SIEMENS EDA

Calibre® WORKbench™ Topography Modeling User's and Reference Manual

Software Version 2024.1

SIEMENS

About Siemens Digital Industries Software

Siemens Digital Industries Software is a global leader in the growing field of product lifecycle management (PLM), manufacturing operations management (MOM), and electronic design automation (EDA) software, hardware, and services. Siemens works with more than 100,000 customers, leading the digitalization of their planning and manufacturing processes. At Siemens Digital Industries Software, we blur the boundaries between industry domains by integrating the virtual and physical, hardware and software, design and manufacturing worlds. With the rapid pace of innovation, digitalization is no longer tomorrow’s idea. We take what the future promises tomorrow and make it real for our customers today. Where today meets tomorrow. Our culture encourages creativity, welcomes fresh thinking and focuses on growth, so our people, our business, and our customers can achieve their full potential.

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

Table of Contents

List of Figures

List of Tables

Chapter 1
Topographical Modeling User's Guide

Topographical modeling models the reflection effects from underlying resist topography caused by the absence of anti-reflection coatings in implant layers. The result of this methodology is a set of optical models that predict the characteristics of different films under the resist with a set of signals that model secondary reflection and diffraction from these films.

Introduction to Topographical Modeling

Typically, a bottom anti-reflection coating (BARC) insulates the resist from reflections from underlying films. Implant layers frequently do not use a BARC because of the cost and complexity of removing the BARC. The absence of the BARC exposes the resist to reflections from underlying films and can significantly influence OPC results.

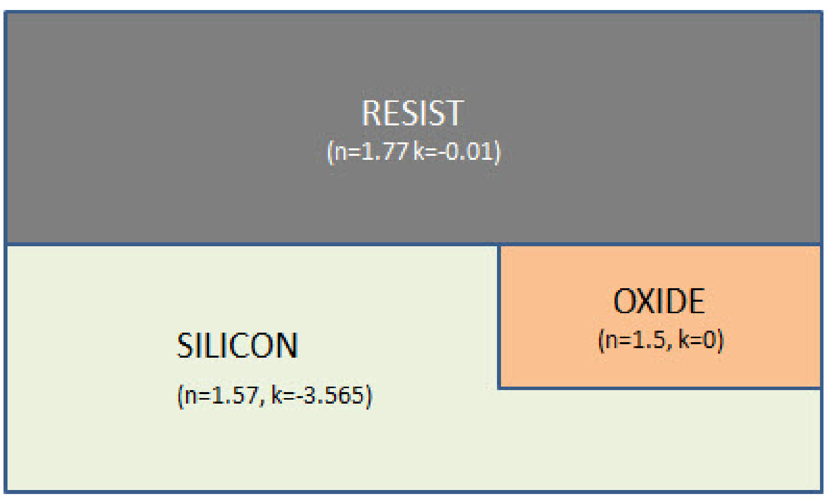
The Calibre® WORKbench™ topographical modeling methodology handles these processes, specifically by using two or more optical models to model film inhomogeneity in the XY directions (transitional optics) and by using additional signals to model secondary reflections and diffraction from the underlying films.

Key Concepts

The following key concepts are important to learning how to implement topographical modeling in Calibre WORKbench.

- **Implant Layers** — The primary motivation for topographical modeling is to model the intralayer interference effects on an implant layer that does not use BARC.
- **Rigorous Model** — A rigorous model is used for implant layers to find the intensity distribution in a film stack by solving Maxwell's equations with the configuration provided by a user, such as the one shown in Figure 1-1. The rigorous model serves as an accurate reference to which a compact model can be compared, but is too slow to use for OPC purposes.
- **Optical Models** — Optical models characterize the optical system used in photolithography. They include film reflection effects assuming films are infinite and homogeneous in the X and Y dimensions. Optical models can be combined together to model inhomogeneities and complemented with additional edge signals to model additional reflections and scattering from the underlying films.
- **Film Stack** — The materials stacked on top of the silicon wafer all the way to the resist-air interface. In optical models, the "film stack" is defined from the top down, with each layer having a thickness (n) and refractive index (k) as shown in Figure 1-1.

Figure 1-1. Topographical Modeling Cross-Section Showing Two Film Stacks



Either two or three distinct types of film stacks are used in this methodology:

- **Oxide Stack** — Resist on top of oxide (blank field).

  In this manual, an optical model with an oxide stack defined is referred to as an oxide optical model.
- **Silicon Stack** — Resist on top of the RX layer, also known as the active area. The silicon stack may also include fins.

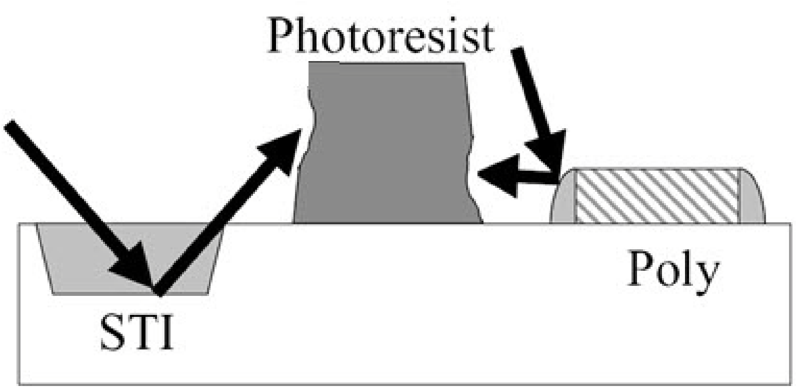
In this manual, an optical model with a silicon stack defined is referred to as a silicon optical model.

- **PC Stack** — Poly on top of the stack, commonly referred to as the PC stack.

  In this manual, an optical model with a PC stack defined is referred to as a PC optical model.

  The PC stack is required only when a poly layer is present (post-PC configurations) as shown in Figure 1-2. It is used to model interactions between the oxide and the resist, such as the shallow trench isolation (STI) oxide shown in the figure.

Figure 1-2. Post-PC Stack Interaction



  Pre-PC configurations require only two film stacks defined. The interaction is between the silicon and oxide stacks.

- **Transitional Region** — The area of the implant mask near a film stack boundary in the X and Y dimensions. The effects within a transitional region are a hybrid of multiple stack effects, which can be dramatically different from separate stack models because of the optical interference effects between them. Transitional regions are also referred to as density layers.

  For example, Figure 1-3 shows a single density layer requiring two optical models, one for oxide and one for silicon, using a a brightfield mask. Internally, Calibre uses a density function u(x) (as shown in Figure 1-4) to weight the two optical models and calculate the final intensity I(x).

Figure 1-3. Single Density Layer, Top View (left) and Side View (right)

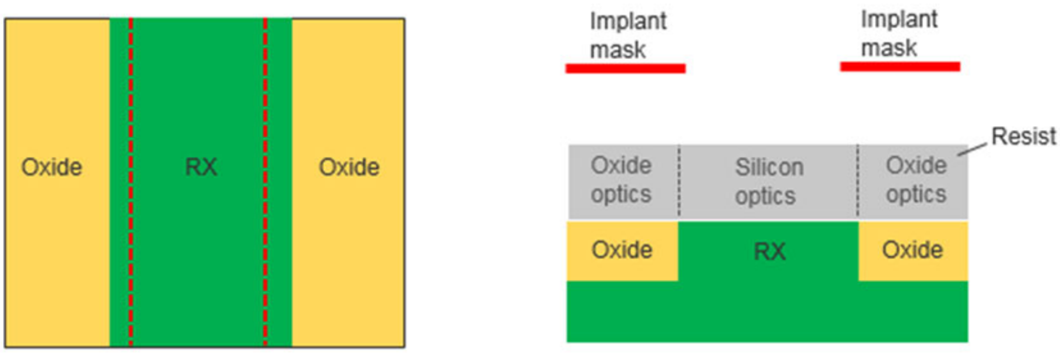
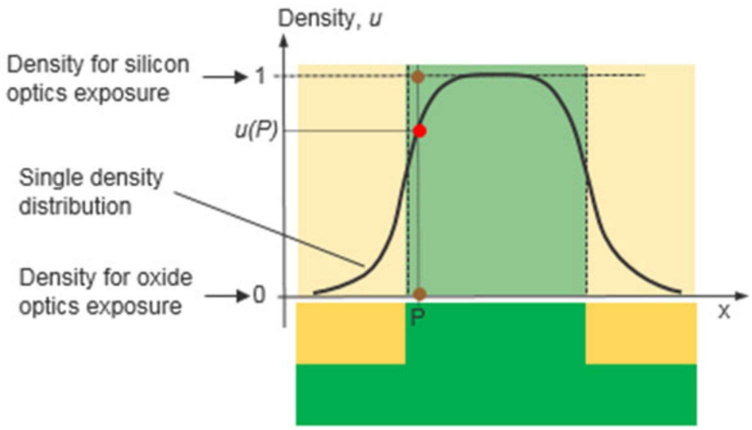


Figure 1-4. Density Function for Topography Modeling



In this case, the implant makes the oxide regions dark and the silicon region bright. Correspondingly, the density u ranges from 0 to 1.

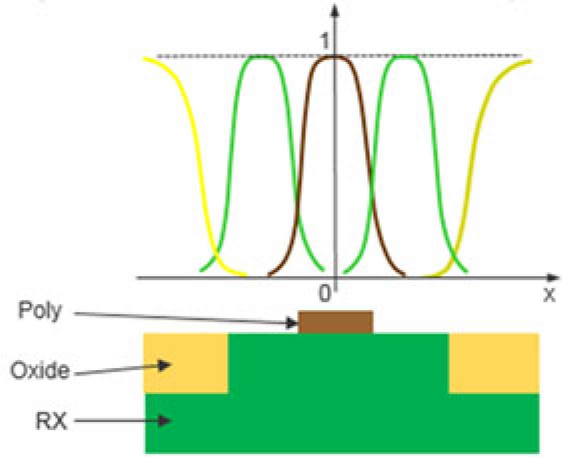
- In the oxide regions, the density is 0, and the final intensity corresponds to the intensity from the oxide optical model.
- In the silicon regions, the density is 1, and the final intensity corresponds to the intensity from the silicon optical model.
- In the transition regions, the intensity is obtained by weighting the contribution between the two optical models according to the formula:

  $$I(x) = u(x)I_{Si}(x)+(1-u(x))I_{Ox}(x)$$

A post-PC case would have multiple density layers and three transitional optical models. The intensity is calculated according to the formula:

$$I(x) = u_{Ox}(x)I_{Ox}(x) + u_{Si}(x)I_{Si} + u_{Poly}(x)I_{Poly}(x)$$

Figure 1-5. Multiple Density Function for Three Transitional Optical Models



- **Density Optical Model** — A specialized optical model used as a density convolution kernel for topographical modeling.
- **Density Exposure** — A measurement percentage for the transmission value. Because the transmission value of a transitional region ranges from 1.0 (100%, representing clear) to 0.0 (0%, representing dark), the optical models need to have a distributed exposure level to generate a smoother image intensity curve. The density values are calculated as part of the topo model.
- **Edge Signal** — Any source of reflected or scattered light to be modeled. All edge signals are represented by a basis kernel, which consists of sums of Gauss-Laguerre kernels. Gauss-Laguerre kernels are orthogonal kernels used to represent arbitrary optical signals. Each edge signal has a characteristic source function $H_s(x)$. When convolved with the basis kernel Hs(x) gives the contribution of an edge signal to the light energy at x. A quadratic term is added to model non-linearities.

  The electric field at position x,y from an edge signal S is given by the formula:

$$E_S(x,y) = \sum_{t=0}^{T-1} \varepsilon_t G_t(x,y) + \varepsilon_{quad}\left(\sum_{t=0}^{T-1} s_t G_t(x,y)\right)^2$$

$$G_t(x,y) = G_{gl,t}(x,y) \otimes H_S(x,y)$$

  The edge signal can be a complex value with real and imaginary components. The linear, quadratic, and diffusion length terms are calibrated to measured CDs. The final electric field is found by adding the edge electric field to the optical field, calculated from SOCS kernels. The final intensity is found by multiplying the final electrical field by its complex conjugate giving the equation:

$$I(x,y) = \left(\sqrt{I_{opt}(x)} + \sum_{s=1}^{N\_signals} E_s^{Re}(x,y)\right)^2 + \left(\sum_{s=1}^{N\_signals} E_s^{Im}(x,y)\right)^2$$

  The edges of the layers are a typical source of edge signals, as shown in Figure 1-6.

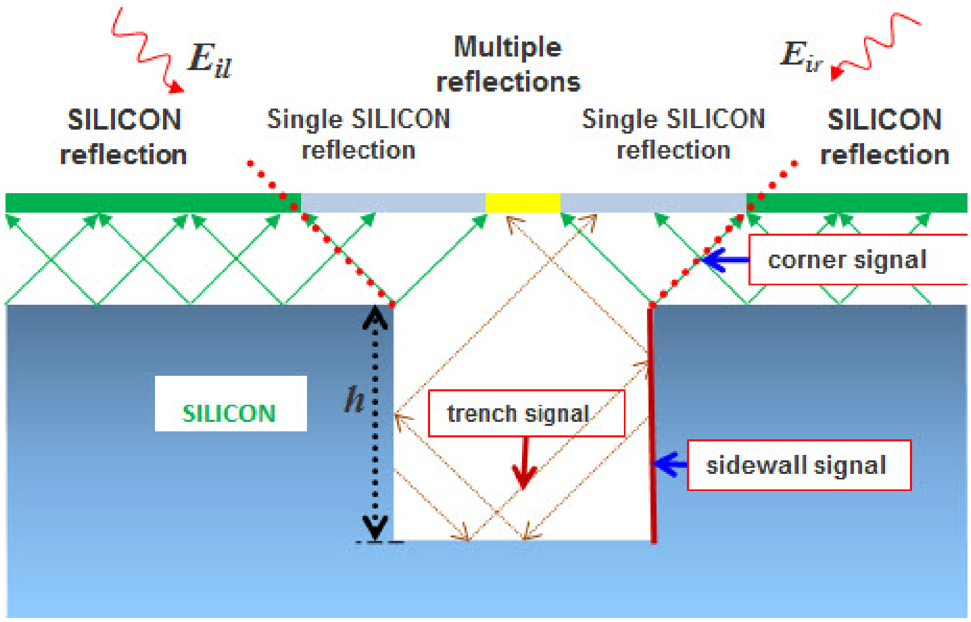
The following types of edge signals are used to model the active layer:

- Sidewall reflection signal (S-term), which can also include an imaginary component (Sim-term)
- Corner reflection signal (C-term)
- Trench radiation effect signal (T-term), which can also include a second-order term (T2-term), fourth-order term (T4-term), and an imaginary component (Tim-term)
- Vertical corner effect signal (V-term)

The following types of edge signals are used to model the poly layer:

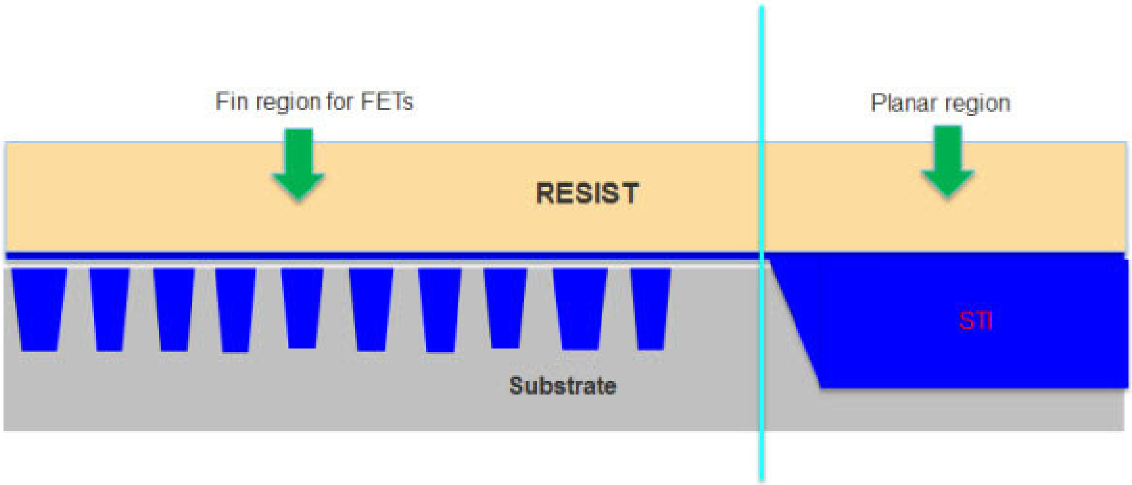
- Gate scattering signal (G-term)
- Poly interconnect scattering signal (I-term)

Figure 1-6. Signal Types for Topo Models



- **FinFET** — A type of transistor where the gate wraps around a channel formed by fins. The fins add additional topology as shown in Figure 1-7. A FinFET is modeled by adding a second film stack to an optical model, and by adding Calibre WORKbench FinFET edge signals (F-term and Fw-term for the sidewall) in the modelform.

Figure 1-7. FinFET Cross-Section



- **Variable Oxide Bias** — Size-dependent sidewall angles in oxide trenches. Oxide trench sidewall angles can vary with the size of the trenches. This causes a variability in the effective width of the trenches. This effect is calculated by applying a variable oxide bias to all RX or SOI edges adjacent to oxide layers in the layout.

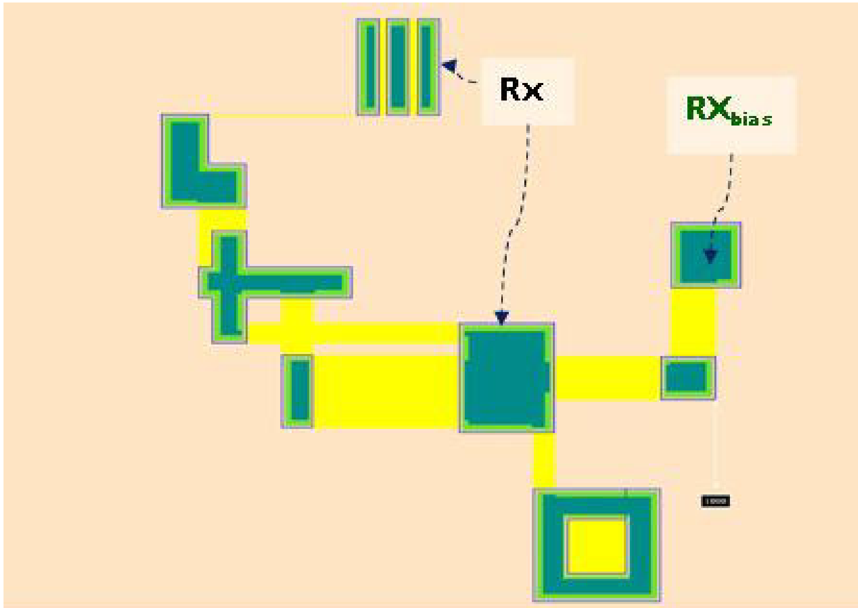
  Note

  Starting with the 2021.2 release, oxide bias is only applied to boundaries between RX or SOI layers and oxide layers, not oxide2 layers. This is because oxide2 is used for shallow trenches, and oxide is used for deep trenches.

  Note

  Calibrating models for variable oxide bias replaces calibrating models for the active layer (topo_dupsize).

Figure 1-8. Variable Oxide Biasing



The total oxide bias b_{oxide} is calculated as follows.

$$b_{oxide} = b_{oxide\_init} + b_{oxide\_var}(w)$$

where

- b_{oxide_init} is the constant oxide bias applied to all RX or SOI edges (or parts of edges) adjacent to the oxide layer in the layout, regardless of the corresponding trench width.
- $b_{oxide\_var}(w)$ is an oxide bias that is dependent on the trench width *w*. A trench is formed by paired RX or SOI edges, or by paired parts of RX or SOI edges adjacent to an oxide layer. Any edge of these types can be considered as a side of a trench having width *w*. If any RX or SOI edge (or partial edge adjacent to an oxide layer) does not have a paired edge, Calibre WORKbench assumes that the corresponding trench has an infinite width.

The absolute value for the variable part of the oxide bias b_{oxide_var} increases monotonically with the width *w*.

The dependency $b_{oxide\_var}(w)$ is calculated as a decomposition over polynomials. The coefficients of the decomposition a_i, i=0,1,2 are calibrated parameters.

The values of b_{oxide}, b_{oxide_init}, and b_{oxide_var} can be positive or negative. A positive value means that an RX or SOI edge moves inside the RX or SOI feature, and a negative value means that an RX or SOI edge moves outside its original location.

In Calibre WORKbench, six parameters are used to calibrate variable oxide bias:

- **obiasinit** — Calibrates oxide_init, an upsize value for oxide bias.
- **obiaswsat** — Calibrates the saturation width of a variable trench bias for oxide bias.
- **obiasmax** — Calibrates the maximum bias for saturation values of a variable trench for oxide bias.
- **oterm0, oterm1, and oterm2** — Calibrates the oxide bias polynomial first, second, and third-order multipliers, respectively.

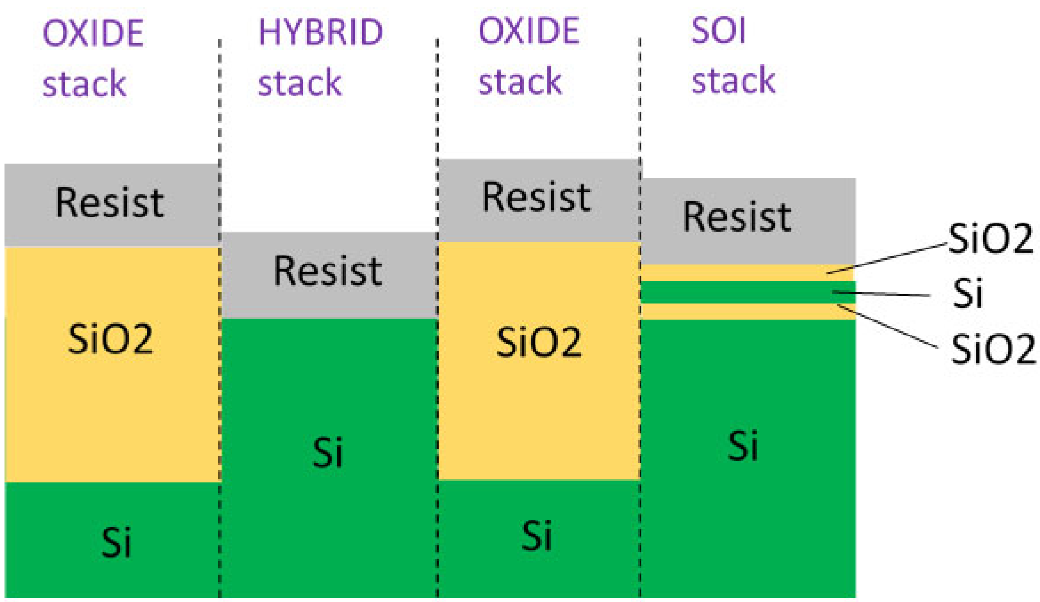
Note

In this manual, the term *oxide bias* is used to refer to the set of six variable oxide bias parameters listed above.

For example, "calibrate oxide bias" means in Calibre® nmModelflow™ to add the parameters TOPO_OBIASINIT, TOPO_OBIASMAX, TOPO_OBIASWSAT, TOPO_OTERM_0, TOPO_OTERM_1, and TOPO_OTERM_2.

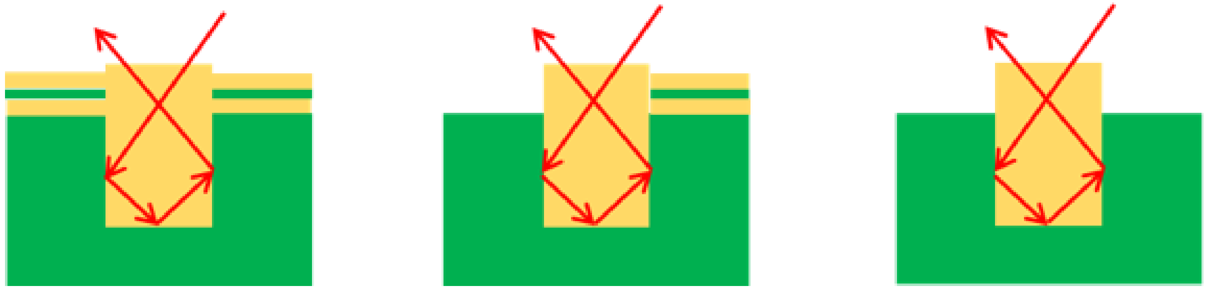
- **Silicon on Oxide (SOI)** — A film configuration where a thin layer of silicon is sandwiched between oxide. The SOI stack on the right needs to be simulated separately from an oxide stack.

Figure 1-9. Silicon on Oxide (SOI)



In Figure 1-9 the regular (active) silicon stack is denoted as "HYBRID". At the boundary between the SOI stack and the oxide stackm is a silicon step that is a source of sidewall signal. This signal is similar to the sidewall signal that appears at the boundary between a HYBRID (active) stack and the oxide stack.

Figure 1-10. Variants of Trenches Formed With SOI/Hybrid/Oxide Stacks



To simulate topographical effects caused by an SOI stack, use the underlying layer keyword "active2." It has the same properties as a regular underlying layer (active) except that the FinFET layer cannot intersect the active2 layer in the layout.

The underlying layer must be specified in the setup file as "underlying active2". There are two topo signals that are used with active2:

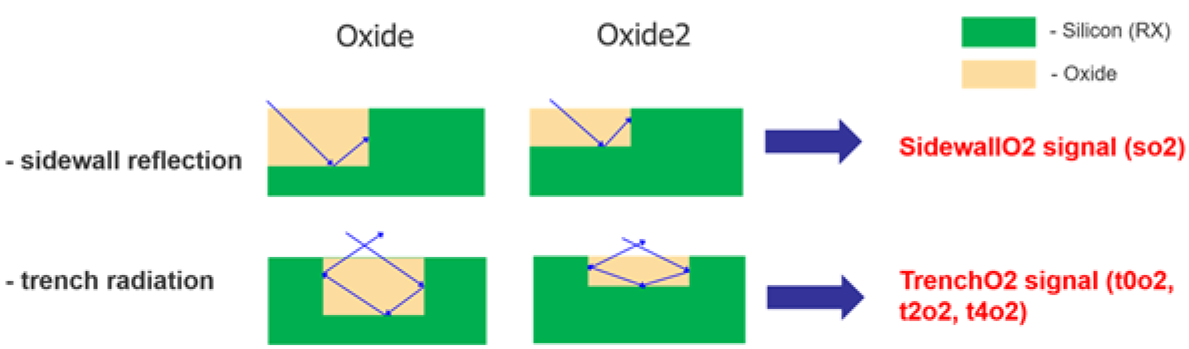
- SOI corner (corner2) diffraction signal (c2-signal), which is represented by c2-terms.
- SOI sidewall (sidewall2) reflection signal (s2-signal), which is represented by s2-terms.

There are no special trench signals required for SOI stacks. Trenches formed with any possible configurations presented in Figure 1-10 are simulated by standard trench signals: t-signal, t2-signal, and t4-signal. These signals are represented in the topo model by t-terms, t2-terms, and t4-terms correspondingly.

- **oxide2** (Ox2) — An additional shallow trench isolation (STI) stack that can be simulated separately. It must be used with a regular oxide stack, which is the corresponding normal deep trench isolation (DTI).

  Declare an oxide2 layer in an underlying statement as 'underlying oxide2'.

Figure 1-11. Oxide and Oxide2 Comparison



There are four signals associated with oxide2:

- oxide2 sidewall (sidewallo2) reflection signal (so2-signal), which is represented by so2-terms.

    - oxide2 trench0 (trencho2) radiation signal (to2-signal), which is represented by to2-terms.
    - oxide2 trench2 (trench2o2) radiation signal (t2o2-signal), which is represented by t2o2-terms.
    - oxide2 trench4 (trench4o2) radiation signal (t4o2-signal), which is represented by t4o2-terms.

- **Inner and Outer Loop** — Refers to the two-phase approach that topo model calibration uses to find the best model when the default fasttopo 2 mode is activated.

    - The inner loop calibrates a set of specified signals for each iteration of the outer loop.

      Linear and quadratic signal terms (sterm, cterm, tterm, and so on) are calibrated in the inner loop, based on the number of terms that are present in the topo model or topo modelform. The only inner loop parameters allowed for signals are ranges for each term.

    - The outer optimization loop calibrates more non-linear and non-quadratic topo parameters, such as oxide bias parameters, sigma-parameters, and upsize parameters.

  The number of outer loop iterations is controlled by the "topoiterations" parameter; the inner loop limits are controlled by the individual linear term parameters "topo_<*x*>termrange" and "topolimits_quadratic".

- **Implicit and Explicit Parameters** — The method of selecting linear and quadratic signal term parameters during the topo calibration stage.

    - **Implicit Parameters** — (Fasttopo mode set to 2) Topo parameters determined using a modelform. The signal terms to be calibrated are automatically generated based on a loaded topo model or selected modelform in Calibre nmModelflow. Linear and quadratic parameters are optimized implicitly in the inner loop.

    - **Explicit Parameters** — (Fasttopo mode set to 0 or 1) Topo parameters that name specific signals. Each signal term to be calibrated is listed separately as a calibration parameter named TOPO_*x*TERM_*N* in the Calibre nmModelflow Stage wizard. Linear and quadratic parameters are thusly optimized explicitly in the outer loop.

  Explicit and implicit parameters are mutually exclusive options.

Topo Model Development Flow in Calibre nmModelflow

Implementing a topographical model in Calibre WORKbench nmModelflow uses a specific development flow.

Note
The instructions in this manual assume a general knowledge of the concepts and tasks described in the *Calibre nmModelflow User's and Reference Manual*.

Table 1-1. Topographical Modeling Flow (Calibre nmModelflow)

Step	Task	Described In
1	Create a litho model and add required model files and configuration information.	"Creating Litho Models for Topography Modeling" on page 22
2	Add gauge files to the database.	Described in the *Calibre nmModelflow User's and Reference Manual*
3	Add stages to the database for each model in the litho model.	"Creating Calibration Stages for Topo Models" on page 33
4	Set up calibration jobs for the stages and run them.	"Creating and Running Calibration Jobs" on page 46
5	Set up verification jobs for the stages and run them.	"Creating and Running Calibration Jobs" on page 46
6	Use a calibrated model in Calibre nmOPC for simulation.	"Using the Topographical Model in Calibre nmOPC" on page 67
7	Use a calibrated model in Calibre OPCverify for design verification.	"Using the Topographical Models in Calibre OPCverify" on page 68

Syntax Conventions

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

Table 1-2. Syntax Conventions

Convention	Description
Bold	Bold fonts indicate a required item.
Italic	Italic fonts indicate a user-supplied argument.
`Monospace`	Monospace fonts indicate a shell command, line of code, or URL. A bold monospace font identifies text you enter.
<u>Underline</u>	Underlining indicates either the default argument or the default value of an argument.
UPPercase	For certain case-insensitive commands, uppercase indicates the minimum keyword characters. In most cases, you may omit the lowercase letters and abbreviate the keyword.

Table 1-2. Syntax Conventions (cont.)

<table>
<tr><th>Convention</th><th>Description</th></tr>
<tr><td>[ ]</td><td>Brackets enclose optional arguments. Do not include the brackets when entering the command unless they are quoted.</td></tr>
<tr><td>{ }</td><td>Braces enclose arguments to show grouping. Do not include the braces when entering the command unless they are quoted.</td></tr>
<tr><td>‘ ’</td><td>Quotes enclose metacharacters that are to be entered literally. Do not include single quotes when entering braces or brackets in a command.</td></tr>
<tr><td>| or ||</td><td>Vertical bars indicate a choice between items. Do not include the bars when entering the command.</td></tr>
<tr><td>…</td><td>Three dots (an ellipsis) follows an argument or group of arguments that may appear more than once. Do not include the ellipsis when entering the command.</td></tr>
<tr><td colspan="2">Example:
DEVice {<i>element_name</i> [‘(’<i>model_name</i>‘)’]}
<i>device_layer</i> {<i>pin_layer</i> [‘(’<i>pin_name</i>‘)’] …}
[‘<’<i>auxiliary_layer</i>‘>’ …]
[‘(’<i>swap_list</i>‘)’ …]
[<u>BY NET</u> | BY SHAPE]</td></tr>
</table>

Creating Litho Models for Topography Modeling

Litho models simplify access to a collection of process models. The Calibre nmModelflow tool requires litho models for simulation and calibration runs. The tool contains a Litho Model Creation wizard to simplify the process regardless of whether you have existing models or not.

Table 1-3. Models Used for Topography Modeling

Model Type	Description
silicon optical	Required. Describes the resist over silicon stack.
oxide optical	Required. Describes the resist over oxide stack.
FinFET optical	Optional. Replaces silicon optical if used. Describes two film stacks: a resist over silicon stack, and resist over oxide stack.
Post-PC (poly) transitional	Optional. Used for multiple density configurations. Describes the poly over resist over silicon stack.
topo	Required. Describes a topo model.
resist	Required. Describes a resist model.

Creating a Topo-Enabled Litho Model (Part 1: Resist and Topo)

The first part of creating a topo-enabled litho model in Calibre nmModelflow involves setting the output directory and adding the resist and topo models.

Prerequisites

- Calibre WORKbench invoked from a working directory and the Calibre nmModelflow window open.
- A resist model matching your resist configuration is recommended.
- A topo model matching your topo configuration is recommended.

Procedure

1. In Calibre nmModelflow, select **File > Create Lithomodel** to open the Lithomodel Creation Wizard. The wizard lists each step that requires input (Output Lithomodel Directory, Lithomodel Components, Exposure, and Finish).

2. In the Output Lithomodel Directory page, enter or navigate to an empty directory to contain the completed litho model. (You can create a new directory from the file chooser dialog.)

3. In the Lithomodel Components page, select options as follows:

   - **Resist Model or Resist Model Form** — If you have a resist model, choose Resist Model and fill in the directory path to the model. Choose Resist Model Form if you do not have a resist model. If you choose a Resist Model Form, Siemens EDA suggests using Modelform 21 to start.

     Do not select the 3D Resist checkbox. 3D Resist is not used for topo models.

   - **Etch Model or Etch Model Form** — Leave the setting at Etch Model and do not fill in an etch model file; etch models are not used for topo modeling.

   - **Topo Model or Topo Model Form** — If you have a topo model available, choose Topo Model and fill in the directory path to your topo model. Choose Topo Model Form if you do not have a topo model.

     Note

     For post-PC configurations, you must pick a Topo Model Form that contains the prefix "topo3" in order to be prompted for the third optical model, or import a topo model that contains the keyword "transit 0 1 2".

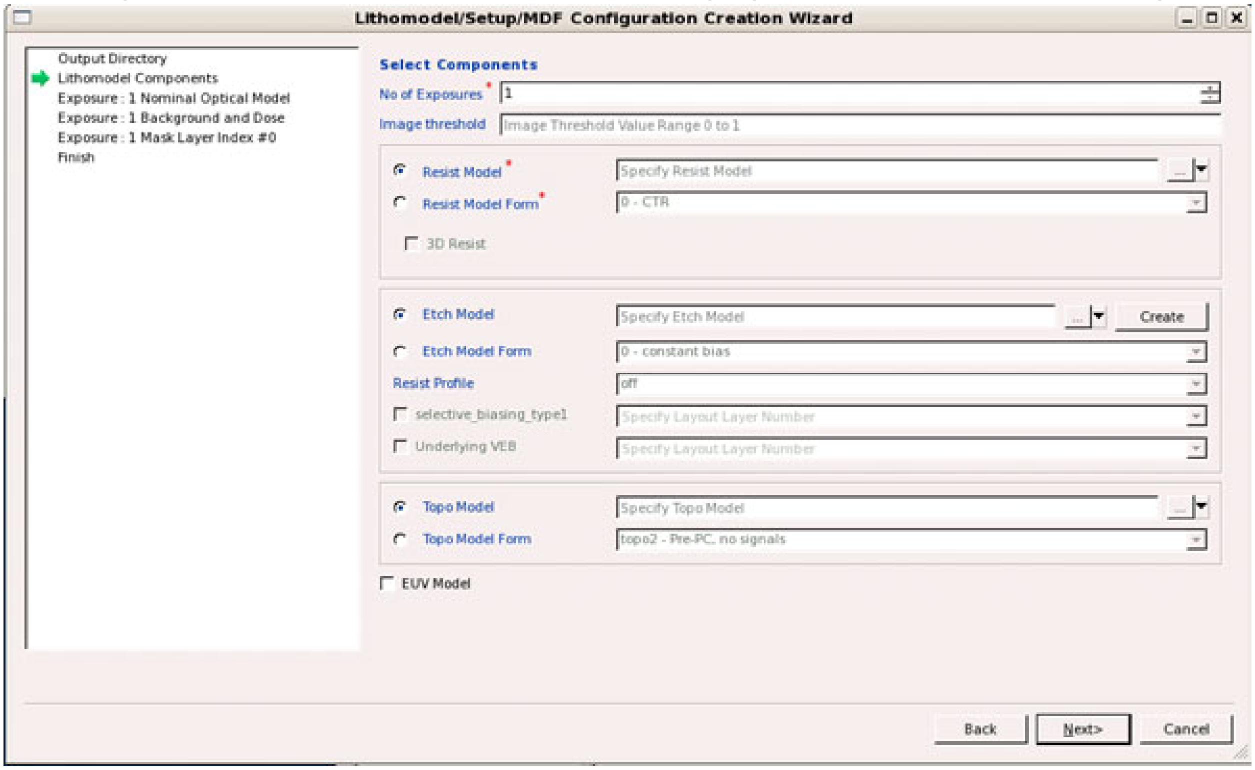
     If you pick Topo Model Form, it is helpful if you pick a modelform that is similar to your configuration; the wizard adjusts future prompts based on your choice. For example, picking a Topo Model Form of "topo2STF" creates an additional Exposure entry and a topo model that contains S-terms, T-terms, and F-terms.

     Note

     Starting with the 2017.4 release, you can reuse litho models that contain Topo Modelforms with extra signals, as long as you deactivate the extra signals with the "mdf param topo_<*signal*>termnum 0" command before running a calibration job with those litho models.

   - **EUV Model** — Do not select the EUV Model checkbox.

Figure 1-12. Lithomodel Components Page (Resist and Topo Selection)



4. Click **Next** to continue, and go to the next task.

Editing an Existing Topo Model

Topo modelforms in Calibre nmModelflow use templates, which come with preset signals and parameters that you can edit in the interface.

Use this task when you need a customized topo model, such as one having additional signals or non-sequential signals.

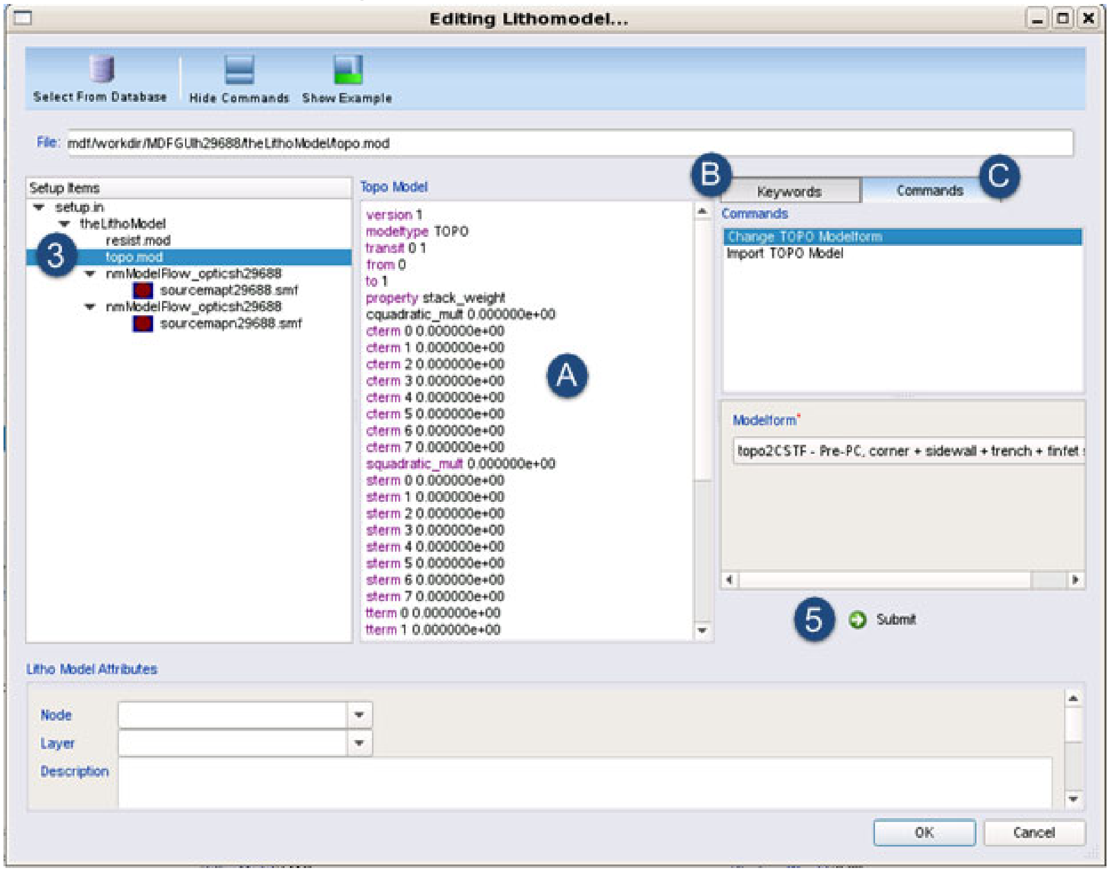
Prerequisites

- Calibre nmModelflow invoked
- An active litho model that contains topo layer information

Procedure

1. In the Calibre nmModelflow Database Browser, select a Litho Model from the list with a litho model.
2. Click **Lithomodel Editor**.
3. Click on the topo model name (the default filename is *topo.mod* if you used the Lithomodel Creation tool) to open the topo model in the editor.

Figure 1-13. Lithomodel Editor



4. Modify your topo model specification using any of the following methods:

- Modify the model text directly in the center pane (A in the figure).
- Use the **Keywords** tab (B in the figure), which contains a list of individual topo model parameters. Clicking on a parameter adds them to the displayed topo model.
- Use the **Commands** tab (C in the figure), which contains macros that let you change the current topo modelform or import a new topo model file. Both of these actions replace the existing topo model.

Note

A full list of modelforms can be found in "Topographical Modelform Parameters (topomf)" on page 102.

- Use the Command Window, found at the bottom left corner of the Calibre nmModelflow window. For example, you can change the total number of signals using one or more mdf param topo_*signal*termnum *number* commands. For example, the following commands modify the base CST model to be CS with only C-terms 0 and 1:

mdf param topo_ctermnum 2

mdf param topo_ttermnum 0

- Each mdf param command should be on its own line.
- The most common custom command is to change the default number of terms for a signal. The default is 8, which generates terms 0 to 7. Specifying 0 removes the signal from the model.

Note

You can also change the number of terms per signal in the Flow Stage Wizard.

5. Click **Submit** when you are finished making changes, then click **OK** to close the window.

Results

The edited model is saved to the database. You can confirm your changes by selecting the edited model and then clicking **View** again.

Tip

You can also use the Command Window to inspect the active litho model.

- mdf gui params topo modelinfo nominal — Shows information about the active topo model.
- mdf gui params topo nominal — Shows information about the stage parameters associated with the active topo model.

Topo Model Examples

Topography models contain mostly parameters based on the signal information you are interested in modeling.

Topo Model Concepts

- All linear and quadratic parameters specified in the initial topo model are optimized.
- A topo model can be created using a modelform or manually. They are commonly referenced by their modelform codes.

Example 1: topo3CSTGIF

A post-PC topo model that contains corner, sidewall, trench, gate, interconnect, and FinFET signals.

```
transit 0 1 2 3
density finfet active poly oxide
from 0
to 1
property stack_weight
transit_dose 1.0 0.9 1.1 1.0
linearity intensity
dsigma 1.4969
dupsize 0.0
cterm 0 0.1
cterm 1 -0.05
sterm 0 -0.1
sterm 1 0.05
tterm 0 -0.1
tterm 1 0.05
gterm 0 -0.1
gterm 1 0.02
iterm 0 0.1
iterm 1 -0.12
fterm 0 0.15
fterm 1 -0.07
```

Creating a Topo-Enabled Litho Model (Part 2: Optical)

Topo model calibration typically requires at least two optical models, and may use more optical models depending on your configuration.

Prerequisites

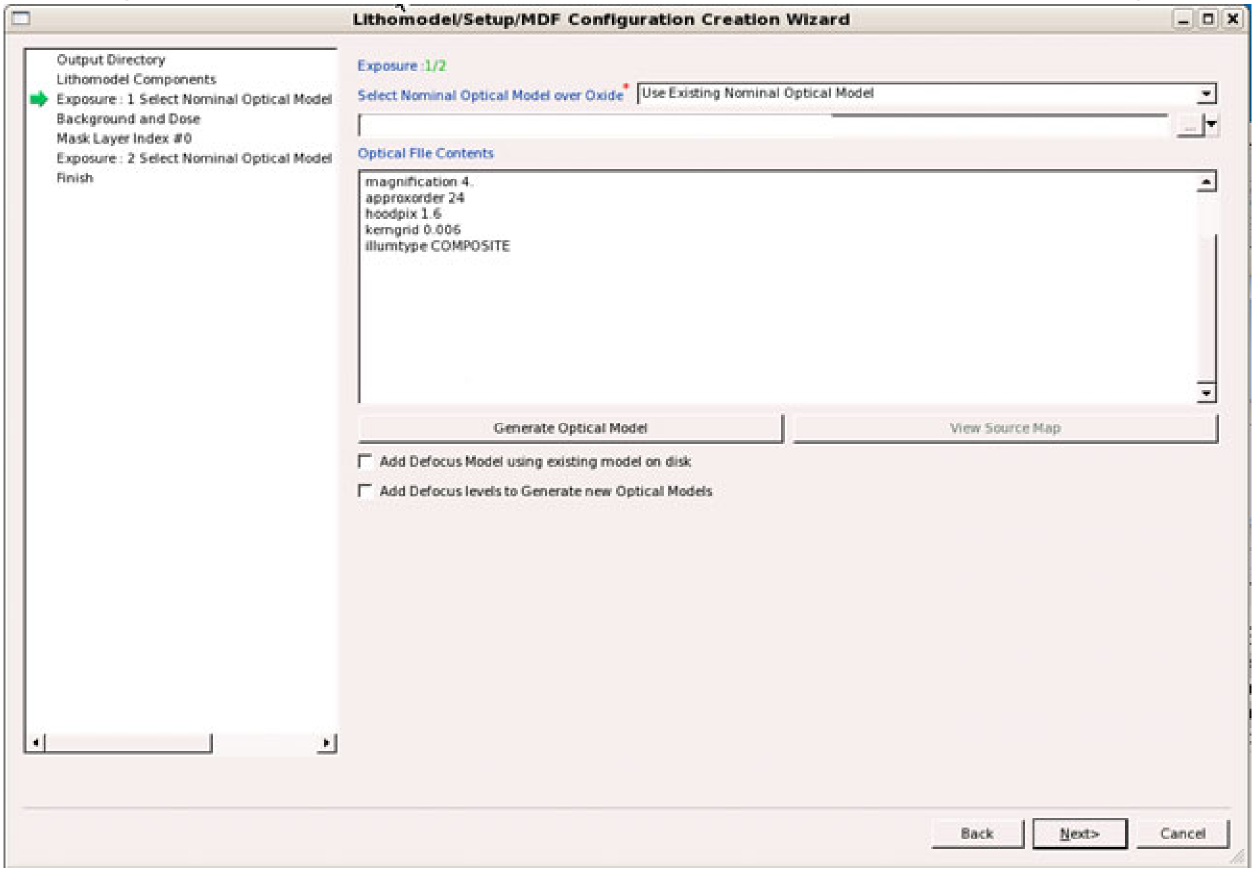
- Completed "Creating a Topo-Enabled Litho Model (Part 1: Resist and Topo)" on page 22.
- An understanding of how many optical models your topo configuration uses. At least two optical models are used for topo model calibration.

Procedure

1. In the Exposure 1: Select Nominal Optical Model page, set the oxide optical model.
    - If you have an optical model file already generated, select "Use Existing Nominal Optical Model" and select it using the file navigator.
    - If you have only the Optical file available without the generated files, select "Generate Using Existing Optical File," then navigate to the *Optical* file.
    - Experienced users who are familiar with the Optical Model File Format can select "Generate Using New Optical File" and type in the active text field.
    - If you do not have an optical model to load, select "Generate Using Optical Model Tool" and fill in the tabs using the GUI that appears.

    The optical model appears in the viewing area.

Figure 1-14. Lithomodel Creation Wizard, Optical Model Selection Page



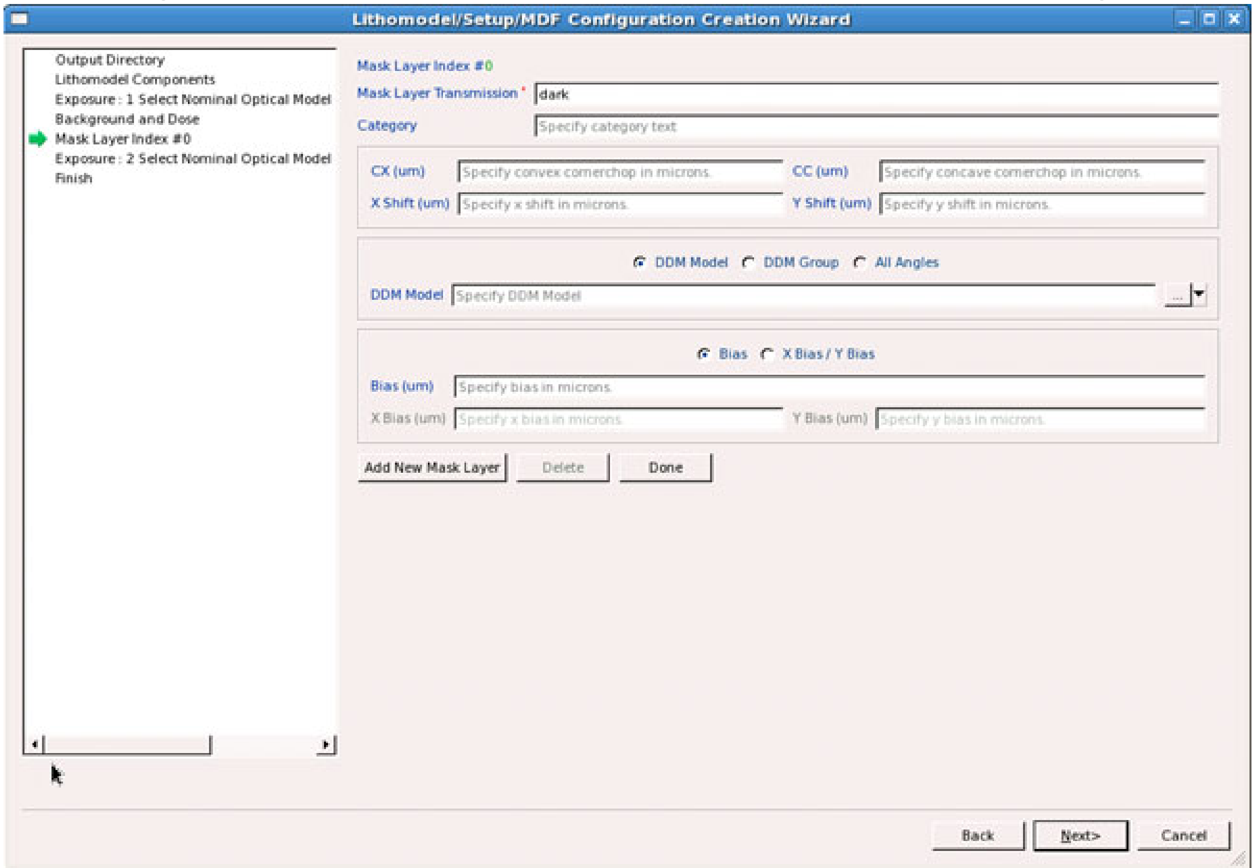
2. In the Background and Dose page, accept the default values unless you know your background and dose are different.

3. In the Mask Layer Index page, accept the default values unless your mask configuration has additional features such as cornerchop bias.

Note

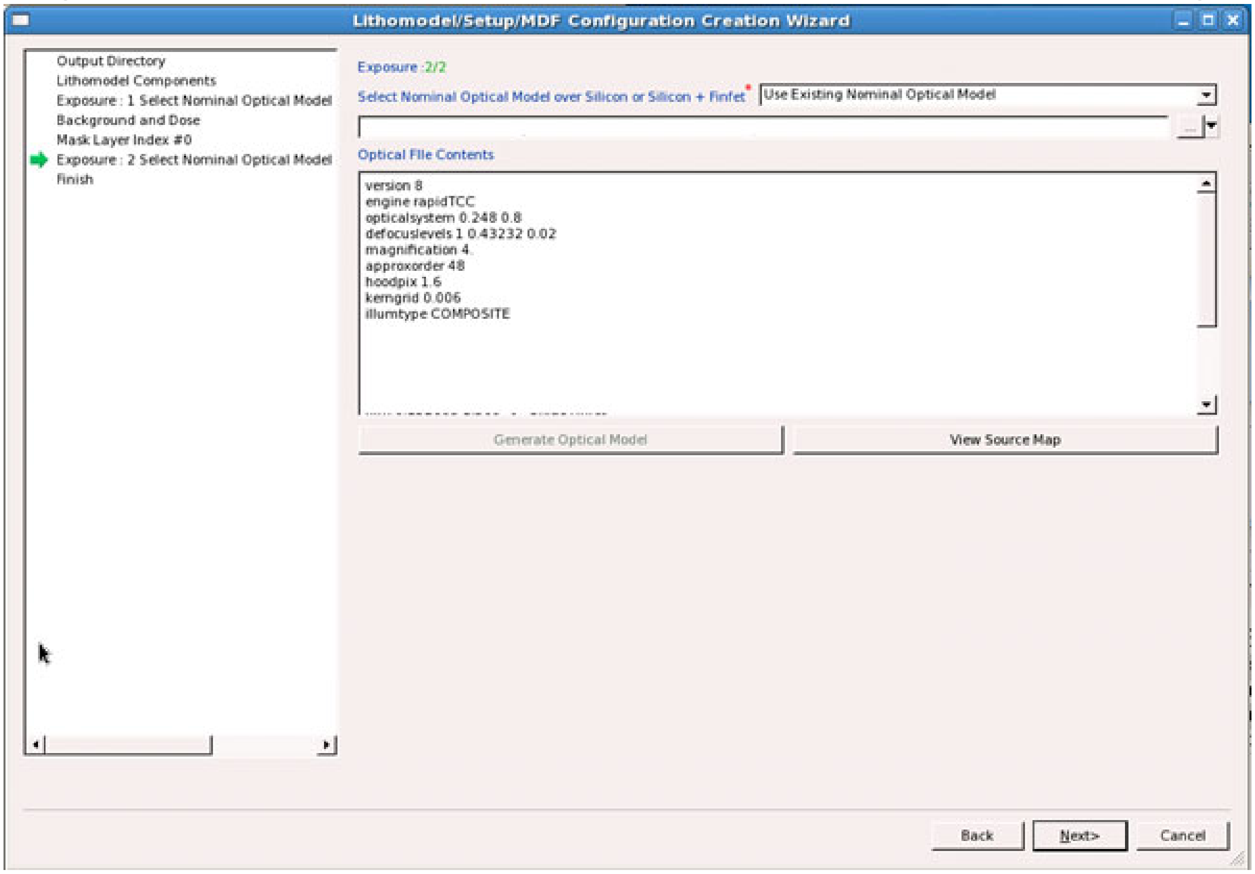
DDM models are not used in topo modeling.

Figure 1-15. Lithomodel Creation Wizard, Mask Specification Page



4. In the Exposure 2: Select Nominal Optical Model page, select the silicon optical or silicon and FinFET model using methods similar to Step 1.

Figure 1-16. Lithomodel Creation Wizard, Second Optical Model Selection Page



5. If you loaded or selected a topo model with three exposures in the task "Creating a Topo-Enabled Litho Model (Part 1: Resist and Topo)" on page 22, load or define the model for the transitional optical model, similar to "In the Exposure 1: Select Nominal Optical Model page, set the oxide optical model." on page 27.

6. On the Finish page, ensure that Activate Resulting Lithomodel and Import Lithomodel to Database are both selected, then click **Finish**.

Results

The new litho model is added to the Calibre nmModelflow database.

Topo Optical Model Examples

Optical models used with topo models are primarily differentiated by their film, film2, and film3 keywords. These film keywords define the stack specific to that optical model exposure.

Example Concepts

Multiple optical models are used for topo calibration. The number of models is based on the nature of your design. Each optical model is designed to be used over a specific part of the film stack, and will differ from similar optical models accordingly.

Example: Single Oxide Film Stack

A typical optical model with a single oxide film stack would look similar to the following:

```
version 10
engine rapidTCC
opticalsystem 0.248 0.8
...
STANDARD SHARP weight sigma 0.5
imagediffusion 0.0
film 0.2 1.4 -0 "200 nm Resist"
film 0.16 1.2 -0 "Oxide"
substrate 1.7 -2.4
...
```

Example: Single Silicon Film Stack

A typical optical model with a single silicon film stack would look similar to the following:

```
version 10
engine rapidTCC
opticalsystem 0.248 0.8
...
STANDARD SHARP weight sigma 0.5
imagediffusion 0.0
film 0.2 1.4 -0 "200 nm Resist"
film 0.25 1.577 -3.2 "Silicon"
substrate 1.577 -3.2
...
```

Example: FinFET Film Stack

A typical optical model used for FinFET calibration. In this case, two film stacks are used.

- Set film to describe the FinFET stack component (resist, oxide, then oxide FinFET).
- Set film2 to describe the oxide stack component (resist, then oxide).

- The stack_weight keyword is only used when more than one film stack is present in the optical model, as it is in the FinFET example following:

```
version 10
engine rapidTCC
opticalsystem 0.248 0.8
...
STANDARD SHARP weight sigma 0.5
imagediffusion 0.0
film 0.2 1.4 -0 "200 nm Resist"
film 0.16 1.2 -0 "Oxide"
film 0.14 1.6 -0 "Oxide FinFet"
film2 0.2 1.4 -0 "200 nm Resist"
film2 0.16 1.2 -0 "Oxide"
stack_weight 0.5
substrate 1.7 -2.4
```

- Some situations benefit from a transitional third film stack. In this case, set film to describe the FinFET oxide stack, film2 to describe the FinFET stack, and film3 to describe the bulk oxide stack. Use stack_weight and stack2_weight to control relative weighting. (See "film2, film3" on page 74" and "stack_weight, stack2_weight" on page 85" for details.)

Example: Poly (Post-PC) Film Stack

A typical optical model used for multiple-density post-PC configurations. A single film stack is used to describe the configuration.

```
version 8
engine rapidTCC
opticalsystem 0.248 0.75
defocuslevels 1 0.633467 0.02
magnification 4.0
approxorder 4
hoodpix 1.280
kerngrid 0.005
illumtype COMPOSITE
STANDARD SHARP weight 1 sigma 0.799853
imagediffusion 0.050048
film 0.76 1.77 -0.01 "760 nm Resist"
film 0.053 2.221 -0.043 "SiN"
film 0.055 1.69 -2.76 "Poly"
film 0.23 1.505 -0 "Oxide"
substrate 1.57 -3.565
beamfocus 0.300001
vectormodelflag 1
```

Creating Calibration Stages for Topo Models

Topo models are calibrated using multiple individual stages in Calibre nmModelflow.

Creating an Optical Model Calibration Stage

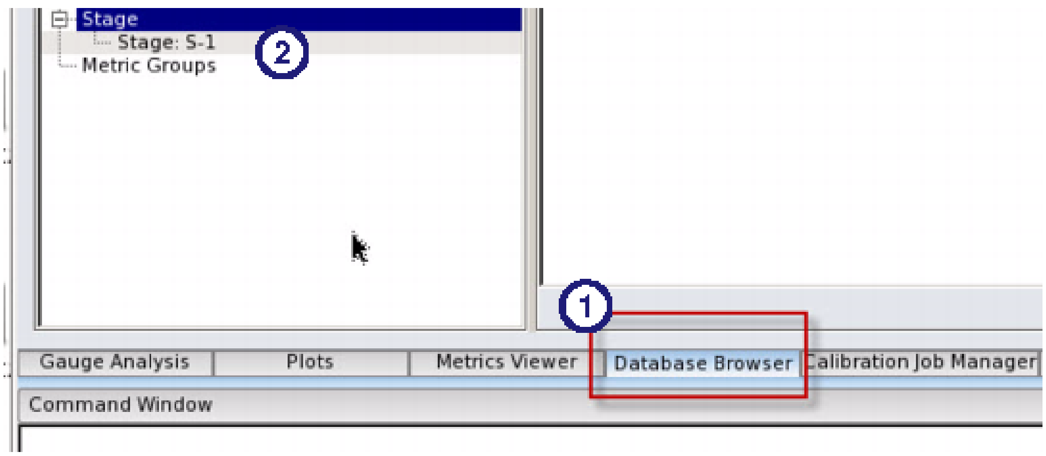
The optical model is calibrated as a separate stage from the topo and resist models. Optical models are calibrated as a set.

Prerequisites

- Calibre WORKbench and Calibre nmModelflow invoked.

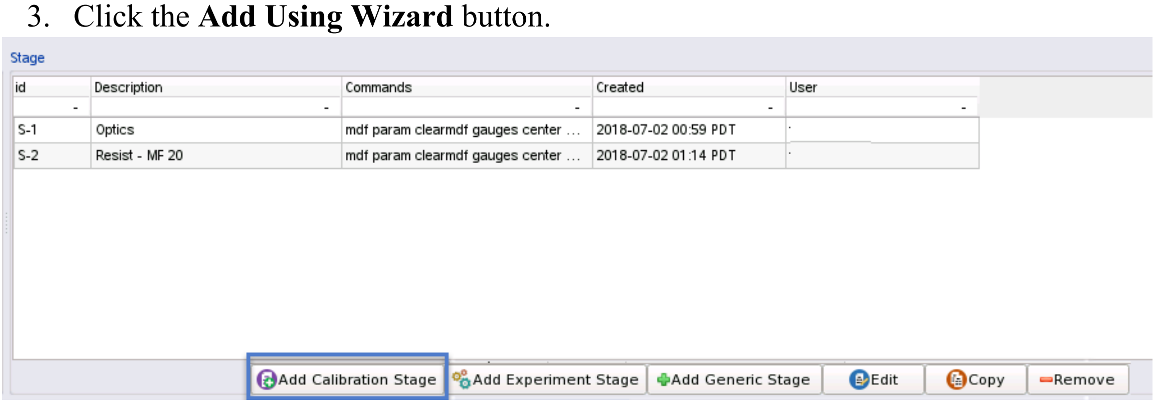
Procedure

1. Raise the **Database Browser** tab in the list of primary display tabs.

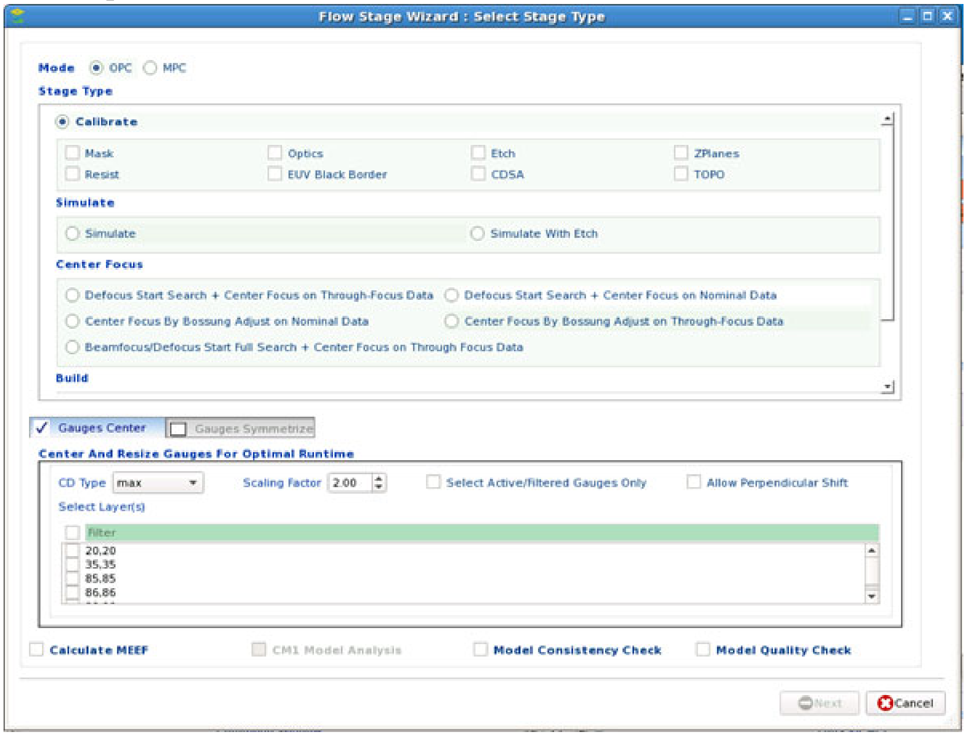


2. Click **Stage** in the left column.

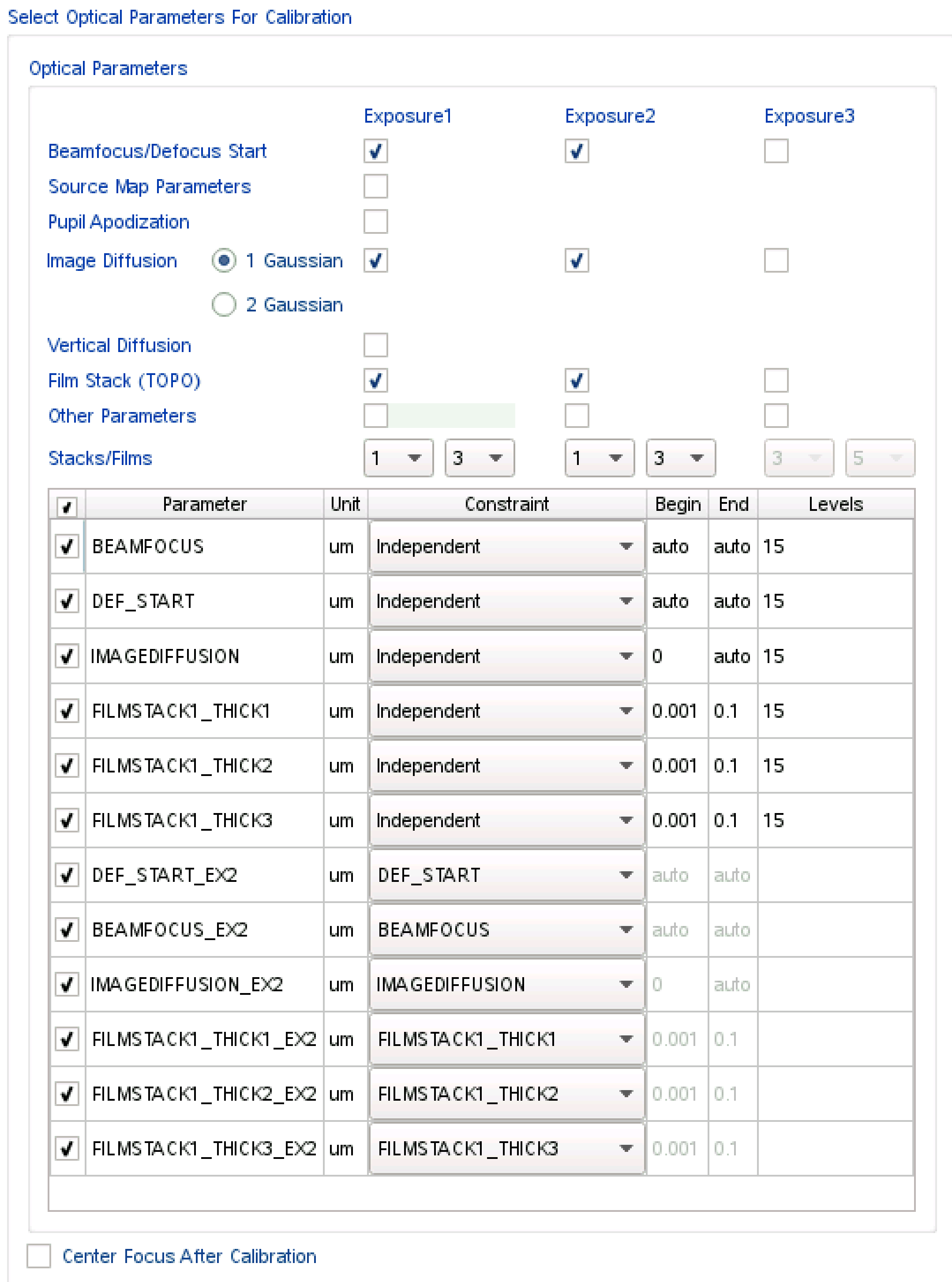
3. Click the **Add Using Wizard** button.



4. Choose Optics, then click **Next**.



The Select Optical Parameters page appears, similar to the following figure.



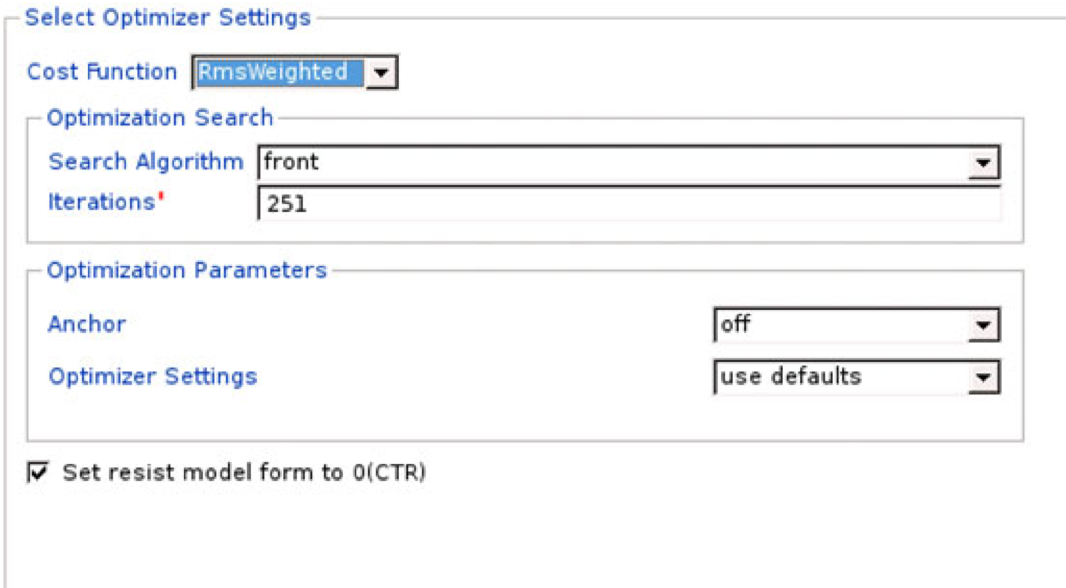
5. Select parameters using the following guidelines for a pre-PC (two optical model) configuration, then click **Next** to continue:

Table 1-4. Optical Parameter Calibration Settings (pre-PC)

Optical Parameter	Description
Beamfocus / Defocus Start checkbox	Select both Exposure 1 and Exposure 2 to enable the BEAMFOCUS, DEF_START, DEF_START_EX2, and BEAMFOCUS_EX2 parameters.
Image Diffusion selection set	Select both Exposure 1 and Exposure 2 to enable the IMAGEDIFFUSION and IMAGEDIFFUSION_EX2 parameters.
Film Stack (TOPO)	Select both Exposure 1 and Exposure 2 to enable the Stacks/Films controls and the STACK_WEIGHT_EX*N* and FILMSTACK*S*_THICK*F*_EX*N* parameters.
Stacks/Films	Set the stacks and films corresponding to the number of each item in the relevant optical model. For example, a typical oxide exposure would use Stacks of 1 and Films of 2; a typical FinFET would use Stacks of 2 and Films of 3.
BEAMFOCUS	Select the checkbox to optimize this parameter.
DEF_START	Select the checkbox to optimize this parameter.
IMAGEDIFFUSION	Select the checkbox to optimize this parameter. Change the defaults to 0 0.5 5.
DEF_START_EX2	Select the checkbox to optimize this parameter and set the constraint to DEF_START.
BEAMFOCUS_EX2	Select the checkbox to optimize this parameter and set it to BEAMFOCUS.
IMAGEDIFFUSION_EX2	Select the checkbox to optimize this parameter and set the constraint to IMAGEDIFFUSION.

.

6. On the Select Optimizer Settings page, examine the current settings and change them if needed. In most cases, the default settings are appropriate; however, a best practice is to set Iterations to around 50 to 100 per variable optimized.



Click **Next** to reach the Finish screen with the list of commands for Calibre nmModelflow. Accept the settings to create the stage.

Results

The optical model calibration stage is added to the database.

Creating a Topo Model Calibration Stage

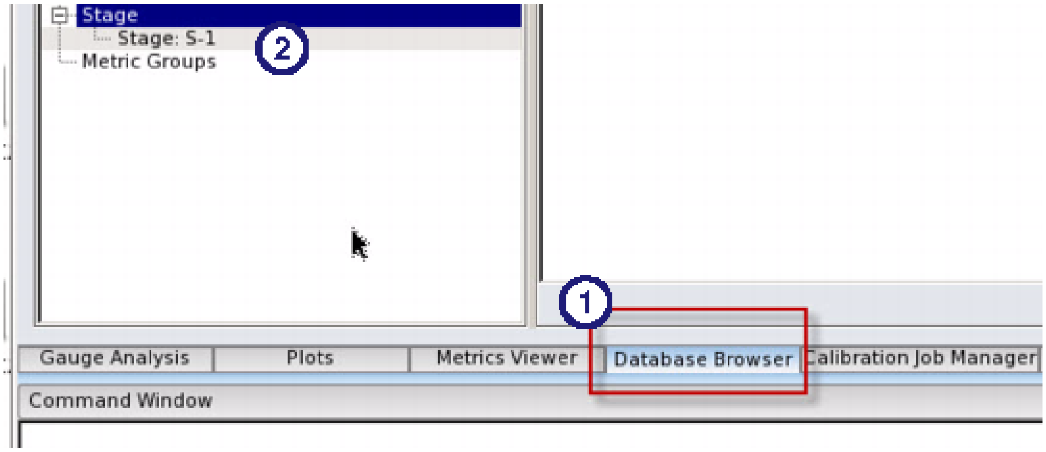
The topo model is calibrated as a separate stage from the optical and resist models.

Prerequisites

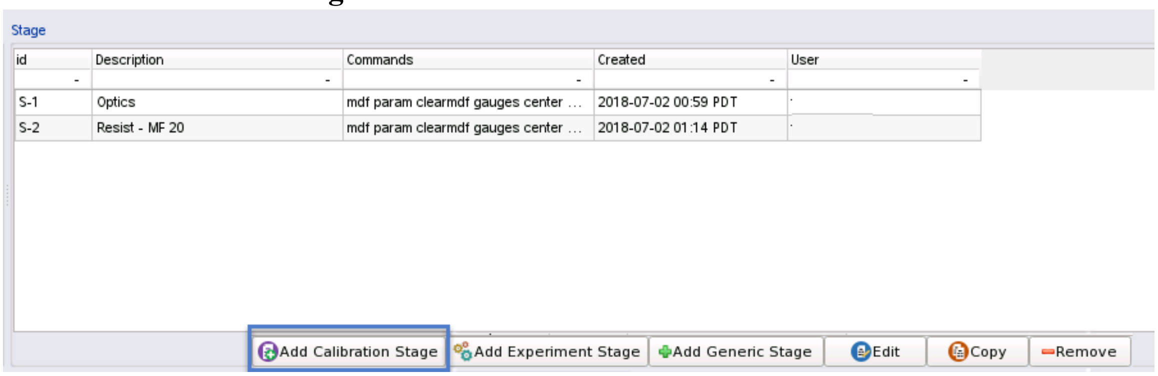
- A litho model containing a topo model.
- Calibre WORKbench and Calibre nmModelflow invoked.

Procedure

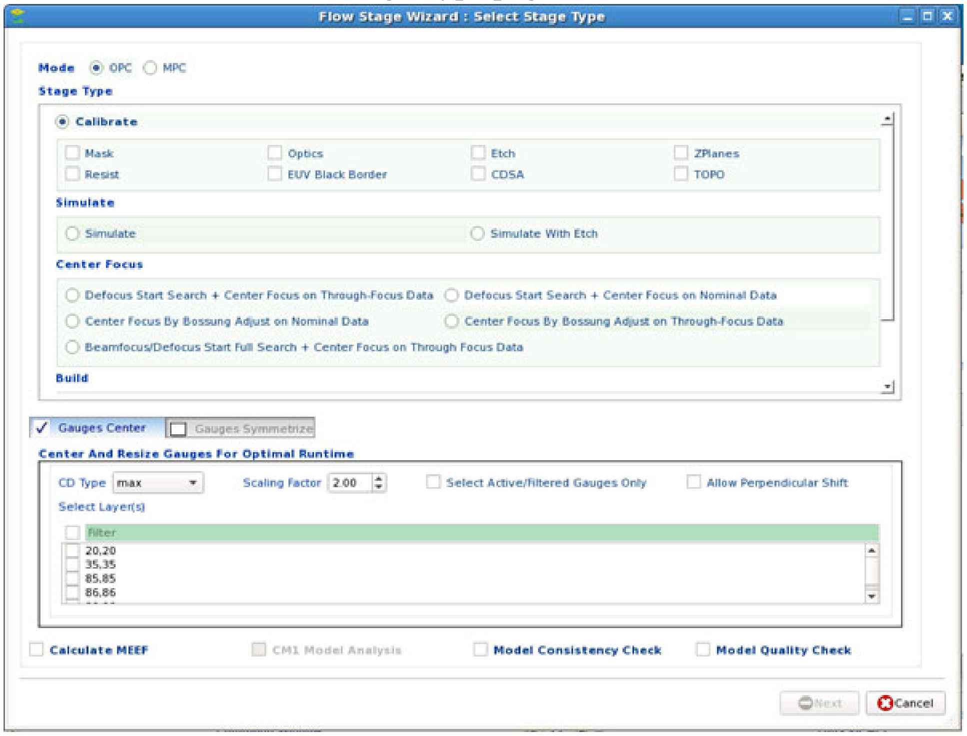
1. Raise the **Database Browser** tab in the list of primary display tabs.



2. Click **Stage** in the left column.
3. Click the **Add Using Wizard** button.



4. Choose TOPO in the Select Stage Type page, then click **Next**.



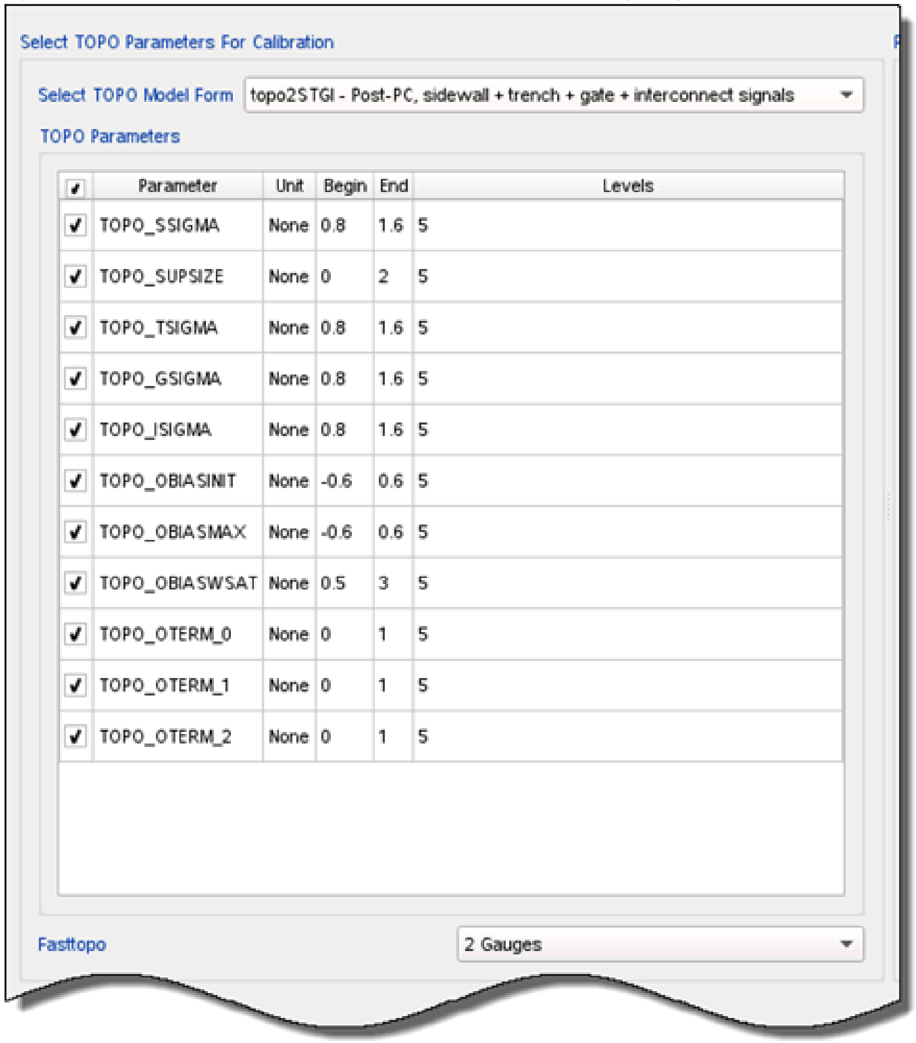
The Calibration page populates its parameters based on the modelform you select on the first page (Figure 1-17).

5. Use the Select Topo Model Form dropdown to ensure that your selection matches your topo model inside the litho model. In most cases, you can leave the default calibration parameters at the defaults; adjust parameters as needed and click **Next**.

Tip

Choosing "Active Input Lithomodel" uses the loaded litho model if you activated one, or if you created a custom litho model. (See "Editing an Existing Topo Model" on page 24.).

Figure 1-17. Topo Calibration Parameters (Page 1 - Outer Loop)

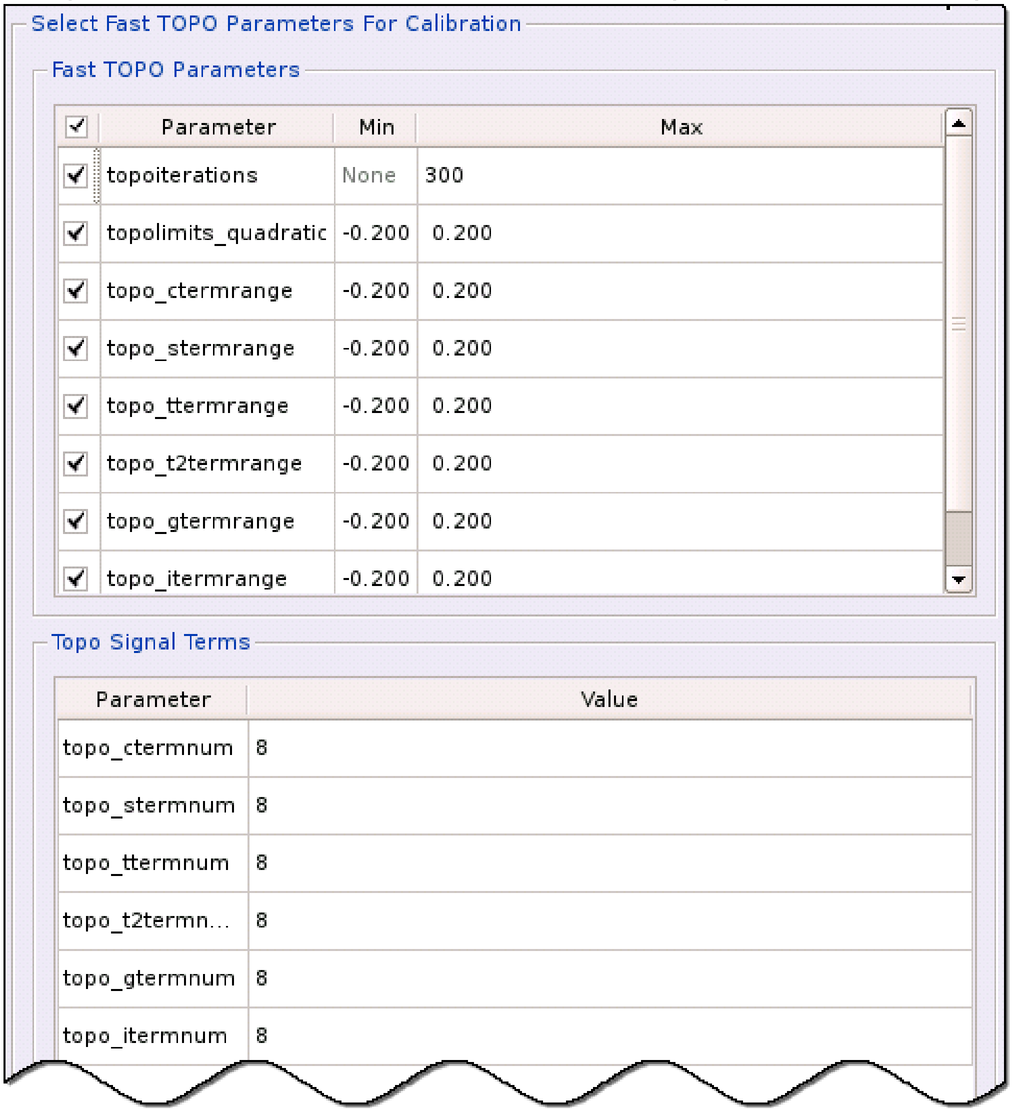


Note

On page 1, the oxide bias parameters (TOPO_OBIAS_INIT, TOPO_OBIASMAX, TOPO_OBIASWSAT, TOPO_OTERM_0, TOPO_OTERM_1, and TOPO_OTERM_2) always appear regardless of the TOPO Model Form you select, and should always be calibrated.

6. Use the controls on the second page (Figure 1-18) to control the iterations and signal term ranges for the inner loop.

Figure 1-18. Topo Calibration Parameters (Page 2 - Inner Loop)

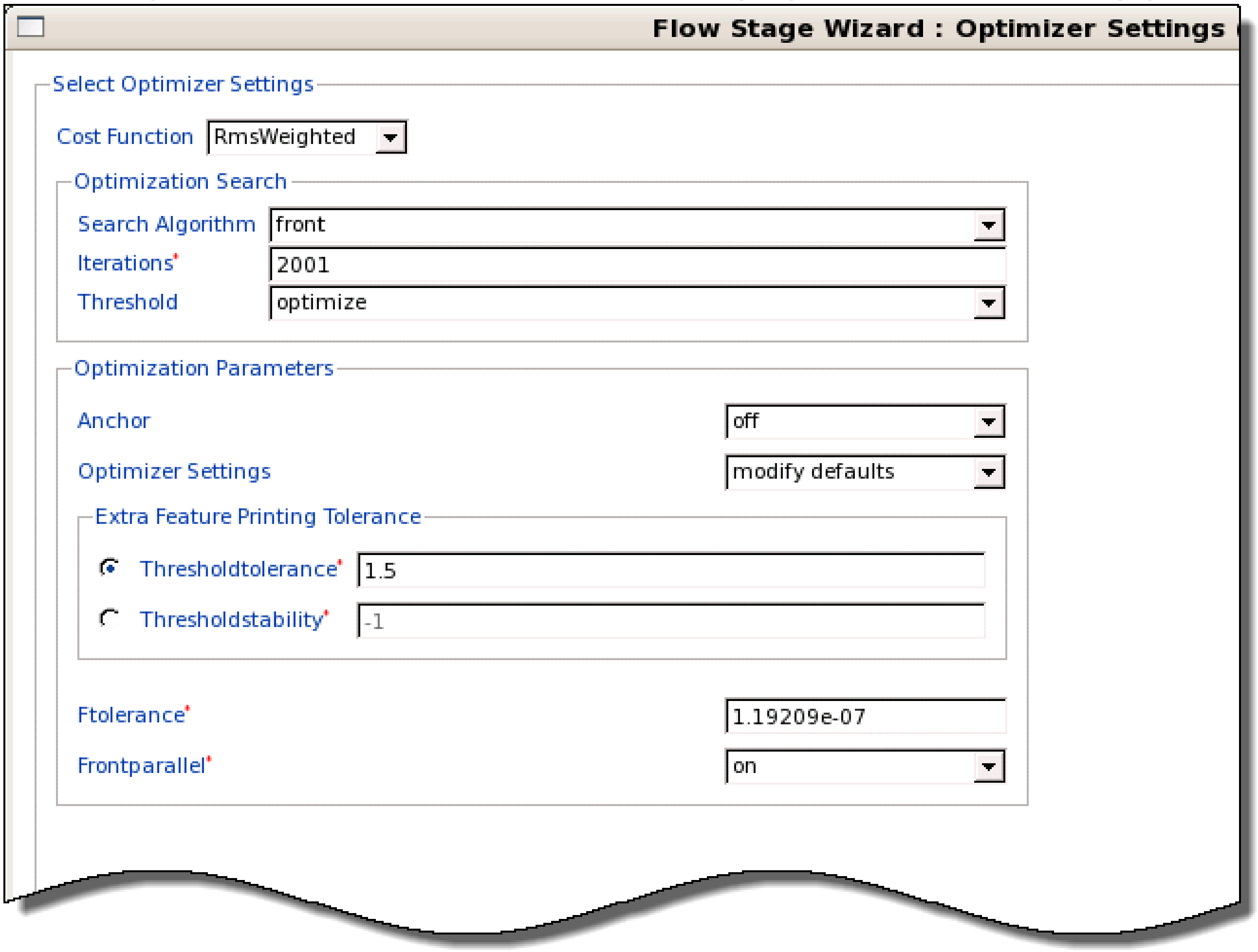


- The topoiterations parameter sets the number of search iterations performed on the inner loop for all topo_*x*termrange calibrations during each pass of the inner loop.
- The Topo Signal Terms section in the lower half of the page sets the number of terms for each active topo signal. By default, an active signal has 8 terms.

Adjust the parameters as needed, then click **Next**.

7. In the Optimizer Settings page, set the Search Algorithm and Iterations fields (Figure 1-19).

Figure 1-19. Topo Calibration Parameters (Page 3 - Optimizer Settings)



Click **Next** to reach the Finish screen with the list of commands for Calibre nmModelflow. Accept the settings to create the stage.

Results

The topo model calibration stage is added to the database.

Creating a Resist Model Calibration Stage

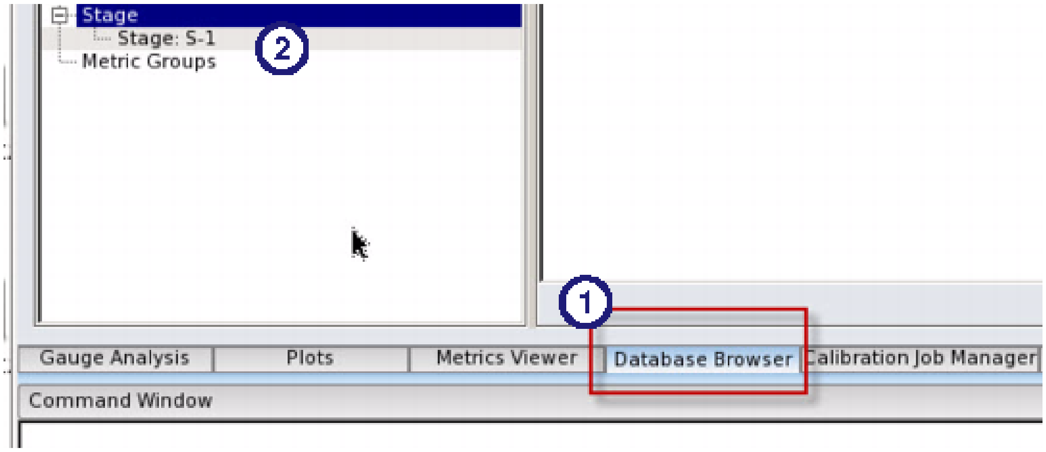
The resist model is calibrated separately from the optical and topo models.

Prerequisites

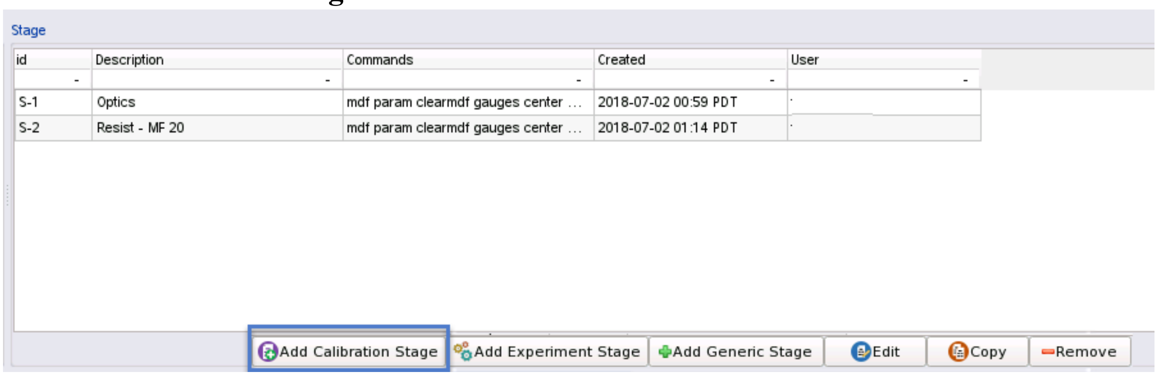
- Calibre WORKbench invoked, and the Calibre nmModelflow tool running

Procedure

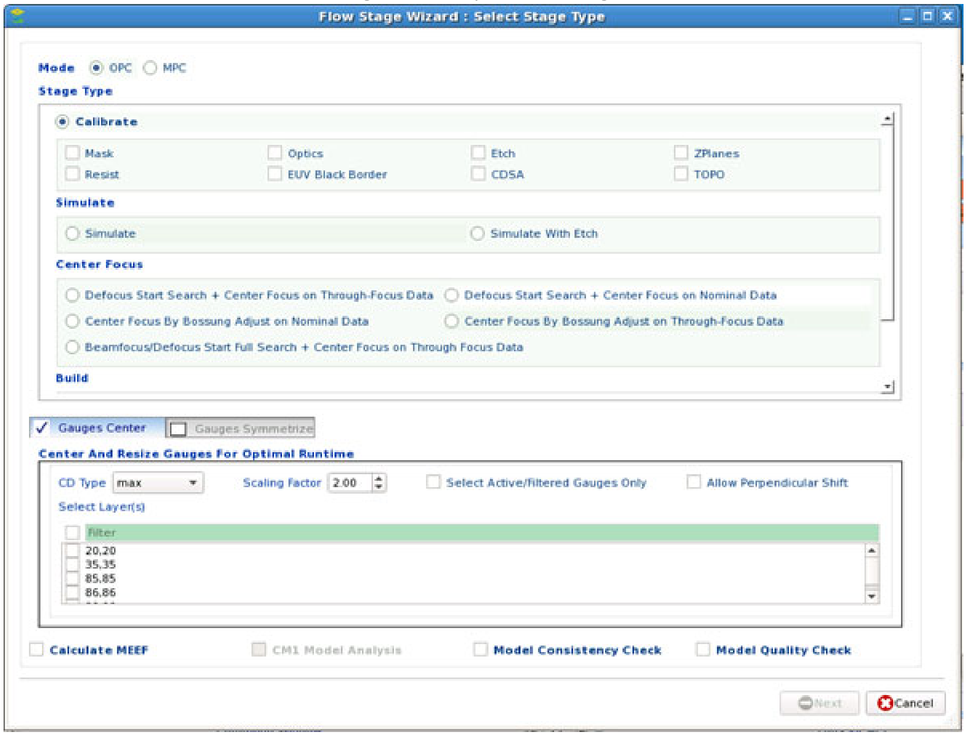
1. Raise the **Database Browser** tab in the list of primary display tabs.



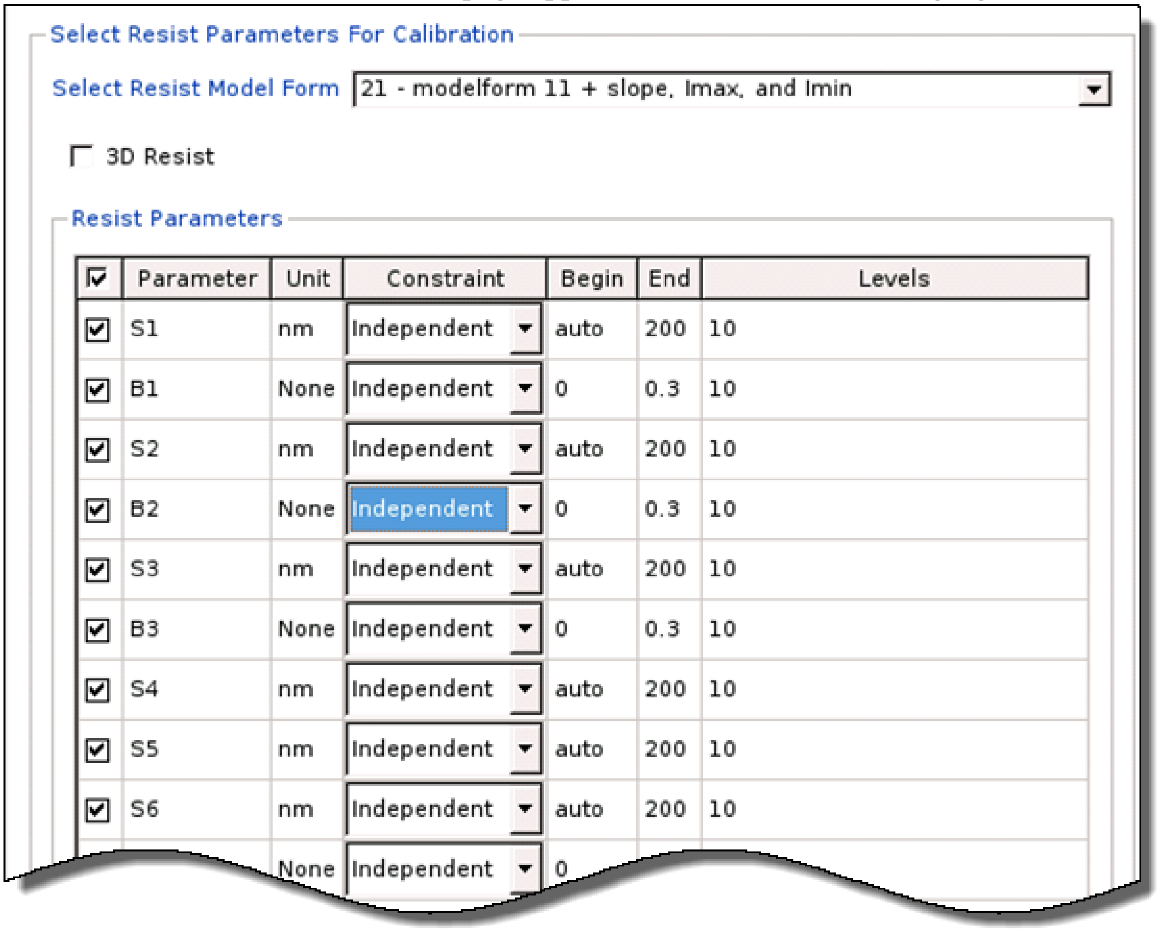
2. Click **Stage** in the left column.
3. Click the **Add Using Wizard** button.



4. Create a Resist calibration stage first by choosing Resist, then click **Next**.



The Select Resist Parameters page appears, similar to the following figure.



5. Select the modelform that matches the resist model in the litho model (usually modelform 21 or 22), adjust any calibration parameters as needed, then click **Next**.
6. Set search optimizer settings on the next screen as needed. In most cases, you should use the default settings.

   Click **Next**, then **Finish** on the final screen to confirm your selections.

Results

The resist model calibration stage is added to the database.

Creating and Running Calibration Jobs

Calibrating topo model stages is performed using calibration jobs. In the case of topo modeling, each calibration stage is run in a specific order, and the resulting model is used to calibrate another model.

Creating a Topo Calibration Job

Calibration jobs are the core of optimization for Calibre nmModelflow. You must set up a calibration job in order to generate output.

Prerequisites

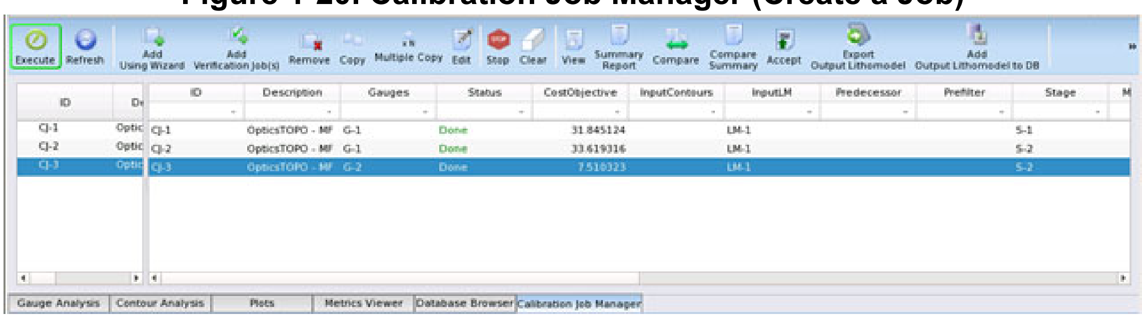
The database must contain each of the following items:

- A gauge object or contour object.
- A litho model.
- A stage.

Procedure

1. Raise the **Calibration Job Manager** tab in the main window.

Figure 1-20. Calibration Job Manager (Create a Job)



2. Click the **Add Using Wizard** button.
3. Fill out the wizard that appears. On the initial screen, if you have contours in your database, you can choose Contours (allowing you to skip the gauge selection steps) or Gauges and Contours (allowing you to use both gauge and contours during calibration). Otherwise, you must choose Gauges; click **Next** to continue.

   The components list appears as shown in the following figure.

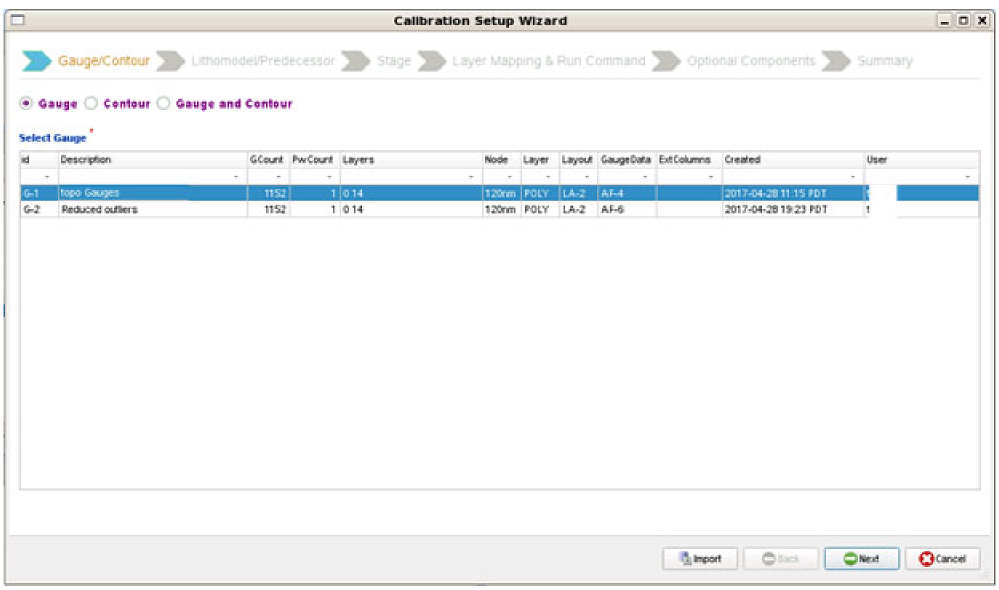
Tip

Components that are marked with a red asterisk are required. All other components are optional and you can safely click **Next** without adding any items from the list for that component.

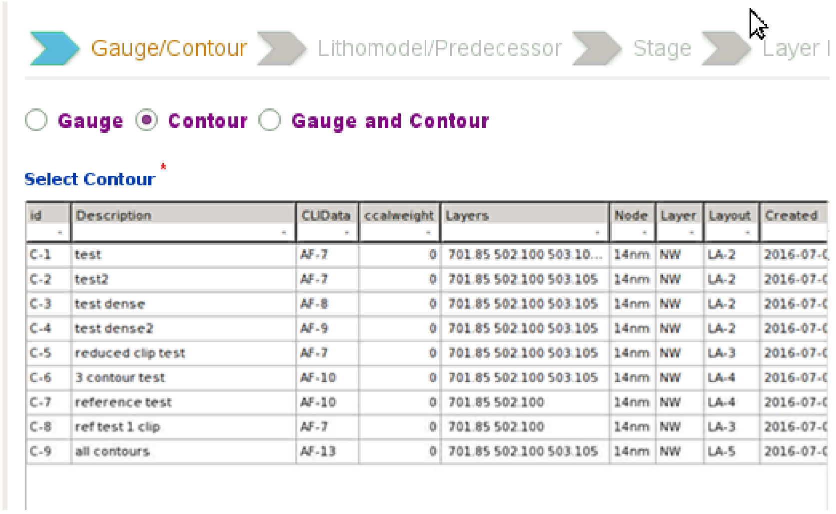
a. If you chose Gauges or Gauges and Contours, select a gauge object from the gauges list, then click **Next**.

   The gauge object you select should correspond to the stage type you intend to calibrate using this job:

   - **Optical Model Calibration** — Select a set of planar feature gauges made up of horizontal, vertical, and 2D features.
   - **Topo Calibration** — The set of planar gauges and topo gauges (usually many more than just the 2Rx and 3Rx features).
   - **Resist Calibration** — The set of planar gauges and topo gauges.
   - **Verification Job** — A subset of the planar and topo gauges.



b. If you chose Contours or Gauges and Contour, select a contour file, then click **Next**.



id	Description	CLIData	ccalweight	Layers	Node	Layer	Layout	Created
C-1	test	AF-7	0	701.85 502.100 503.10...	14nm	NW	LA-2	2016-07-(
C-2	test2	AF-7	0	701.85 502.100 503.105	14nm	NW	LA-2	2016-07-(
C-3	test dense	AF-8	0	701.85 502.100 503.105	14nm	NW	LA-2	2016-07-(
C-4	test dense2	AF-9	0	701.85 502.100 503.105	14nm	NW	LA-2	2016-07-(
C-5	reduced clip test	AF-7	0	701.85 502.100 503.105	14nm	NW	LA-3	2016-07-(
C-6	3 contour test	AF-10	0	701.85 502.100 503.105	14nm	NW	LA-4	2016-07-(
C-7	reference test	AF-10	0	701.85 502.100	14nm	NW	LA-4	2016-07-(
C-8	ref test 1 clip	AF-7	0	701.85 502.100	14nm	NW	LA-3	2016-07-(
C-9	all contours	AF-13	0	701.85 502.100 503.105	14nm	NW	LA-5	2016-07-(

c. Choose between two options from the selection screen:

- **Litho Model from Database** — Displays a list to select a litho model from those available in the database, followed by a wizard to assign mask layers described in the litho model to layers in the layout.

  Note

  If you are creating a job for a litho model that contains a topo model, and the topo model has no FinFET signals, do not map the FinFET layers in the design file.

- **Result of a Calibration Job** — Displays the Job list to select the output of another calibration job as input, in place of a litho model in the database. This option cannot be selected if there are no other calibration jobs defined.

d. Select a Stage from the list of previously-defined stages.

Note

For topo model contour verification, choose a stage that has simulation only.

If you do not have any stages in the list, you can click **Add** to create one.

e. Select a run command type from the dropdown list. A full list of builtin runscripts is available by clicking the "?" button to access the online help.

f. (Optional) Add one or more filters to the job. You would do this to reduce the number of gauges that need to be calibrated, limiting the gauges to test patterns of interest.

g. (Optional) Add one or more metrics to the job. You would do this to assess the quality of the results by more than just the cost objective.

h. (Optional) Add one or more plot groups to the job.

i. On the Summary screen, add a description of the job, then click **OK**.

Results

The new job appears in the Calibration Job Manager list in the main window. Continue to the next task, "Running a Topo Calibration Job."

Tip

If you need to change the parameters of a job, double-click on its ID. When the Attributes viewer appears for the job, click the **Edit** button. Clicking on a field puts the cursor there for editable text fields, or brings up a choice list for items in the database. However, only jobs in the Initial state can be edited.

Running a Topo Calibration Job

Calibration jobs are explicitly executed. This allows you to set up multiple experiments before running any job(s).

Prerequisites

- A set of calibration jobs in the database (as described in "Creating a Topo Calibration Job").

Procedure

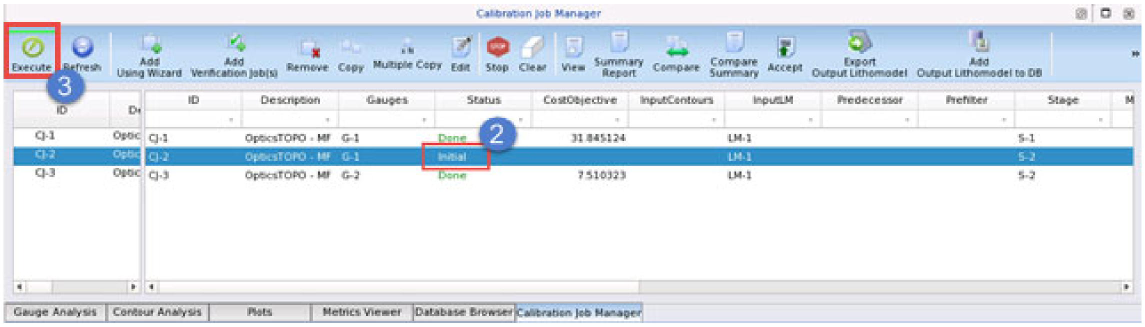
1. Raise the **Calibration Job Manager** tab.
2. Select a job from the list. Ensure that the Status field is "Initial" or "Ready". The order of jobs you execute should be as follows:

Table 1-5. Order of Calibration for Topo Modeling

Order	Job
1	Optical models
2	Topo model
3	Resist model
4	Verification

Each job should use the resulting models from the previous calibration job as their input.

3. Click the **Execute** button to start the selected job(s).



Results

The Status field for the job switches to one of the following states:

- **Running** — The job is currently active and being run.
- **Queued** — The job is waiting for the dispatcher to become available.
- **Waiting** — The job is dependent on the results of another calibration job, and will not be run until the parent job completes.

The calibration job run ends in one of the following states:

- **Done** — The cost objective for the calibration run is added to the calibration job entry.
- **Error** — An error occurred during processing. To find out what caused the error, click the **View** button in the **Calibration Job Manager**.
- **Stopped** — Job was canceled with the **Stop** button.
- **Dead** — Only occurs if the user killed the Calibre nmModelflow or Calibre WORKbench process.

Common calibration run error messages and their resolutions are as follows:

- **Write: bad etch model mapping** — This error occurs when the litho model used with topo contains an etch model. Resolve this by regenerating the litho model without an etch model.
- **The Lithomodel contains an optical model that refers to one or more non-existent source maps or pupil map files** — This error occurs when Calibre WORKbench was started in a directory that does not correctly resolve a relative path to the source or pupil map files. Resolve this by using any of the following methods:
    - Restart Calibre WORKbench in the directory that contains the models and map files in the expected location.
    - Adjust the optical model to point to the correct path.
    - Move the missing map files to the correct location.

- **Nominal value of S1 (*N*) is less than the minimal allowed value (*M*)** — This error occurs if a resist model parameter is below the lower bound. Correct this by editing the resist model file. (In this specific case, you adjust the DEFINE M1 line, changing the s parameter.)

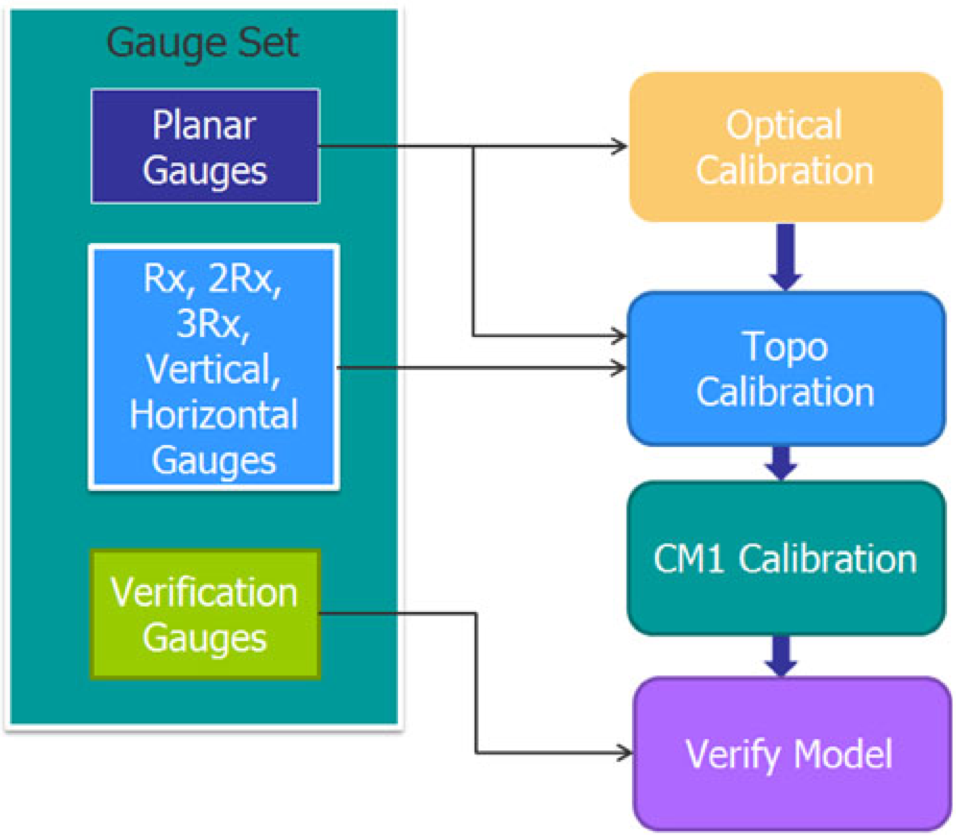
Topo Calibration Best Practices

The best practices for topo modeling represent the results of testing performed at Siemens EDA. They are intended as a guideline for most topo configurations.

The flow for calibrating topo models involves splitting up the test pattern gauges into three groups (Figure 1-21). Around 1000 gauges should be sufficient for the total number of gauges.

- **Planar Gauges** — About 200 to 400 gauges should be on planar features, made up of horizontal, vertical, and 2D features.
- **Rx Gauges** — The test pattern should also contain various enclosure distances of 2Rx and 3Rx features, which are used during the topo modeling step. These gauges should be disabled during the optical calibration step.
- **Verification Gauges** — A subset of the gauges should be designated for verification purposes.

Figure 1-21. Gauge Set Calibration Flow for Topo Modeling



Topo Calibration Best Practices for Optical Models

Optical models are calibrated as the first step of the topo calibration flow. They are calibrated using only planar gauges, in order to best characterize the optical source.

Use these best practice recommendations to calibrate optical models.

Test Pattern Guidelines for Optical Models

Use 200 to 400 planar gauge features. Planar gauges are the gauges that do not contain transitional effects. These are gauges that are typically deep inside the oxide and silicon regions.

Optical Model Guidelines

The following optical model parameters should be calibrated during this phase:

Table 1-6. Optical Calibration Recommendations

Parameter	Recommended Setting	Notes
BEAMFOCUS	auto auto 5	
DEF_START	auto auto 5	
IMAGEDIFFUSION	0 0.05 5	You should also specify the imagediffusion3d 1 option.
Search Algorithm	front	
Explore Iterations	1000	The GUI default is 251.
STACK_WEIGHT_EX1	0 1 5	Use this option only if the Exposure 1 optical model has more than one stack, and then set options for multiple exposures.
Multiple Exposures		
BEAMFOCUS_EX*N* (*N*=2,3)	BEAMFOCUS auto auto	Set BEAMFOCUS_EX*N* to be dependent on BEAMFOCUS.
DEF_START_EX*N* (*N*=2,3)	DEF_START auto auto	Set DEF_START_EX*N* to be dependent on DEF_START.
IMAGEDIFFUSION_EX*N* (*N*=2,3)	IMAGEDIFFUSION 0 50	Set IMAGEDIFFUSION_EX*N* to be dependent on IMAGEDIFFUSION.

Table 1-6. Optical Calibration Recommendations (cont.)

Parameter	Recommended Setting	Notes
FILMSTACK*F*_THICK1_EX*N* (*F*=2,3, *N*=2,3)	FILMSTACK1_THICK1 - 0.1 0.1 10	The additional exposures for multiple layer filmstacks should be dependent on the matching filmstack in exposure 1.

Topo Calibration Best Practices for Topo Models

Topography models are calibrated as the second step of the calibration flow. They are calibrated using the best calibrated optical model.

Calibrating for topo models is an iterative process to find the best model.

- Test Pattern Guidelines for Topo Models
- Initial Calibration Best Practices
- Iterative Topo Calibration

Test Pattern Guidelines for Topo Models

Topo calibration test patterns calibrate both planar gauges and Rx-type gauges.

- The drawn CD of an Rx-type gauge is the implant size, and is dependent on the mode (minimum rule size) for the design. Because the minimum size may not resolve well, at least three larger size gauges should be added for calibration purposes. For example, for a 400nm design, you would have 400nm, 450nm, 500nm, and 550nm patterns.

  The enclosure distance for test pattern structures between the Rx layer and the edge of the topo pattern depends on the density of the pattern. Your 2Rx and 3Rx implant patterns should have some variety according to the following enclosure samples:

  - For 0 to 400nm, use a spacing of 40nm, starting with -40 (-40, 0, 40, 80, 120...).
  - For 400nm to 1.5 microns, use a spacing of 100nm.

  Note

  “2Rx” and “3Rx” refers to the number of Rx shapes present on the test pattern as shown in the figure. 2Rx shapes vary their spacing distances measured from the center of the gauge, moving outwards. A notation of -40 for the spacing indicates that the Rx shapes are exposed -40nm from the edge of the implant. A 3Rx shape has a constant Rx shape in the center, plus two Rx shapes that vary moving outwards.

Figure 1-22. 2Rx Patterns Showing Enclosure Distance Variations

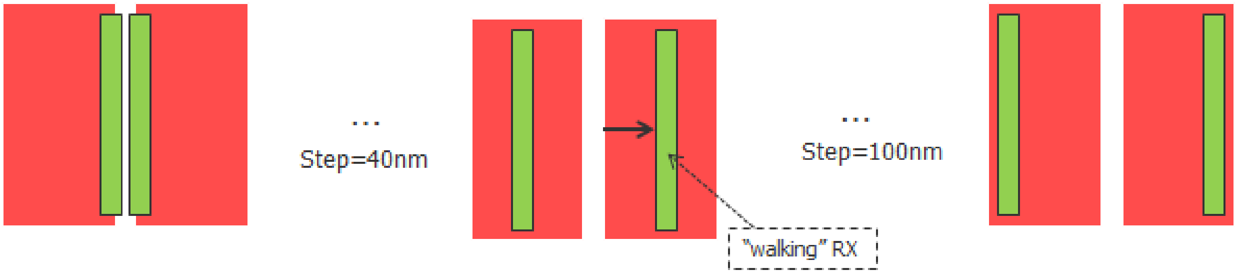


Figure 1-23. 2Rx Side View Showing Signals

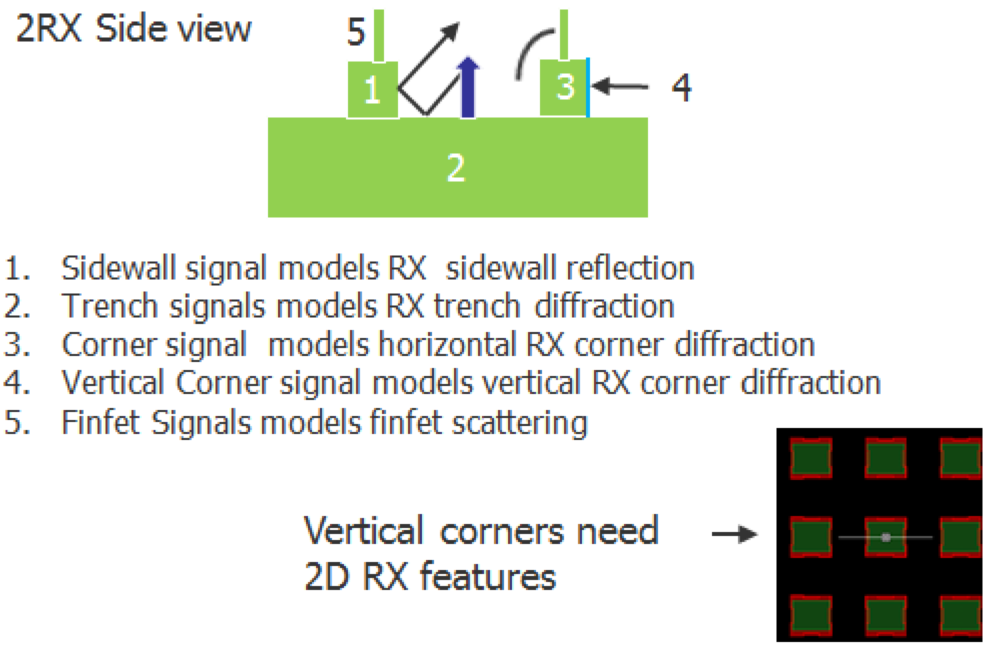
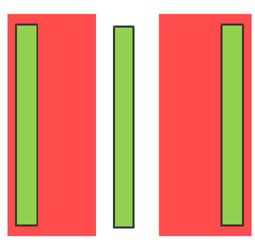


Figure 1-24. 3Rx Pattern



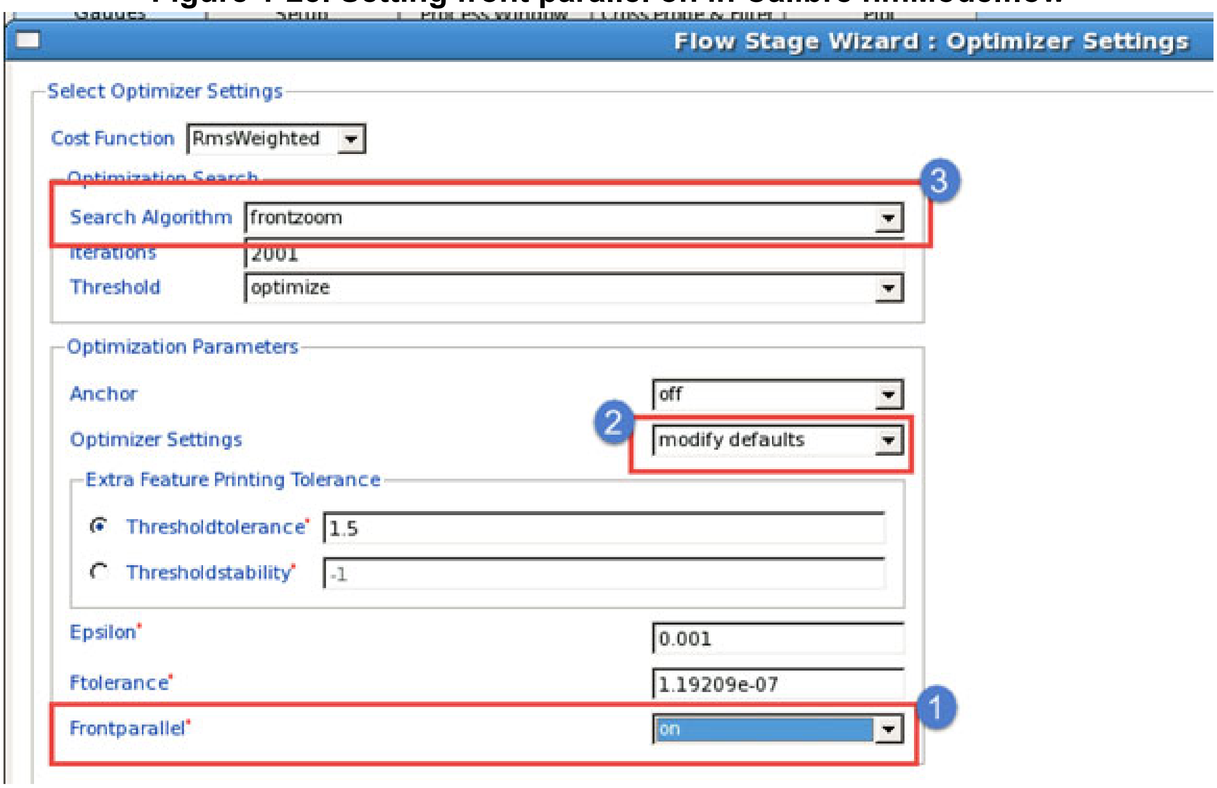
- The number of gauges should be approximately three times the number of calibrated parameters. For example, a topo CSTFV + oxide with 8 linear terms has six signals (Corner, Sidewall, Trench, FinFET, Vertical corner, and oxide) with about 48 calibrated parameters, which requires about 150 topo gauges.

Initial Calibration Best Practices

Using a set of simple and fast calibration methods allows you to find various outlier errors and correct them before full calibration.

- Use a simple topo model.
    - For pre-PC configurations, use topo2CST or topo2CSTF if FinFETs are present.
    - For post-PC configurations, use topo3STGI, or topo3STGIF if FinFETs are present.
- Optimize the oxide bias parameters for both pre-PC and post-PC configurations.
- For linear terms, the recommended search ranges are -0.15 ... 0.15 for clear backgrounds and -0.2 ... 0.2 for dark backgrounds.
- Use the optimizer setting for front search during the topo calibration stage. Set the frontparallel "on" option (1 in the following figure) in the Optimizer Settings screen of the Stage wizard. You may also need to select the "modify defaults" setting (2) in the Optimizer Settings field.

Figure 1-25. Setting front parallel on in Calibre nmModelflow



- Use the frontzoom search for the Search Algorithm field (3 in the figure above) with an explore setting of 2001.
- Use a CTR resist model for the topo calibration phase.

During initial calibration, investigate any large CD errors and correct setup file errors. Possible sources of error include:

- A gauge with an incorrect location flag
- Layers not being mapped correctly

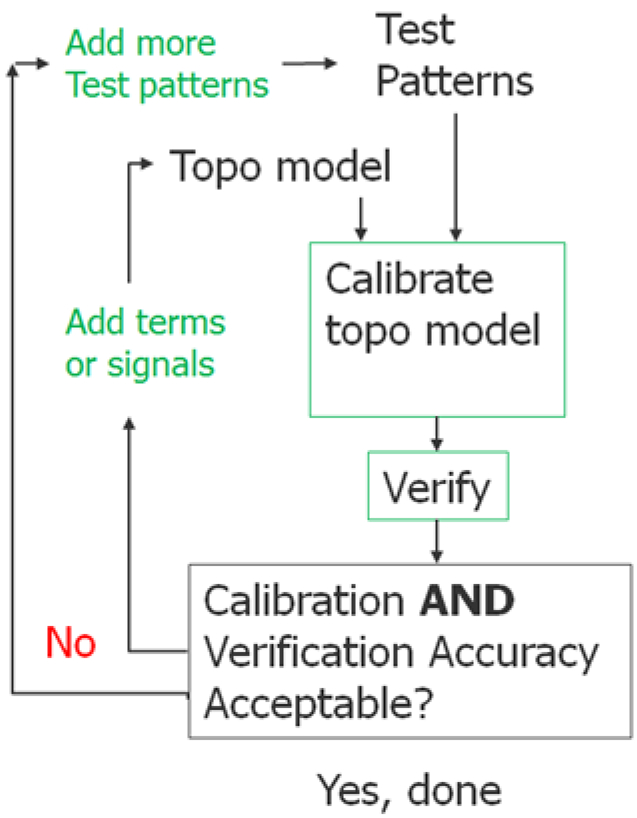
- Incorrect optical models used

Check the plots of CDs for the 2Rx and 3Rx features over the enclosure distances. Remove any outliers after investigating them for inaccuracies.

Iterative Topo Calibration

Once you have found a simple topo model, you improve the accuracy by using the following methods. The best model is the model that minimizes both calibration error and verification errors.

Figure 1-26. Topo Iterative Flow



- Experiment with a copy of the Calibre nmModelflow stage, modifying it as follows:
    - Increase the number of terms from 8 (the default) to 16 terms in the Flow Stage Wizard or by editing the topo model file.

      Note
      Starting with the 2016.3 release, you can also create a custom topo modelform in the Calibre nmModelflow GUI (see "Editing an Existing Topo Model" on page 24).

    - Add additional signals (such as vertical corners or trench 2) as appropriate. Additional signals can improve calibration accuracy, but also risk overfitting. You should always check verification accuracy after adding more signals. Additional signals also need more iterations, which adds additional runtime.

Table 1-7. Suggested Iterations Per Signal Parameter

Signal Parameters	Outer Loop (explore)	Inner Loop (topoiterations)
2	250	300 (8 terms)
3	500	300
4	1200	300 (8 terms), 3000 (16 terms)
5	2500	300, 3000
8	5000	300, 3000
9	7500	300 (8 terms), 3000 (16 terms), 5000 (24 terms)
10	10000	300, 3000, 5000

- Increase your explore iteration settings cautiously. Siemens EDA has found RMS to decrease smoothly as explore values are increased from 6000 iterations to 10000 iterations, but the runtime increases linearly; in most cases, you may be satisfied with the RMS obtained at 6000 iterations.

  For topoiterations settings, values higher than 6000 iterations do not provide additional accuracy.

- Simulate contours on test structures that closely resemble actual features, and compare them to SEM images of those features.

  For 2D features, verification using extracted SEM contours is recommended. This method requires you to load a contour, Contour Library Information file, and a layout that contains SEM clips. For more information, see the following sections:

  - "Loading Contours" in the *Calibre nmModelflow User's and Reference Manual*
  - "Contour Layer Info (CLI) File Format" in the *Calibre WORKbench User's and Reference Manual*

- For runtime performance improvement, Siemens EDA recommends using the "front parallel on" option in Calibre nmModelflow. At least 400 CPUs should be used with these options when possible.

- Calibrate the final model with the setting "front parallel off" in Calibre nmModelflow.

Chapter 2
Topo Test Pattern User's Guide

A standard Calibre WORKbench modeling test pattern only contains an implant layer. Topo-specific test patterns contain additional layers: a required active layer, and optional FinFET and poly layers.

Creating a Basic Topo Test Pattern

A basic topography test pattern closely resembles the standardized test pattern available in Calibre WORKbench, but includes at least one additional layer for the active (Rx) topo overlay.

Prerequisites

- General knowledge of your design file precision and node size.

Procedure

1. In a terminal window, navigate to your working directory and invoke Calibre WORKbench.
2. In the console window, enter the following command:

   tepCreate topo -ggfile *filename* *optional_parameters*

   where the optional parameters are one or more of the following:

Table 2-1. tepCreate topo Parameters

Parameter	Description	Default
-cdwidth *value*	Defines the implant layer (target CD) width in nm.	140
-cdspace *value*	Defines the implant layer (target CD) space in nm.	140
-finfetwidth *fwidth* -finfetspace *fspace* [-finfetorientation {v| h}	Adds a FinFET layer to the output. Both width and space must be specified if this option is used.	Not active by default. The default orientation for the FinFET lines is vertical.

Table 2-1. tepCreate topo Parameters (cont.)

Parameter	Description	Default
-gdsfile *filename*	Specifies an alternative name for the output design file, other than the default generated name.	"*topo.<cdwidth>.<cdspace>.<dbunit>.<grid>.gds*"
-ssfile *filename*	Specifies an alternative name for the output spreadsheet file, other than the default.	"*topo.<cdwidth>.<cdspace>.<dbunit>.<grid>.ss*"
-ggfile *filename*	Specifies a name for an output gauge file.	Not generated by default, but this option is *required* for Calibre nmModelflow purposes.
-csvfile *filename*	Specifies a name for the output spreadsheet file. Used only for the Calibre CMI tool.	Not generated by default.
-markers {none | cross | odd | even | sem }	Sets the marker type for the pattern.	No markers are generated by default. (none)
-topcellname *name*	Sets the name of the topcell.	TOP
-dbinnm *dbunit*	Sets the database unit size in nm.	1
-gridinnm *gridsize*	Sets the grid unit size in nm.	5

Results

Three files are created in the working directory if you used the default options and specified the -gaugefile option: a design file (*.gds* file) and a gauge file (*.gg* file), which you use in the gauge file loading task, and a sample spreadsheet file (*.ss* file), which is not used in Calibre nmModelflow.

Creating a Custom Topo Test Pattern

Topo test patterns extend the Calibre WORKbench test pattern format. Running the tepCompile batch command on a topo test pattern file results in a layout file with customized layers for topo models.

Prerequisites

- Understanding the process described in the "Test Pattern Creation" section of the *Calibre WORKbench User's and Reference Manual*

Procedure

1. Open a file in a text editor and start by defining the PATTERN header to set the baseline information for the output OASIS®[1] layout and gauge file, similar to the following:

```
PATTERN "Top 3RX Example"
   dbgrid    0.001
   oasisfile tpf_3RX_topo.oas
   ggfile    tpf_3RX_topo.gg
```

   A topo test pattern file includes one or more of the following building blocks as shown in the table.

Table 2-2. Topo Test Pattern Keywords Summary

Step	Keyword	Notes
2	FINFET	Optional. Used only if your topo configuration uses FinFETs.
3	FEATURE	Required. Places the implant layer.
4	PC_FEATURE	Optional. Places poly lines over the implant layer.
5	RX_FEATURE	Required. Places the topo pattern.

2. If you are creating a FinFET layer, add the FINFET Keyword and associated parameters, similar to the following code. The FINFET keyword must appear before the FEATURE keyword.

```
FINFET
   Widths 0.014
   Pitchs 0.06
   Orientation h
   Centerings space
```

3. Define a FEATURE block, using the templates from the *Calibre WORKbench User's and Reference Manual* to set the patterns for the implant layer.

```
FEATURE rect_v_array
   Titles     inv_linend neck_x pitch_0 bridge_0 linelen
   Counts     3           3      17      9        9
   Ewidths    4           0.6    0.2     4        3.8
   Eheights   0.3         1.2    0.2     0.2      0.2
   Spaces     0          -0.4    0.09    0.1      0.1
   Midwidths  0.1         0.2    0.2     0.2      3.2
```

Note
The Titles field sets what structure names the features will have in the gauge file.

1. OASIS® is a registered trademark of Thomas Grebinski and licensed for use to SEMI®, San Jose. SEMI® is a registered trademark of Semiconductor Equipment and Materials International.

4. If you are creating a poly layer, add the PC_FEATURE Keyword and associated parameters, similar to the following code.

```
PC_FEATURE pc_rect
  NumSections 5
  OrientationS v
  Offset_Y 0
  SpacesS 0.8
  WidthsS 0.2
  HeightsS 0.5
  Widths 0.02
  Heights 0.4
  Pitchs 0.15
  Orientation v
```

5. Define an RX_FEATURE Keyword block similar to the following code, using the templates from "Topo Model Test Patterns" on page 109.

```
RX_FEATURE 3rx_rect
   Orientation h h h v
   Widths 0.4 0.2 0.4 0.2
   Spaces 0.5 0.2 0.5 0.2
   Offset_X 0 0 0 0
   Offset_Y 0 0 0 0
   Heights 2.0 2.0 2.0 2.0
```

6. Repeat steps 2 through 5 until you have created all the test patterns you need. Note that if you are using FINFET keywords, you can skip the FINFET step to automatically re-use the previous FINFET settings, define a new FINFET block, or specify FINFET off to deactivate FINFET structures being applied for the current test pattern set.
7. Save the file, then generate the test pattern with the following command:

```
calibrewb -a tepCompile filename.tpf
```

Results

If the command completes without errors, tepCompile creates a layout file and a gauge file for use with Calibre WORKbench CM1 Center or Calibre nmModelflow.

Tip

The most common issues if the compile fails are either a missing keyword or a mistyped keyword, both of which specify which keyword is an error.

Related Topics

FINFET Keyword

PC_FEATURE Keyword

RX_FEATURE Keyword

Topo Model Test Patterns

Analyzing Enclosure Distances Using mdf topoenclosure

In topo modeling, obtaining the distances between an oxide and the nearest active edge in a layout assists in plotting CD errors versus distance. Calibre nmModelflow provides a console command (mdf topoenclosure) and visualization tools to perform this optional analysis.

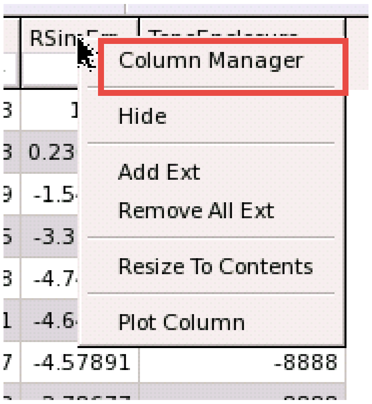
Prerequisites

- Calibre WORKbench and Calibre nmModelflow invoked
- The relevant design file loaded in Calibre WORKbench
- The associated gauge file in the Calibre nmModelflow database
- The associated litho model file in the Calibre nmModelflow database

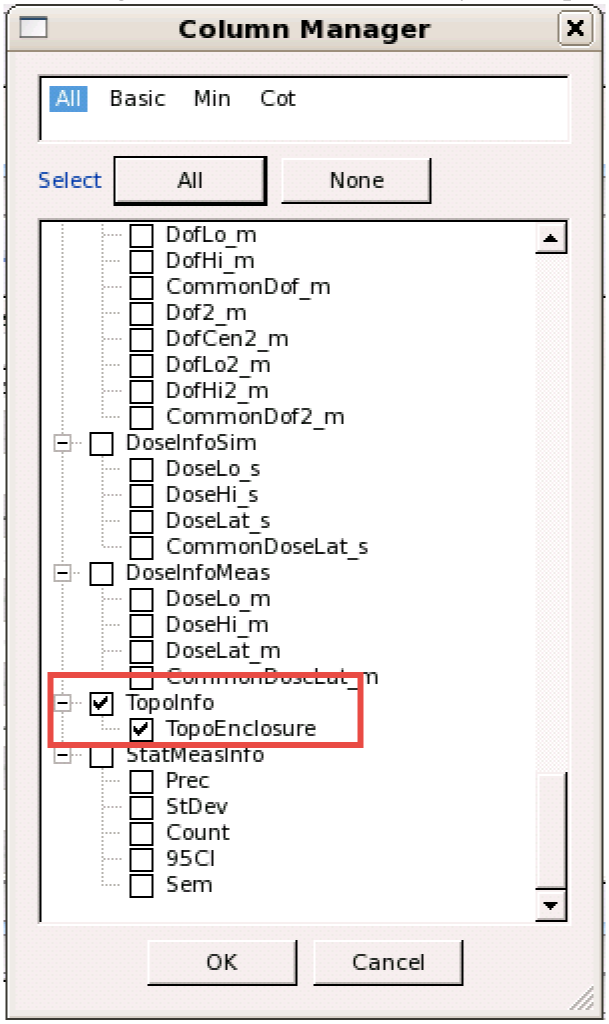
Procedure

1. In Calibre nmModelflow, activate the gauge file and litho model.
2. Switch to the **Gauge Analysis** tab. If the Topoenclosure column is not visible:

    a. Right-click any column header in the gauges list, then select **Column Manager** from the popup menu.

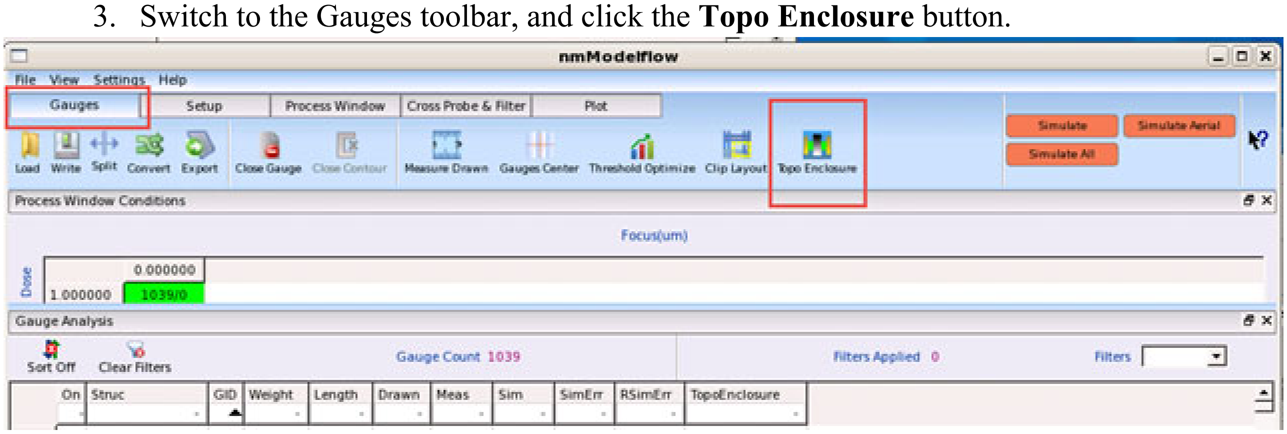


b. In the Column Manager, scroll down to the entry for TopoEnclosure, and select it.



Click **OK** to close the dialog box and add the column to the **Gauge Analysis** tab.

3. Switch to the Gauges toolbar, and click the **Topo Enclosure** button.



4. In the Measure Topoenclosure Distance dialog box, select options as appropriate, then click **OK**.

Table 2-3. Measure Topoenclosure Distance Parameters

Field	Description
Underlying	Specifies the type of underlying layer (active or poly) corresponding to your litho model.
Mode	Specifies the measurement mode. • **Meas** — Computes enclosure for CD measurement points on a gauge. • **Edge** — Computes enclosure for intersection points between the cutline and edges of implant features.
Max	Specifies the maximum search enclosure distance in microns.

Results

The command runs. On completion, it adds a Topoenclosure column to the **Gauge Analysis** tab. The new column contains enclosure distance values for all gauges it found to be enclosed, or a value of -8888 for gauges where an enclosure distance was not found.

Measuring Polygon Widths Around a Gauge

Knowing the underlying polygon widths for silicon (RX), poly (PC), and layer shapes around a gauge is useful information during topo model calibration. Measurement of polygons is supported in Calibre nmModelflow via the command line interface.

Prerequisites

- Calibre WORKbench and Calibre nmModelflow invoked
- The relevant design file loaded in Calibre WORKbench
- The associated gauge file in the Calibre nmModelflow database
- The associated litho model file in the Calibre nmModelflow database

Procedure

1. In Calibre nmModelflow, activate a gauge from the **Database Browser** tab by selecting Gauges from the listing tree, clicking a gauge in the list, and then clicking **Activate**.
2. Similarly, select a litho model from the Litho Model list and activate a litho model that contains a topo model of interest.
3. Switch to the **Gauge Analysis** tab.

4. In the Command Window, enter the following command:

   mdf gauges topomeas [-underlying {active | poly}]
   [-max *dmax*] [-active_gauges]

   where

   -underlying {active | poly} — Sets which underlying layer is being scanned. The default layer is the active layer defined in the litho model.

   -max dmax — Sets the maximum search distance between features along each enabled gauge.

   -active_gauges — Computes values only for the active gauge set, which may have been modified by other commands. By default, the calculation is performed on the original loaded gauges.

Results

The command computes the minimum width of the underlying features and spaces along each gauge in the active gauge file. Outputs are written to the gauge file as the new columns "RxWidth" and "RxSpace" for the underlying active layer (if -underlying active was specified), or "PolyWidth and "PolySpace" for the underlying poly layer (if -underlying poly was specified).

Chapter 3
Using Topo Models in Other Calibre Tools

Calibrated topography models are typically used with the Calibre nmOPC and Calibre OPCverify tools.

Using the Topographical Model in Calibre nmOPC

In this task, you include topographical model awareness for Calibre nmOPC for situations where you have a litho model.

Implementing topo model support in Calibre nmOPC requires minor additions to existing Calibre nmOPC setup files. Inside the denseopc_options command block, you modify the layer command and add the topo_model command.

Prerequisites

- A litho model containing calibrated optical, topo, and resist models.
- A Calibre nmOPC setup file.

Procedure

1. In the Calibre nmOPC setup file, set the modelpath and layers.

   ```
   modelpath models
   layer implant
   layer RX
   ```

2. In the denseopc_options block, declare the Implant layer as “opc” for the type, and the RX layer as “underlying active.” Assign layers to the masks as follows:

   ```
   denseopc_options A {
         version 1
         layer Implant opc mask 0 mask 1
         layer RX underlying active
      }
   ```

3. Code image statements to include the litho model.

   ```
   image models_directory
   ```

4. Code other additions to the denseopc_options block as needed.
5. Add a setlayer denseopc command to use the denseopc_options block:

```
setlayer mask_output = denseopc Implant RX MAP Implant OPTIONS A
```

6. Code other Calibre nmOPC checks as needed and save the file.

Using the Topographical Models in Calibre OPCverify

Modifying Calibre OPCverify code to include awareness of topo models contained in a litho model requires several steps to work with the pointers inside the litho model.

The primary tasks to configure Calibre OPCverify for topo modeling include the topo_model_load command and a modified image command.

Prerequisites

- A litho model containing calibrated optical, topo, and resist models
- A Calibre OPCverify setup file

Procedure

1. Edit the Calibre OPCverify setup file, setting the modelpath and layers.
2. Create an image_options block that has the following items:
    - A litho_model statement to indicate the directory containing the Lithomodel file.
    - Underlying layers using the “underlying” modifier after the layer declaration along with a type of active, active2, poly, finfet, or oxide2.
    - A mask number if you are using multiple underlying density layers. The mask number comes from the litho model.

    ```
    image_options A {
          layer Implant visible mask 0 mask 1
          layer Active underlying active
          litho_model models_directory
          }
    ```

3. Code image statements to include the image options block:

    ```
    setlayer cntr = image A
    ```

4. Code other Calibre OPCverify checks as needed.
5. Save the file and run it as normal.

Example Topo Files for Calibre OPCverify

This section provides examples of single- and multiple-density topographical models and the corresponding Calibre OPCverify setup files in litho model syntax.

Example 1 (Pre-PC; Single Density)

Given the following litho model definition of a single density configuration:

```
version 1
resist  resist_model_file
topo    topo_model_file
mask 0 {
   background  dark
   mask_layer  0 TRANS 1.0 0.0 CATEGORY OXIDE
   optical oxide_optical_file  }
  mask 1 {
   background  dark
   mask_layer  0 TRANS 1.0 0.0 CATEGORY SILICON
optical silicon_optical_file
}
```

The related Calibre OPCverify setup file would be similar to the following code:

```
LITHO FILE topo_opcv_lm [
   modelpath models
   layer Implant
   layer RX
   image_options A {
      layer Implant visible           mask 0 mask 1
      layer RX      underlying active
   litho_model   LITHO_MODEL
}
setlayer cntr = image A]
```

The related topo model would be similar to the following code:

```
transit 0 1
from 0 to 1
property stack_weight
transit_dose 1 1
```

Example 2 (Post-PC; Transit Exposures for Oxide, Silicon, and Poly Optics; Multiple Density Exposures)

The following example shows a litho model setup for a post-PC configuration, using transit exposures for oxide, silicon, and poly optics with multiple density exposures using litho model syntax.

The model information for a single-density configuration can be expressed in the following litho model:

```
version 1
resist resist_model_file
topo topo_model_file
mask 0 {
   background dark
   mask_layer 0 TRANS 1.0 0.0 CATEGORY OXIDE
   optical oxide_optical_file
}
mask 1 {
   background dark
   mask_layer 0 TRANS 1.0 0.0 CATEGORY SILICON
   optical silicon_optical_file
}
mask 2 {
   background dark
   mask_layer 0 TRANS 1.0 0.0 CATEGORY POLY
   optical poly_optical_file
}
mask 3 {
   background dark
   mask_layer 0 TRANS 1.0 0.0 CATEGORY DENSITY
   optical density_optical_file
}
mask 4 {
   background dark
   mask_layer 0 TRANS 1.0 0.0 CATEGORY DENSITY
   optical density_optical_file
}
```

The related Calibre OPCverify setup file would be similar to the following code:

```
LITHO FILE topo_opcv_lm [
modelpath models
layer Implant
layer RX
layer Poly

image_options A {
   layer Implant visible mask 0 mask 1 mask 2
   layer RX   underlying active mask 3
   layer Poly underlying poly mask 4
   litho_model LITHO_MODEL
}
setlayer cntr = image A
]
```

The topo model is the same as in the previous example. As in the previous example, you can introduce FinFET effects with the fterm topo signal by adding the FinFET layer.

```
LITHO FILE topo_opcv_lm [
modelpath models
layer Implant
layer RX
layer Poly
layer Finfet
image_options A {
   layer Implant visible mask 0 mask 1 mask 2
   layer RX     underlying active mask 3
   layer Poly   underlying poly mask 4
   layer Finfet underlying finfet
   litho_model LITHO_MODEL
}
setlayer cntr = image A
]
```

Example 3 (Post-PC; Transit Exposures for Oxide, Silicon, Poly, and FinFET Optics)

The following code example shows a litho model set up for a single density, but a topo model that refers to four transit exposures. Setups like this simulate the corresponding density exposures internally.

The litho model definition:

```
version 1
resist resist_model_file
topo topo_model_file
mask 0 {
   background dark
   mask_layer 0 TRANS 1.0 0.0 CATEGORY OXIDE
   optical oxide_optical_file
}
mask 1 {
   background dark
   mask_layer 0 TRANS 1.0 0.0 CATEGORY SILICON
   optical silicon_optical_file
}
mask 2 {
   background dark
   mask_layer 0 TRANS 1.0 0.0 CATEGORY POLY
   optical poly_optical_file
}
mask 3 {
   background dark
   mask_layer 0 TRANS 1.0 0.0 CATEGORY FINFET
   optical finfet_optical_file
}
```

Part of the related Calibre OPCverify setup file:

```
LITHO FILE topo_opcv_lm [
modelpath models
layer Implant
layer RX
layer Poly
layer Finfet
image_options A {
   layer Implant visible mask 0 mask 1 mask 2 mask 4
   layer RX     underlying active
   layer Poly   underlying poly
   layer Finfet underlying finfet
   litho_model LITHO_MODEL
}
setlayer cntr = image A
]
```

The topo model:

```
transit 0 1 2 4
density oxide active Poly Finfet
from 0 to 1
property stack_weight
transit_dose 1 1 1 1
...
```

The topo model refers to four transit exposures for oxide, silicon, poly, and FinFET optics. The corresponding density exposures are simulated internally.

Note

Previous to the 2017.3 release, the layer names in the declaration in the Calibre OPCverify setup file needed to match the names in the topo model files “density underlying” parameter. In this example preceding, the topo model would have required a “density underlying -RX RX Poly Finfet” statement in it.

Starting with the 2017.3 release, this restriction has been removed, and the new syntax has been simplified to “density oxide active [poly] [finfet]”, corresponding to the “layer underlying [*type*]” statements in the Calibre OPCverify setup file.

Chapter 4
Topographical Model Reference

The commands for topographical modeling are either a subset of the commands from Calibre WORKbench or a command specifically created for topo modeling.

Note

If a command you are looking for is not found in this chapter, see the *Calibre WORKbench User's and Reference Manual*.

Table 4-1. Topographical Modeling-Related Commands

Command	Description
film2, film3	Defines additional film stacks in an optical model file for use with topographical modeling.
FINFET Keyword	Specifies FinFET layer parameters for RX (active) templates following this keyword.
PC_FEATURE Keyword	Specifies a poly layer to be added to a topo test pattern layout.
RX_FEATURE Keyword	Specifies an active (RX) layer to add to a test pattern layout.
stack_weight, stack2_weight	Defines the stack weight of topographical modeling film stacks relative to other film stacks in an optical model file.
tepCreate topo	Generates a Version 11 topo test pattern and associated spreadsheet file.
topo_model Format	Describes topographical models inside model-layer files or litho model files.
Topographical Modelform Parameters (topomf)	Specifies a topo modelform that specifies a template set of signals as an ease of use convention.
Topo Model Test Patterns	The topo model test patterns are an extension of the standard test pattern formats described in the *Calibre WORKbench User's and Reference Manual*.
topoenclosure	Outputs distances from the measurement CD sites to the nearest edge of the active layer.
toposignal	Outputs the toposignal data for a specified signal from the input topo model.

film2, film3

Optical model parameter

Defines additional film stacks in an optical model file for use with topographical modeling.

Usage

film2 *thickness n k* “*comment*” [film3 *thickness n k* “*comment*”]

Arguments

- ***thickness***

  A required parameter specifying the film thickness in microns.

- ***n***

  A required parameter specifying the index of refraction for the material.

- ***k***

  A required parameter specifying the absorption value of the material.

- **“ *comment* ”**

  A required parameter describing what the material is. The ***comment*** is an arbitrary string, but it must be enclosed in quotes (“ ”).

Description

Note

These parameters require the related stack weight parameter (stack_weight, stack2_weight) to also be specified in the optical model.

These Calibre WORKbench optical model parameters define additional film stacks such as may be required for modeling FinFET processes. They are used in conjunction with the stack_weight and stack2_weight parameters.

- film2 and stack_weight define the second film stack and its weight relative to the first film stack (film keyword).
- film3 and stack2_weight define the third film stack. They can only be specified if film2 and stack_weight are also specified.

Each film stack can, and usually does, contain multiple films. The film, film2, and film3 statements apply to a film in a film stack, but the stack_weight and stack2_weight parameters apply to the whole stack.

In order to accurately model a transitional optical state, an optical model must take into account the effects of a silicon area versus oxide-over-silicon area (or silicon area versus FinFET area) on the chip. The effects are dependent on the position of the measurement relative to the “pure” areas.

Multiple film, film2, and film3 keywords are allowed; each describes one part of its stack. That is, each film2 keyword describes one part of the second film stack. Similarly, multiple film3 keywords are used to define the third film stack.

Examples

Example 1: Oxide and Silicon Stacks for Transitional Optical Model

Consider the case of the first stack over silicon (Rx regions on the layout) and a second stack over oxide (no Rx regions in this area of the layout). Then the transitional optical model (from stack 1 to stack 2) defined using the film and film2 keywords with a stack_weight of 0.2 would be as follows:

```
stack_weight 0.2
film 0.280 1.774 -0.035 "Resist"
film2 0.280 1.774 -0.035 "Resist"
film2 0.265 1.56 0.0 "Oxide"
substrate 1.57 -3.565
```

Example 2: FinFET Transitional Optical Model

Although FinFET processes are commonly modeled with only two stacks (a FinFET stack and an oxide or silicon stack), occasionally it is useful to define a transitional stack as in the following code.

```
//FinFET Oxide Stack
film 0.73  1.8 -0.01 "Resist"
film 0.003 1.5 -0.00 "Oxide"
film 0.1   1.5 -0.00 "Oxide Finfet"

//FinFET Silicon Stack
film2 0.73  1.8 -0.01 "Resist"
film2 0.003 1.5 -0.00 "Oxide"

//Oxide Stack
film3 0.73  1.8 -0.01 "Resist"
film3 0.003 1.5 -0.00 "Oxide"
film3 0.1   1.5 -0.00 "Oxide"
```

FINFET Keyword

Test Pattern Command (topo only)

Specifies FinFET layer parameters for RX (active) templates following this keyword.

Usage

FINFET
    Width *line_width*
    Pitch *pitch*
    [Orientation] {horiz | vert}]
    [Centering {line | space}]

[FINFET OFF]

Arguments

- **Width *line_width***

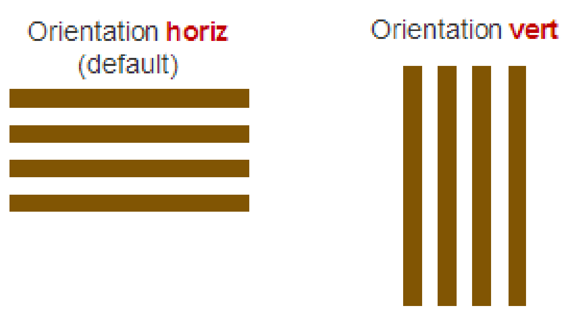
  A required argument that sets the line width of an individual FinFET rectangle in microns.

- **Pitch *pitch***

  A required argument that sets the pitch for the FinFETs in microns.

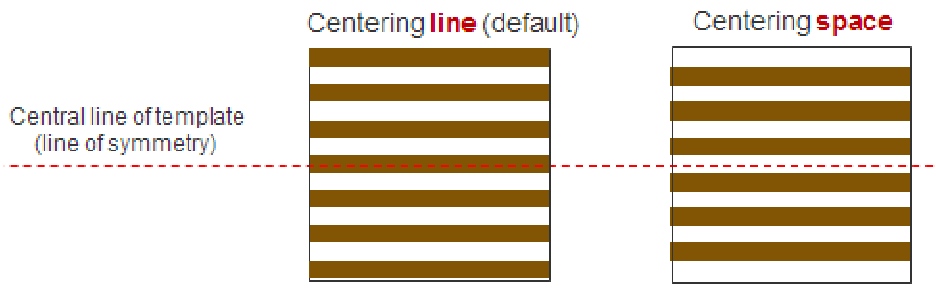
- Orientation {horiz | vert}

  An optional argument that sets the orientation of all FinFETs for this block. Default is horiz, which specifies horizontal.



- Centering {line | space}

  An optional argument specifying whether the center line of the FinFETs symmetry is covered by a line or space. Default is line.



Description

The FINFET keyword defines a FinFET line layer block for use with RX (active) features. It must come as the first keyword in a test pattern definition block, even before FEATURE.

FINFET definitions are considered in effect for all following FEATURE declarations until either a FINFET OFF keyword is specified or a new FINFET keyword is defined.

Examples

Declares three FEATURES with two different FINFET definitions.

```
PATTERN test
   //keywords for PATTERN here
FINFET // FinFET #1 creates horizontal lines
   Width 0.010
   Pitch 0.032
FEATURE iso
   // keywords for standard "iso" test structure
RX_FEATURE 2rx
   // keywords for RX structure associated with the FinFET and iso feature
FINFET // FinFET #2 creates vertical lines instead of horizontal
   Width 0.010
   Pitch 0.032
   Orientation vert
FEATURE inv_iso
   // keywords for standard "inv_iso" test structure
RX_FEATURE 3rx
   // keywords for the RX structure associated with the second FinFET and
inv_iso feature
FINFET OFF
FEATURE iso_pad
   // keywords for the "iso_pad" standard test structure; no FinFET
```

Related Topics

Creating a Custom Topo Test Pattern

PC_FEATURE Keyword

Test Pattern Command (topo only)

Specifies a poly layer to be added to a topo test pattern layout.

Usage

PC_FEATURE {**pc_rect** | **pc_planar_oxide** | **pc_planar_rx**}
(The following set of keywords are required for pc_rect only:)
NumSections ***p***
OrientationS {**horiz** | **vert**}
Offset_x ***opc_x***
Offset_y ***opc_y***
WidthsS ***w_ps***
HeightsS ***h_ps***
SpacesCS ***sc_ps***
SpaceS ***s_ps***
(The remaining set of keywords are required for all PC_FEATURE types:)
Widths ***w_p***
Heights ***h_p***
Pitchs ***p_p***
Orientation {**horiz** | **vert**}

Arguments

- **pc_rect**

  A required keyword (and the default if no PC_FEATURE keyword is specified) that indicates that a full definition of the poly layer sections is included.

- **pc_planar_oxide**

  A required keyword that sets most of the standard parameters to fixed values, creating a single poly section with rectangles covering the entire block.

- **pc_planar_rx**

  A required keyword for that when specified sets most of the PC_FEATURE parameters to fixed values. It creates an active (RX) layer with rectangles covering the entire block.

- **NumSections** ***p***

  A required pc_rect keyword that specifies the number of poly sections. An odd value places a poly section in the center of the template; an even value leaves a space in the center.

- **OrientationS** {**horiz** | **vert**}

  A required pc_rect keyword that specifies the orientation of all poly sections.

- **Offset_x** ***opc_x***

  A required pc_rect keyword that specifies the poly section offset on the X axis in microns, relative to the center.

- **Offset_y *opc_y***

  A required pc_rect keyword that specifies the poly section offset on the Y axis in microns, relative to the center.

- **WidthsS *w_ps***

  A required pc_rect keyword that specifies the width of each poly section in microns.

- **HeightsS *h_ps***

  A required pc_rect keyword that specifies the height of each poly section in microns.

- **SpacesCS *sc_ps***

  A required pc_rect keyword that specifies the space between the center two poly sections in microns.

- **SpaceS *s_ps***

  A required pc_rect keyword that specifies the space between all poly sections other than the centermost two sections, in microns.

- **Widths *w_p***

  A required keyword that specifies the width of an individual poly rectangle within a section, in microns.

- **Heights *h_p***

  A required keyword that specifies the height of an individual poly rectangle within a section, in microns.

- **Pitchs *p_p***

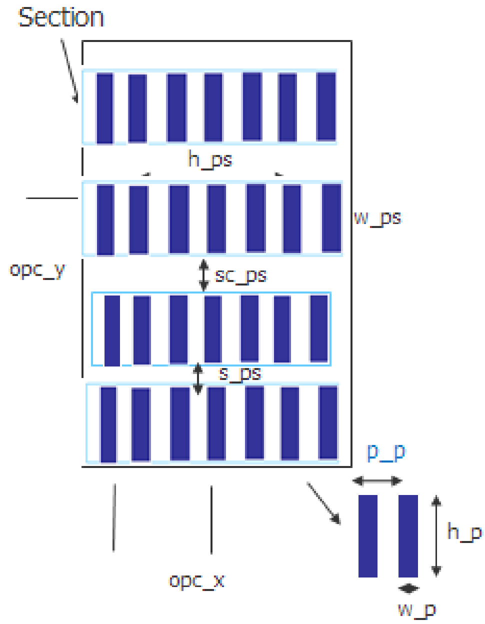
  A required keyword that specifies the pitch for all poly rectangles, in microns.

- **Orientation** {**horiz** | **vert**}

  A required keyword that specifies the orientation for all poly rectangles in the layer, horizontal or vertical. Ensure that this is set perpendicular to the FinFET layer, if present.

Description

The PC_FEATURE keyword creates a poly test pattern layer in the output topo test pattern layout, consisting of one or more sections of parallel rectangles. In the default (pc_rect) mode, all parameters are enabled, allowing you to customize multiple aspects of the output pattern.



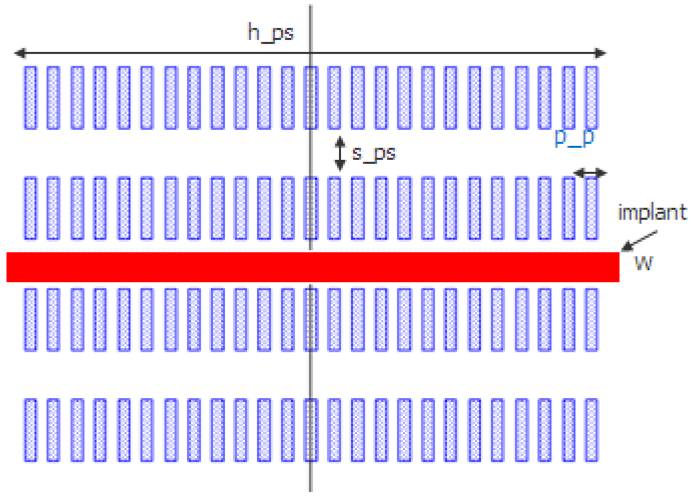
Two templates (pc_planar_oxide and pc_planar_rx) are also provided, which reuse the PC_FEATURE functions with a subset of the keywords to create block-level patterns.

Examples

Example 1 (iso_pc_rect)

A pc_rect type PC_FEATURE over a standard iso test pattern structure using the following keywords results in the pattern shown in Figure 4-1; red is the iso layer and blue is the poly layer. Notice that the set of poly sections is oriented horizontally (OrientationS parameter), but the individual rectangles are oriented vertically (Orientation parameter) in this example.

Figure 4-1. PC_FEATURE iso_pc_rect Example

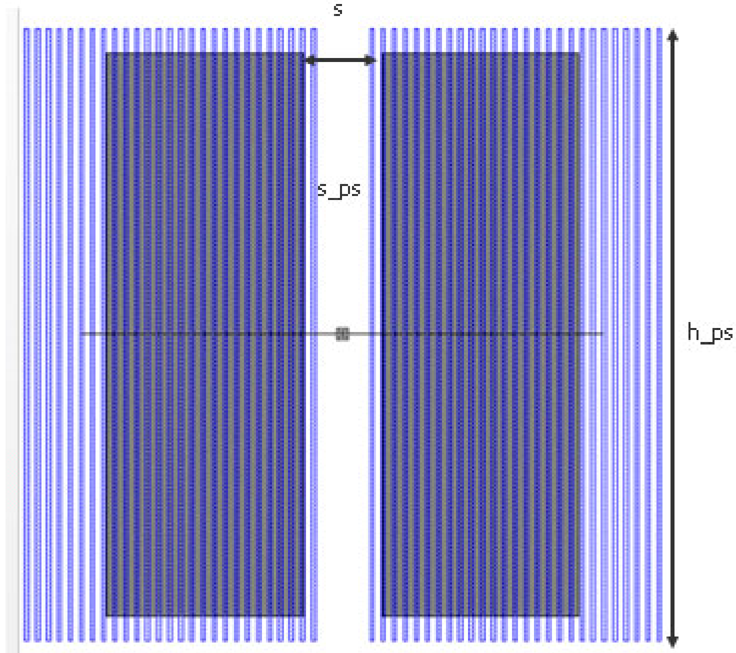



```
FEATURE iso
   Orientation h
   Widths w
PC_FEATURE
   NumSections 4
   OrientationS horiz
   WidthsS w_ps
   HeightsS h_ps
   SpaceS s_ps
   Widths w_p
   Heights h_p
   Pitchs p_p
   Orientation vert
```

Example 2 (inv_iso_pc_rect)

An example showing how the center spacing between two sections fits over an inverse iso feature (Figure 4-2) uses the following keywords:

Figure 4-2. PC_FEATURE inv_iso_pc_rect Example




```
FEATURE inv_iso
   Orientation h
   Spaces s
PC_FEATURE
   NumSections 2
   OrientationS vert
   Offset_X 0
   Offset_Y 0
   WidthsS w_ps
   HeightsS h_ps
   SpaceS s_ps
   Widths w_p
   Heights h_p
   Pitchs p_p
   Orientation vert
```

Related Topics

Creating a Custom Topo Test Pattern

Topo Model Test Patterns

RX_FEATURE Keyword

Test Pattern Command (topo only)

Specifies an active (RX) layer to add to a test pattern layout.

Usage

RX_FEATURE *feature_name*
    [*feature_parameter value*]
    ...

Arguments

- ***feature_name***

  Specifies the name of a topo feature template, as described in "Topo Model Test Patterns" on page 109.

- *feature_parameter value*

  Specifies parameters for the feature specified in the RX_FEATURE argument.

Description

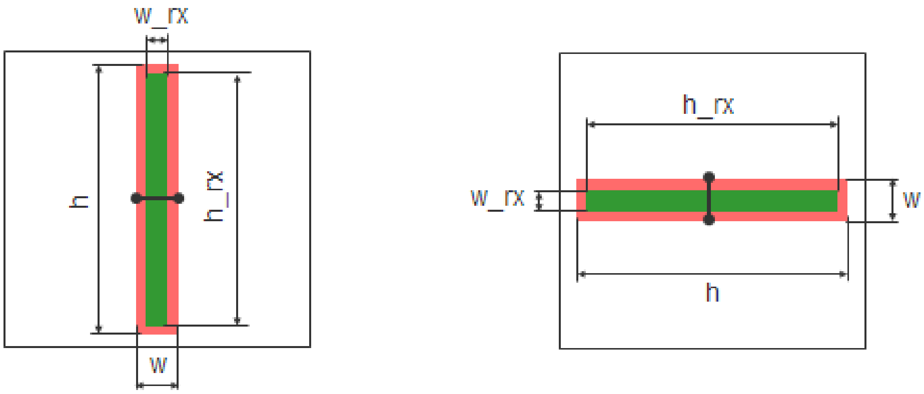
The RX_FEATURE keyword defines an instance of a topo test pattern structure in a test pattern file. The named test structure is placed on the RX (active) layer in the output file when a tepCompile command is run on the file.

The RX pattern uses the same block size as the associated implant pattern by default.

Examples

In the following example, an iso implant layer (red) is paired with an active rx_rect feature (green).

```
FEATURE iso
   Orientation v|h
   Widths w
   Heights h
RX_FEATURE rx_rect
   Widths w_rx
   Heights h_rx
   Orientation v|h
```



Related Topics

Creating a Custom Topo Test Pattern

stack_weight, stack2_weight

Optical model parameter

Defines the stack weight of topographical modeling film stacks relative to other film stacks in an optical model file.

Usage

stack_weight *weight* [stack2_weight *weight2*]

Arguments

- ***weight***

  A required argument for stack_weight that sets the weight of the first film stack. It is used in conjunction with the film2 keyword. The value specified must be between 0.0 and 1.0.

- *weight2*

  A required argument to stack2_weight that sets the weight of the second film stack. It is used in conjunction with the film3 keyword. The value specified must be between 0.0 and 1.0, and *weight2* + ***weight*** must be less than or equal to 1.

Description

These parameters are part of the optical model file. They can only be used when the associated film2 (stack_weight) or film3 (stack2_weight) parameter is also specified. The stacks are weighted according to the following calculation:

Table 4-2. Stack Weight Calculations

first film stack (film)	***weight***
second film stack (film2)	(three stacks) *weight2* or (two stacks) 1 - ***weight***
third film stack (film3)	1 - (***weight*** + *weight2*)

Use the stack_weight parameter to weight transitional optical models. A transitional optical model closer to the first stack would have a ***weight*** value closer to 1.0, such as 0.7; a transitional optical model closer to the second film stack would have a ***weight*** closer to 0.0, such as 0.3.

Tip

One method of creating a set of transitional optical models is to use a baseline optical model that has both film and film2 defined, and only varying stack_weight and the underlying substrate information in order to create different models.

You can optimize ***weight*** and *weight2* using Calibre nmModelflow STACK_WEIGHT_EX*N* and STACK2_WEIGHT_EX*N* parameters.

Examples

Example 1: Two Stacks

The following code has two film stacks: stack 1 (silicon stack) contains only a “Resist” film, and stack 2 (oxide stack) contains “Resist” and “Oxide” films. When calculating the total pupil function, the resist-only stack is given 4x the prominence of the resist-and-oxide stack, since stack_weight sets their relative values to 0.8 and 0.2, respectively.

```
stack_weight 0.8
film 0.280 1.774 -0.035 "Resist"
film2 0.280 1.774 -0.035 "Resist"
film2 0.265 1.56 0.0 "Oxide"
substrate 1.57 -3.565
```

Example 2: Three Stacks

The following code has three film stacks. Stack 1 has a weight of 0.46; stack 2 has a weight of 0.3; and stack 3 has a weight of 0.24.

```
//Stack 1
film 0.73  1.8 -0.01 "Resist"
film 0.003 1.5 -0.00 "Oxide"
film 0.1   1.5 -0.00 "Oxide Finfet"

//Stack 2
film2 0.73  1.8 -0.01 "Resist"
film2 0.003 1.5 -0.00 "Oxide"

//Stack 3
film3 0.73  1.8 -0.01 "Resist"
film3 0.003 1.5 -0.00 "Oxide"
film3 0.1   1.5 -0.00 "Oxide"

stack_weight 0.46
stack2_weight 0.3
```

If the code did not include film3 and stack2_weight, stack2 would have a weight of 0.54 instead of 0.3.

tepCreate topo

Test Pattern Commands

Generates a Version 11 topo test pattern and associated spreadsheet file.

Usage

tepCreate topo [-cdwidth *cdwidth*] [-cdspace *cdspace*]
[-finfetwidth *fwidth* -finfetspace *fspace* [-finfetorientation {v | h}]]
[-gdsfile *gdsfile*] [-ssfile *ssfile*] [-ggfile *ggfile*] [-csvfile *csvfile*] [-markers *marker*]
[-topcellname *cellname*] [-dbinnm *dbunit*] [-gridinnm *grid*]

Arguments

- -cdwidth *cdwidth*

  An optional keyword/value pair defining the implant target CD width, in nm. The default is 140.

- -cdspace *cdspace*

  An optional keyword/value pair defining the implant target CD space, in nm. The default is 140.

- -finfetwidth *fwidth* -finfetspace *fspace* [-finfetorientation {v | h}]

  An optional keyword/value set defining the FinFET target width and space in nm, and orientation (vertical or horizontal). No FinFET layer is added by default; when a FinFET layer is added, it is in vertical orientation by default.

- -gdsfile *gdsfile*

  The name of the GDS output file, specified as either an absolute path or relative to the directory from which you invoke the Calibre WORKbench application. The default is system-generated, created by concatenating test pattern parameters: "*topo.<cdwidth>.<cdspace>.<dbunit>.<grid>.gds*."

- -ssfile *ssfile*

  The name of the spreadsheet output file, specified as either an absolute path or relative to the directory from which you invoke the Calibre WORKbench application. The default is system-generated, created by concatenating test pattern parameters: "*topo.cdwidth.cdspace.dbunit.grid.ss*."

- -ggfile *ggfile*

  An optional keyword/value pair specifying the name of a gauge file to be output with the test pattern structure. This file is intended to be used with CM1 Center or the Calibre nmModelflow tool. It is not generated by default.

Note

While it is possible to convert spreadsheet files to gauge files with the Calibre WORKbench ss2gauge command, it is recommended that you use the -ggfile option to create a gauge file and fill in the measurement values in the gauge file that are taken from the sample spreadsheet file.

- -csvfile *csvfile*

  An optional keyword/value pair specifying the name of a comma-separated value (CSV) file output for use with the Calibre Metrology Interface. It is not generated by default.

- -markers *marker*

  An optional keyword/value pair indicating the type of marker pattern to include in the test pattern. *marker* must be one of {none | cross | oddeven | sem}. The default is none.

- -topcellname *cellname*

  An optional keyword/value pair that names the pattern structure cell. The default is TOP.

- -dbinnm *dbunit*

  An optional keyword/value pair defining the database unit, in nm. The default is 1.

- -gridinnm *grid*

  An optional keyword/value pair defining the grid unit, in nm. The default is 5.

Examples

The following command creates a topo test pattern using the default values with an additional gauge file output. The GDS and spreadsheet output files are *topo.140.140.1.5.gds* and *topo.140.140.1.5.ss*, respectively.

```
% tepCreate topo -ggfile topo_gauges.gg
```

topo_model Format

model_layer file block

Describes topographical models inside model-layer files or litho model files.

Defines a topographical model as a block inside a litho model file. The litho model file is optimized with Calibre nmModelflow. The information defined in the topographical model instructs Calibre WORKbench to combine multiple optical model characteristics in order to model transitional exposure effects.

The optimized topo_model interpolates multiple transitional optical models between multiple film stacks (film and film2, and potentially film3).

Format

```
topo_model '{'
version 1
modeltype TOPO
transit te1 te2 …teT
from umin
to umax
[dsigma value] | [density {exposure  pe1 [pe2 … peT]} |
   {layer1 layer2 … layerT}]
[property name]
[mtype {efield | zeroorder}]
[dupsize dupsize_value]
[transit_dose dte1 dte2 … dteT]
[csigma csigma_value]
[cterm cindex cterm_value]
[cquadratic_mult cquad_value]
[fsigma fsigma_value]
[fterm findex fterm_value]
[fwsigma fwsigma_value]
[fwterm fwindex fterm_value]
[fwupsize fwupsize_value]
[fwasym fwasym_value]
[gsigma gsigma_value]
[gterm gindex gterm_value]
[gupsize gupsize_value]
[isigma isigma_value]
[iterm iindex iterm_value]
[iupsize iupsize_value]
[obiasinit globalrxbias]
[obiasmax obiassaturation_value]
[obiaswsat obiassaturation_width]
[oterm oindex otermvalue]
[squadratic_mult squad_value]
[squadratic_mult_im squadim_value]
[ssigma ssigmavalue]
sterm sindex sterm_value
[stermim sindex stermim_value]
[supsize supsize_value]
[tsigma tsigmavalue]
[tsurface {active | field}]
tterm tindex tterm_value
[ttermim tindex ttermim_value]
[t2sigma t2sigma_value]
[t2term t2index t2value]
[t4sigma t4sigma_value]
[t4term t4index t4value]
[vsigma vsigma_value]
[vterm vindex vterm_value]
[c2term c2index c2term_value]
[c2quadratic_mult c2quad_value]
[c2sigma c2sigma_value]
[s2term s2index s2term_value]
[s2quadratic_mult s2quad_value]
[s2sigma s2sigma_value]
[so2sigma so2sigma_value]
[so2term so2index so2term_value]
[to2sigma to2sigma_value]
```

```
[to2surface {active | field}]
[to2term to2index to2term_value]
[t2o2sigma t2o2sigma_value]
[t2o2term t2o2index t2o2term_value]
[t4o2sigma t4o2sigma_value]
[t4o2term t4o2index t4o2term_value]
[execute_mode execute_mode_number]
[linearity {linear | efield}]
'}'
```

Parameters

Note

Some parameters are expressed in "P1" units. One P1 unit corresponds to 1 optical wavelength in photoresist. For example, a 248 nm laser illumination lambda and a 1.75 index of refraction results in a P1 unit that is equal to 141.7 nm.

- version 1

  A required parameter specifying the model version. The only accepted value is 1.

- modeltype TOPO

  A required parameter specifying the model type. The only accepted value for topographical models is TOPO.

- **transit *te1 te2 … teT***

  A required parameter that references topographical optical models by number. Each number used must reference one of the optical models defined with the optical_model statement inside the litho model.

  - For single densities, the ordering of optical models is required. The oxide exposure must be first and the silicon exposure must be second. The topmost layer exposure must be last.

  - For multiple densities, the only ordering requirements are that each transit value specified corresponds with the order of the specified density exposures.

- **from *umin***

  to *umax*

  A required pair of parameters specifying the endpoints of the optical model list to be sampled by transitional exposures. ***umin*** must be less than ***umax***, and both must be non-negative integers.

- dsigma *value*

  An optional value used to specify the density diffusion length in P1 units, used for single density configurations. In this case, converting a diffusion length from um to P1 units uses the formula:

  ```
  P1 = sqrt(2) * length_um * K_ref / optical_lambda
  ```

where *K_ref* is the resist refractive index. The default value is sqrt(2)*0.05*1.75/0.248 = 0.498966, corresponding to 50nm.

- density {exposure *pe* [*pe2* … *peT*]} | {*layer1* [*layer2* … *layerT*]}

Note
This option is only used for post-PC configurations. Single density configurations should use dsigma instead.

An optional parameter that references density optical models by exposure number or by underlying layer name or number. These references correspond to the values specified for transit *te1*… *teT*. Calibre WORKbench calculates density exposures as convolutions of the specified wafer stack layers with the Gaussian kernel. You specify the kernel parameters in a density optical model.

  - For multiple density exposures, you must specify a single negative value of the density parameter as part of the list. This negative density value indicates that the density corresponds to the "transitional type" exposure that belongs to a field area, *usually the oxide stack.* The field density is calculated using the following formula:

    $$u_{-pef}(x, y) = 1 - u_{pef}(x, y)$$

    An example for two density exposures might have the following code:

    ```
    transit 0 1
    density exposure -2 2
    from 0
    to 1
    ```

    where:

    - The 0th transitional exposure corresponds to oxide optics.
    - The 1st transitional exposure corresponds to silicon optics.
    - Masks 0 and 1 are specified as the implant layer.
    - Mask 2 is specified as the RX layer.

    The density of the 0th transitional exposure is modulated by the density exposure 1-u(x,y) and the 1st transitional exposure is modulated by the density exposure u(x,y).

  - For multiple density underlying layers (which is different from the underlying layers specification in the model-layer file), specify the type of layer for each transit exposure using the following keywords:

    - oxide — Specifies an underlying oxide (field) stack.
    - oxide2 — Specifies an underlying oxide2 (field) stack.
    - active — Specifies an underlying active silicon (RX) stack.

- active2 — Specifies an underlying active2 (SOI) stack.
- poly — Specifies an underlying poly stack.
- finfet — Specifies an underlying FinFET stack.

An example for density underlying layers thusly resembles the following code:

```
transit 0 1 2
density oxide active poly
from 0
to 1
```

- property *name*

  An optional parameter specifying an optical property (such as beamfocus or stack_weight) to be sampled for the transitional exposures between the ***umin*** and ***umax*** values.

- mtype efield | zeroorder

  An optional parameter specifying that all signals are multiplied by the electrical field (efield argument) or by the zeroth order harmonic of the mask (zeroorder argument).

- dupsize *dupsize_value*

  An optional parameter that upsizes the density layer, usually the RX layer, by the *dupsize_value* in P1 units. The upsize function happens before convolution. The default is 0.

  Note

  The dupsize option is only used for internal density calculations. It is not allowed when calibrating external density.

- transit_dose *dte1 dte2 … dteT*

  An optional parameter specifying a list of real numbers, one per dose, that are used as transitional intensity multipliers.

- csigma *csigma_value*

  An optional parameter specifying the diffusion length of the corner diffraction kernels in P1 units. The default value is 1.

- cterm *cindex cterm_value*

  An optional argument specifying the corner diffraction of a single kernel. Each cterm kernel gets its own line.

  cindex is an integer value between 0 and 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  cterm_value is a multiplier for the corner diffraction kernel used with the corresponding *cindex*.

- cquadratic_mult *cquad_value*

  An optional argument specifying a multiplier for the quadratic part of the corner diffraction signal. The default is 0.

- fsigma *fsigma_value*

  An optional parameter that specifies the size in P1 units for all FinFET kernels. The default is 1.

- fterm *findex fterm_value*

  An optional parameter that specifies the multiplier for the *n*th term for a FinFET scattering signal kernel, corresponding to *findex*. This kernel is only used when underlying finfet layers are present. The kernels affect “FinFET over active” layout regions and “active not FinFET” regions of active layout features that interact with FinFET structures.

- fwasym *fwasym_value*

  An optional parameter that specifies the value of the asymmetry multiplier for the FinFET sidewall signal. *fwasym_value* is a dimensionless floating point integer between 0 and 1, and defines the power ratio of the signal sources between the horizontal and vertical fin sidewalls. The power of sources on the vertical fin sidewalls is computed as (1-*fwasym_value*) for the linearity intensity, and sqrt(1-*fwasym_value*^2) for the linearity efield.

  The default value is 1, which assumes horizontal fin sidewalls only.

  Setting *fwasym_value* to 0 assumes vertical fin sidewalls only.

  Values between 0 and 0.5 correspond to cases of continuous vertical sidewalls and perforated horizontal sidewalls.

  Values between 0.5 and 1.0 correspond to cases of perforated vertical sidewalls and continuous horizontal sidewalls.

- fwsigma *fwsigma_value*

  An optional parameter that specifies the size in P1 units for all FinFET sidewall scattering kernels. The default is 1.

- fwterm *fwindex fterm_value*

  An optional parameter that specifies the multiplier for the *n*th term for a FinFET sidewall scattering signal kernel, corresponding to *findex*. This kernel is only used when underlying finfet layers are present. The kernels affect “FinFET over active” layout regions and “active not FinFET” regions of active layout features that interact with FinFET structures.

- fwupsize *fwupsize_value*

  An optional parameter that moves the FinFET sidewall layer outward in P1 units by the specified amount before topo signal calculation. Must be positive. The default is 0.

- gsigma *gsigma_value*

  An optional parameter that specifies the size of the gate scattering kernels in P1 units. The default is 1.

- gterm *gindex gterm_value*

  An optional parameter that specifies a multiplier for the kernel of the gate scattering signal corresponding to *gindex*. These kernels affect only layout regions with poly over active (the PC stack).

  gindex is an integer value between 0 and 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  gterm_value is the multiplier.

- gupsize *gupsize_value*

  An optional parameter that upsizes the gate part of the poly layer by *gupsize_value* in P1 units. The default is 0. Gate polygons are detected as the result of a logical AND between poly and RX polygons.

- isigma *isigma_value*

  An optional parameter that specifies the size of the poly interconnect kernels in P1 units. The default is 1.

- iterm *iindex iterm_value*

  An optional parameter that specifies a multiplier for the kernel of the poly interconnect scattering signal corresponding to *iindex*. These kernels affect regions that are not poly over active.

  iindex can be between 0 and 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  iterm_value is the multiplier used with the corresponding *iindex*.

- iupsize *iupsize_value*

  An optional parameter that upsizes the interconnect part of the poly layer by *iupsize_value* in P1 units. The default is 0. Interconnect polygons are detected as the result of a logical NOT between poly and RX polygons.

- obiasinit *globalrxbias*

  An optional parameter specifying the upsize value in P1 units. This value is applied to all original RX and SOI features before internal density simulations and topo signal computations.

  After the obiasinit operation completes, all edges (or part of edges) of original RX and SOI features adjacent to the oxide layer are moved by the distance *globalrxbias*. The *globalrxbias* value can be positive or negative. Specifying a positive value means that an RX or SOI edge adjacent to an oxide feature moves to the inside of the RX or SOI feature; specifying a negative value moves it to the outside of the original feature. The typical absolute value of *globalrxbias* is between 0.15 and 0.65 P1 units. The recommended range is from -0.6 to 0.6. The default is 0.

- obiasmax *obiassaturation_value*

  An optional parameter that specifies the saturation value of the variable part of the oxide bias in P1 units. It assumes that any RX or SOI edge (or part of an edge) adjacent to an oxide layer forms a trench with a paired RX or SOI edge (or part of an edge) that is also adjacent to an oxide layer. If any RX or SOI edge has no paired RX or SOI edge, then it is considered that the corresponding trench has infinite width.

  The absolute value of the variable oxide bias increases monotonically with the width reaching a saturation value of *obiassaturation_value* at the width of *obiassaturation_width*.

  The *obiassaturation_value* can be positive or negative. Specifying a positive value means that an RX or SOI edge (or partial edge) moves to the inside of the RX or SOI feature; specifying a negative value moves it to the outside of the original feature. A zero value means that there is no variable part of the oxide bias. The recommended range is from -0.6 to 0.6. The default is 0.

- obiaswsat *obiassaturation_width*

  An optional parameter that specifies the width in P1 units at which the variable trench bias stops increasing.

  Trenches that are wider than *obiassaturation_width* are biased by *obiassaturation_value*. The *obiassaturation_width* value must be positive. This argument is only used when *obiassaturation_value* is not equal to zero. The recommended range of *obiassaturation_width* is between 0.5 and 3.0. The default is 2.0 P1 units.

- oterm *oindex oterm_value*

  An optional parameter that specifies a multiplier for the kernel of the oxide bias polynomial corresponding to *oindex*.

  oindex is an integer value between 0 and 2, inclusive. Each oterm kernel must be specified on its own line.

  oterm_value is the multiplier.

- squadratic_mult *squad_value*

  An optional parameter specifying a multiplier for the quadratic part of the sidewall reflection signal. The default is 0.

- squadratic_mult_im *squadim_value*

  An optional parameter specifying a multiplier for the quadratic imaginary part of the sidewall reflection signal. The default is 0.

- ssigma *sigmavalue*

  An optional parameter specifying the diffusion length of the sidewall reflection kernels in P1 units. The default is 1.

- **sterm *sindex sterm_value***

  A required parameter specifying the sidewall reflection of a single kernel. Each sterm kernel gets its own line.

  sindex is an integer value between 0 and 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  sterm_value is a multiplier for the sidewall kernel used with the corresponding *sindex*.

- stermim *sindex stermim_value*

  An optional parameter specifying the imaginary part of the sidewall signal for the *sindex*'th kernel. It is added to the final intensity.

- supsize *supsize_value*

  An optional parameter specifying the upsize value for the active layer to be used in sidewall reflection signal calculation. Active layer edges are shifted outwards by the specified value in P1 units before sidewall signals are added.

- tsigma *tsigmavalue*

  An optional parameter specifying the diffusion length of the trench radiation kernel in P1 units. The default is 1.

- tsurface {active | field}

  An optional parameter designating the region in which to apply trench radiation signal modifiers to the image intensity. "active" specifies the RX region, and "field" specifies that it is not the RX region. The default value is "field".

- **tterm *tindex tterm_value***

  A required parameter specifying a multiplier for the kernel of a trench radiation signal.

  tindex is an integer value between 0 and 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  tterm_value is the multiplier value.

- ttermim *tindex ttermim_value*

  An optional parameter specifying the imaginary part of the trench radiation signal for the *tindex*'th kernel. It is added to the final intensity.

- t2sigma *t2sigma_value*

  An optional parameter specifying the diffusion length of the trench2 radiation signal kernel in P1 units.

- t2term *t2index t2term_value*

  An optional parameter specifying a multiplier for the kernel of a trench2 radiation signal. It describes the height along the trench width with a Chebyshev second order polynomial.

t2index is an integer value between 0 and 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

t2term_value is the multiplier value.

- t4sigma *t4sigma_value*

  An optional parameter specifying the diffusion length of the trench4 radiation signal kernel in P1 units. The default value is 1.

- t4term *t4index t4term_value*

  An optional parameter specifying a multiplier for the kernel of a trench4 radiation signal. It describes the height along the trench width with a Chebyshev fourth order polynomial.

  t4index is an integer value between 0 and 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  t4term_value is the multiplier value.

- vterm *vindex vterm_value*

  An optional parameter specifying a kernel multiplier for a vertical corner diffraction signal.

  vindex is an integer number from 0 to 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  vterm_value is a multiplier for the vindex vertical corner diffraction signal kernel.

- vsigma *vsigma_value*

  An optional parameter specifying the diffusion length of the vertical corner diffraction kernel in P1 units. The default is 1.

- c2term *c2index c2term_value*

  An optional parameter used for corner2 signal kernels (for SOI processes). *c2index* is an integer from 0 to 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  c2term_value is the multiplier for the *c2index* corner2 diffraction signal kernel.

- c2quadratic_mult *c2quad_value*

  An optional parameter that specifies the multiplier in front of the quadratic part of the corner2 diffraction signal (for SOI processes). Default is 0.

- c2sigma *c2sigma_value*

  An optional parameter that specifies the diffusion length of the corner2 diffraction kernels in P1 units (for SOI processes). Default is 1.

- s2term *s2index s2term_value*

  An optional parameter that specifies a sidewall2 reflection signal (for SOI processes).

  s2index is an integer *i* from 0 to 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  s2term_value is a multiplier for the *s2index* of the *i*th sidewall2 kernel. The sidewall2 signal is added to the final intensity in linear and quadratic fashions.

- s2quadratic_mult *s2quad_value*

  An optional parameter that specifies the multiplier for the quadratic part of the sidewall2 reflection signal (for SOI processes). Default is 0.

- s2sigma *s2sigma_value*

  An optional parameter that specifies the diffusion length of the sidewall2 reflection kernels in P1 units (for SOI processes). Default is 1.

- so2sigma *so2sigma_value*

  An optional parameter that specifies the diffusion length of the sidewallo2 reflection kernels in P1 units (for STI processes). Default is 1.

- so2term *so2index so2term_value*

  An optional parameter that specifies a sidewallo2 reflection signal (for STI processes).

  so2index is an integer *i* from 0 to 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  so2term_value is a multiplier for the *so2index* of the *i*th sidewallo2 kernel. The sidewallo2 signal is added to the final intensity in linear and quadratic fashions.

- to2sigma *to2sigma_value*

  An optional parameter that specifies the diffusion length of the trencho2 radiation kernel in P1 units (for STI processes). Default is 1.

- to2surface {active | <u>field</u>}

  An optional parameter that specifies the region where the trench radiation signal is added to the image intensity.

  active — Applies to the RX layout region.

  <u>field</u> — Applies to the not-RX layout region (default).

- to2term *to2index to2term_value*

  An optional parameter that specifies a trencho2 radiation signal (for STI processes).

  to2index is an integer *i* from 0 to 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

to2term_value is a multiplier for the *to2index* of the *i*th trencho2 kernel. The trencho2 signal is added to the final intensity in linear and quadratic fashions.

- to2o2sigma *to2sigma_value*

  An optional parameter that specifies the diffusion length of the trench2o2 radiation kernels in P1 units (for STI processes). Default is 1.

- t2o2term *t2o2index t2o2term_value*

  An optional parameter that specifies a trench2o2 radiation signal (for STI processes). For this signal, the trench form (height) along the trench width is defined with a Chebyshev polynomial of the 2nd order.

  t2o2index is an integer *i* from 0 to 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  t2o2term_value is a multiplier for the *t2o2index* of the *i*th trench2o2 kernel. The trench2o2 signal is added to the final intensity in linear and quadratic fashions.

- t4o2sigma *t4o2sigma_value*

  An optional parameter that specifies the diffusion length of the trench4o2 radiation kernels in P1 units (for STI processes). Default is 1.

- t4o2term *t4o2index t4o2term_value*

  An optional parameter that specifies a trench4o2 radiation signal (for STI processes). For this signal, the trench form (height) along the trench width is defined with a Chebyshev polynomial of the 4th order.

  t4o2index is an integer *i* from 0 to 23, inclusive (24 signals). The number of terms for each signal can be up to 24 for fasttopo 2 mode, but the recommended (default) number of terms is 8 (index of 7). You cannot use index values higher than 7 for fasttopo 0 mode.

  t4o2term_value is a multiplier for the *t4o2index* of the *i*th trench4o2 kernel. The trench4o2 signal is added to the final intensity in linear and quadratic fashions.

- execute_mode {<u>0</u> | 1}

  An optional parameter that sets the execution mode for the topo model.

  0 — Specifies “ordinary” mode, where standard routines for separate fast fourier transform (fft) operations are used for topo signal calculations. This is the default setting.

  1 — Specifies “transposed” mode, where special routines that aggregate fft operations and spectra processing are used. Usually “transposed” mode is 20-30% faster than “ordinary” mode.

- linearity {intensity | <u>efield</u>}

  An optional parameter that controls how the signals are added to the intensity.

  intensity — Adds signals linearly to intensity.

efield — Adds signals linearly to square root of the intensity, then the result is squared. The default is “efield”.

Examples

This example describes a topo_model for a pre-PC stack. Its sterms and tterms have already been optimized, since the sterms and tterms have non-zero values.

```
transit 0 1
from 0
to 1
transit_dose 1 1
squadratic_mult 0.0031580031
sterm 0 -0.0015999079
sterm 1 0.0047998875
sterm 2 -0.0015999079
sterm 3 -0.0073601007
sterm 4 0.0041601658
sterm 5 -0.0047995001
sterm 6 0.0015990734
sterm 7 0.0022394657
tsurface active
tterm 0 0.1200005
tterm 1 -0.12320461
tterm 2 0.043184131
tterm 3 -0.024000928
tterm 4 -0.12000256
tterm 5 -0.02400133
tterm 6 -0.15520051
tterm 7 -0.12000051
```

Topographical Modelform Parameters (topomf)

Calibre nmModelflow command

Specifies a topo modelform that specifies a template set of signals as an ease of use convention.

Usage

mdf param

…

topomf ***string***
[topomf_termrange {auto | *value*} [{auto | *value*}]]
…
[topomf_cupsize { auto | *value*} [{auto | *value*}]]
[topomf_cquad_mult {auto | *value*} [{auto | *value*}]]
[topomf_ctermnum *value*]
[topomf_ctermrange {auto | *value*} [{auto | *value*}]]
[topomf_density_exposure *value1 value2* …]
[topomf_dsigma {auto | *value*} [{auto | *value*}]]
[topomf_dupsize {auto | *value*} [{auto | *value*}]]
[topomf_fasym {auto | *value*} [{auto | *value*}]]
[topomf_fsigma {auto | *value*} [{auto | *value*}]]
[topomf_ftermnum *value*]
[topomf_ftermrange {auto | *value*} [{auto | *value*}]]
[topomf_fupsize {auto | *value*} [{auto | *value*}]]
[topomf_fwasym {auto | *value*} [{auto | *value*}]]
[topomf_fwsigma {auto | *value*} [{auto | *value*}]]
[topomf_fwtermrange {auto | *value*} [{auto | *value*}]]
[topomf_fwtermnum *value*]
[topomf_fwupsize {auto | *value*} [{auto | *value*}]]
[topomf_gsigma {auto | *value*} [{auto | *value*}]]
[topomf_gtermnum *value*]
[topomf_gtermrange {auto | *value*} [{auto | *value*}]]
[topomf_gupsize {auto | *value*} [{auto | *value*}]]
[topomf_isigma {auto | *value*} [{auto | *value*}]]
[topomf_itermnum *value*]
[topomf_itermrange {auto | *value*} [{auto | *value*}]]
[topomf_iupsize {auto | *value*} [{auto | *value*}]]
[topomf_obiasinit {auto | *value*} {auto | *value*}]
[topomf_obiasmax {auto | *value*} {auto | *value*}]
[topomf_obiaswsat {auto | *value*} {auto | *value*}]
[topomf_otermnum *value*]
[topomf_simtermnum *value*]
[topomf_simtermrange {auto | *value*} [{auto | *value*}]]
[topomf_squad_mult {auto | *value*} [{auto | *value*}]]

[topomf_ssigma {auto | *value*} [{auto | *value*}]]
[topomf_stermnum *value*]
[topomf_stermrange {auto | *value*} [{auto | *value*}]]
[topomf_timtermnum *value*]
[topomf_timtermrange {auto | *value*} [{auto | *value*}]]
[topomf_transit *value1 value2* …]
[topomf_transit_dose *value1 value2* …]
[topomf_tsigma {auto | *value*} [{auto | *value*}]]
[topomf_tsurface *value*]
[topomf_ttermnum *value*]
[topomf_ttermrange {auto | *value*} [{auto | *value*}]]
[topomf_t2sigma {auto | *value*} [{auto | *value*}]]
[topomf_t2termnum *value*]
[topomf_t2termrange {auto | *value*} [{auto | *value*}]]
[topomf_t4sigma {auto | *value*} [{auto | *value*}]]
[topomf_t4termnum *value*]
[topomf_t4termrange {auto | *value*} [{auto | *value*}]]
[topomf_vsigma {auto | *value* } [{auto | *value*}]]
[topomf_vtermnum *value*]
[topomf_vtermrange {auto | *value*} [{auto | *value*}]]
[topomf_c2termrange {auto | *value*} [{auto | *value*}]]
[topomf_s2termrange {auto | *value*} [{auto | *value*}]]
…

Arguments

- **topomf *string***

  A required argument that specifies the topo model form to be calibrated. The name of the model form defines which topo signals should be used to generate model and optimization settings. Each signal is signified with a code number or letter (or two letters in the case of FinFET sidewall, "Fw"), and the available modelforms are specific combinations of the codes. For example, "topo3STGI" is "Post-PC, density underlying, sidewall, trench, gate, and interconnect signals."

  Note

  Pre-2017.2 versions of the topomf option listed additional signals, such as "T2" (trench2), "Sim"(sidewall imaginary), and "Tim" (trench imaginary). Modelforms that included those signals (such as "topo2SSim") are no longer accepted, but the individual signals can still be added as custom signals.

Table 4-3. topomf Signal Codes

Code	Description
2	Pre-PC
3	Post-PC, density underlying

Table 4-3. topomf Signal Codes (cont.)

Code	Description
4	Post-PC, density underlying (for SOI)
5	Post-PC, density underlying (for SOI and Oxide2)
S	Sidewall
S2	Sidewall2 (for SOI)
SO2	SidewallO2 (for STI)
T	Trench
TO2, T2O2, T4O2	TrenchO2, Trench2O2, Trench4O2
C	Corner
C2	Corner2 (for SOI)
F	FinFET
Fw (lowercase W)	FinFET sidewall
V	Vertical corner
G	Gate
I	Interconnect

Table 4-4. Supported Topo Modelform Codes

topo2	topo2S	topo2ST	topo2CS
topo2CST	topo2STF	topo2STFw	topo2STFFw
topo2CSTF	topo2CSTFw	topo2CSTFFw	topo2CSTFV
topo2SGI	topo2STGI	topo3	topo3STGI
topo3STGIF	topo3STGIFw	topo3STGIFFw	topo3CSTGI
topo3CSTGIF	topo3CSTGIFw	topo3CSTGIFFw	topo3STGIV
topo3STGIFV	topo3CSTGIV	topo3CSTGIFV	topo3CSTGIFwV
topo3CSTGIFFwV	topo3STS2	topo3CSTC2S2	topo3CSTC2S2V
topo3CSTC2S2GI	topo4CSTGIC2S2	topo4STS2O2TO2	topo4STVS2SO2TO2
topo4CSTC2S2SO2TO2	topo4CSTT2C2S2SO2TO2T2O2	topo4CSTVC2S2SO2TO2	topo4CSTT2VC2S2SO2TO2T2O2
topo4CSTGIC2S2SO2TO2	topo4CSTGIVC2S2SO2TO2	topo5CSTGIC2S2SO2TO2	topo5CSTGIVC2S2SO2TO2

Argument Configurations

- For most of the following arguments, three types of argument configurations are permitted:
    - Single *value* (for example, topomf_dsigma 1.1) — Sets the term to be the specified *value*, and the term is not calibrated. "auto" may not be used in single *value* mode.
    - Double *value* (for example, topomf_dsigma 0.7 1.3) — Sets the search values explicitly to the specified range.
    - Mixed *value* and "auto", or both "auto" (for example, topomf_dsigma 0.7 auto) — The "auto" keyword tells the tool to use the default search arguments for the range *value*.
- topomf_termrange{auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for every topo term's arguments, like sterm, tterm, and so on. This is a generic term configuration argument; when you explicitly define specific terms, it overrides this argument for that term.
- topo_cupsize {auto | *value*} [{auto | *value*}]

  An optional argument that specifies the lower and upper bounds of search intervals for the cupsize parameter.
- topomf_cquad_mult {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the cquadratic_mult parameter. The keyword auto tells the tool to use the default search intervals.
- topomf_ctermnum *value*

  An optional argument that specifies the created number of cterms.
- topomf_ctermrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies the lower and upper bounds of search intervals for the cterm argument. Using this argument overrides the value for topomf_termrange.
- topomf_density_exposure *value1 value2* …

  An optional argument that specifies values for the density exposure argument in the topo model.

  For the topo modelform, a single density exposure case is assumed, and in those cases, this argument is not used. However, you can specify multiple densities by passing the topomf_density_exposure argument with multiple values.
- topomf_density_underlying *list_of_values*

  An optional argument used only for topo3 models (because it requires multiple densities, and topo2 models are single density). Use this option to specify the layer names written to the density underlying option in the final output topo model. The default list is:

  ```
  -RX RX PC
  ```

- topomf_dsigma {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the dsigma argument.

- topomf_dupsize {auto | *value*} [{auto | *value*}]

  An optional argument that specifies the lower and upper bounds of search intervals for the dupsize argument.

- topomf_fasym {auto | *value*} [{auto | *value*}]

  An optional argument that specifies the lower and upper bounds of search intervals for the fasym parameter.

- topomf_fsigma {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the fsigma argument.

- topomf_ftermnum *value*

  An optional argument that specifies the created number of fterms.

- topomf_ftermrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the fterm arguments.

- topomf_fupsize {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the fupsize argument.

- topomf_fwasym {auto | *value*} [{auto | *value*}]

  An optional argument that specifies the lower and upper bounds of search intervals for the fwasym parameter.

- topomf_fwsigma {auto | *value*} [{auto | *value*}]

  Use this option to specify lower and upper bounds of search intervals for the fwsigma parameter.

- topomf_fwtermrange {auto | *value*} [{auto | *value*}]

  Use this option to specify lower and upper bounds of search intervals for the fwterm parameters.

- topomf_fwtermnum *value*

  Use this option to specify the created number of fwterms.

- topomf_fwupsize {auto | *value*} [{auto | *value*}]

  Use this option to specify lower and upper bounds of search intervals for the fwupsize parameter.

- topomf_gsigma {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the gsigma argument.

- topomf_gtermnum *value*

  An optional argument that specifies the created number of gterms.

- topomf_gtermrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the gterm arguments.

- topomf_gupsize {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the gupsize argument. The default is no upsize (gupsize = 0).

- topomf_isigma {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the isigma argument.

- topomf_itermnum *value*

  An optional argument that specifies the created number of iterms.

- topomf_itermrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the iterm arguments.

- topomf_iupsize {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the iupsize argument. The default is no upsize (iupsize = 0).

- topomf_obiasinit {auto | *value*} {auto | *value*}

  Use this option to specify the lower and upper bounds of the obiasinit topo model parameter.

- topomf_obiasmax {auto | *value*} {auto | *value*}

  Use this option to specify the lower and upper bounds of the obiasmax topo model parameter.

- topomf_obiaswsat {auto | *value*} {auto | *value*}

  Use this option to specify the lower and upper bounds of the obiaswsat topo model parameter.

- topomf_otermnum *value*

  Use this option to specify the number of oxide bias terms for optimization.

- topomf_simtermnum *value*

  An optional argument that specifies the created number of imaginary sterms.

- topomf_simtermrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the imaginary sterm arguments.

- topomf_squad_mult {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the squadratic_mult argument.

- topomf_ssigma {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the ssigma argument.

- topomf_sternmnum *value*

  An optional argument that specifies the created number of sterms.

- topomf_stermrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the sterm arguments.

- topomf_timtermnum *value*

  An optional argument that specifies the created number of imaginary tterms.

- topomf_timtermrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for imaginary tterm arguments.

- topomf_transit *value1 value2 …*

  An optional argument that specifies values for the transit argument in the topo model.

- topomf_transit_dose *value1 value2 …*

  An optional argument that specifies values for the transit_dose argument in the topo model. At least two values must be specified.

- topomf_tsigma {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for the tsigma argument.

- topomf_tsurface *value*

  An optional argument that specifies the value of the tsurface argument in the topo model.

- topomf_ttermnum value

  An optional argument that specifies the created number of tterms.

- topomf_ttermrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for tterm arguments.

- topomf_t2sigma {auto | *value*} [{auto | *value*}]

  An optional argument that specifies the optimization range for the t2term sigma parameter.
- topomf_t2termnum *value*

  An optional argument that specifies the created number of t2terms.
- topomf_t2termrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for t2term arguments.
- topomf_t4sigma {auto | *value*} [{auto | *value*}]

  An optional argument that specifies the optimization range for the t4term sigma parameter.
- topomf_t4termnum *value*

  An optional argument that sets the number of t4term signals to be optimized. The default value is zero.
- topomf_t4termrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies lower and upper bounds of search intervals for t4term arguments.
- topomf_vsigma {auto | *value*} [{auto | *value*}]

  An optional argument that specifies the diffusion length of the vertical corner diffraction kernels in P1 units. The default value is 1.
- topomf_vtermnum *value*

  An optional argument that sets the number of vertical corner signals to be optimized. The default value is zero.
- topomf_vtermrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies the optimization range for all vterms. This option overrides the default range value.
- topomf_c2termrange {auto | *value*} [{auto | *value*}]

  An optional argument that specifies the optimization range for all corner2 terms. This option overrides the default range value.
- topomf_s2termrange {auto | *value*} [{auto | *value*}]

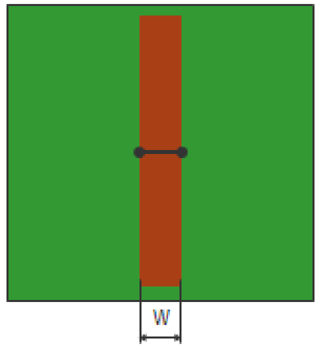
  An optional argument that specifies the optimization range for all sidewall2 terms. This option overrides the default range value.

Topo Model Test Patterns

The topo model test patterns are an extension of the standard test pattern formats described in the *Calibre WORKbench User's and Reference Manual*.

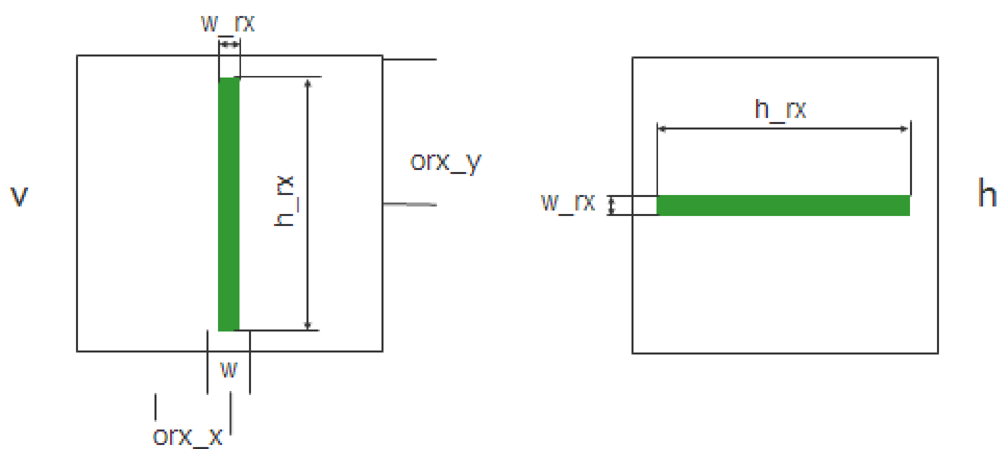
rx_planar

Adds a feature to the active layer in the same size as the block size. The green in the figure is the active layer. (In this figure, w is the width of the implant.)



rx_rect

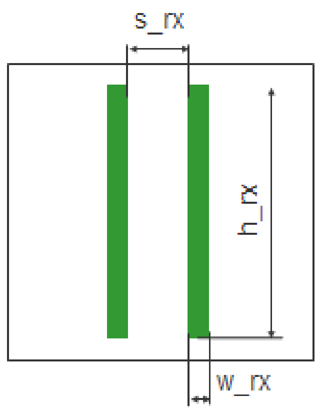
Adds an active feature of one rectangle, vertically or horizontally. It uses the following parameters:

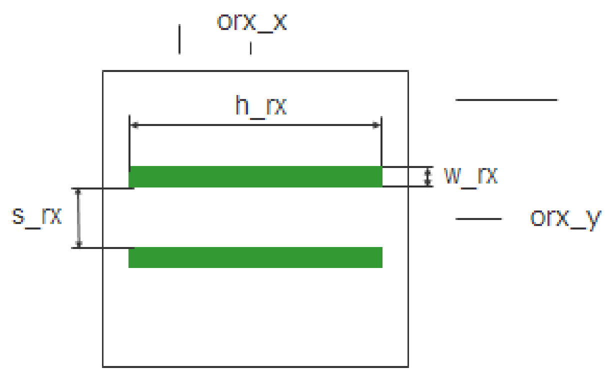


- **Orientation v | h** — Vertical or horizontal orientation
- **Widths *w_rx*** — Width of the RX rectangle
- [Heights] *h_rx*— Height of the RX rectangle (default is the block height)
- **Offset_X *orx_x*** — X offset of the center of the RX rectangle from the center of the gauge
- **Offset_Y *orx_y*** — Y offset of the center of the RX rectangle from the center of the gauge

2rx_rect

Adds two active features as vertical or horizontal rectangles with space between them. It uses the following parameters:

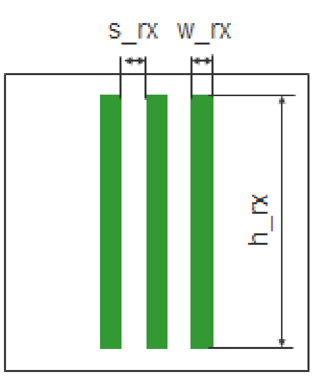


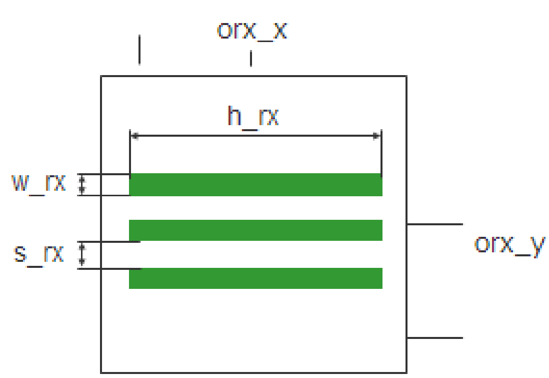


- **Orientation** {**h** | **v**} — Vertical or horizontal orientation
- **Widths *w_rx*** — Width of the RX rectangles
- **Spaces *s_rx*** — Spacing between the RX rectangles
- [Heights *h_rx*] — Height of the RX rectangles (default is the block height)
- **Offset_X *orx_x*** — X offset of the centers of the RX rectangles from the center of the gauge
- **Offset_Y *orx_y*** — Y offset of the centers of the RX rectangles from the center of the gauge

3rx_rect

Adds three active features as vertical or horizontal rectangles with equal space between them. It uses the following parameters:



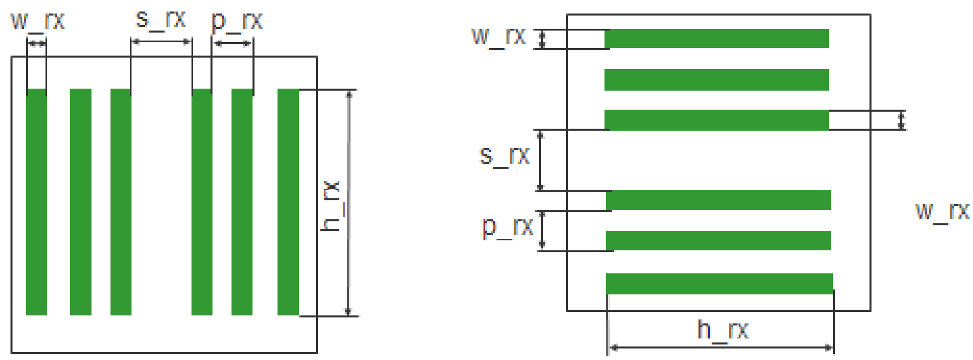


- **Orientation** {**v** | **h**} — Vertical or horizontal orientation
- **Widths *w_rx*** — Width of the RX rectangles

- **Spaces *s_rx*** — Spacing between the RX rectangles; the same space is used on either side of the center rectangle
- [Heights *h_rx*] — Height of the RX rectangles (default is the block height)
- **Offset_X *orx_x*** — X offset of the center of the middle RX rectangle from the center of the gauge
- **Offset_Y *orx_y*** — Y offset of the center of the middle RX rectangle from the center of the gauge

rx_rect_array

Creates multiple sets of active features in the vertical or horizontal direction. It uses the following parameters:



- **Orientation** {**v** | **h**} — Orientation of the arrays, vertical or horizontal
- **Widths *w_rx*** — Width of the RX rectangles
- **Pitchs *p_rx*** — Pitch for an RX array
- **Spaces *s_rx*** — Spacing between the two RX arrays nearest to the center
- **Counts *c*** — Number of RX arrays, which must be an even number
- [Heights *h_rx*] — Height of the RX rectangles (default is the block height)
- **Offset_X *orx_x*** — X offset of the center of the RX arrays from the center of the gauge
- **Offset_Y *orx_y*** — Y offset of the center of the RX arrays from the center of the gauge

rx_line_end

Creates two abutting lines with a gap between them. It uses the following keywords:

- **Widths *w*** — Width of the RX rectangles
- **Gaps *g*** — Space between the abutting lines

rx_inv_line_end

Creates two abutting spaces around a central inverse gap. It uses the following keywords:

- **Widths *w*** — Width of the RX spaces
- **Gaps *g*** — Width of the fill between the abutting spaces

rx_dense_line_end

Similar to line end, but adds two long lines on either side of the abutting line. It uses the following keywords:

- **Widths *w*** — Width of the RX rectangles
- **Spaces *s*** — Spaces between the lines and the middle abutting line
- **Gaps *g*** — Space between the abutting lines

rx_dense_contact

Creates an array of square contacts.

- **Sizes *sz*** — The length of one side of the contact
- **Spaces *sp*** — The space between the contacts

rx_sandwich

Creates a custom pattern in a sandwich style, supporting up to three separately configurable layer types (P first, then E, then Q). Each layer type has its own set of keywords that set the characteristics of that layer type only. If no Q type keywords are supplied, Q uses the matching P keyword settings. If no P settings are supplied, E settings are used.

- **Layers *ln*** — Specifies the number of layers in the sandwich

  When ***ln*** is divisible by three, one P, E, and Q layer is created. Additional layers are created on top, starting with an additional E layer.
- **Orientation** {**h** | **v**} — Orientation of the sandwich, horizontal or vertical
- **Spaces *sp*** — Space between layers
- **Ewidths *e_w*** — Width of a rectangle on an "E" layer
- **Ethicks *e_t*** — Thickness of a rectangle on an "E" layer
- [Espace *e_sp*] — Space between "E" layer rectangles
- [Ecounts *e_c*] — Number of rectangles on an "E" layer
- [Offsets *of*] — Distance offset from the center of the pattern for offsets

- [Offtypes *ot*] — Selects an offset pattern (shear, slide, and alter (for alternating))
- [Pwidths *p_w*] — Width of a rectangle on a “P” layer
- [Pthicks *p_t*] — Thickness of a rectangle on a “P” layer
- [Pspaces *p_sp*] — Space between “P” layer rectangles
- [Players *p_l*] — Number of layers using “P” layer settings
- [Pcounts p_c] — Number of rectangles on a “P” layer
- [Qwidths q_w] — With of a rectangle on a “Q” layer
- [Qthicks q_t] — Thickness of a rectangle on a “Q” layer
- [Qspaces q_sp] — Space between “Q” layer rectangles
- [Qlayers q_l] — Number of layers using “Q” layer settings
- [Qcounts q_c] — Number of rectangles on a “Q” layer

Related Topics

Creating a Custom Topo Test Pattern

PC_FEATURE Keyword

topoenclosure

Tcl batch command

Outputs distances from the measurement CD sites to the nearest edge of the active layer.

Note

This command can be used to find the sigma coverage for the squadratic value in the topo_model Format specification.

Usage

topoenclosure -l *layout* -active *active* -implant *implant* -gauge *gaugefile* [-out *outputfile*] [-mode {edge | meas}] [-max *dmax_um*] [-active_gauges]

Arguments

- **-l *layout***

  A required argument specifying a Calibre WORKbench Tcl handle for an input layout.

- **-active *active***

  A required argument specifying the layer number for the active (Rx) layer.

- **-implant *implant***

  A required argument specifying the layer number for the implant layer.

- **-gauge *gaugefile***

  A required argument specifying an input super gauge or gauge file with gauge coordinates and measurement CDs.

- -out *outputfile*

  An optional argument specifying an output file name. If this option is not specified, then the standard output is used.

- -mode {edge | meas}

  An optional argument that sets the measurement mode. If the mode is "edge", then the distances are measured from the edge of the implant layer and measurement CDs are ignored. If the mode is "meas", then the distances are calculated from the measurement point on the gauges. The default is "meas".

- -max *dmax_um*

  An optional argument specifying the maximum search distance in microns. If an active edge is not found, or is further away from the CD measurement point than this value, then the distance for that gauge is reported as *dmax_um**1000. Default is 1 um.

- -active_gauges

  An optional argument that when specified computes values only for the active gauge set, which may have been modified by other commands. By default, the calculation is performed on the original loaded gauges.

Description

The output columns represent the following data values:

Table 4-5. topoenclosure Data Values By Column

Column	Description
1	Gauge number. Only enabled gauges are output, sequentially numbered from 1.
2	Gauge row number, similar to an input gauge (super gauge) file.
3	Gauge column number, similar to an input gauge (super gauge) file.
4	Distance from the measurement site on the gauge cutline (near the edge of the implant layer) to the nearest edge of the active layer, in nm. The distance is positive when the measurement site of the gauge is outside of active features and negative otherwise.

The output file data can be visualized in Calibre nmModelflow, or by plotting in a spreadsheet program, or in MathWorks MATLAB[1]using the following code:

```
d = load('outputfile');
plot(d(:,1), d(:,4),'*');
```

The following limitations must be observed:

- The layout geometry must be Manhattan-style.
- Gauge cutlines must be horizontal or vertical.

Notes:

- At least two intersections (defined as where the gauge crosses the contour of the implant feature) with implant features must be between gauge endpoints. If the number of intersections is larger then two, then the nearest points on the left and on the right from the gauge center are considered as the drawn CD site points. Measured CD sites are supposed to be at equal distances from drawn CD sites.
- It is assumed that the nearest active edge is intersected by a line passing through the gauge endpoints. Active edges that are not intersected by this line are not considered.

1. MATLAB and MathWorks are registered trademarks of The MathWorks, Inc.

Examples

This example measures the topo enclosure distances for the loaded layout, using layer 1 as the active layer, and reading the gauges from an external gauge file. Edge mode is selected, meaning that distances are measured from the edge of the implant layer and measurement CDs are ignored. Output is written to *file.dat*.

```
set  L_handle  [layout  create input_layout_file.oas]
topoenclosure  \
        -l $L_handle  \
        -active 1 \
        -implant 0 \
        -gauge gauge_file.gg  \
        -mode edge \
        -out topoenclosure_results_file.dat \
        -max 5
```

toposignal

Tcl batch command

Outputs the toposignal data for a specified signal from the input topo model.

Usage

toposignal -model *modelfilename* -signal *signaltype* [-out *outputfilename*]

Arguments

- **-model *modelfilename***

  A required argument specifying a file containing a topo model in topo_model Format. However, model-layer files are not read by this command; in order to use this command, you must copy only the contents of the topo_model block to a separate file.

- **-signal *signaltype***

  A required argument specifying the type of signal to generate. Accepted values are corner, corner2, finfet, finfet_sw, gate, interconnect, sidewall, sidewall2, sidewallo2, trench, trench2, trench4, trencho2, trench2o2, trench4o2, and vcorner.

- -out *outputfilename*

  An optional argument specifying the name of an output file. If one is not specified, then standard output (stdout) is used.

  The data has two columns. The first column is the distance from the source point in microns. The second column is the value of the topo signal. The topo signal is axially symmetrical.

  You can visualize the data by plotting in a spreadsheet program or MathWorks MATLAB[2]. For MATLAB, use the following code:

  ```
  s = load('outputfilename');
  plot(s(:,1), s(:,2));
  ```

Examples

This example outputs the sidewall signal data for the topo model in the file mytopo.txt. Output is written to the file toposignal.txt.

```
toposignal -model mytopo.txt -signal sidewall -out toposignal.txt
```

2. MATLAB and MathWorks are registered trademarks of The MathWorks, Inc.

Index

Third-Party Information

Details on open source and third-party software that may be included with this product are available in the *<your_software_installation_location>/legal* directory.