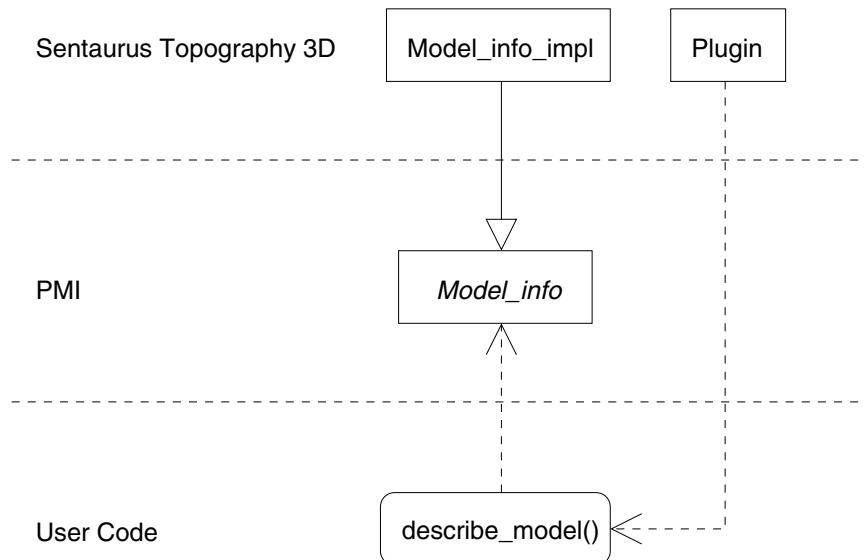
- `Model_info` defines the model name, the model-specific command file parameters, and the plot datasets generated by the model.
- `Model_rate` calculates the surface rate and the plot datasets for each time step.

- `Global_data` provides access to the values of the command file parameters and other data that does not change during the time steps.
- `Time_step_data` provides access to surface-related data and data that changes between time steps.

The developer of a PMI-based model must implement a class that is derived from `Model_rate` and must implement three free functions.

The function `describe_model()`, which must be implemented by users, is called during the initialization of a PMI-based model. It defines the model name, the model parameters, and the plot datasets generated by the model. [Figure 35](#) shows a unified modeling language (UML) diagram of the classes used to initialize a PMI-based model.

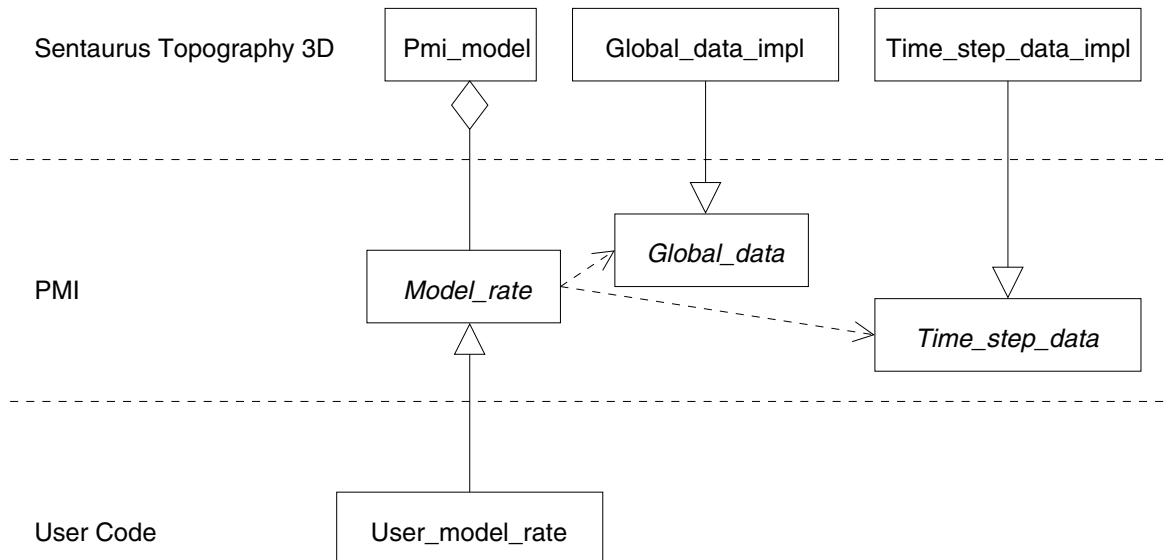
Figure 35 UML diagram for initializing a PMI model



The class derived from `Model_rate` is used in each time step to calculate the surface rate and to initialize the plot datasets. [Figure 36](#) shows a UML diagram of the PMI-related classes used for the surface rate calculation in each time step.

Detailed documentation of the abstract classes is available in the corresponding header files, which are located in the directory `${STROOT} /tcad/${STRELEASE}/lib/sptopo3d/include`.

Figure 36 PMI class hierarchy for calculating surface rates



Implementing a New Model

As previously mentioned, to implement a PMI-based model, it is necessary to implement the function `describe_model()`. The only argument of this function is a reference to an object derived from `Model_info`. This object is used to define the model name and whether the model is used for deposition, or etching, or simultaneous etching and deposition. Besides that, it is used to define all model-specific command file parameters (name, type, and default value if applicable), and the names and units of the datasets that are generated by the model.

A class derived from `Model_rate` is used to implement the surface rate calculation for each time step. An instance of the class derived from `Model_rate` is created when the `deposit` or `etch` command is executed. For each time step, the function `Model_rate::calculate_rate()` is called to calculate the surface rates and the plot datasets. The free functions `model_rate_factory()` and `model_rate_delete()` are used to create and delete instances of the class derived from `Model_rate`, respectively.

Information Available to a Model

All input data available to a PMI-based model is provided by the classes `Global_data` and `Time_step_data`.

The class `Global_data` makes the values of model-specific command file parameters available. In addition, it contains the names of all the materials of the input structure and

other data that does not change between time steps. For example, for etching models, a list of all materials for which parameters have been defined with the `add_material` command is available.

The class `Time_step_data` provides information about the current surface and its properties. It also contains other data that changes between time steps, for example, the size of the previous time step.

For each surface vertex, a list of indices in the list of structure materials at the vertex is available. If the vertex is located at a material interface, the list of structure material indices will contain more than one entry, and you are responsible for choosing which material will be used for the rate calculation.

Information about the surface material is available in all three modes. This information allows models to calculate material-specific rates also for the deposition mode.

The arguments of the free function `model_rate_factory()` are a reference to an object derived from `Global_data` and a reference to an object derived from `Time_step_data`. Therefore, all information is available to the constructor of the class derived from `Model_rate`, which is called in `model_rate_factory()`. The reference to the object derived from `Global_data` is guaranteed to be valid for the entire lifetime of the PMI model instance.

In `Model_rate::calculate_rate()`, a reference to an object derived from `Time_step_data` is available. In addition, you obtain a reference to an object derived from `Plot_data`, which provides access to the plot data arrays.

Additional Input Data

Additional input data that is too complicated to be read from the command file might be required by a model, for example, complex chemical reactions. This kind of data can be read and stored as member data by the constructor of the derived class for later use in `Model_rate::calculate_rate()`.

In some models, it might be necessary to store data from previous time steps. This additional data also can be stored as member data of the derived class object.

Data Types Used in Interface

To avoid linking problems, only C-style data types or types for which the corresponding header files are provided are used in the interface. While this makes handling the data more cumbersome, it helps to prevent problems that are difficult to debug.

Error Handling

The only safe way to signal to the calling code that an error has occurred in a constructor is to throw an exception. However, C++ does not allow exceptions to cross interfaces as they are used for the PMI.

An exception that is thrown in a PMI model and that is not caught and handled in the PMI model will terminate the simulation immediately. For a PMI model, this means that exceptions that were thrown in the constructor must be caught in the factory functions. If there is an error, the null pointer is returned and the simulation is stopped in a controlled way.

Similarly, exceptions must not leave `Model_rate::calculate_rate()`. Here, an error is signaled to the calling code by returning the value `false`. The simulation is then stopped in a controlled way.

The function `Global_data::message()` can be used to output messages to the standard output and the log file. For error messages, the verbosity level should be set to low; while for other messages, a verbosity level of medium or higher should be used.

Compiling the Source Code

The source code can be compiled and linked using the `cmi` tool, which calls the C++ compiler and linker with the correct command-line options for each supported platform.

The `cmi` tool is contained in each TCAD release and is mainly used for Sentaurus Device CMI-based and PMI-based models:

```
cmi -I${STROOT}/tcad/${STRELEASE}/lib/sptopo3d/include User_model_rate.cc
```

See *Compact Models User Guide*, Compilation of C++ (.C) Files.

Using Additional Source Files or Libraries

A PMI-based model is not limited to the functionality declared in the abstract base classes of the PMI or directly derived from it. You can use additional source files or libraries, which provide additional functionality, to build the shared library for a model. For example, a model might need to solve a linear equation system. In this case, it is possible to link with a library that provides this functionality.

Additional source files or libraries can be specified as command-line parameters for the `cmi` tool, which will combine them into one shared library.

Using the C++ Standard Library

When using the features of the C++ Standard Library in the implementation of a PMI-based model, depending on the platform, it might be necessary to link an additional library that provides this functionality. For example, on 64-bit Linux, it is necessary to link `libstdc++` by adding `-lstdc++` to the command-line parameters of the `cmi` tool.

To avoid problems at link-time or runtime, it is necessary to use *exactly* the same version of the C++ compiler for the compilation of the PMI code as used for the compilation of Sentaurus Topography 3D.

See *Compilers for CMI and PMI* in the TCAD Sentaurus release notes for information about the compiler versions used in Sentaurus Topography 3D.

Loading the Shared Library

During the initialization of the simulator, the current directory and the path defined with the environment variable `SPTOPO3D_PMI_PATH` are searched for shared libraries that conform to the CMI-naming or PMI-naming convention (platform-specific suffix, for example `.so.linux64`).

These files are loaded and searched for the three free functions: `describe_model()`, `model_rate_factory()`, and `model_rate_delete()`. If all three functions are found, the model is added to the list of available models and can then be used for an arbitrary number of deposition or etching steps. Otherwise, the shared library is closed again.

Note:

The shared library containing a PMI model is unloaded only when the simulator is stopped. This must be taken into account when using static variables.

Debugging

Debugging the PMI code is more difficult than debugging a standard program for several reasons:

- Only the PMI code created by users is available as source code, and debugging symbols are not available for Sentaurus Topography 3D.
- The library containing the PMI code is loaded dynamically whenever the conditions mentioned in the previous section are fulfilled.
- Typically, Sentaurus Topography 3D is started by a wrapper script that sets some environment variables. When using a debugger, it is necessary to set these environment variables manually and then to start the debugger with the executable of Sentaurus Topography 3D without the wrapper script.

Chapter 10: Physical Model Interface for Etching and Deposition

Input and Output Parameters

The details of debugging are highly platform dependent and tool dependent, especially when using shared libraries.

Input and Output Parameters

This section describes the input parameters available to a PMI-based model.

The following input data is available from objects derived from `Global_data`, `Time_step_data`, and `Plot_data` for model initialization and rate calculation:

- Model-specific command file parameters
- Default values of model-specific command file parameters
- Model-specific plot datasets
- Surface vertices
- Surface normal at vertices
- Curvature at vertices
- Area associated with surface vertices
- Material at vertices
- Mutual visibility between surface vertices
- Point classification (inside, on surface, outside)
- Size of the previous time step
- Machine materials
- Structure materials
- Conversion between material indices and material names
- Conversion between material names and material indices
- Direct flux at vertices for analytic flux distributions
- Visibility of vertices from source vertically above
- Neighbor elements information

The output parameter for the rate calculation is an array containing the rate at each surface vertex.

Chapter 10: Physical Model Interface for Etching and Deposition

Input and Output Parameters

Each surface vertex belongs to a discretization element of the surface. For each surface vertex, information about the neighbor elements is available.

The `Plot_data` parameter provides access to the plot datasets and to a query function that determines whether plotting is activated.

11

Known Issues and Limitations

This chapter provides information about known issues and limitations in the use of Sentaurus Topography 3D.

Description of Known Issues and Limitations

The following known issues and limitations apply to this version of Sentaurus Topography 3D:

- There is no support for energy-dependent flux integration in level set-based models.
- Reflective sputtering is not supported when processing 2D structures.
- Reflection is not supported for the rate formula module (RFM) when processing 2D structures.
- Boundary conditions specified by the command `define_boundary_conditions` have no effect on the indirect flux computations for 2D structures.
- Flux integration in 2D is performed with the radiosity method exclusively. There is no Monte Carlo engine in 2D.
- The radiosity method does not provide accurate results for machines with a tilt angle greater than approximately 70° for 3D structures. The Monte Carlo method is recommended for such machines.
- Built-in models do not support machines with `rotation=continuous`.
- The RFM functions `defined_yield(<flux>)`, `defined_yield(<flux, material>)`, `theta()`, and `visible()` are not supported by etching or deposition machines with `rotation=continuous`.
- When `rotation=continuous` is used, the actual equivalent error for any species always approaches 0.5 as the tilt angle approaches 90°.
- When defining a machine that uses a reaction model, `rotation=continuous` can be used only if no source species of the reaction model has an energy-dependent

Chapter 11: Known Issues and Limitations

Description of Known Issues and Limitations

distribution computed by HPEM, the Benoit-Cattin plasma sheath model, or the Edelberg–Aydil plasma sheath model, or imported from an EAD file.

- As described in [Discretization Size and Accuracy When Using the Level-Set Method on page 26](#), the use of the level-set method to discretize and move the surface of a structure introduces artifacts. When using models for simultaneous etching and deposition, these artifacts can cause the creation of thin layers of deposited material. Depending on the values of the model parameters, this might prevent etching and cause even more deposition.
- Only reflective boundary conditions are supported when using the built-in electrodeposition, spin_on, or wet deposition model.
- The `add_litho_command`, `define_litho_machine`, and `litho` commands cannot be used when you start Sentaurus Topography 3D with a value greater than 1 for the command-line option `--processes` (see [From the Command Line on page 17](#)).
- The `define_deposit_machine` and `define_etch_machine` commands do not support the parameters `tilt` and `rotation` when `model=crystal`.
- The `cfl`, `compute_first_plot`, `compute_last_plot`, `extraction_interval`, `extraction_times`, `extraction_times_unit`, `file`, `merge`, `plot_interval`, `plot_times`, `plot_times_unit`, `plot_type`, `stop_plane`, `stop_point`, and `update_scheme` parameters are not supported by the `deposit` command when using a machine having `model=crystal`.
- The `etch` command does not support the `abs_min_deposition_thickness`, `cfl`, `compute_first_plot`, `compute_last_plot`, `extraction_interval`, `extraction_times`, `extraction_times_unit`, `file`, `min_deposition_thickness`, `plot_interval`, `plot_times`, `plot_times_unit`, `plot_type`, `region_query_accuracy`, `stop_plane`, `stop_point`, and `update_scheme` parameters when using a machine having `model=crystal`.
- The `define_boundary_conditions` command has no effect on the built-in deposition and etching models named `crystal`. Boundary conditions are always reflective in simulations using deposition and etching machines having `model=crystal`.
- The `define_deposit_machine` command does not support the parameters `tilt` and `rotation` when `model=electrodeposition` or `model=spin_on`.
- The parameters `tilt` and `rotation` are not supported by the `define_etch_machine` command when `model=wet`.
- The parameters `tilt` and `rotation` of the `define_deposit_machine` and `define_etch_machine` commands do not support the radian measurement unit.
- The parameters `stop_plane`, `stop_point`, and `update_scheme` are not supported by the `deposit` command for machines having `model=spin_on`.

Chapter 11: Known Issues and Limitations

Description of Known Issues and Limitations

- For machines having `model=spin_on`, the `deposit` command does not support structures whose exposed surface cannot be described as a single-valued function of the x-and y-coordinates in three dimensions, or of the x-coordinate in two dimensions. However, topologically connected exposed surfaces are allowed to contain elements parallel to the vertical direction.
- The `etch` command does not support the `mask` parameter for 2D structures.
- The `pattern` command is not available for 2D structures.
- Sentaurus Lithography integration only works in three dimensions.
- The `slice_angle` parameter of the `define_structure` command does not support the radian measurement unit.
- The `filter_structure` command with `type=convert_to_pmc`, `type=smooth`, or `type=rediscretize_boundary` is not available for 2D structures.
- Boundary conditions set with the `define_boundary_conditions` command are ignored when using RFM models that use the RFM function `pad_pressure()` and periodic boundary conditions are always applied.
- Models using the RFM function `pad_pressure()` cannot use these RFM functions: `direct_flux(<flux>)`, `sputter_depo_flux(<flux>)`, `sputtered_flux(<flux>)`, `total_flux(<flux>)`.
- The parameter `structure` of the following commands cannot specify a PMC structure:
 - `deposit` (when using a level set-based model)
 - `etch` (when using a level set-based model)
 - `filter_structure` (when `type=convert_to_pmc`, `type=decimate`, `type=merge_regions`, `type=rediscretize_boundary`, `type=remove_disconnected_top_parts`, `type=remove_region`, `type=rename_region`, `type=replace_material`, or `type=smooth`)
 - `litho`
 - `save` (when `type=closed_surface`, or `type=exposed_surface`, or `type=final_structure`)

[Integration With Other Sentaurus Topography 3D Functionality on page 519](#) explains how to convert the structure in such a way that these commands can be used.

- When `type=gc`, `type=pmc`, `type=vbe`, or `type=volume_fractions`, the parameter structure of the `save` command must specify a structure obtained as the result of a PMC-based simulation.
- The `flat_orientation` and `vertical_orientation` parameters of the `etch` command are not supported when `method=pmc`.

Chapter 11: Known Issues and Limitations

Description of Known Issues and Limitations

- Boundary conditions of type `none` are not supported when processing a structure with the PMC method.
- Nonuniform spacing of the simulation grid is not supported when using the `etch` command with `method=pmc`.
- When using a built-in model, the `add_interface_layer` command is not supported for the sticking and reflection parameters.
- When using the `etch method=pmc` command to process a structure obtained as the result of a previous PMC-based simulation, the spacing specified with the `spacing` parameter is ignored.
- When running a simulation with a machine using a reaction model that allows deposition, the vertical extent of the computational domain must be large enough to contain the deposited material.
- The parameter structure of the `extract` command can specify a PMC structure only in any of the following cases:
 - When `type=bounding_box`, `type=dimension`, or `type=interface` is used to extract the interface between two materials (that is, when also parameters `material1` and `material2` are specified)
 - When `type=material_names` and `exposed_only=false`
 - When `type=probe` and `property=material`
 - When `type=probe` and `property=length`
- The parameter `stop_material` of the `etch` command can be used only when `method=pmc`.
- When you start Sentaurus Topography 3D with the `--processes` command-line option with a value greater than 1 (see [Table 1 on page 17](#)):
 - The parameter structure of the following commands must not denote a structure obtained from the command `etch method=pmc averaging_runs=1`:

```
etch method=levelset
etch shape=<c>
etch mask=<c>
extend_structure, extract, fill
filter_structure (except when type=copy)
litho, pattern
save (except when type=gc)
truncate
```
 - The parameters `extraction`, `plot_interval`, `plot_times`, `stop_material`, `stop_plane`, and `stop_point` of the command `etch method=pmc averaging_runs=1` are not supported.

Chapter 11: Known Issues and Limitations

Description of Known Issues and Limitations

- The parameter `compute_process_graph=true`, machines using the parameter `damage` or volumetric source species, and structures using periodic boundary conditions are not supported by the command `etch method=pmc averaging_runs=1`.
- The parameter `compute_process_graph` is not supported when the `noise_reduction` parameter is set to a value other than 0.

A

Examples

This appendix provides examples that demonstrate how to use Sentaurus Topography 3D with different models. Three-dimensional input structures can be created using either Sentaurus Topography 3D itself or Sentaurus Structure Editor. Examples of how to create 3D structures are available in the Sentaurus™ Structure Editor User Guide.

Initial Structure Generation

The following examples show how to use geometric etch and deposition steps to generate simple initial structures in Sentaurus Topography 3D.

Simple Trench

This example shows how to create a simple silicon trench:

```
# Define a silicon unit cuboid as the initial bulk structure.  
define_structure material=Silicon point_min={0 0 0} \  
    point_max={0.24 0.05 0.33}  
  
# Define a cuboid shape that is used to etch a trench out  
# of the initial structure.  
define_shape type=cube name=trench point_min={0.06 0.0 0.15} \  
    point_max={0.18 0.05 0.33}  
  
# Use the shape to etch a trench.  
etch shape=trench  
  
# Save the final structure to file.  
save file="Structures/trench.tdr"
```

Fill

This example illustrates the use of the `fill` command when generating an initial structure:

```
# Define a silicon unit cube as the initial bulk structure.  
define_structure material=Silicon point_min={0 0 0} \  
    point_max={0.35 0.05 0.07}  
  
# Fill the initial structure with aluminum with a thickness  
# of 0.08 micrometers.  
fill material=Aluminum thickness=0.08  
  
# Fill again with Photoresist with a thickness of 0.18 micrometers.  
fill material=Photoresist thickness=0.18  
  
# Define a first cuboid shape that is used to etch a first trench  
# out of the previously generated structure.  
define_shape type=cube name=cuboid1 point_min={0.07 0.0 0.07} \  
    point_max={0.14 0.05 0.33}  
  
# Define a second cuboid shape that is used to etch a second trench  
# out of the previously generated structure.  
define_shape type=cube name=cuboid2 point_min={0.21 0.0 0.07} \  
    point_max={0.28 0.05 0.33}  
  
# Etch the two cuboids.  
etch shape=cuboid1  
etch shape=cuboid2  
  
# Save the final structure to file.  
save file=Structures/etch_input.tdr
```

Mask or Patterning

This example illustrates the use of the `define_mask` and `pattern` commands:

```
# Define an initial substrate.  
define_structure material=Silicon point_min={0 0 0} point_max={1 1 1}  
  
# Define a mask  
# - the mask is loaded from the file 'mask.gds'.  
# - the Sentaurus Topography 3D internal name is 'mask_1'.  
# - the layer used from the file 'mask.gds' is called '1:0'.  
# - the domain cut from the layer '1:0' is the rectangle {0 0} {1 1}.  
define_mask name=mask_1 file=mask.gds layer=1:0 \  
    domain_min={0 0} domain_max={1 1}  
  
# Pattern using 'mask_1' and material Photoresist.  
pattern mask=mask_1 material=Photoresist thickness=0.1
```

Appendix A: Examples

Initial Structure Generation

```
# Save the resulting structure to TDR.  
save
```

Multi-Tapered 3D Shapes

The following example defines a multi-tapered 3D shape with tilting angles:

```
# Define an initial structure.  
define_structure material=Nitride point_min={0 0 0} point_max={1 1 3}  
  
# Define a mask based on 2D polygon coordinates.  
define_mask name=m polygon={-5 0.7 -5 0.3 5 0.3 5 0.7}  
  
# Define the vertical tapered mask by specifying tapering angles.  
define_shape name=s type=mask mask=m \  
    z_coordinates={3.0 2.5 2.0 1.5 1.0} taper_angles={-10 15 0 -5}  
  
# Etch the mask.  
etch shape=s  
  
# Save the final structure.  
save
```

The following example defines multi-tapered 3D shapes by a 2D mask and a series of sidewall profile coordinates:

```
# Define an initial structure.  
define_structure material=Nitride point_min={0 0 0} point_max={1 1 3}  
  
# Define a mask based on 2D polygon coordinates.  
define_mask name=m polygon={-5 0.7 -5 0.3 5 0.3 5 0.7}  
  
# Define the sidewalls of the mask with pairs of lateral offsets  
# and z-coordinates.  
define_shape name=s type=mask mask=m \  
    profile={ 0.000 3.0 \  
        -0.088 2.5 \  
        0.046 2.0 \  
        0.046 1.5 \  
        0.002 1.0 }  
  
# Deposit the Photoresist mask.  
deposit shape=s material=Photoresist  
  
# Save the final structure.  
save
```

Appendix A: Examples

Deposition

Deposition

The following examples show the use of the deposition models in Sentaurus Topography 3D.

Atomistic Layer Deposition

This example illustrates the atomistic layer deposition (ALD) model:

```
# Define an ALD machine and set the deposited material, the sticking
# coefficient, the anisotropy, the conformality, the growth per cycle,
# the cycle duration, and the ion angular distribution exponent.
define_deposit_machine model=ald material=Tungsten \
    sticking=1e-3 anisotropy=0 conformality=10 \
    growth_per_cycle=0.0001 cycle_duration=1 exponent=1

# Create an initial structure with a hole by removing a tapered cuboid
# from a block of Nitride.
define_structure material=Nitride \
    point_min={-0.1 -0.4 -5.1} point_max={0.1 0.4 0}
define_shape type=cube name=trench \
    point_min={-0.1 -0.1 -5} point_max={0.1 0.1 0} scale_bottom_y=0.9
etch shape=trench
save

# Start the ALD process.
deposit spacing=0.02 cycles=500 engine=monte_carlo

# Save the final structure to a file.
save
```

Crystallographic Orientation-Dependent Deposition

This example illustrates the crystallographic orientation-dependent deposition model:

```
# Define a crystal deposition machine, set the deposition rates along the
# <100>, <110>, and <111> directions of the lattice, set the deposited
# material, and set the material where deposition will occur.
define_deposit_machine model=crystal material=Silicon \
    selective_materials=Silicon rate_100=0.3 rate_110=0.1 rate_111=0.01

# Create the initial structure and save it.
define_structure material=Silicon point_min={0 0 0} \
    point_max={0.03 0.06 0.18} slice_angle=-90
define_shape type=cube name=c0 point_min={0.01 0.02 0.14} \
    point_max={0.02 0.04 0.18}
define_shape type=cube name=c1 point_min={0 0 0.05} \
    point_max={0.03 0.02 0.18}
```

Appendix A: Examples

Deposition

```
define_shape type=cube name=c2 point_min={0 0.04 0.05} \
    point_max={0.03 0.06 0.18}
define_shape type=cube name=c3 point_min={0 0 0.05} \
    point_max={0.03 0.02 0.15}
define_shape type=cube name=c4 point_min={0 0.04 0.05} \
    point_max={0.03 0.06 0.15}
define_shape type=cube name=c5 point_min={0 0. 0.15} \
    point_max={0.01 0.06 0.2}
define_shape type=cube name=c6 point_min={0.02 0. 0.15} \
    point_max={0.03 0.06 0.2}
etch shape=c0
etch shape=c1
etch shape=c2
deposit shape=c3 material=Oxide merge=true
deposit shape=c4 material=Oxide merge=true
deposit shape=c5 material=Nitride
deposit shape=c6 material=Nitride
save

# Start the silicon deposition process and set the crystallographic
# orientation of the deposited region.
deposit spacing={0.002 0.002 0.002} time=0.25 flat_orientation={1 1 0} \
    vertical_orientation={0 0 1}

# Save the final structure to a file.
save
```

Electrodeposition

This example illustrates the electrodeposition model:

```
# Define an electrodeposition machine and set its physical parameters.
define_deposit_machine model=electrodeposition material=Copper \
    overpotential=-0.21 bulk_distance=0.05 temperature=300 \
    depo_bulk_concentration=2.4e-4 exchange_current_density=1 \
    exchange_current_density_inh=0.039 \
    exchange_current_density_acc=2.5 inh_bulk_concentration=5e-8 \
    inh_diffusivity=5e-7 inh_adsorption_rate=6e4 \
    inh_displacement_rate=0 acc_bulk_concentration=0.64e-7 \
    acc_adsorption_rate=6e2 depo_diffusivity=4e-6 acc_diffusivity=1e-5

# Create an initial structure with a hole by removing a cuboid
# from a block of silicon.
define_structure material=Silicon point_min={-0.2 0} point_max={0.2 2}
define_shape name=hole type=rectangle point_min={-0.1 1.5} \
    point_max={0.1 2}
etch shape=hole

# Start the copper deposition process and create a plot every
# 0.1 minutes.
deposit time=1.5 spacing=0.005 plot_interval=0.1
```

Appendix A: Examples

Deposition

```
# Save the final structure to a file.  
save
```

Electroplating Deposition

This example illustrates the electroplating deposition model:

```
# Define an electroplating deposition machine and set the deposited  
# material, the rate, and the initial vertical change of the  
# accelerator concentration.  
define_deposit_machine model=electroplating material=Copper \  
    rate=0.1 delta=5  
  
# Create an initial structure with a hole by removing a cuboid  
# from a block of silicon.  
define_structure material=Silicon point_min={0 0 0} point_max={1 1 1} \  
    region=bulk  
define_shape type=cube name=hole point_min={0.3 0.3 0.5} \  
    point_max={0.7 0.7 1.0}  
etch shape=hole  
  
# Start the copper deposition process and create a plot every 0.1 min.  
deposit time=1 spacing={0.03 0.03 0.03} plot_interval=0.1  
  
# Save the final structure to a file.  
save
```

High-Density Plasma Deposition

This example illustrates the high-density plasma deposition model:

```
# Define a deposition machine that uses the hdp model  
# and set the anisotropy, the ion angular distribution exponent,  
# the deposited material, the rate, the redeposition  
# coefficient, the sputtering coefficients,  
# the sputter rate, and the sticking coefficient.  
define_deposit_machine model=hdp material=Oxide rate=2.0 \  
    anisotropy=0.7 exponent=100 redeposition=0.1 sputter_rate=0.95 \  
    s1=5.5 s2=-6 sticking=0.1  
  
# Load a structure from a file.  
define_structure file=Structures/trench.tdr  
  
# Start the oxide deposition process.  
deposit spacing={0.015 0.015 0.015} time=0.1  
  
# Save the final structure to a file.  
save
```

Appendix A: Examples

Deposition

High-Density Plasma 2 Deposition

This example illustrates the high-density plasma 2 deposition model:

```
# Define a deposition machine that uses the hdp2 model
# and set the anisotropy, the ion angular distribution exponent,
# the deposited material, the rate, the redeposition coefficient,
# the sputtering coefficients, the sputter rate,
# the sticking coefficient, the sputtering distribution type,
# the sputtering distribution exponent, and the reflection coefficient.
define_deposit_machine model=hdp2 material=Oxide rate=2.0 \
    anisotropy=0.7 exponent=100 redeposition=0.1 sputter_rate=0.95 \
    s1=5.5 s2=-6 sticking=0.1 sputter_type="reflective" \
    sputter_exponent=100 reflection=0.05

# Load a structure from a file.
define_structure file=Structures/trench.tdr

# Start the oxide deposition process.
deposit spacing={0.015 0.015 0.015} time=0.1

# Save the final structure to a file.
save
```

Low-Pressure Chemical Vapor Deposition

This example illustrates the low-pressure chemical vapor deposition model:

```
# Define a deposition machine that uses the lpcvd model
# and set the deposited material, the rate,
# and the sticking coefficient.
define_deposit_machine model=lpcvd material=Oxide rate=1.0 \
    sticking=0.35

# Load a structure from a file.
define_structure file=Structures/trench.tdr

# Start the oxide deposition process.
deposit spacing={0.015 0.015 0.015} time=0.2

# Save the final structure to a file.
save
```

Appendix A: Examples

Deposition

Physical Vapor Deposition

This example illustrates the physical vapor deposition model:

```
# Define a deposition machine that uses the pvd model
# and set the deposited material, the rate, and the ion
# angular distribution exponent.
define_deposit_machine material=Oxide model=pdf rate=1.0 exponent=1

# Load a structure from a file.
define_structure file=Structures/trench.tdr

# Start the oxide deposition process.
deposit spacing={0.015 0.015 0.015} time=0.15

# Save the final structure to a file.
save
```

Plasma-Enhanced Chemical Vapor Deposition

This example illustrates the plasma-enhanced chemical vapor deposition model:

```
# Define a deposition machine that uses the pecvd model
# and set the anisotropy, the ion angular distribution exponent,
# the deposited material, the rate, and the sticking coefficient.
define_deposit_machine model=pecvd material=Oxide rate=1.0 \
    anisotropy=0.6 exponent=30 sticking=0.05

# Load a structure from a file.
define_structure file=Structures/trench.tdr

# Start the oxide deposition process.
deposit spacing={0.015 0.015 0.015} time=0.2

# Save the final structure to a file.
save
```

Simple Deposition

This example illustrates the use of the simple deposition model:

```
# Define a deposition machine that uses the simple deposition model
# and set the deposited material, the curvature, and the rate.
define_deposit_machine model=simple material=Oxide rate=1.0 \
    anisotropy=0.3 curvature=0.1

# Load a structure from a file.
define_structure file=Structures/trench.tdr
```

Appendix A: Examples

Deposition

```
# Start the deposition process.  
deposit spacing={0.015 0.015 0.015} time=0.1  
  
# Save the final structure to a file.  
save  
  
# Extract the intersection points of the final structure  
# with a line passing through the point (0.05 0.05 0.0)  
# and parallel to the z-axis.  
extract type=1d_cut axis=z point={0.05 0.05 0.0}
```

Spin-on-Glass Deposition

This example illustrates the spin-on-glass deposition model:

```
# Define a deposition machine that uses the spin_on model and set the  
# deposited material, the viscosity, the density, the initial_thickness,  
# the radial_distance, the surface_tension, the angular_velocity,  
# the evaporation_rate, and the angular_position.  
define_deposit_machine model=spin_on material=Oxide viscosity=0.015 \  
    density=1 initial_thickness=2 radial_distance=2e4 surface_tension=27 \  
    angular_velocity=4000 evaporation_rate=0 angular_position=0  
  
# Create the initial structure and save it.  
define_structure material=Silicon point_min={0 0 0} point_max={100 100 1}  
define_shape type=cube name=11 point_min={40 0 1} point_max={60 100 4}  
define_shape type=cube name=12 point_min={0 60 1} point_max={60 80 2.5}  
define_shape type=cube name=13 point_min={60 30 1} point_max={100 50 1.5}  
deposit shape=11 material=Aluminum  
deposit shape=12 material=Aluminum  
deposit shape=13 material=Aluminum  
save  
  
# Start the oxide deposition process.  
deposit spacing={10 10} time=0.01 cfl=0.025  
  
# Save the final structure to a file.  
save
```

Appendix A: Examples

Etching

Etching

The following examples show the use of the etching models in Sentaurus Topography 3D.

Crystallographic Orientation–Dependent Etching

This example illustrates the crystallographic orientation–dependent etching model:

```
# Define a crystal etch machine, set the etching rates along the <100>,
# <110>, and <111> directions of the lattice, and set the material to be
# etched.
define_etch_machine model=crystal rate_100=1 rate_110=1.2 rate_111=0.1 \
    etchable_material=Silicon
# Create the initial structure and save it.
define_structure material=Silicon point_min={0 0 0} \
    point_max={0.4 0.2 0.2} region=Silicon_region
fill thickness=0.05 material=Photoresist
define_shape type=cube point_min={0.1 0.05 0.2} point_max={0.3 0.2 0.3} \
    name=cuboid
etch shape=cuboid
save

# Set the crystallographic orientation of the region with the material
# to be etched.
set_orientation region=Silicon_region flat_orientation={1 1 0} \
    vertical_orientation={0 0 1}

# Start the etch simulation process.
etch spacing={0.004 0.004 0.004} time=0.2

# Save the final structure to a file.
save
```

Dry Etching

This example illustrates the dry etching model:

```
# Define a dry-etch machine.
# Define the material that is redeposited and set
# the sticking coefficient and the redeposition rate.
define_etch_machine model=dry deposit_material=Anyinsulator \
    sticking=0.035 rate=0.2

# Multimaterial etch.
# Define materials and set the machine-dependent properties:
# rates and sputtering coefficients.
add_material material=Oxide rate=2.0 s1=5.5 s2=-6
add_material material=Resist rate=0.1 s1=5.5 s2=-6
add_material material=Anyinsulator rate=0.1 s1=5.5 s2=-6
```

Appendix A: Examples

Etching

```
# Load a structure from a file.  
define_structure file=Structures/dry_etch_input.tdr  
  
# Start etch simulation process.  
etch spacing={0.015 0.015 0.015} time=0.1  
  
# Save the final structure to a file.  
save
```

Wet Etching

This example illustrates the wet etching model:

```
# Define a wet etch machine and set the etchant diffusivity and the  
# distance of the source plane from the topmost point of the structure  
# under process.  
define_etch_machine model=wet diffusivity=1e-2 source_distance=1  
  
# Define materials to be etched and their rates.  
add_material material=Silicon rate=1  
  
# Create an initial structure with a hole by removing a cuboid  
# from a block of silicon.  
define_structure material=Silicon point_min={0 0 0} point_max={2 3 2}  
fill material=Photoresist thickness=0.5  
define_shape type=cube name=c point_min={1 1 2} point_max={2 2 2.5}  
etch shape=c  
  
# Start etch simulation process.  
etch time=0.5 spacing=0.1  
  
# Save the final structure to a file.  
save
```

Simultaneous Etching and Deposition

This example illustrates the simultaneous etching and deposition (etchdepo) model:

```
# Define an etchdepo machine and set the ion angular distribution  
# exponent. Define the material that is redeposited and set the  
# sticking coefficient and the redeposition rate.  
define_etch_machine model=etchdepo exponent=100 \  
deposit_material=Anyinsulator sticking=0.035 rate=0.2  
  
# Multimaterial etch.  
# Define materials and set the machine-dependent properties:  
# rates and sputtering coefficients.  
add_material material=Oxide rate=2.0 s1=5.5 s2=-6  
add_material material=Anyinsulator rate=0.1 s1=5.5 s2=-6
```

Appendix A: Examples

Etching

```
# Load a structure from a file.  
define_structure file=Structures/dry_etch_input.tdr  
  
# Start etch simulation process.  
etch spacing={0.01 0.01 0.01} time=0.1  
  
# Save the final structure to a file.  
save
```

High-Density Plasma Etching

This example illustrates the high-density plasma etch model:

```
# Define a high-density plasma etch machine and set  
# its ion angular distribution exponent.  
define_etch_machine model=hdp exponent=15  
  
# Multimaterial etch.  
# Define materials and set the machine-dependent properties:  
# rates, anisotropy, and sputtering coefficients.  
add_material material=Aluminum rate=1.13 anisotropy=0.04 s1=5.5 s2=-6  
add_material material=Photoresist rate=0.663 anisotropy=0.95 s1=5.5 s2=-6  
  
# Load a structure from a file.  
define_structure file=Structures/etch_input.tdr  
  
# Start etch simulation process.  
etch spacing={0.015 0.015 0.015} time=0.1  
  
# Save the final structure to a file.  
save
```

High-Density Plasma 2 Etching

This example illustrates the high-density plasma 2 etch model:

```
# Define a high-density plasma 2 etch machine and  
# set its ion angular distribution exponent.  
define_etch_machine model=hdp2 exponent=100  
  
# Multimaterial etch.  
# Define materials and set for each of them the machine-dependent  
# properties: rates, anisotropy, sputtering coefficients,  
# sticking coefficient, and sputter rate.  
add_material material=Aluminum rate=1.13 anisotropy=0.04 s1=5.5 s2=-6 \  
sticking=0.5 sputter_rate=1.0  
add_material material=Photoresist rate=0.663 anisotropy=0.95 s1=5.5 \  
s2=-6 sticking=0.0 sputter_rate=0.0
```

Appendix A: Examples

Etching

```
# Load a structure from a file.  
define_structure file=Structures/etch_input.tdr  
  
# Start etch simulation process.  
etch spacing={0.015 0.015 0.015} time=0.1  
  
# Save the final structure to a file.  
save
```

Ion-Enhanced Etching

This example illustrates the ion-enhanced etching model:

```
# Define an ion-enhanced etch machine and set  
# its ion angular distribution exponent.  
define_etch_machine model=ion_enhanced exponent=10  
  
# Multimaterial etch.  
# Define materials and set the machine-dependent properties:  
# rates, anisotropy, sputtering coefficients, and sticking coefficient.  
add_material material=Aluminum rate=0.7 anisotropy=0.5 sticking=0.5 \  
    s1=5.5 s2=-6  
add_material material=Photoresist rate=0.5 anisotropy=0.5 sticking=0.0 \  
    s1=5.5 s2=-6  
  
# Load a structure from a file.  
define_structure file=Structures/etch_input.tdr  
  
# Start etch simulation process.  
etch spacing={0.015 0.015 0.015} time=0.1  
  
# Save the final structure to a file.  
save
```

Ion-Milling

This example illustrates the ion-milling model:

```
# Define an ion-mill machine.  
define_etch_machine model=ionmill  
  
# Multimaterial etch.  
# Define materials and set the machine-dependent properties:  
# rates, anisotropy, and sputtering coefficients.  
add_material material=Aluminum rate=0.505 anisotropy=1.0 s1=5.5 s2=-6  
add_material material=Photoresist rate=0.18 anisotropy=1.0 s1=5.5 s2=-6  
add_material material=Tantalum rate=0.18 anisotropy=1.0 s1=5.5 s2=-6  
  
# Load a structure from a file.  
define_structure file=Structures/etch_input.tdr
```

Appendix A: Examples

Etching

```
# Start etch simulation process.  
etch spacing={0.015 0.015 0.015} time=0.1  
  
# Save the final structure to a file.  
save
```

PMI Simple Etching

This example illustrates the PMI simple etching model:

```
# Define a PMI simple etching machine.  
define_etch_machine model=pmi_simple_etch  
  
# Multimaterial etch.  
# Define materials and set the machine properties.  
add_material material=Silicon rate=0.3 anisotropy=0.3  
add_material material=Oxide rate=0.1 anisotropy=0.0  
  
# Load a structure from a file.  
define_structure file=Structures/etch_input.tdr  
  
# Start etch simulation process.  
etch spacing={0.005 0.005 0.005} time=0.1 plot_interval=0.03  
  
# Save the final structure to a file.  
save
```

The source code for the PMI model is available in the file:

```
`${STROOT}/tcad/${STRELEASE}/lib/sptopo3d/examples/pmi_simple_etch/  
pmi_simple_etch.cc
```

Reactive Ion Etching

This example illustrates the reactive ion etch model:

```
# Define a reactive ion etch machine and set  
# its ion angular distribution exponent.  
define_etch_machine model=rie exponent=100  
  
# Multimaterial etch.  
# Define materials and set the machine-dependent properties:  
# rates and anisotropy.  
add_material material=Aluminum rate=0.7 anisotropy=1.0  
add_material material=Photoresist rate=0.5 anisotropy=1.0  
  
# Load a structure from a file.  
define_structure file=Structures/etch_input.tdr
```

Appendix A: Examples

Etching

```
# Start etch simulation process.  
etch spacing={0.015 0.015 0.015} time=0.1  
  
# Save the final structure to a file.  
save
```

Reactive Ion Etching 2

This example illustrates the reactive ion etch 2 model:

```
# Define a reactive ion etch 2 machine and set  
# its ion angular distribution exponent.  
define_etch_machine model=rie2 exponent=100  
  
# Multimaterial etch.  
# Define materials and set the machine-dependent properties:  
# rates, anisotropy, and the sticking coefficient.  
add_material material=Aluminum rate=0.7 anisotropy=1.0 sticking=0.5  
add_material material=Photoresist rate=0.5 anisotropy=1.0 sticking=0.0  
  
# Load a structure from a file.  
define_structure file=Structures/etch_input.tdr  
  
# Start etch simulation process.  
etch spacing={0.015 0.015 0.015} time=0.1  
  
# Save the final structure to a file.  
save
```

This example creates a structure from the beginning and shows how to set the parameter reflection. Microtrenching at the bottom of the trench is formed:

```
# Define the initial bulk structure.  
define_structure material=Silicon point_min= {0 0 0} \  
    point_max= {0.1 0.02 0.01}  
  
# Create an oxide and a photoresist layer.  
fill material=Oxide thickness=0.15  
fill material=Photoresist thickness=0.02  
  
# Remove part of the photoresist.  
define_shape type=cube point_min= {0.025 0.0 0.16} \  
    point_max= {0.075 0.02 0.19} name=subtract  
etch shape=subtract  
  
# Save the initial structure.  
save  
  
# Define the rie2 machine.  
define_etch_machine model=rie2 exponent=100
```

Appendix A: Examples

Tilt and Units

```
# Set and define the material-dependent parameters.  
# Ion reflection is only considered to happen in the resist.  
# No neutrals are considered.  
add_material material=Photoresist rate=0.0 anisotropy=0.0 sticking=0.0 \  
    reflection=0.05  
add_material material=Oxide rate=1.0 anisotropy=1.0 sticking=0.0  
etch spacing= {0.005 0.005 0.005} time=0.05  
  
# Save the simulation results, appending them  
# in the previously generated TDR file.  
save
```

Simple Etching

This example illustrates the simple etching model:

```
# Define an etch machine using a simple model.  
define_etch_machine model=simple  
  
# Multimaterial etch.  
# Define materials and set the machine-dependent properties:  
# rate, anisotropy, and curvature factor.  
add_material material=Silicon rate=0.5 anisotropy=0.3 curvature=0.0  
add_material material=Oxide rate=0.1 anisotropy=0.0 curvature=0.0  
  
# Load a structure from a file.  
define_structure file=Structures/etch_input.tdr  
  
# Start etch simulation process.  
etch spacing={0.015 0.015 0.015} time=0.1  
  
# Save the final structure to a file.  
save
```

Tilt and Units

This example shows how to tilt the wafer using the parameters `tilt` and `rotation`. At the same time, nondefault units are set:

```
# Define a deposition machine that uses a simple deposition model  
# and set the deposited material, the anisotropy, the curvature,  
# and the rate. Tilt and rotation are also set for the machine.  
define_deposit_machine model=simple material=Oxide anisotropy=1.0 \  
    curvature=0.0 rate=0.05<um/min> rotation=90 tilt=45  
  
# Load a structure from a file.  
define_structure file=Structures/trench.tdr  
  
# Start the oxide deposition process.
```

Appendix A: Examples

Simulations With 2D Structures

```
deposit spacing={10<nm> 10<nm> 10<nm>} time=60<s>

# Save the final structure to a file.
save
```

Simulations With 2D Structures

This example shows how to use Sentaurus Topography 3D to run simulations on two-dimensional structures:

```
# Define a 2D initial structure.
define_structure point_min={0 0} point_max={1 1} material=Silicon

# Fill works exactly as for a 3D structure.
fill material=Photoresist thickness=0.1

# Define a rectangle.
define_shape type=rectangle name=rect point_min={0.25 0.5} \
    point_max={0.75 1.5}

# Etch a trench geometrically, exactly as for a 3D structure.
etch shape=rect

# Define a simple deposition machine, exactly as for a 3D structure.
define_deposit_machine model=simple material=Oxide rate=1.0 \
    anisotropy=0.7 curvature=0.0

# Deposit. Note that the size of the 'spacing' parameter value
# must match the dimension of the structure.
deposit spacing={0.025 0.025} time=0.2

# Save the 2D structure to TDR.
save
```

Integration With Sentaurus Process

This example shows how to use the functionality of Sentaurus Topography 3D from within Sentaurus Process. This is a Sentaurus Process command file. Note the use of the `init` and `struct` commands instead of `define_structure` and `save`, respectively. The `topo` command of Sentaurus Process calls all the functionality related to Sentaurus Topography 3D:

```
# Initialize Sentaurus Process with a trench.
init tdr=Structures/trench.tdr

# Define a deposition machine that uses the simple model.
# Note the 'topo' prefix.
topo define_deposit_machine model=simple material=Oxide rate=2.0 \
```

Appendix A: Examples

Integration With Sentaurus Process

```
anisotropy=0.7 curvature=0.0

# Start the Oxide deposition process.
# Note the 'topo' prefix.
topo deposit spacing= {0.015 0.015} time=0.1

# Save the structure.
struct tdr=simple_deposition
```

B

EAD File Format

This appendix describes the energy angular distribution (EAD) file format.

Description of the EAD File Format

EADs can be stored in a tabular file format, defining the values of the EAD at discrete positions (E, θ), where E denotes the energy and θ is the polar angle.

EAD values are given in units of flux per energy and angle, that is, mol/m²/s/eV. A global conversion factor allows you to easily convert the values to the prescribed unit. In addition, you can specify the mathematical format of the distribution, for example, if the distribution includes an isotropy factor $\cos\theta$.

Figure 37 Format of the EAD file

EAD values arranged in a matrix of size:
<number of energy values> × <number of angle values>

1	ead_table 1.0
2	energies[eV] : 195 200 205 210 215 220
3	angles[deg] : 0 20 40 60 80
4	conversion_factor: 1e-18
5	distribution_format: standard
6	5.54e-04 5.57e-5 3.24e-7 5.89e-10 0
7	3.67e-04 6.25e-5 1.56e-7 3.01e-10 0
8	3.29e-04 9.71e-5 1.03e-7 2.04e-10 0
9	3.36e-04 1.72e-5 7.93e-7 1.71e-10 0
10	3.94e-04 3.47e-5 6.97e-7 1.67e-10 0
11	6.21e-04 9.69e-5 8.34e-7 2.09e-10 0

Energy ↓ → Angle

Appendix B: EAD File Format

Mathematical Conventions

The file consists of the following:

- The *header section* contains meta information about the data. The first line must be the file format specifier, followed by the version number. In [Figure 37](#), line numbering is provided for demonstration purposes only and refers to the following:

Line 1: Format specifier (for example, ead_table) followed by version number

Line 2: List of monotonically increasing energy values

Line 3: List of monotonically increasing polar angle values

Line 4: Global scaling factor for all EAD values

Line 5: Mathematical form of the distribution

The lines of the header section can be given in arbitrary order and must contain, at least, the energy and angle values. In general, empty lines are ignored.

- The *value section* contains the EAD values.

Mathematical Conventions

The total flux Γ of the imported EAD is computed by integrating the EAD. Different mathematical conventions are possible as follows:

$$\text{distribution_format=standard: } \int_{E_{\min}}^{E_{\max}} dE \int_0^{2\pi} d\phi \int_0^{\pi/2} \sin\theta d\theta \text{ EAD}(E, \theta) = \Gamma$$

- *distribution_format=cos_convention:*

$$\int_{E_{\min}}^{E_{\max}} dE \int_0^{2\pi} d\phi \int_0^{\pi/2} \sin\theta d\theta \text{ EAD}(E, \theta) \cos\theta = \Gamma$$

The following example uses the standard distribution format. If you specify the parameter `flux` when defining the species distribution, the EAD is normalized to the specified flux:

$$\int_{E_{\min}}^{E_{\max}} dE \int_0^{2\pi} d\phi \int_0^{\pi/2} \sin\theta d\theta \text{ EAD}(E, \theta) = \text{flux}$$

Appendix B: EAD File Format

Mathematical Conventions

Special Cases

If you specify only one energy value E , then the flux is computed as:

$$\int_0^{2\pi} d\phi \int_0^{\pi/2} \sin\theta d\theta \text{ EAD}(E, \theta) = \Gamma$$

If you specify only one angle value θ , the flux is computed as:

$$\int_{E_{\min}}^{E_{\max}} dE \int_0^{2\pi} d\phi \text{ EAD}(E, \theta) = \Gamma$$

If you specify only one energy value E and only one angle value θ , then the flux is computed as: $2\pi \text{ EAD}(E, \theta) = \Gamma$

The same convention is used to save special particle Monte Carlo (PMC) species distributions, such as:

- Unidirectional distributions with only one angle value
- Energy-independent distributions (angular distributions), where the energy is stored as 0
- Species distributions that are saved with `energy_min=energy_max` (in which case, only one slice of the distribution, that is, an angular distribution with `energy E=energy_min`, is saved)

Sampling From the Distribution

The following methods are available to extract energy and angle samples from the given EAD:

- Rejection method
- Inverse cumulative distribution function (CDF) method

You can specify the sampling method using the parameter `sampling_method` of the `define_species_distribution` command (see [define_species_distribution on page 290](#)). For example:

```
sampling_method=inverse_cdf  
sampling_method=rejection
```

Glossary

ALD

Atomic layer deposition.

anisotropy

The ratio of the directional deposition or etch rate to the total deposition or etch rate, which is a combination of a directional component and an isotropic component.

CCP

Capacitively coupled plasma.

CDF

Cumulative distribution function.

chemical vapor deposition (CVD)

The growth of the material from a gaseous medium containing a mixture of reactants that chemically interact and convert to the required material.

curvature-dependent deposition/etch

The deposition rate along a curved surface is a function of the local curvature, which smooths the surface as it evolves.

conformal structure

If all the regions of a structure share the same vertices and faces along region boundaries that touch each other, the structure is *conformal*. If vertices or faces on shared region borders are not part of both regions, the structure is *nonconformal*. Sentaurus Topography 3D needs conformal structures as input. Nonconformal structures can be made conformal when read into Sentaurus Topography 3D by setting the parameter `conformalize=true` in the `define_structure` command.

CVD

Chemical vapor deposition.

dry etching

An etching process that uses plasma gas as the reactive source.

EAD

Energy angular distribution.

etching

The process that removes materials from the wafer surface.

HDP

High-density plasma.

high-density plasma (HDP) deposition

A deposition process that uses high-density plasma sources to produce deposition precursors. Usually, this process results in simultaneous deposition and sputter etching of thin film on the wafer surface, due to highly energetic ions.

IAD

Ion angular distribution. It describes the angular flux distribution of ions in flux models, that is, the probability of an ion traveling at a certain angle.

ICP

Inductively coupled plasma.

IEAD

Ion energy and angular distribution. It describes the energy and angular flux distribution of ions in flux models, that is, the probability of an ion traveling at a certain energy and angle.

ion-milling

A synonym for ion-mill etch.

ion-mill etch

An etching process purely caused by physical sputtering of the surface by highly energetic ions. The sputter yield (the number of particles knocked off the surface per incoming ion) is a function of the angle between the surface normal and the direction of the incoming ion.

IUPAC

International Union of Pure and Applied Chemistry.

low-pressure chemical vapor deposition (LPCVD)

The deposition environment is at such a low pressure that particle-particle collision based on gas-vapor deposition is negligible compared to particle-surface collision. The particles travel between surfaces in a ballistic trajectory.

LPCVD

Low-pressure chemical vapor deposition.

MPI

Message passing interface.

NAD

Neutral angular distribution.

ODE

Ordinary differential equation.

PECVD

Plasma-enhanced chemical vapor deposition.

physical vapor deposition (PVD)

The condensation of material from its own vapors.

plasma-enhanced chemical vapor deposition (PECVD)

A chemical vapor deposition technique that uses a plasma chemical vapor discharge to enhance the deposition characteristics.

PMC

Particle Monte Carlo.

PVD

Physical vapor deposition.

reactive ion etch (RIE)

An etching process in which chemically reactive ions are the major source of the reaction that removes particles from the surface.

RF

Radio frequency.

RFM

Rate formula module.

RIE

Reactive ion etch.

Sentaurus Lithography

A microlithography process simulator.

shadowed element

When a surface element has no vertical line of sight to the source because another surface element blocks its view, the element is *shadowed*.

SLO

File format used by Sentaurus Lithography.

sticking coefficient

The statistical probability that an incoming precursor will stay on the surface and contribute to the deposition of thin film, contrary to bouncing back to the gas phase. It controls the conformality in a reemission process.

STP

Standard temperature and pressure. Since 1982, IUPAC has defined STP as a temperature of 273.15 K or 0°C and a pressure of 10^5 Pa.

TDR

File format used for TCAD Sentaurus tools.

VBE

Volume boundary extraction.