TopSim

G. Hobler
T. Zahel, S. Lindsey

TU Vienna, AUSTRIA

December 2011

Draft Version (September 14, 2012)

Contents

Chapter 1

Overview

Chapter 2

Files

Chapter 3

Input Parameters

The configuration file `xxx.cfg` is read using Python's `ConfigParser` module. The syntax therefore has to adhere to the `ConfigParser` rules, which basically demand the configuration file to be written in the Windows INI format, like

```
# A comment
[section1]
option1 = value1
option2 = value2

[section2]
option1 = value3
option3 = value4
...
```

Note that same option names appearing in different sections will be distinguished.

One pitfall that requires attention when writing a configuration file is that TopSim will check the type of the option values against its defaults. Since Python distinguishes between tuples (lists) and scalar values, it is essential that tuples are specified as tuples and scalars (e.g. floats) as scalars (floats). A tuple parameter can be specified as

```
option = (item1,item2,...,itemn)
```

with the dots replaced by other item values. The parenthesis may be omitted, but the commas are essential. Several TopSim parameters are tuples, among them parameters that have components in x- and y-direction. While in a 2D simulation only the x-component is used, the parameter still is a tuple and must be specified as a tuple like

```
option = item1,
```

Incomplete tuples will have their missing elements duplicated from the last given component. If in doubt, consult the `[Output]LOG_FILE`, where a copy of all parameters after any substitutions can be found.

3.1 [Include]

INCLUDE_FILE

This parameter specifies the name of a nested configuration file. If specified, the nested configuration file will be read before the configuration file specifying the INCLUDE_FILE.

Type: character string
Default: ''

3.2 [Setup]

ANGLE_BISECTOR

This parameter specifies whether the angle bisector is used to determine the direction of point advancement. If `ANGLE_BISECTOR=False`, point advancement is perpendicular to the line connecting the two neighboring points. Only valid if `SURFACE_TYPE='rectilinear'`.

Type: boolean
Default: `False`

DEPOSITION

This parameter enables beam induced deposition.

Type: boolean
Default: `False`

DIMENSIONS

This parameter specifies the number of spatial dimension of the coordinate system.

Type: integer
Default: required
Range: `1`, `2`, `3`

ETCHING

This parameter enables beam induced etching.

Type: boolean
Default: `False`

INTERPOLATE

This parameter specifies whether the grid points are interpolated back to the x- and y-positions of the initial grid. Only valid if `SURFACE_TYPE='rectilinear'`.

Type: boolean
Default: `True`

ISOTROPIC

This parameter specifies whether isotropic sputtering is used or the sputter yield is read from a table (see the [Physics] section). If ISOTROPIC=True, the first order sputter fluxes are set equal to the beam flux. Otherwise, the first order sputter fluxes are calculated as the product of the beam fluxes, the cosine of the incidence angle, and the sputter yield.

Type: boolean
Default: False

REDEP_1

This parameter specifies whether first order redeposition is considered, i.e. redeposition of atoms ejected by the beam (first order sputtering).

Type: boolean
Default: False

REDEP_2

This parameter specifies whether second order redeposition is considered, i.e. redeposition of atoms ejected in second order sputtering.

Type: boolean
Default: False

SPUTTER_2

This parameter specifies whether second order sputtering is considered, i.e. sputtering caused by backscattered ions.

Type: boolean
Default: False

SHADOWING

This parameter specifies whether complex shadowing, in addition to simple visibility, is applied to the simulation of the surface. It is currently only availabe in 2D.
Simple visibility finds points that are shadowed by the limits of ejection, i.e. the ejection angles cannot be larger than 90 degrees.
Complex shadowing finds points that are shadowed by other points being in the way by finding local maxima in the cosine (Source to Destination). Such a Point of Interest is the starting point of a shadow, and the next point with a higher cosine is the corresponding endpoint.

Type: boolean
Default: `False`

SURFACE_TYPE

This parameter specifies the type of the grid for the surface.

Type: character string
Default: `'rectilinear'`

3.3 [Initial Conditions]

DELTA_X

This parameter specifies the initial grid spacing in x direction. The units of DELTA_X are nm. This parameter is ignored if the initial surface is read from a file (see SURFACE_FILE), or if the simulation is 1D (see DIMENSIONS in the [Setup] section).

Type: real
Default: 5.

DELTA_Y

This parameter specifies the initial grid spacing in y direction. The units of DELTA_Y are nm. This parameter is ignored if the initial surface is read from a file (see SURFACE_FILE), or if the simulation is 1D or 2D (see DIMENSIONS in the [Setup] section).

Type: real
Default: 5.

SURFACE_FILE

This parameter specifies the name of the surface file to be read at at the beginning of the simulation. The format of the file must be consistent with the DIMENSIONS and SURFACE_TYPE parameters of the [Setup] section. If SURFACE_FILE='', a flat surface at z=0 is assumed.

Type: character string
Default: ”

XMAX

This parameter specifies the maximum x value of the initial surface grid. The units of XMAX are nm. This parameter is ignored if the initial surface is read from a file (see SURFACE_FILE), or if the simulation is 1D (see DIMENSIONS in the [Setup] section).

Type: real
Default: 100.

XMIN

This parameter specifies the minimum x value of the initial surface grid. The units of XMIN are nm. This parameter is ignored if the initial surface is read from a file (see SURFACE_FILE), or if the simulation is 1D (see DIMENSIONS in the [Setup] section).

Type: real
Default: -100.

YMAX

This parameter specifies the maximum y value of the initial surface grid. The units of YMAX are nm. This parameter is ignored if the initial surface is read from a file (see SURFACE_FILE), or if the simulation is 1D or 2D (see DIMENSIONS in the [Setup] section).

Type: real
Default: 100.

YMIN

This parameter specifies the minimum y value of the initial surface grid. The units of YMIN are nm. This parameter is ignored if the initial surface is read from a file (see SURFACE_FILE), or if the simulation is 1D or 2D (see DIMENSIONS in the [Setup] section).

Type: real
Default: -100.

3.4 [Regions]

Regions define the geometry of spatial domains for the definition of material properties (see the Physics] section).

NUMBER_OF_REGIONS

This parameter specifies the number of regions. (soon to be depreciated)

Type: integer
Default: required

REGIONS

This parameter specifies the boundaries of the regions. (not yet used). [This has been replaced in 2D by the concept of primitives and domains]

Type: real
Default: $(-\infty, \infty)$

DOMAINS

This parameter specifies a dictionary that is used to define combinations of primitives. Currently the only Domains that can be used are material domains. The indexed order of the material names is specified with the Physics.MATERIAL_NAMES parameter. Each entry to the dictionary is made up of two strings: the key, and the equation. The key is the name of the Domain, this name should match one of the materials used in Physics.MATERIAL_NAMES, unmatched keys will be ignored. The equation instructs the simulator on how to combine spatial primitives. Equations have two operations +(plus) and -(minus), plus indicates the union of two spatial primitives, minus indicates the subtraction of the second primitive from the first. All domain equations are evaluated from left to right such that a+b-c = (a+b)-c, the leading operator is assumed to be positive and any attempt to use unary operators will result in an error. These operations (+/-) operate on spatial primitives as defined by thier keys, the spatial primitives are specified with the PRIMITIVES parameter.

Type: dictionary
Default: $(-\infty, \infty)$

PRIMITIVES

This parameter specifies the spatial primitives used in the simulation. Format: key: primitive-parameters.

Type: real
Default: $(-\infty, \infty)$

3.5 [Beam]

CENTER

This parameter specifies the x and y coordinates of the beam center. CENTER may also be considered the origin of the scan coordinate system, i.e. the position of the first pixel in case of a predefined scan or an offset applied to the coordinates in the pixel file. If only one value of CENTER is given, the second value is assumed to be identical to the first value. The units of CENTER are nm.

Type: (real,)
Default: (0.,0.)

CURRENT

This parameter specifies the beam current in A. For DIMENSIONS=0 the current density is CURRENT/(SCAN_WIDTH[0] · SCAN_WIDTH[1]) (units A/nm^2). For DIMENSIONS=1 the line current density is CURRENT/SCAN_WIDTH[1] (units A/nm).

Type: real
Default: required

DIMENSIONS

This parameter specifies the dimensions of the beam current density function, i.e. in how many directions the beam current density varies. For DIMENSIONS=2 the current density varies in both x- and y-direction. For DIMENSIONS=1 the current density varies in x-direction only. For DIMENSIONS=0 the current density does not vary in any direction.

Type: integer
Default: 0 for TYPE='constant'
1 otherwise
Range: 0 for TYPE='constant'
1, 2 otherwise

DIVERGENCE

This parameter specifies the beam divergence as the standard deviation of a Gaussian distribution function. If only one value of DIVERGENCE is given, the second value is assumed to be identical to the first value. The units of DIVERGENCE are degrees.

Type: (real,)
Default: (0.,0.)

ERF_BEAM_WIDTH

This parameter specifies the displacement $x_2 - x_1$ of the two error functions defining the error function beam (see TYPE). Note that for ERF_BEAM_WIDTH $\ll$ FWHM the error function beam tends towards a Gaussian beam with FWHM equal to FWHM. For increasing ERF_BEAM_WIDTH the top of the distribution gets flatter and the FWHM increases. For ERF_BEAM_WIDTH $\gg$ FWHM the error function beam is a constant beam with blurred edges, where the amount of blur depends on FWHM. If only one value of ERF_BEAM_WIDTH is given, the second value is assumed to be identical to the first value. The units of ERF_BEAM_WIDTH are nm. Ignored unless TYPE='error function'.

Type: (real,)
Default: (1.e-10,1.e-10)

FWHM

This parameter specifies the full width at half maximum (FWHM) of the beam distribution function, except for TYPE='error function' where FWHM specifies the FWHM of the corresponding Gaussian beam. If only one value of FWHM is given, the second value is assumed to be identical to the first value. The units of FWHM are nm.

Type: (real,)
Default: (30.,30.)

TILT

This parameter specifies the beam incidence angle with respect to the z-axis. Positive values mean the ions have a velocity component in positive x-direction. The units of TILT are degrees.

Type: real
Default: 0.

TOTAL_TIME

This parameter specifies the total time the beam is on, i.e. the total simulation time. The units of `TOTAL_TIME` are seconds.

Type: real
Default: required if `[Scan]DWELL_TIME` is not specified

TYPE

This paramter specifies the type of the beam current density function. Note that the type of the beam is fully specified only with its `DIMENSION` unless `TYPE='constant'`. An error function beam corresponds to overlapped Gaussian beams if the pixel spacing is small enough and is given by $f(x) \sim \mathrm{erf}((x - x_2)/(\sqrt{2}\sigma)) - \mathrm{erf}((x - x_1)/(\sqrt{2}\sigma))$.

Type: integer
Default: `'Gaussian'`
Range: `'constant','Gaussian','error function'`

3.6 [Scan]

DWELL_TIME

This parameter specifies the dwell time per pixel in seconds. If TYPE='pixel file' the dwell time is multiplied by the value found in the PIXEL_FILE. The total simulation time in case of a raster or serpentine scan is given by PASSES $\times$ PIXELS[0] $\times$ PIXELS[1] $\times$ DWELL_TIME. If DWELL_TIME is not specified, it is calculated from [Beam]TOTAL_TIME. If both DWELL_TIME and [Beam]TOTAL_TIME are specified, [Beam]TOTAL_TIME will be overwritten by the value calculated from DWELL_TIME.

Type: real
Default: required if [Beam]TOTAL_TIME is not specified

OVERLAP

This parameter specfies whether all the beams of a pass should be overlapped to a single beam. The advantage of an overlapped beam is that the time step is not limited to the pixel dwell time. This approximation is only valid if the change in the surface caused by a single pixel during DWELL_TIME is small enough.

Type: boolean
Default: False

OVERLAP_Y

This parameter specfies whether all the beams along the line of a scan in y-direction should be overlapped to a single beam. The advantage of an overlapped beam is that the time step is not limited by the pixel dwell time. This approximation is only valid if the change in the surface caused by a single pixel is small enough.

Type: boolean
Default: False

Y_OVERSCAN

This parameter instructs the simulation to scan each Y line in a 3D simulation a preset number of times. This is used to produce a marching line scane in 3D mode without using overlap modes. A value of 1 indicates that the line is to be scanned once and then the beam will step in the x direction. A larger value causes the beam to scan that Y line a larger number of times before stepping in the x direction.

Type: integer
Default: 1

PASSES

This parameter specifies the number of passes, i.e. the number of times the pattern defined by `TYPE` is repeated.

Type: integer
Default: 1

PIXEL_FILE

This parameter specifies the name of the pixel file, containing a list of pixels coordinates and dwell times. The dwell times will be multiplied with `DWELL_TIME`.

Type: character string
Default: required if `TYPE='pixel file'`

PIXEL_SPACING

This parameter specifies the pixel spacing in x- and y-direction for `TYPE='raster'` and `TYPE='serpentine'`. If only one value of `PIXEL_SPACING` is given, the second value is assumed to be identical to the first value. Ignored if `TYPE='none'` or `TYPE='pixel file'`. The units of `PIXEL_SPACING` are nm.

Type: (real,)
Default: `0.,0.` if `TYPE='none'`
`SCAN_WIDTH/max(PIXEL-1,1)` otherwise

PIXELS

This parameter specifies the number of pixels in x- and y-direction for TYPE='raster' and TYPE='serpentine'. If only one value of PIXELS is given, the second value is assumed to be 1. Ignored if TYPE='pixel file'.

Type: (integer,)
Default: 1,1

SCAN_WIDTH

This parameter specifies the scan width in x and y direction for TYPE='raster' and TYPE='serpentine'. If only one value of SCAN_WIDTH is given, the second value is assumed to be identical to the first value. Ignored if TYPE='pixel file'. The units of SCAN_WIDTH are nm.

Type: (real,)
Default: PIXEL_SPACING*max(PIXELS-1,1) if PIXEL_SPACING is specified
1.,1. otherwise

TYPE

This parameter specifies the scan type. TYPE='raster' means a scan along parallel lines starting each line on the the same side. TYPE='serpentine' means a scan along parallel lines starting every other line on the the same side. TYPE='pixel file' means the pixel positions are specified in the PIXEL_FILE. For TYPE='none' no scan is performed, being equivalent to TYPE='raster' and PIXELS=1,1.

Type: character string
Default: 'raster'
Range: 'none','raster','serpentine','pixel file'

3.7 [Physics]

BACKSCATTER_ANG_DIST_FILES

This parameter specifies the filenames of the angular distribution tables for backscattered ions, i.e. the distribution as a function of incidence and ejection angle. Actual values of the distribution function are obtained by interpolation in the tables. Paths relative to the work directory or the physics/tables directory of the code have to be specified. If a file is not specified or `''`, the file name is set to the `TABLE_DIRECTORY` joined with the file name `'bang.npz'`.

Type: (character string,)
Default: required if corresponding `TABLE_DIRECTORY` is undefined

BACKSCATTER_YIELD_FILES

This parameter specifies the filenames of the backscatter yield tables, i.e. the backscatter yields as a function of incidence angle. Actual backscatter yields are obtained by interpolation in the tables. Paths relative to the work directory or the physics/tables directory of the code have to be specified. If a file is not specified or `''`, the file name is set to the `TABLE_DIRECTORY` joined with the file name `'byield.npz'`.

Type: (character string,)
Default: required if corresponding `TABLE_DIRECTORY` is undefined

DENSITIES

This parameter specifies the atomic density for each material region. If less `DENSITIES` are specified than `MATERIAL_NAMES`, the missing `DENSITIES` are assumed to be indentical to the last specified `DENSITY`. The units of `DENSITIES` are atoms per nm^3.

Type: tuple of real
Default: `(49.94,)`

MATERIAL_NAMES

This parameter specifies the material name for each region.

Type: tuple of character string
Default: `('Si',)`

SPUTTER_ANG_DIST_FILES

This parameter specifies the filenames of the angular distribution tables for sputtered atoms, i.e. the distribution as a function of incidence and ejection angle. Actual values of the distribution function are obtained by interpolation in the tables. Paths relative to the work directory or the physics/tables directory of the code have to be specified. If a SPUTTER_ANG_DIST_FILE is not specified or '' and , the file name is set to the TABLE_DIRECTORY joined with the file name 'sang.npz'. If both SPUTTER_ANG_DIST_FILE='' and TABLE_DIRECTORY='', a cosine distribution function is used.

Type: (character string,)
Default: required if corresponding TABLE_DIRECTORY is undefined

SPUTTER_YIELD_FILES

This parameter specifies the filenames of the sputter yield tables, i.e. the sputter yields as a function of incidence angle. Actual sputter yields are obtained by interpolation in the tables. Paths relative to the work directory or the physics/tables directory of the code have to be specified. If a file is not specified or '', the file name is set to the table directory joined with the file name 'syield.npz'.

Type: (character string,)
Default: required if corresponding TABLE_DIRECTORY is undefined

TABLE_DIRECTORIES

This parameter specifies the directories for the sputter/backscatter yield/angular distribution tables. Paths relative to the work directory or the physics/tables directory of the code have to be specified. It is only used, if the corresponding table file is unspecified. If less TABLE_DIRECTORIES are specified than MATERIAL_NAMES, the missing TABLE_DIRECTORIES are assumed to be indentical to the last specified TABLE_DIRECTORY.

Type: (character string,)
Default: '',

3.8 [Numerics]

ADAPTIVE_GRID

This parameter specifies whether the grid is dynamically adapted.

Type: logical
Default: `False`

GRID_LAZINESS

This parameter specifies the tolerated percentage deviation from the maximum segment length `MAX_SEGLEN` for point/line insertion. The parameter is intended to avoid repeated insertions and deletions at the same position. Ignored if `ADAPTIVE_GRID=False`.

Type: real
Default: `10.`

GRID_REGIONS

This parameter specifies the x and y range of the grid to be refined. Ignored if `ADAPTIVE_GRID=False`.

Type: tuple of tuples
Default: `(XMIN,XMAX),` if `DIMENSIONS=2`
`(XMIN,XMAX,YMIN,YMAX),` if `DIMENSIONS=3`

MAX_POINTS

This parameter specifies the maximum number of total points in the simulation. Ignored if `ADAPTIVE_GRID=False`.

Type: integer
Default: `10000`

MAX_SEGLEN

This parameter specifies the maximum segment length. A point/line is inserted when the/any segment length exceeds MAX_SEGLEN plus a tolerance given by GRID_LAZINESS. A point/line is deleted when the/any segment length after deletion falls below MAX_SEGLEN. The units of MAX_SEGLEN are nm. Ignored if ADAPTIVE_GRID=False.

Type: real
Default: 10.

MIN_DELTA

This parameter specifies the minimum grid spacing in x and y. Points/lines are not inserted if (any of) the grid spacings would fall below MIN_DELTA. The units of MIN_DELTA are nm. The parameter is intended to prevent the formation of kinks in the contour due to excessive grid refinements. Ignored if ADAPTIVE_GRID=False. If only an x value is specified, the x value is copied and used as the y value.

Type: tuple (x,y)
Default: 1.

TIME_STEP

This parameter specifies the maximum time step in seconds. The time step may be reduced by the program in case of numerical problems or if it would extend over the end of a pixel dwell time.

Type: real
Default: required

3.9 [Output]

DISPLAY_SURFACE

This parameter specifies whether the plots should be displayed at the end of the simulation. A window opens, and various options are available through keystrokes while the window is in focus. A short help is printed to stdout. The same functionality is available through the plot.py script after the simulation has finished.

Type: boolean
Default: `False`

LOG_FILE

This parameter specifies whether a copy of the input parameters and the data written to the screen should also be written to a log file. The name of the log file is identical to that of the configuration file, but with the extension `'.log'`.

Type: boolean
Default: `False`

SAVE_ALL

This parameter specifies whether all available fields should be written to the `SURFACE_FILE`. Equivalent to all `SAVE_xxx=True`.

Type: boolean
Default: `False`

SAVE_ANGLES

This parameter specifies whether the local angle between the surface normal and the beam should be written to the `SURFACE_FILE`. Ignored if `SAVE_ALL=True`.

Type: boolean
Default: `False`

SAVE_BEAM_FLUXES

This parameter specifies whether the local beam fluxes should be written to the SURFACE_FILE. Ignored if SAVE_ALL=True.

Type: boolean
Default: False

SAVE_FLUXES

This parameter specifies whether the local surface fluxes should be written to the SURFACE_FILE. Ignored if SAVE_ALL=True.

Type: boolean
Default: False

SAVE_POSITIONS

This parameter specifies whether the positions of the surface vertices should be written to the SURFACE_FILE. Ignored if SAVE_ALL=True.

Type: boolean
Default: True

SAVE_PRECURSOR

This parameter specifies whether the local values of the precursor coverage should be written to the SURFACE_FILE. Ignored if SAVE_ALL=True.

Type: boolean
Default: False

SURFACE_FILE

This parameter specifies the name of the surface file, which is the main output file.

Type: character string
Default: 'input_file.srf'

VERBOSE

This parameter specifies whether more information should be printed and written to the log file.

Type: boolean
Default: `False`

WRITE_TIME_STEP

This parameter specifies the time between two interim outputs to the `SURFACE_FILE`. **WARNING:** The ratio of the total simulation time (`[Beam]TOTAL_TIME` or as calculated from `[Scan]DWELL_TIME`) and `WRITE_TIME_STEP` determines the number of frames written to the `SURFACE_FILE`. Careless choice of `WRITE_TIME_STEP` may result in excessive hard disk space requirements.

Type: real
Default: `1.`

SAVE_BINARY

This paramter specifies wether the output surface should be saved as hexadecimal numbers in order to provide better accuracy when loading a surface for another simulation.

Type: boolean
Default: `False`

Chapter 4

Examples

Chapter 5

Algorithms