SIEMENS EDA

Calibre® Metrology API (MAPI) User's and Reference Manual

Software Version 2021.2

SIEMENS

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction to Calibre MAPI

The Calibre® Metrology API (MAPI) is a set of iTcl (incr Tcl) classes that assist in the support of manufacturing machines (metrology machines) with Calibre WORKbench or Calibre MDPview. Calibre MAPI makes possible the rapid development of drivers and analysis tools, and can serve as middleware between tools or data and Calibre analysis programs.

Calibre MAPI is implemented as a set of iTcl classes in Calibre® WORKbench™ or Calibre® MDPview™. The classes provide the maximum amount of common functionality without limiting further extension.

The classes can be organized into the following functional groups:

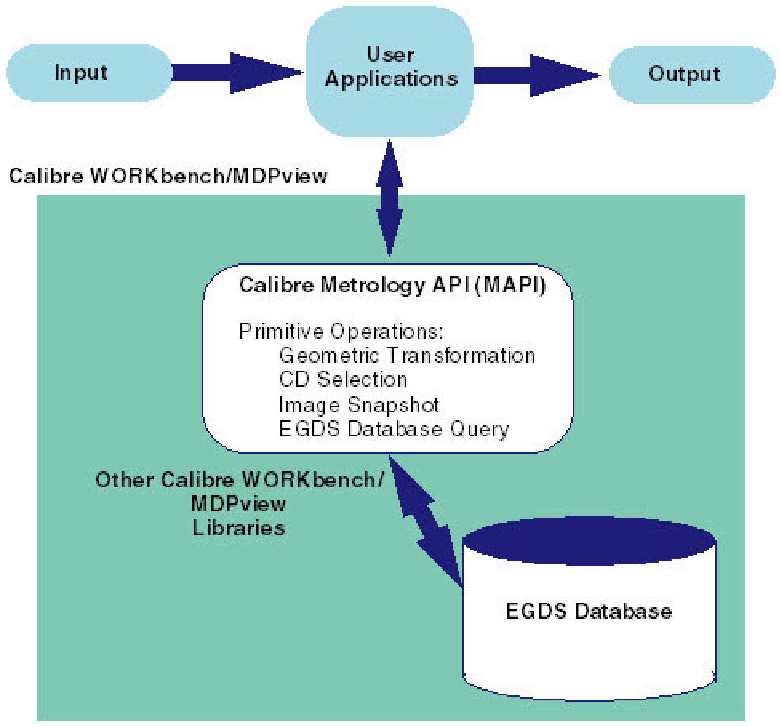
- Site Creation (MAPI_Site) — Create a basic data structure (also known as a “site”) in a Calibre MAPI application.
- Site Processing (MAPI_ProcessSite) — Process a single site.
- Output Objects (MAPI_Output) — Control all output.
- Transforms (MAPI_Transform) — Apply geometric transformations to your layout such as mirror, rotation, scaling, and x-y origin shifting. The transform set provides a uniform representation and interface for affine transforms.
- Image Generation (MAPI_Image) — Generates snapshots of geometry or aerial images.
- Database Query (MAPI_Egds_Query) — Query information the layout database in a neighborhood of a point may sometimes be necessary
- Flat Database Traversal (MAPI_RefFilter) — Flat database traversal provides efficient and easy access to flattened elements of hierarchical databases.
- Site Filtering (MAPI_SiteFilter) — Reduce, qualify, or group sites based on their two-dimensional location. Filter classes permit various types of site sampling and organization.
- Alignment Point Detection (MAPI_FindAlignmentPoint) — Detect two-dimensional geometrical configurations inside of a specified query extent.
- Extract Clips from a Layout — Clip portions of an input layout and output them either as a file or directly to a layout viewer.

Calibre MAPI Workflow

Calibre MAPI allows you to drive and interact with wafer and mask inspection equipment. The information required includes location sets on the photomask, assistance for finding them through pattern recognition, as well as specific characteristics on the structures to be measured (for example, target value).

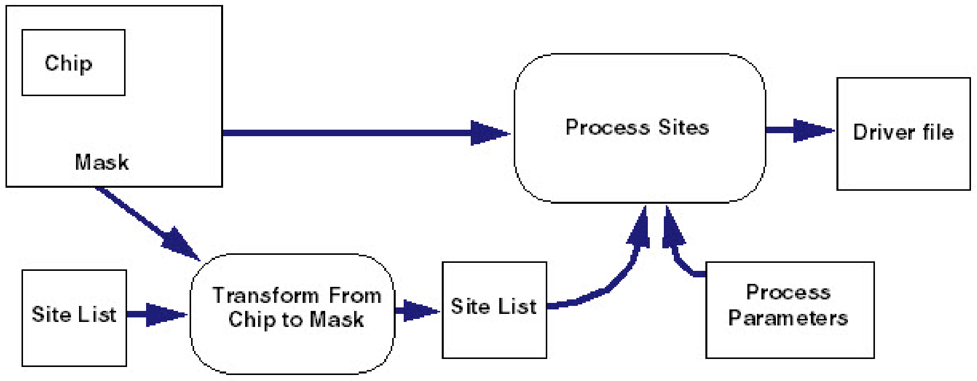
The Calibre MAPI platform is built on Calibre WORKbench and Calibre MDPview (see Figure 1-1 for a relational diagram). Thus, many core features of Calibre WORKbench and Calibre MDPview are accessible to Calibre MAPI. In particular, Calibre MAPI relies on the EGDS database of Calibre WORKbench and Calibre MDPview that stores and queries, both mask and design data efficiently. Most importantly, you can quickly develop and customize their metrology applications using the operations and functionality provided by the Calibre MAPI platform.

Figure 1-1. Calibre MAPI Platform Architecture



Mask metrology is an example of the need to drive a manufacturing machine using data derived from a layout. Figure 1-2 demonstrates a typical structure and behavior of such a system: a list of “measurement sites” are specified for a layout.

Figure 1-2. A Typical Machine Interface System



The sites may need to be transformed into the coordinate system of the layout or some related physical medium. The system then refines or validates each measurement site based on the local context at the specified location in the layout. The list of "elaborated" sites is then formatted into a driver file. The basic elements of this process are common to varying degrees in all measurement-type manufacturing processes and even in more general manufacturing processes such as mask repair. Application programs written in isolation from each other would likely duplicate much effort and require the same or very similar basic services from the layout database and the broader development environment.

Table 1-1 lists the documents associated with related Calibre tools.

Table 1-1. Related Products and Their Manuals

Related Products	Documentation
Calibre® FRACTUREc™ Calibre® FRACTUREh™ Calibre® FRACTUREi™ Calibre® FRACTUREj™ Calibre® FRACTUREm™ Calibre® FRACTUREn™ Calibre® FRACTUREp™ Calibre® FRACTUREt™ Calibre® FRACTUREv™ Calibre® MDPmerge™ Calibre® MDPstat™ Calibre® MDPverify™ Calibre® MPCpro™ Calibre® MASKOPT™ Calibre® MDP Embedded SVRF	*Calibre Mask Data Preparation User's and Reference Manual* *Calibre Release Notes*
Calibre® MDPview™	*Calibre MDPview User's and Reference Manual* *Calibre Release Notes*
Calibre® Interactive™ Calibre® RVE™	*Calibre Interactive User's Manual* *Calibre RVE User's Manual*
Calibre® nmDRC™ Calibre® nmDRC-H™	*Calibre Release Notes* *Calibre Verification User's Manual* *Standard Verification Rule Format (SVRF) Manual*
Calibre® WORKbench™	*Calibre WORKbench User's and Reference Manual*
Tcl/Tk Batch Commands	*Calibre DESIGNrev Reference Manual*
Calibre® Metrology API (MAPI)	*Calibre Metrology API (MAPI) User's and Reference Manual*
Calibre® Job Deck Editor	*Calibre Job Deck Editor User's Manual*
Calibre® MDPDefectAvoidance™	*Calibre MDPDefectAvoidance User's Manual*

Table 1-1. Related Products and Their Manuals (cont.)

Related Products	Documentation
Calibre® nmMPC™ Calibre® nmCLMPC	*Calibre nmMPC and Calibre nmCLMPC User's and Reference Manual*
Calibre® MPCverify	*Calibre MPCverify User's and Reference Manual*
Calibre® DefectReview™	*Calibre DefectReview User's Manual*
Calibre® MDPAutoClassify™	*Calibre MDPAutoClassify User's Manual*
Calibre®DefectClassify™	*Calibre DefectClassify User's Manual*

Calibre MAPI Prerequisites

Before you begin using Calibre MAPI, there are several prerequisites.

- **Platform support** — Calibre MDPview is available on all supported platforms found in the *Calibre Administrator's Guide*. Refer to that document for instructions on how to install Calibre software.
- **Licensing** — To use Calibre MAPI you must have a Calibre MAPI license as well as a license for either Calibre WORKbench or Calibre MDPview. For more information on licensing, refer to the *Calibre Administrator's Guide*.

Calibre MAPI Invocation

The scripting capabilities provided by Calibre WORKbench and Calibre MDPview allow you to write an external script that will manipulate one or more layout databases. You write these scripts using the iTcl (an object-oriented extension to Tcl), referencing the Viewer layout object.

Note

Writing scripts for use with the Calibre WORKbench and Calibre MDPview data viewer requires knowledge of iTcl.

The basic syntax is as follows:

```
calibrewb -mapi -s scriptname.tcl
```

or

```
calibremdpv -mapi -s scriptname.tcl
```

For example:

```
calibrewb -mapi -s myscript.tcl
```

The use of this scripting capability is fully documented in the *Calibre DESIGNrev Reference Manual*.

Syntax Conventions

The command descriptions in this document 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.
Underline	Underlining indicates either the default argument or the default value of an argument.
UPPercase	For certain case-insensitive commands, uppercase indicates the minimum keyword characters. In most cases, you may omit the lowercase letters and abbreviate the keyword.
[ ]	Brackets enclose optional arguments. Do not include the brackets when entering the command unless they are quoted.
{ }	Braces enclose arguments to show grouping. Do not include the braces when entering the command unless they are quoted.
‘ ’	Quotes enclose metacharacters that are to be entered literally. Do not include single quotes when entering braces or brackets in a command.
| or ||	Vertical bars indicate a choice between items. Do not include the bars when entering the command.
…	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.

Table 1-2. Syntax Conventions (cont.)

Convention	Description
Example: **DEVice** {***element_name*** [‘(’*model_name*‘)’]} ***device_layer*** {***pin_layer*** [‘(’*pin_name*‘)’] …} [‘<’*auxiliary_layer*‘>’ …] [‘(’*swap_list*‘)’ …] [BY NET | BY SHAPE]	

Note

When creating Calibre MAPI scripts using the commands in this chapter, the –mapi switch must be used at invocation (for example, calibremdpv –mapi), otherwise an error will be issued.

Chapter 2
Calibre MAPI Reference

The Calibre MAPI interface is composed of Tcl-based batch scripting commands.

Introduction to iTcl Calibre MAPI Constructs

This chapter provides a listing of certain key iTcl-based Calibre MAPI constructs.

The Calibre MAPI constructs provided fall into the following categories:

- Class: A class is a “template” containing common characteristics. When this is adapted for a specific instance, it is known as an “object.”
- Constructor: A constructor is a member function that is called when an object of the class is declared but does not have a return type.
- Method: A method is a function that has access to the members of the class.

Class

The fundamental construct in iTcl is the class definition. Each class acts as a template for actual objects that can be created. The class itself contains characteristics common to all objects. Each object has its own unique set of data that contains instances of the variables defined in the class definition. Classes can also have “common” data members that are shared by all objects in a class.

The basic syntax for an iTcl class definition is:

class className definition

This syntax provides the *definition* (the characteristics) for a class named className. If a class or command are both called className, this command returns an error. When a class is instantiated, it becomes a known as an “object”.

Constructor

A constructor is a function called when an object is declared and never has a return type. Constructors are special instance methods that are called automatically upon the creation of an object (instance of a class). The basic syntax for an iTcl constructor definition is:

constructor *args* [*init*] *body*

This syntax declares the *args* argument list and *body* used for the constructor, which is automatically invoked whenever an object is created (the instantiation of a class).

Prior to executing the instructions in *body*, the optional *init* statement can be used to invoke any base class constructors that require arguments. Variables in the *args* specification are accessed in the *init* code fragment and passed to base class constructors.

If construction is successful, the constructor will always return the object name and the object name then becomes a command.

Method

A method is a sequence of statements to perform an action, a set of input parameters, or a return value associated with a class or object. The basic syntax for an iTcl method definition is:

method name [*args*] [*body*]

This syntax declares a method called name. When the method body is executed, it will have automatic access to object-specific variables and common data members.

If the args list is specified, it establishes the usage information for this method.

Site Creation (MAPI_Site)

A basic data structure in a Calibre MAPI application is the “site”, implemented by the MAPI_Site class. This data structure is a spatial marker with several annotation fields. For instance, it can be used to indicate the location and type of a measurement site. TCL lists of Calibre MAPI sites are a typical construct in Calibre MAPI applications.

Complete details of the site creation structures can be found in Appendix A, “MAPI Interfaces.”

This section covers the following:

- MAPI_Site class — Defines Calibre MAPI sites.
- method cdSpec — Specifies a basic subset of the class fields.

MAPI_Site

Defines Calibre MAPI sites.

Usage

MAPI_Site {
 public_variables
 }

Arguments

- ***public_variables***

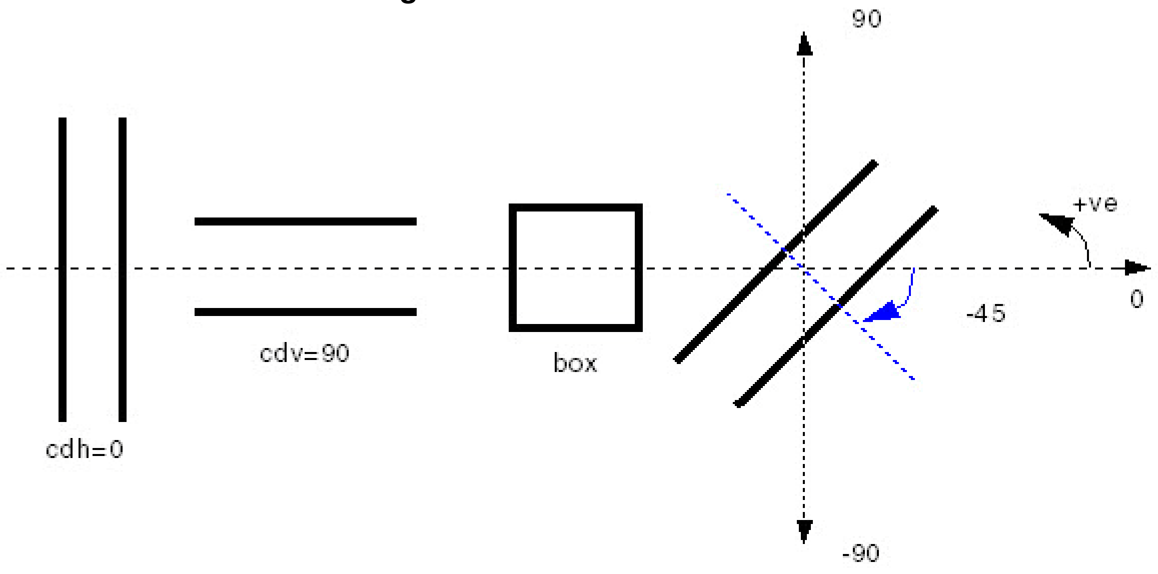
  This can be any of the following:

  _center_x *x*: Specifies site center point x coordinate in database units (dbu).

  _center_y *y*: Specifies site center point y coordinate in dbu.

  _cd_orientation *site_orientation*: Specifies site orientation. Possible values are cdh (horizontal), cdv (vertical), box or an angle value from -90 to 90 as shown in Figure 2-1.

Figure 2-1. Site Orientations



  _cd_type *site_type*: Specifies the CD type. Possible values are width or space. Those values refer to as-drawn layout polygons; width measures the internal distance between polygon edges, while space measures the external one.

  _cd_feature_type *feature_type*: This combines with _cd_orientation to support any angle orientation of *feature_type*: CD or box.

  _cd_opacity *site_opacity*: Specifies the opacity of the site. Possible values are dark or clear. This value does is for reference only, it has no effect on current object behavior.

Note

This setting does not invert the *site_type* when it is set to clear.

_cd_min *value*: Specifies the site's minimum value in the direction of measurement in dbu.

_cd_max *value*: Specifies the site's maximum value in the direction of measurement in dbu.

_cd_h *value*: Specifies site horizontal dimension value in dbu.

_cd_v *value*: Specifies site vertical dimension value in dbu.

_validate_flag *flag*: A flag used specify whether the site is validated in the layout or not. Possible values are 0 (site not validated) or 1 (site validated). All values of MAPI_Site data members can be obtained or modified using the following code:

Site_object configure -*data_member* -*new_value*

set value [*Site_object* cget -*data_member*]

_reason_not_validated *reason*: Specifies the reason for CD validation failure when the _validate_flag is set to 1 in the MAPI_CD_Validate command. Note that all values of MAPI_Site data members can be obtained or modified using:

Site_object configure -*data_member* -*new_value*

set value [*Site_object cget* -*data_member*]

method cdSpec

The method cdSpec specifies a basic subset of the class fields.

Usage

method cdSpec {***cd_orientation cd_opacity***
cd_min cd_max center_x center_y feature_type}

Arguments

- ***cd_orientation***

  Specifies the orientation of the CD (as shown in Figure 2-1).

- ***cd_opacity***

  Specifies opacity of the CD, dark, or clear.

- ***cd_min***

  Specifies the minimum CD value. Possible values are cdh (horizontal), cdv (vertical), box, or an angle value from -90 to 90 as shown in Figure 2-1.

- ***cd_max***

  Specifies the maximum CD value in dbus.

- ***center_x***

  Specifies the center x coordinate position in dbus.

- ***center_y***

  Specifies the center y coordinate position in dbus.

- ***feature_type***

  Specifies box or CD. Specification of feature_type is required when cd_orientation is an angle_value from -90 to 90.

Site Processing (MAPI_ProcessSite)

The processing of a single site is always accomplished by deriving a variant of the MAPI_ProcessSite class. The ProcessSite member function takes as arguments a Calibre MAPI site, a Calibre WORKbench or Calibre MDPview layout, and an output object. This member function should never be invoked directly, rather, it should be called indirectly through the function MAPI_ProcessSites.

This function iterates through a list of sites, hiding any internal parallelism or other computational mechanisms. It may always be considered to have the behavior of the following definition:

```
proc MAPI_ProcessSites { siteList processor L {outputMap ""}}
{
   foreach site $siteList {
      $processor ProcessSite $site $L $outputMap
   }
}
```

There are three classes derived from MAPI_ProcessSite:

- MAPI_CD_Extraction — This class searches for sites based on geometric directives.
- MAPI_CD_Validate — Given a complete site specification, this class determines whether that structure is in the layout.
- MAPI_GetCdDirectionAndValue — This class obtains the value and direction of a measurement around a site center.

Each are used to process sites as defined by the MAPI_ProcessSite class.

MAPI_CD_Extraction

This derived class searches for sites based on geometric directives.

Usage

MAPI_CD_ Extraction ***object_name***
-over_size ***value***
-over_size2 ***value***
-clear_space ***value***
-clear_space2 ***value***
-feature_length ***value***
-search_area_w ***value***
-search_area_h ***value***
-search_area_w_div ***integer***
-search_area_h_div ***integer***
-cd_count_limit ***value***
-cd_pick_mode ***value***
-layer_num ***value***

Arguments

- **-over_size** ***value***

  Specifies over size value in the direction of measurement in database units (dbu).

- **-over_size2** ***value***

  Specifies over size value in dbu at the right side of the feature (while -over_size specifies the over size value in dbu at the left side of the feature). The default value is the same as the value of -over_size.

- **-clear_space** ***value***

  Specifies clear space value on all sides in dbu.

- **-clear_space2** ***value***

  Specifies clear space value in dbu at the top of the feature (while -clear_space specifies the clear space value in dbu at the bottom of the feature). The default value is the value of -clear_space.

- **-feature_length** ***value***

  Specifies min. feature length in dbu.

- **-search_area_w** ***value***

  Specifies the width of the search region in dbu.

- **-search_area_h** ***value***

  Specifies the height of the search region in dbu.

- **-search_area_w_div *integer***

  Number of divided sub-areas in horizontal dimension.

- **-search_area_h_div *integer***

  Number of divided sub-areas in vertical dimension.

- **-cd_count_limit *countmax***

  The maximum number of output sites.

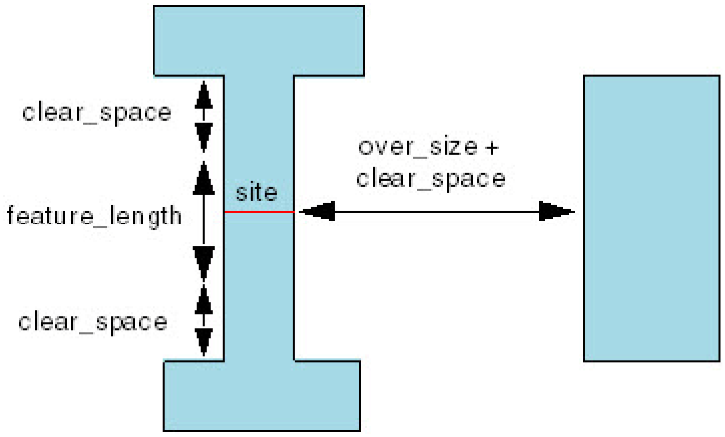
- **-cd_pick_mode *mode***

  Specifies the CD selection mode, in case the sites found by the engine exceeds the maximum value of cd_count_limit (*countmax*). Valid values are: first, uniform, random.

  - If mode is “first”, then the first *countmax* sites are selected.
  - If mode is “uniform”, then *countmax* number of sites are selected uniformly across the search area.
  - If mode is “random”, then *countmax* number of sites are selected randomly across the search area.

Figure 2-2 illustrates the clearance values for a horizontal site or CD.

Figure 2-2. Clearance Values for Horizontal Site/CD



MAPI_CD_Validate

Given a complete site specification (center_x, center_y, cd_min, cd_max, cd_orientation), this class determines whether such a site actually exists in a layout with the clearance values defined in the class arguments.

Usage

MAPI_CD_Validate ***object_name***
-cell_name*top_cell*
-layer_num *value*
-feature_length *value*
-over_size *value*
-over_size2 *value*
-clear_space *value*
-clear_space2 *value*

Arguments

- **-cell_name *top_cell***

  Specifies the top cell name.

- **-layer_num *num***

  Specifies the layer number.

- **-feature_length *value***

  Specifies minimum feature length in database units (dbu).

- **-over_size *value***

  Specifies the over size value in the direction of measurement in dbu.

- **-over_size2 *value***

  Specifies over size value in dbu at the right side of the feature (while -over_size specifies the over size value in dbu at the left side of the feature). The default value is the same as the value of -over_size.

- **-clear_space *value***

  Specifies the clear space value on all sides in dbu.

- **-clear_space2 *value***

  Specifies clear space value in dbu at the top of the feature (while -clear_space specifies the clear space value in dbu at the bottom of the feature). The default value is the value of -clear_space.

Examples

The following is an example of using validation functionality on a given site.

```
proc validateCD { data_file_name } {
# Create a workbench layout from a layout file.
set L [layout create $data_file_name]
# Define a MAPI site list containing a single site.
MAPI_Site site
site cdSpec "cdh" "clear" 600 4000 74174764 74158288
lappend siteList site
# Instantiate a validation object.
MAPI_CD_Validate processor -cell_name TOP_CELL -layer_num 1
-feature_length 100 -over_size 0 -clear_space 0
# Call the top level processor with a NULL output object.
MAPI_ProcessSites $siteList processor $L
set validateFlag [site cget -_validate_flag]
if {$validateFlag == 1} {
# Site is validated
} else {
# site is not validated
}
# Cleanup.
delete object processor
layout delete $L
```

MAPI_GetCdDirectionAndValue

This class extracts information about the input site knowing its approximate center point this information include site direction, site value, type (width || space) and updates its center point to the exact value.

Usage

MAPI_GetCdValueAndDirection ***object_name***
  -layoutLayer ***layer_num***
  -searchRadius ***value_in _dbu***

Arguments

- **-layoutLayer** ***top_cell***

  Specifies the layer number of the input feature.

- **- searchRadius** ***value_in_dbu***

  Specifies the region extents in which the function will search for feature edges. If its dimension exceeds this value, measurement will fail. The default value 3000 dbu.

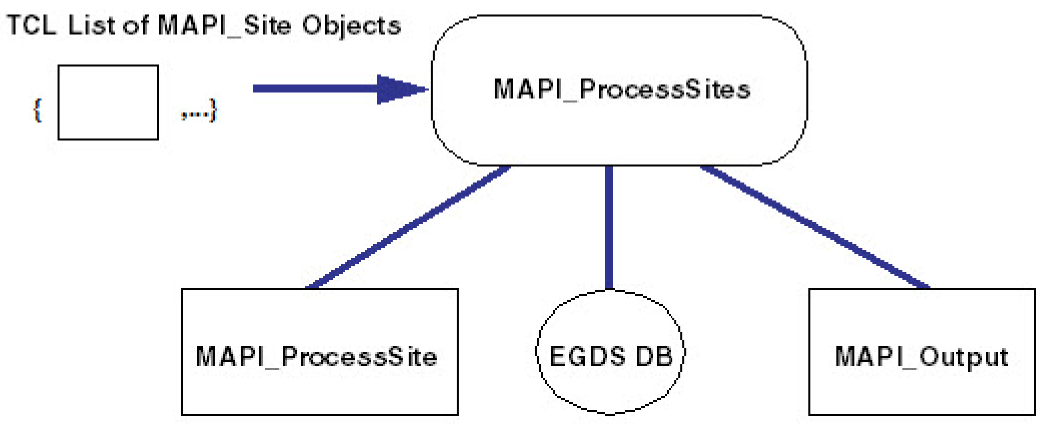
Output Objects (MAPI_Output)

Objects derived from MAPI_Output are intended to abstractly control all output. The constructor and destructor should handle any general processing or setup, and the single OutputSite method should be the only interface to application code that generates output from a MAPI_Site object.

A derived class MAPI_Output_File opens a text file in its constructor. The OutputSite method generates a generic spreadsheet row in that file for a site object. Complete details of the structures can be found in Appendix A, “MAPI Interfaces.”

Figure 2-3 demonstrates how sites, site processors, and output functions work together.

Figure 2-3. Site Processing Prototype



There are two classes derived from MAPI_ProcessSite.

- MAPI_Output_List — Stores sites in a Tcl list.
- MAPI_Output_File — Opens a text file in its constructor.

MAPI_Output_List

The MAPI_Output_List derived class stores sites in a Tcl list. The OutputSite method generates a generic spreadsheet row in that file for a site object.

Another example for a Tcl function also implements a random CD search but outputs are stored in a Tcl list instead of being written in a text file.

```
proc searchCD2_gds { data_fileName } {
# Create a layout object from the given layout file.
set L [layout create $data_fileName]
# Define an output object.
MAPI_Output_List mapiOut
# Create an empty site list
set siteList [list]
# Add several MAPI sites to the list.
MAPI_Site site1
site1 cdSpec "cdh" "clear" 30 3000 500 500
MAPI_Site site2
site2 cdSpec "cdv" "dark" 30 3000 500 500
MAPI_Site site3
site3 cdSpec "box" "clear" 30 4000 500 500
lappend siteList site1 site2 site3
# Instantiate a CD search processor.
MAPI_CD_Extraction processor -over_size 120 \
-clear_space 120 -feature_length 1000 \
-search_area_w 1000 -search_area_h 1000 \
-search_area_w_div 1 -search_area_h_div 1 \
-cd_count_limit 10 -cd_pick_mode random -layer_num 10
# Call the top level site list processor.
MAPI_ProcessSites $siteList processor $L mapiOut
# Get the output sites
set outList [mapiOut GetOutList]
# Cleanup.
delete object mapiOut
}
```

MAPI_Output_File

A derived class MAPI_Output_File opens a text file in its constructor. The OutputSite method generates a generic spreadsheet row in that file for a site object.

Figure 2-3 demonstrates how sites, site processors, and output functions work together.

This example TCL function implements a random CD search and outputs the results to the generic text file format.

```
proc searchCD1_gds { data_fileName } {
   # Create a layout object from the given layout file.
   set L [layout create $data_fileName]
   # Define a text-file output object.
   MAPI_Output_File mapiOut test8.txt
   # Create an empty site list
   set siteList [list]
   # Add several MAPI sites to the list.
   MAPI_Site site1
   site1 cdSpec "cdh" "clear" 30 3000 500 500

   MAPI_Site site2
   site2 cdSpec "cdv" "dark" 30 3000 500 500

   MAPI_Site site3
   site3 cdSpec "box" "clear" 30 4000 500 500

   lappend siteList site1 site2 site3

   # Instantiate a CD search processor.
   MAPI_CD_Extraction processor -over_size 120 \

      -clear_space 120 -feature_length 1000 \
      -search_area_w 1000 -search_area_h 1000 \
      -search_area_w_div 1 -search_area_h_div 1 \
      -cd_count_limit 10 -cd_pick_mode random -layer_num 10

   # Call the top level site list processor.
   MAPI_ProcessSites $siteList processor $L mapiOut

   # Cleanup.
   delete object mapiOut
}
```

A typical output file is:

```
unit: micron
center_x     center_y     cd_type   target_cd_h target_cd_v opacity
67.300000    65.400000    H         0.520000    NA          clear
52.480000    548.780000   V         NA          0.520000    clear
152.480000   55.780000    B         0.500000    0.520000    clear
```

For angled features, a typical output file is:

```
units: microns
center_x     center_y     cd_orientation target_cd_h target_cd_v
-81.420000   -81.20000    45.000000      0.144       NA
-78.752000   -82.213500   45.000000      0.144       0.200

opacity  width/space feature_type
clear    width       cd
clear    width       box
```

Transforms (MAPI_Transform)

The Calibre MAPI Transform class allows you to apply geometric transformations to your layout such as mirror, rotation, scaling, and x-y origin shifting. These controls are to ensure that the data you are comparing have the same scale, orientation, and relative origin of the original layout.

The class MAPI_Transform encapsulates a two dimensional affine transform with rotations limited to multiples of 90 degrees.

The class transforms a single point to a new point. Complete details of the structures can be found in Appendix A, "MAPI Interfaces."

This section covers the following:

- MAPI_Transform Constructor — Used to construct the MAPI_Transform class.

MAPI_Transform Constructor

The following is the constructor used by MAPI_Transform. The method Transform is later used to process the MAPI_Transform object.

Usage

MAPI_Transform ***transformObj*** **-shiftx** ***x0_dbu*** **-shifty** ***y0_dbu*** **-rotate** ***angle*** \
-scale ***scaling*** **-mirror** ***mvalue***

Arguments

- ***transformObj***

  Specifies the name of the MAPI_Transform output object.

- **-shiftx** ***x0***

  Specifies the shift value in the x-direction of the layout coordinate system in database units (dbu). The default value is 0.

- **-shifty** ***y0***

  Specifies the shift value in the y-direction of the layout coordinate system in dbu. The default value is 0.

- **-rotate** ***angle***

  Specifies the rotation angle in degrees (which must be a multiple of 90 degrees). The default value is 0.

- **-scale** ***scaling***

  Specifies the scaling factor. The default value is 1.

- **-mirror** ***mvalue***

  Specifies a mirror value. Possible values are x, y, xy, and no value specified (no mirror). The default value is no value is specified.

Examples

The following example creates a MAPI_Transform object

```
# Define a transformation of rotation by 180 degrees.
MAPI_Transform transformObj -rotate 180
# Apply the defined transformation to a location (-100, 800) and store the
# result in x and y.
# x = 100, y = -800 after applying the transformation.
set transformedPoint [transformObj Transform -100 800]
set xt [lindex $transformedPoint 0]
set yt [lindex $transformedPoint 1]
```

Image Generation (MAPI_Image)

Frequently a machine interface application must form an image of the local context of a point in the layout.

The MAPI_Image class simulates the image of a layout. This is used in conjunction with the Image member function (also known as a Method) to take the snapshot and output the results to a file.

One class, MAPI_Aerial_Image, derived from MAPI_Image, has been provided to simulate the aerial image of a layout.

This section covers the following:

- MAPI_Image Constructor — Takes a simulated image of the layout.
- method Image — Takes a snapshot of the mask data (layout data).
- MAPI_Aerial_Image — A derived class used to create a simulated aerial image of the layout and to capture a snapshot image.
- method Image (Aerial Image) — Creates an aerial image around the point (x,y) in the top cell of the layout.

Complete details of the structures can be found in Appendix A, “MAPI Interfaces.”

MAPI_Image Constructor

The following is the MAPI_Image constructor that takes a simulated image of the layout.

Usage

**MAPI_Image *imageObj* -layout_db *layout_object* -layer_num *num* -halo_dbu pW\
-image_width *mW* -image_heigh**t ***mH*** **-image_format *value***

Arguments

- ***imageObj***

  Specifies the name of the image.

- **-layout_db *layout_object***

  Specifies the EGDS layout database, where *layout_object* is a handle of the layout object corresponding to the intended input layout.

- **-layer_num *num***

  Specifies the desired EGDS layer number.

- **-halo_dbu *pW***

  Specifies the physical size (width and height) in database units (dbu).

- **-image_width *mW***

  Specifies the image width in pixels. The default value is 400.

- **-image_height *mH***

  Specifies the image height in pixels. The default value is 400.

- **-image_format *value***

  Specifies the image format. Currently gif, png, tiff and jpeg formats are supported. The default format is gif.

method Image

The method Image is a member function that takes a snapshot of the mask data (layout data).

Usage

method Image {*x y* ***imagefilename***}

Arguments

- *x y*

  Specifies the center point of the desired image, database units (dbu).

- ***imagefilename***

  Specifies the name of the output image file.

Examples

The following example creates an image object called "mimage."

```
# load the input layout
set L [layout create lab.gds]

# create a MAPI_Image object called mimage
MAPI_Image mimage -layout_db $L -layer_num 2 -halo_dbu 10000

# Run the Image method on the mimage object.
# This simulates an aerial image, takes a snapshot and outputs the
# image into a file named 'am.gif'
mimage Image 9100 10250 am.gif
```

MAPI_Aerial_Image

A derived class of MAPI_Image, this class simulates the aerial image of a layout. This is used in conjunction with the Image member function (also known as a Method) to take the snapshot and output the results to a file.

MAPI_Aerial_Image uses the following:

- MAPI_Aerial_Image Constructor — Creates a simulated aerial image of the layout to capture a snapshot image.
- method Image (Aerial Image) — Referenced by the MAPI_Aerial_Image constructor.

MAPI_Aerial_Image Constructor

The MAPI_Aerial_Image constructor used to create a simulated aerial image of the layout to capture a snapshot image.

Note

Complete details of the structures can be found in Appendix A, “MAPI Interfaces.”

Usage

**MAPI_Aerial_Image *aImageObj* -layout_db *layout_object* -layer_num *num* -halo_dbu *pW*\
-image_width *mW* -setup *setup* -appearance *value* -image_format *value***

Arguments

- ***-aImageObj***

  Specifies the name of the aerial image object.

- **-layout_db *layout_object***

  Specifies the input layout database. Where ***layout_object*** is a handle of the layout object corresponding to the intended input layout.

- **-layer_num *num***

  Specifies the layer number.

- **-halo_dbu *pW***

  Specifies the physical size in database units (dbu).

- **-image_width *mW***

  Specifies the image width in pixels. The default value is 400.

- **-image_height *mH***

  Specifies the image height in pixels. The default value is 400.

- **-image_format *value***

  Specifies the image format. Currently gif, png, tiff and jpeg formats are supported. The default format is gif.

- **-setup *setup***

  Specifies the setup file for aerial image simulation.

- **-appearance *value***

  Specifies the type of aerial image. Choose 1 for contour image and 2 for an intensity map.

method Image (Aerial Image)

The method Image is used to create an aerial image around the point (x,y) in the top cell of the layout.

Usage

method Image {*x y imagefilename*}

Arguments

- *x y*

  Specifies the center point of the desired image, database units (dbu).

- *imagefilename*

  Specifies the name of the output image file.

Examples

```
# Load the input layout
set L [layout create lab.gds]

# Create a MAPI_Aerial_Image object called aImage

MAPI_Aerial_Image mimage -layout_db $L -layer_num 2 -halo_dbu 10000 \
   -setup setup.in -appearance 1

# Run the Image method on the aImage object
# This simulate .saerial image, take snapshot and output the image into
# a file named ‘am.gif’
aImage Image 9100 10250 am.gif
```

Database Query (MAPI_Egds_Query)

Direct queries of the layout database in a neighborhood of a point may sometimes be necessary. For instance, prototype or proprietary CD search functions may process the geometry in a separate process. The database query class generates a layout file for the geometry in a rectangular extent near the query point.

The MAPI_Egds_Query class is used to get geometric data inside a rectangular extent and write it to a GDS file.

Complete details of the structures can be found in Appendix A, "MAPI Interfaces."

This section covers the following:

- MAPI_Egds_Query Constructor — Queries a GDS file.
- method Query — Referenced by the MAPI_Egds_Query constructor.

MAPI_Egds_Query Constructor

The MAPI_Egds_Query constructor is used to query a GDS file.

Usage

MAPI_Egds_Query -layout_db *L* -layer_num *num*

Arguments

- **-layout_db** ***L***

  Specifies the input layout database to be used for the query.

- **-layer_num** ***num***

  Specifies the EGDS layer to be queried.

method Query

The following method Query is referenced by the MAPI_Egds_Query constructor.

Usage

method Query {***region fileName*** {**topCellName** "***topcellname***"}}

Arguments

- ***region***

  A TCL list containing the lower left and upper right corners of the query extent, in dbu, in the format "x1 y1 x2 y2".

- ***filename***

  Specifies the name of the output GDS file to be created.

- **topCellName** ***topcellname***

  Specifies the name of the starting layout cell for the query. The default is "TOP".

Examples

```
set L [layout create lab.gds]
MAPI_Egds_Query mquery -layout_db $L -layer_num 1
set rect "300000 753000 301000 755000"
set outputGdsFile "simple.gds"
mquery  Query $rect $outputGdsFile
```

Flat Database Traversal (MAPI_RefFilter)

Frequently, it becomes necessary to transform some element of a hierarchical database to the top-level coordinate system. For instance, it becomes necessary to find all flat-perspective placements of a particular cell. The MAPI_Flatten function and MAPI_RefFilter class provide a flexible mechanism for such flat data traversal.

To initiate flat data traversal, submit a layout, starting cell, starting transform, and “reference filter” to the MAPI_Flatten function. That function recursively flattens every placement in the layout to the top coordinate system. For each such placement, it calls the “FilterRef” function of the filter object with the cell and placement transform. You can perform whatever action is necessary as indicated by the information, as well as return a signal that will halt the recursion into the sub-hierarchy of that placement.

Complete details of the structures involved can be found in Appendix A, “MAPI Interfaces.”

The MAPI_RefFilter class uses the method FilterRef function to filter and generate application-specific output for flattened EGDS references. In this case, “cell” is the EGDS cell name and “xform” is the MAPI transform from the cell coordinate system to the output coordinate system. The function will return a Boolean value; if it is true, it will not flatten the sub-hierarchy, otherwise, it will return false.

The following is covered in this section:

- MAPI_Flatten Constructor — Recursively flattens the layout.

MAPI_Flatten Constructor

The MAPI_Flatten constructor recursively flattens the layout. This function can be modified by the choice of the *filter* input parameter.

Usage

MAPI_Flatten {
db cell X filter
}

Arguments

- ***db***

  Specifies the input layout database to be flattened.

- ***cell***

  Specifies the starting cell.

- ***X***

  Specifies the starting transform.

- ***filter***

  Specifies a placement filter object.

Examples

The following example code notes all flattened placements of a subset of cells.

```
#
# Test the flatten function.
#
# Create a layout.
set testLayout [layout create m8051.oas]
# Make an array of cells of interest.
set cellsOfInterest [list ndrv_power ICV_48]
array set targetCells ""
foreach cell $cellsOfInterest {
   set targetCells($cell) 1
}
# Make an array of cells that contain a target cell
# in their subhierarchy. This is an important performance
# assistant. This assumes that that the output of
# cells from "testLayout cells" is in bottom-up topological
# order.
array set targetDescendants ""
foreach cell $cellsOfInterest {# Add the target cells.
   set targetDescendants($cell) 1
}
foreach cell [$testLayout cells] {
# Scan placements to propagate the 'sub-hierarchy
#containment' property.
foreach place [$testLayout iterator ref $cell range 0 10000] {
      set pCell [lindex $place 0];# The placed cell.
      if { [array get targetDescendants $pCell] != "" } {
         set targetDescendants($cell) 1
         break;# No need to continue the scan.
      }
   }
}
# Define a filter that prints occurences of target cell placements.
class RefFilter_printCells {
   inherit MAPI_RefFilter
   method FilterRef { cell xform } {
   global targetDescendants
   global targetCells
# First prevent flattening of sub-hierarchies that don't
# contain any target cells.
   if { [array get targetDescendants $cell] == "" } {
   return true
   }
# Process targets.
   if { [array get targetCells $cell] != "" } {
# Get the translation by transforming the point (0,0).
      set translation [$xform Transform 0 0]
      puts "Target $cell at <[lindex $translation 0], \
      [lindex $translation 1]>."
      }
   return false
   }
}
# Invoke.
set IX [MAPI_Transform \#auto];# Identity starting transform.
set refFilter [RefFilter_printCells \#auto]
MAPI_Flatten $testLayout [$testLayout topcell] $IX $refFilter
# Cleanup
layout delete $testLayout
```

Site Filtering (MAPI_SiteFilter)

Several metrology applications require some instantiation of a filtering mechanism to reduce, qualify, or group sites based solely on their two-dimensional location. For instance, some wafer measurement tools can use one alignment operation for all measurement sites in a vicinity. Since alignment operations are expensive, minimizing their number becomes a factor. In another context, if you have many sites and want to choose a subset for measurement to meet a time budget, a random sub-sampling filter can be helpful.

This class hierarchy is intended to perform such operations. As usual, an abstract and general interface permits changing the filtering behavior without changing the greater application, facilitates reuse of derived classes, and permits implementation changes without any disturbance to users of the class. The filter base class, MAPI_SiteFilter, has a single member function, “FilterSites”, which takes as input a TCL list of sites and returns a TCL list containing a subset of the objects in the input list.

Complete details of the structures involved can be found in Appendix A, “MAPI Interfaces.”

The filter base class, MAPI_SiteFilter, has a single member function, ProcessSite, which takes as input a TCL list of sites and returns a TCL list containing a subset of the objects in the input list.

Several filter derivatives are provided:

- Grid filter (MAPI_SiteFilter_grid) — Selects the closest site to each grid point. Some grid points may have no associated site. For each grid vertex, the closest site, if one exists within some neighborhood, is chosen.
- method FilterRef — This method filters and generates application-specific output for flattened EGDS references.
- Random distribution filter (MAPI_SiteFilter_randomSample) — This derived class selects a random sub-sample from the site list. The constructor arguments include the desired number of selected sites.
- Area filter (MAPI_SiteFilter_interact) — This derived class selects only sites that interact with the given optionally sized layout layer.
- Cluster filter (MAPI_SiteFilter_cluster) — Cluster the sites according to the given parameters, then annotate the sites with the cluster number using the “groupID” field. Return all sites that fell within the final clustering.
- method FilterSites — This method selects a subset of a specified list of sites.

MAPI_SiteFilter_grid

This derived class allows you to choose a subset of sites based on a grid. The grid parameters are provided in the constructor. For each grid vertex, the closest site, if one exists within some neighborhood, is chosen.

Usage

MAPI_SiteFilter_grid -gridOriginX ***orx_value*** **-gridOriginY** ***ory_value*** **-nCols** ***col_value*** **-nRows** ***row_value*** **-xPitch** ***xvalue*** **-yPitch** ***yvalue***

Arguments

method FilterRef

The following method FilterRef filters and generates application-specific output for flattened EGDS references.

Usage

method FilterRef {***cell xform***} {**return** ***boolean***}

Arguments

- ***cell***

  Specifies the EGDS cell name.

- ***xform***

  Calibre MAPI transform from the cell coordinate system to the output coordinate system

Return Values

The function returns a boolean result. If true, do not flatten the sub-hierarchy.

MAPI_SiteFilter_randomSample

This derived class allows you to reduce a list by choosing a random subset. The constructor arguments include the desired number of selected sites.

Usage

MAPI_SiteFilter_randomSample ***object_name*** **-nSamples** ***value*** **-randomSeed** ***value***

Arguments

- **-nSamples** ***value***

  Specifies the desired number of samples in the output. The default is 0.

- **-randomSeed** ***value***

  Specifies the seed for random processing. The default is 0.

MAPI_SiteFilter_interact

Sites are rejected if they fail to intersect a layer in a specified layout database. Such a filter can be used, for instance, to remove sites which have no nearby alignment structure. For instance, if the measurement site is in the middle of a set of straight lines, a measurement tool that requires 2-D structures will not be able to do fine alignment. In this case, the “permissible” area could be determined in Calibre WORKbench/MDPview, Calibre, or another program and stored as a layer in the database.

Usage

MAPI_SiteFilter_interact ***object_name*** **-layoutDB** ***layout_handler*** **-layoutLayer** ***layer_num***
[-sizeValue *value_in_dbu*]

Arguments

- **-layoutDB** ***layout_handler***

  Specifies the layout database.

- **-layoutLayer** ***layer_num***

  Specifies the layout layer number.

- -sizeValue *value_in_dbu*

  Optional argument to specify size value if the layout is required to be sized. The default value is 0.

MAPI_SiteFilter_cluster

The cluster filter performs clustering on the sites. A typical behavior is to form no more than a maximum number of clusters each with less than a maximum specified diameter (greatest distance between any two members) and reject all sites not in any cluster. The cluster filter satisfies, for instance, the requirement that no more than a given number of alignment operations can be afforded for the inspected object. In this case, one alignment point may be generated and used for all sites in the cluster. The constructor arguments provide the clustering parameters. As a side effect, the filter annotates the output site objects with the containing cluster index in the "groupID" data member of the Calibre MAPI site.

Usage

MAPI_SiteFilter_cluster -maxClusterCount *value* -maxClusterDiameter *value*

Arguments

- **-maxClusterDiameter *value***

  Specifies the maximum value for number of clusters generated. Actual number of clusters found may be less.

- **-maxClusterDiameter *value***

  Specifies the maximum value for the cluster size in database units (dbu).

method FilterSites

The following method FiltersSites selects a subset of a specified list of sites.

Usage

public method FilterSites {***siteList***}

Arguments

- ***siteList***

  Specifies a TCL list of sites.

Return Values

The function returns a list containing a subset of ***siteList***.

Examples

The following is an example of how filtering is used.

```
set siteLocs "3 4 1 2 9 2 11 2 10 10 5 6 2 4 1 7 34 22 22 11 99 100 \
   100 1000"
set siteList ""
foreach { x y } $siteLocs {
set siteObj [MAPI_Site \#auto]
$siteObj configure -_center_x $x -_center_y $y
lappend siteList $siteObj
}
set filter [MAPI_SiteFilter_grid \#auto]
$filter configure -gridOriginX 4 -gridOriginY 2 -nCols 2 -nRows 2 \
   -xPitch 2 -yPitch 2
set filteredList [$filter FilterSites $siteList]
puts $filteredList
produces the output
mAPI_Site0 mAPI_Site1 mAPI_Site2 mAPI_Site5
```

Alignment Point Detection (MAPI_FindAlignmentPoint)

The MAPI_FindAlignmentPoint class detects two-dimensional geometrical configurations inside of a specified query extent. It chooses the point closest to the extent center as the output, or returns a failure signal if no such point could be found. This simple behavior is useful for determining if a given region can be aligned properly on some metrology tools that require two dimensional image signals for correct stage alignment.

Complete details of the structures involved can be found in Appendix A, “MAPI Interfaces.”

The following topics are covered in this section:

- class MAPI_FindAlignmentPoint — Detects two-dimensional geometrical configurations inside of a specified query extent.
- method FindPoints — Returns the 2-D point around which the alignment region should be centered.

class MAPI_FindAlignmentPoint

This class detects two-dimensional geometrical configurations inside of a specified query extent.

Usage

MAPI_FindAlignmentPoint ***object_name*** **-layoutDB** ***layout_handler*** **-layoutLayer** ***layer_num***

Arguments

- **-layoutDB** ***layout_handler***

  Specifies the layout database.

- **-layoutLayer** ***layer_num***

  Specifies the layout layer number.

method FindPoints

The method FindPoints member function is used to return the 2-D point around which the alignment region should be centered.

Usage

public method FindPoints {***x1 y1 x2 y2***}

Arguments

- ***x1 y1 x2 y2***

  Specifies the search extent in database units.

Return Values

The function returns a two-element list containing the point, or “false”.

Extract Clips from a Layout

MAPI provides two functions, **mapiDumpToGds** and **mapiDumpToLayout**, to clip portions of a layout and output the contents (either to an external file or to the layout viewer directly). These functions are commonly used to extract small clips from a layout which are used for alignment reference in metrology machines.

Complete details of the structures involved can be found in Appendix A, "MAPI Interfaces."

The following are covered in this section:

- mapiDumpToGds — Clips a portion of a layout and creates a GDS file containing the shapes from the specified cell on the specified layer within the rectangle.
- mapiDumpToOasis — Clips a portion of a layout and creates an OASIS®[1] file containing the shapes from the specified cell on the specified layer within the rectangle.
- mapiDumpToLayout — Clips a portion of the layout and outputs it to a layout in the viewer rather than an external 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.

mapiDumpToGds

The mapiDumpToGds function clips a portion of a layout and creates a GDS file containing the shapes from the specified cell on the specified layer within the rectangle. The output is single-layer, flattened, and merged. The input layout must be previously loaded into the viewer. The input layout format can be GDSII, OASIS, mask pattern, or mask job deck.

Note
This function does not require the Calibre MAPI license or the -mapi startup switch.

Usage

mapiDumpToGds
layout_obj
selected_layer
{
x1 y1 x2 y2
}
cellname
clipFileName

Arguments

- ***layout_obj***

  Specifies the layout object.

- ***selected_layer***

  Specifies the input layer.

- ***x1 y1 x2 y2***

  Specifies the rectangle border region as four parameters (lower-left-x lower-left-y upper-right-x upper-right-y).

- ***cellname***

  Specifies the name of the cell containing the shapes.

- ***clipFileName***

  Specifies output file name of the clip.

Return Values

None.

Examples

```
mapiDumpToGds $L $selected_layer $rect $cellname $clipFileName
```

mapiDumpToOasis

The mapiDumpToOasis function clips a portion of a layout and creates an OASIS file containing the shapes from the specified cell on the specified layer within the rectangle. The output is single-layer, flattened, and merged. The input layout must be previously loaded into the viewer. The input layout format can be GDSII, OASIS, mask pattern, or mask job deck.

Note
This function does not require the Calibre MAPI license, nor the -mapi startup switch.

Usage

mapiDumpToOasis
layout_obj

selected_layer
{
x1 y1 x2 y2
}
cellname

clipFileName

Arguments

- ***layout_obj***

  Specifies the layout object.

- ***selected_layer***

  Specifies the input layer.

- ***x1 y1 x2 y2***

  Specifies the rectangle border region as four parameters (lower-left-x lower-left-y upper-right-x upper-right-y).

- ***cellname***

  Specifies the name of the cell containing the shapes.

- ***clipFileName***

  Specifies output file name of the clip.

Return Values

None.

Examples

```
mapiDumpToOasis $L $selected_layer $rect $cellname $clipFileName
```

mapiDumpToLayout

The mapiDumpToLayout function clips a portion of the layout and outputs it to a layout in the viewer rather than an external file. The return value is the layout handle of the clip.

Usage

mapiDumpToLayout ***source_handle selected_layer* {*x1 y1 x2 y2*} *cellname*** [*destHandle*]

Arguments

- ***source_handle***

  Specifies the handle of the layout object.

- ***selected_layer***

  Specifies the input layer.

- ***x1 y1 x2 y2***

  Specifies the rectangle border region as four parameters (lower-left-x lower-left-y upper-right-x upper-right-y).

- ***cellname***

  Specifies the name of the cell containing the shape.

- *destHandle*

  An optional argument that allows you to specify a handle name for the layout object. If not specified, Calibre MAPI instead creates a handle name with the automatic naming scheme “layout_dump*n*” with the *n* incremented with each mapiDumpToLayout call (for example, layout_dump0, layout_dump1, and so on). If you are creating a handle, make sure that a pre-existing handle with the same name does not exist, as it will generate an error.

Return Values

This function returns the handle for the layout object that was created by this command.

Examples

```
mapiDumpToLayout $handle $selected_layer $rect $cellname
```

Appendix A MAPI Interfaces

All Calibre MAPI interfaces can be found in the mdp_mapi.itcl file distributed with the Calibre MAPI product.

Interface Example

The following is a detailed listing of all available interfaces for Calibre MAPI.

```
#*******************************************************************CPY11*#
#*Copyright Mentor Graphics Corporation 2009  All Rights Reserved. CPY12*#
#*                                                                 CPY13*#
#* THIS WORK CONTAINS TRADE SECRET AND PROPRIETARY INFORMATION     CPY14*#
#* WHICH IS THE PROPERTY OF MENTOR GRAPHICS CORPORATION OR ITS     CPY15*#
#*   LICENSORS AND IS SUBJECT TO LICENSE TERMS.                    CPY16*#
#*******************************************************************CPY17*#

#
# mdp_mapi.itcl: MAPI class and function declarations.  These specify the
# public interface.
#
#

#
# Measurement site representation.
#

########################################################################
#
# CLASS:MAPI_Site
#
# PURPOSE :A generalized "site".
#
########################################################################
class MAPI_Site {

    public variable _center_x               ;# Center point X coordinate.
    public variable _center_y               ;# Center point Y coordinate.

    # Possible values are "cdh" || "cdv" || "box" || "angle value {-90 ---
> 90}
    # for horizontal(0), vertical(90 || -90), square(box) and any
    # arbitrary angle CDs respectively.
    public variable _cd_orientation
    # To supported rotated box, _feature_type is introduced.
    # Its possible values are "cd" ||"box"
    # In this case, the rotation angle is specified with _cd_orientation
    public variable _feature_type  ""
    public variable _cd_type     ;# CD type. Possible values are
                                  # "width" || "space"
    public variable _cd_opacity  ;# CD tone. Possible values are
                                  # "dark" || "clear"
    public variable _cd_min                ;# Minimum CD value.
    public variable _cd_max                ;# Maximum CD value.
    public variable _units "dbu"           ;# Units, Default is dbu.
    public variable _cd_h                  ;# CD horizontal dimension.
    public variable _cd_v                  ;# CD vertical dimension.
    public variable _validate_flag
    public variable _reason_not_validated ""  ;# reason for validation
                                               # failure
    public variable _image_acquisition "NO" ;
                                 # Whether to acquire and image or not
    public variable _metrology_algorithm "" ;
                                 # What Algorithm is used for measurment

    public variable _automatically_adjust_center_point 1

    # A variable to attach whatever property
    # Expected format a list of { propName1 propVal1 propName2 propVal2}
    # Or could be used with full freedom to the users.
    public variable _properties

    ## Image Mapping Information
    ## These are mainly used for AMAT format.
    public variable _image_content_gds_clip_path   ""
    public variable _image_content_mask_name    "TEST"
    public variable _image_content_layer_number ""
    public variable _image_content_design_type  "DesignPreOPC"
    public variable _image_content_fov_w      "35"
    public variable _image_content_fov_h      "35"
    public variable _image_content_fov_llx    "20"
    public variable _image_content_fov_lly    "30"

    ## Different CD's for different layers
    public variable _target_cd                 0
    public variable _mask_cd                   0
    public variable _sim_cd                    0
    public variable _drawn_cd                  0

    public variable _row                       0
    public variable _col                       0
    public variable _space_pitch               0

    public variable _sa_out_cd

    public variable siteID;# A general purpose index.
    public variable groupID;# A general purpose group index.

    # Specify site status.
    # Possible values are  "input" || "output" . Default value is "none".
    public variable _status

    constructor {} {
   set _status "none"
    }
    destructor {}


######################################################################
    #
    # PROC:MAPI_Site::cdSpec
    #
    # INPUTS:cd_orietation, cd_opacity, cd_min, cd_max, x, y
    #
    # PURPOSE :Specify a basic subset of the class fields.
    #

######################################################################
    method cdSpec {args}
}





#
# Site processing infrastructure.
#

######################################################################
#
# PROC:MAPI_ProcessSites
#
# Inputs:siteList........A list of MAPI_Site objects.
#     processor.......A MAPI_ProcessSite object.
#     L...............An EGDS layout.
#     outputMap.......A MAPI_Output object.
#
# Output:void
#
# Purpose:A computational wrapper for CD site processing.
#
#
######################################################################
# proc MAPI_ProcessSites { siteList processor L {outputMap ""}}

######################################################################
#
# CLASS:MAPI_ProcessSite
#
# Purpose :Do application-specific processing of a given site.
#
#
######################################################################
class MAPI_ProcessSite {
    constructor {} {}
    destructor {}

    method ProcessSite {site L outputMap} {}
}
```

```
######################################################################
#
# CLASS:MAPI_CD_Extraction
#
# Purpose :It is a derived class from MAPI_ProcessSite base
#               class to search for certain CDs inside the layout.
#
#
######################################################################
class MAPI_CD_Extraction {
    inherit MAPI_ProcessSite

    public {
   variable over_size ""          ;# over size value in dbu.
   variable over_size2 ""         ;
    # over size value in dbu on the right side, default is over_size.
   variable clear_space ""        ;# clear space value in dbu.
        variable clear_space2 ""       ;
    # clear space value in dbu at the top. default is clear_space.
   variable feature_length ""     ;# site min feature length in dbu.
   variable search_area_w ""      ;# search area width in dbu.
   variable search_area_h ""      ;# search area height in dbu.
   variable search_area_w_div ""
   variable search_area_h_div ""
   variable cd_count_limit ""     ;# max. number of output sites
   variable cd_pick_mode ""       ;
    # picking criteria, can be first||random||uniform.
   variable layer_num ""
    }

    private {
   variable _mode 0
   variable _fid ""
   variable _fid_gds ""
   variable _fileName ""
   variable _fileNameGds ""
    }


    constructor {args} {
   eval configure $args
   CheckArgs
    }
    destructor {}

    private method CheckArgs {}
    public method ProcessSite { site L outputMap }
}

######################################################################
#
# CLASS:MAPI_CD_Validate
#
# Purpose :It is a derived class from MAPI_ProcessSite class
#              to validate the existance of certain CDs inside the layout.
#
#
######################################################################
```

```
class MAPI_CD_Validate {
    inherit MAPI_ProcessSite

    public variable over_size ""       ;# over size value in dbu.
    public variable over_size2 ""      ;
      # over size value in dbu on the right side, default is over_size.
    public variable clear_space ""     ;# clear space value in dbu.
    public variable clear_space2  ""   ;
      # clear space value in dbu at the top. default is clear_space.
    public variable feature_length ""  ;# site min feature length in dbu.
    public variable cell_name ""       ;# top cell name
    public variable layer_num  ""      ;# layer number to validate


    constructor {args} {
        eval configure $args
   CheckArgs
    }

    private method CheckArgs {}
    method ProcessSite {site L outputMap}
}




######################################################################
#
# CLASS:MAPI_GetCdDirectionAndValue
#
# Purpose :Around a site center (x,y) get the CD value
#                and direction of measurment and updates the input site
#                with the cd value and orientation.
#
#
######################################################################
class MAPI_GetCdDirectionAndValue {
    inherit MAPI_ProcessSite

    public variable layoutLayer ""
    public variable searchRadius 3000  ;# in dbu.

    constructor { args } { eval configure $args }

    public method ProcessSite {site L outputMap}
}


#
# Output infrastructure.
#

####################################################################
#
# CLASS:      MAPI_Output
#
# Purpose :   An abstract output class.
#
```

```
######################################################################
class MAPI_Output {
    constructor {} {}
    destructor {}

    method OutputSite {site} {}
}


####################################################################
#
# CLASS:     MAPI_Output_File
#
# Purpose :  A derived output class to output sites to a text file.
#
#
######################################################################
class MAPI_Output_File {
    inherit MAPI_Output

    public variable _fileName ""     ;# Output file name
    public variable _fileNameGds ""  ;# Output gds file name

    private variable fh ""

    constructor { {fileName ""} {fileNameGds ""} } {
        if {$fileName == ""} { error "No input file specified"
   } else { set _fileName $fileName }
        set fh [open $_fileName w]

        ## Add file header
        puts $fh "units: microns"
        puts $fh [format "center_x\t\tcenter_y\t\tcd_orientation\t\
ttarget_cd_h\ttarget_cd_v\topacity\t\twidth/space\t\tfeature_type"]
        flush $fh
    }

    destructor { close $fh }

    method OutputSite { site }
}


####################################################################
#
# CLASS:     MAPI_Output_List
#
# Purpose:  A derived output class to output the sites into a Tcl list.
#
#
######################################################################
class MAPI_Output_List {
    inherit MAPI_Output

    private variable _outSitesList

    constructor {} { set _outSitesList [list] }
    destructor {}
    method GetOutList {}
    method OutputSite { site }
}

##########################################################################
####
#
# CLASS:     MAPI_OutputAMAT
#
# Purpose:   A derived output class to write AMAT XML files from site
objects.
#
#
##########################################################################
####
class MAPI_OutputAMAT {
    inherit MAPI_Output

    public  variable file_name ""


    private variable fd
    private variable version
    private variable precision
    private variable transform_object
    private variable repeated_dies_object
    private variable gss_object ""

    private variable gds_layers ""
    private variable gds_mask_names ""
    private variable gds_DI_types ""

    ## Specific Info for the AMAT XML format
    private variable dbm_target_list ""

    constructor {fileName toolVersion layoutPrecision \
      {masterMaskNames "TEST"}\
      {gssfilename ""} {keyValueList ""} {transformObj ""} {repeatObj ""}}
{

       ## Open the file
       set file_name $fileName
       set fd [open $file_name "w"]
       set version   $toolVersion
       set precision $layoutPrecision
       set transform_object $transformObj
       set repeated_dies_object $repeatObj

       ## Create gss output file
       set gssFileName [lindex $gssfilename 0]
       set opentype [lindex $gssfilename 1]
       if {$gssFileName != ""} {set gss_object \
                [namespace current]::[Metro_OutputGSS \
      #auto $gssFileName $keyValueList -opentype $opentype]}

       ## Write the file header
       puts $fd {<DBMProfile \
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">}
       puts $fd "<DBMProfileVersion>$version</DBMProfileVersion>"
       puts $fd {<ProductName>OPC-Check</ProductName>}
       puts $fd {<CoordinateSystems>}
       puts $fd {<CoordSysList>}
       puts $fd {<CoordSys>}
       puts $fd {<ToolName>Calibre</ToolName>}
       puts $fd {<CoordSysName>Chip</CoordSysName>}
       puts $fd {<UnitsXY>microns</UnitsXY>}
       puts $fd {<BoundaryXY>}
       puts $fd {<MinX>-20000</MinX>}
       puts $fd {<MinY>-20000</MinY>}
       puts $fd {<MaxX>20000</MaxX>}
       puts $fd {<MaxY>20000</MaxY>}
       puts $fd {</BoundaryXY>}
       puts $fd {<CoordSysMaskSet>}

       foreach maskName $masterMaskNames {
          puts $fd {<MaskAliasNameDefinition>}
          puts $fd "<MaskAliasName>[lindex $maskName 0]</MaskAliasName>"
         puts $fd "<MasterMaskName>[lindex $maskName 0]</MasterMaskName>"
          puts $fd {</MaskAliasNameDefinition>}
          lappend gds_mask_names [lindex $maskName 0]
          lappend gds_layers     [lindex $maskName 1]
          lappend gds_DI_types   [lindex $maskName 2]
       }

       puts $fd {</CoordSysMaskSet>}
       puts $fd {</CoordSys>}
       puts $fd {</CoordSysList>}
       puts $fd {</CoordinateSystems>}

       puts $fd {<MasterMaskSet>}
       foreach maskName $masterMaskNames {
         puts $fd "<MasterMaskName>[lindex $maskName 0]</MasterMaskName>"
       }
       puts $fd {</MasterMaskSet>}
    }

    destructor {}

    public {
       method OutputSite {site}
    }
}

##########################################################################
####
#
# CLASS:     MAPI_OutputHitachi
#
# Purpose: A derived output class to write HITACHI files from site
# objects.
#
#
##########################################################################
####
class MAPI_OutputHitachi {
    inherit MAPI_Output

    public  variable file_name ""
    public variable  design_data_name ""

    private variable fd
    private variable eps_info
    private variable orig_header
    private variable site_counter 1
    private variable max_mp_count 1
    private variable header_till_mpcount ""
    private variable header_from_mpcount ""
    private variable transform_object ""
    private variable repeated_dies_object ""
    private variable precision ""
    private variable gss_object ""

    constructor {fileDir designDataName toolName \
      layoutPrecision topCellName \
      {layoutInfo "TEST"} {gssFileName ""} {idw ""} {keyValueList ""}
{transformObj ""} \
      {repeatObj ""}} {
       ## Open the file
       set design_data_name $designDataName
       set file_name [file join $fileDir ${design_data_name}_HSS_IDP.csv]
       set fd [open $file_name "w"]
       set transform_object $transformObj
       set repeated_dies_object $repeatObj
       set precision $layoutPrecision

       ## Now, Create the XML_BLD file and the IDD_Template
       OutputXMLBLD [file tail $idw]
       OutputIDD $layoutInfo $toolName $topCellName

       ## Only if GSS file name is supplied create a gss object
       if { $gssFileName != "" } {
          set gss_object [namespace current]::[Metro_OutputGSS
          #auto $gssFileName $keyValueList]
       }

       ## Create the epsInfo Header
       ## The header here will have every single piece of info
       ## except for the extra MP's
       set header_till_mpcount [list #EPS_ID EPS_Template \
           EPS_Name AP1_Template AP2_Template EP_Template \
                               Type Move_X Move_Y Mode \
             Type AP1_X AP1_Y AP1_Mag AP1_Rot Type AP1_AF_X \
             AP1_AF_Y AP1_AF_Mag Type AP1_AST_X \
             AP1_AST_Y AP1_AST_Mag \
             Type AP2_X AP2_Y AP2_Mag AP2_Rot Type \
             AP2_AF_X AP2_AF_Y AP2_AF_Mag Type \
             AP2_AST_X AP2_AST_Y AP2_AST_Mag \
             Type EP_X EP_Y EP_Mag_X EP_Mag_Y EP_Rot Type \
             EP_AF_X EP_AF_Y EP_AF_Mag Type EP_AST_X\
             EP_AST_Y EP_AST_Mag Type EP_ABCC_X EP_ABCC_Y]

       set header_from_mpcount  [list MP1_Template Type MP1_X MP1_Y
MP1_DNo1 MP1_DNo2 MP1_Name \
      MP1_Object MP1_Type MP1_OutputData MP1_TargetCD \
         MP1_Edge MP1_PosOffset_X MP1_PosOffset_Y \
      MP1_SA_In MP1_SA_Out \
      MP1_Direction MP1_SumLines MP1_MeaPoints \
         MP1_MeaLeng MP1_Diameters MP1_SumArea]

       set eps_info(header) [join "$header_till_mpcount MP_Count
$header_from_mpcount" ,]
       set eps_info(body) {}

       ## Write the header
       OutputHeader
    }

    destructor {}

    public {
       method OutputXMLBLD {idw}
       method OutputIDD    {layoutInfo toolName topCellName}
       method OutputHeader {}

       method OutputSite   {site}
    }
}



#
# Transform.
#

######################################################################
#
# CLASS:MAPI_Transform
#
# Purpose :Consistent implementation of affine transforms.
#      Any derived class should still use the "_statePacket"
#      member to represent the internal transform.
#
# Notes:After the constructor, the variables correpsonding
#      to the transform specification parameters are no
#      longer guaranteed to be valid.
#
#
######################################################################
class MAPI_Transform {
    public variable shiftx 0
    public variable shifty 0
    public variable rotate 0
    public variable scale 1
    public variable mirror "none" ;#  possible value "x", "y", "xy", and
"none"

    protected variable _statePacket

    constructor { args } {
   eval configure $args
        Init
    }
    destructor {}

    method Init { }
    method Transform { x y }
    method Compose { X XC {sx 0} {sy 0} }

    protected method GetState { }
    protected method SetState { statePacket }
}



#
# Image acquisition.
#

######################################################################
#
# CLASS:MAPI_Image
#
# Purpose :Generate a rectangular digital image from a layout extent.
#
#
######################################################################
class MAPI_Image {
    public variable image_format gif  ;# gif, png, etc.
    public variable layout_db ""      ;# layout data base handle
    public variable halo_dbu ""       ;# image physical size (width &
height) in dbu.
    public variable layer_num ""      ;# layer number to be captured
    public variable image_width  400  ;# In pixels
    public variable image_height 400  ;# In pixels

    protected variable units_mult ""

    constructor {args} { eval configure $args }
    destructor {}


######################################################################
    #
    # PROC:Image
    #
    # Inputs:x,y..............The image center point in db units.
    #        imageFileName.....The name of the output file.
    #
    # Output:none
    #
    # Purpose :Form a digital image around the point (x,y) in the
    # top cell of the layout.
    #
    # Example:  set L [layout create lab.gds]
    # MAPI_Image mimage -layout_db $L -layer_num 2 -halo_dbu 10
    # mimage  Image 87399 112347 ab.gif
    #

######################################################################
```

```
        method Image { x y imageFileName }
    }


    ######################################################################
    #
    # CLASS:MAPI_Aerial_Image
    #
    # Purpose :Form a square digital image of the aerial image in an extent.
    #
    #
    ######################################################################
    class MAPI_Aerial_Image {
        inherit MAPI_Image

        public common _maxSimulationFrame 100000    ;# Max size of sim rect
    side, in dbu
        public variable appearance 2                ;
    # Depreciated variable only "2" is accepted
        public variable intens_min 0.0             ;# Minimum Intensity value
        public variable intens_max 1.2             ;# Maximum Intensity value
        public variable color_contrast 0.5          ;# Gamma Factor
        public variable setup ""                    ;
    # LITHO Setup file used for Simulation.
        public variable simpixsz_dbu ""             ;
    # Simulation pixel size in dbu (if not specified will
    # use the image pixel size for simulation)

        constructor {args} { eval configure $args }
        destructor {}

        private method clipLayout { mode layout_src rect \
          {layout_dst tmp_layout} }



    ######################################################################
        #
        # PROC:Image
        #
        # Inputs:x,y.............The center point of the desired image,
        # db units.
        # imageFileName....The name of the output image file.
        #
        # Output:none
        #
        # Purpose:Form an aerial image around the point (x,y) in the
        # top cell of the layout.
        #
        # Example:  set L [layout create gds/lab5.gds]
        # MAPI_Aerial_Image mimage -layout $L -halo_dbu 4.0 -setup setup/
        # lab5.in -simpizsz_dbu 5
        # mimage  Image 9 10 am.gif
        #

    ######################################################################
        method Image { x y imageFileName }
    }
```

```
#
# Direct layout query.
#

######################################################################
#
# CLASS:MAPI_Egds_Query
#
# Purpose :Get geometric data inside a rectangular extent
#     and write it to a GDS file.
#
#
######################################################################

class MAPI_Egds_Query {
    public variable layout_db ""  ;# layout data base handle
    public variable layer_num ""  ;# layer number to query into.

    constructor {args} { eval configure $args }
    destructor {}


######################################################################
    #
    # PROC:Query
    #
    # Inputs:region.........A TCL list containing the lower left and upper
    # right corners of the query extent in dbu, in the format
    # "x1 y1 x2 y2".
    # fileName.......Name of the output GDS file to be created.
    # topCellName....The name of the starting layout cell for the query.
    # The default is "TOP".
    #
    # Output:none
    #
    # Purpose :Get geometric data inside a rectangular extent
    # and write it to a GDS file.
    #
    # Example:  set L [layout create lab.gds]
    # MAPI_Egds_Query mquery -layout_db $L -layer_num 1
    # set rect "300000 753000 301000 755000"
    # set outputGdsFile "simple.gds"
    # mquery  Query $rect $outputGdsFile
    #
    #

######################################################################
    method Query { region fileName { topCellName "TOP" } }
}



#
# Database flattening and related functionality.
#
```

```
######################################################################
#
# CLASS:MAPI_RefFilter
#
# Purpose :An interface class for processing flattened placements.
#
#
######################################################################
class MAPI_RefFilter {


######################################################################
    #
    # PROC:FilterRef
    #
    # Inputs:cell.....EGDS cell name.
    # xform....MAPI transform from the cell coordinate system
    #     to the output coordinate system.
    #
    # Output:boolean: if true, do not flatten subhierarchy.
    #
    # Purpose :Filter and generate application-specific output for
    # flattened EGDS references.
    #
    #

######################################################################
    method FilterRef { cell xform } { return false }
}


######################################################################
#
# PROC:MAPI_Flatten
#
# Inputs:db......A WB layout object.
#     cell....The starting cell.
#     X.......The starting transform.
#     filter..A placement filter object.
#
# Output:void
#
# Purpose :Recursively flatten the layout.  This function can
#     be modified by the choice of 'filter'.
#
#
######################################################################
# proc MAPI_Flatten { db cell X filter }



#
# Filter class hierarchy.
#

######################################################################
#
```

```
# CLASS:MAPI_SiteFilter
#
# Purpose :Base class for selecting subsets of site lists.
#
#
######################################################################
class MAPI_SiteFilter {


######################################################################
    #
    # PROC:FilterSites
    #
    # Inputs:siteList......A TCL list of sites.
    #
    # Output:A list containing a subset of 'siteList'.
    #
    # Purpose :Select a subset of 'siteList'.
    #
    #

######################################################################
    public method FilterSites { siteList }
}

######################################################################
#
# CLASS:MAPI_SiteFilter_grid
#
# Purpose :Select the closest site to each grid point.  Some
#     grid points may have no associated site.
#
#
######################################################################
class MAPI_SiteFilter_grid {
    inherit MAPI_SiteFilter

    public {
   variable gridOriginX   ;# grid origin x coordinate
   variable gridOriginY   ;# grid origin y coordinate
   variable nCols         ;# number of columns in the grid
   variable nRows         ;# number of rows in the grid
   variable xPitch        ;# grid separation in x direction
   variable yPitch        ;# grid separation in y direction
    }

    constructor { args } { eval configure $args }

    # Use the "_center_[xy]" members of the site as the site location.
    public method FilterSites { siteList }
}



######################################################################
#
# CLASS:MAPI_SiteFilter_randomSample
```

```
#
# Purpose :Select a random subsample of 'siteList'.
#
#
######################################################################
class MAPI_SiteFilter_randomSample {
    inherit MAPI_SiteFilter

    public variable nSamples 0       ;
# Desired number of samples in the output.
    public variable randomSeed 0     ;
# If non-zero, use as seed; else the used seed is written here.

    constructor { args } { eval configure $args }

    public method FilterSites { siteList }
}


######################################################################
#
# CLASS:MAPI_SiteFilter_interact
#
# Purpose :Select only sites which interact with the given,
#     optionally sized layout layer.
#
#
######################################################################
class MAPI_SiteFilter_interact {
    inherit MAPI_SiteFilter

    public variable layoutDB        ;# layout data base handle
    public variable layoutLayer     ;# layer number
    public variable sizeValue 0     ;# Size value in EGDS database units.

    constructor { args } { eval configure $args }

    public method FilterSites { siteList }
}


######################################################################
#
# CLASS:MAPI_SiteFilter_cluster
#
# Purpose :Attempt to cluster the sites according to the given
#     parameters.  Annotate the sites with the cluster number
#     using the "group id" field.  Return all sites
#     that fell withing the final clustering.
#
#
######################################################################
class MAPI_SiteFilter_cluster {
    inherit MAPI_SiteFilter

    public variable maxClusterCount            ;
# Actual number of clusters found may be less.
    public variable maxClusterDiameter "" {
```

```
# L-inf norm, database units.
        if { $maxClusterDiameter < 4 } { error "Diamter too small." }
    }

    protected variable clusterCenters

    constructor { args } { eval configure $args }

    public method FilterSites { siteList }
}



#
# Alignment point selection.
#

######################################################################
#
# CLASS:MAPI_SelectAlignmentPoint
#
# Purpose :Find a suitable 2-D geometry configuration on the
#     given layout layer withing a given extent.
#
#
######################################################################
class MAPI_FindAlignment {
    public variable layoutDB ""      ;# layout data base handle
    public variable layoutLayer ""   ;# layer number

    constructor { args } { eval configure $args }


######################################################################
    # PROC:FindPoint
    #
    # Inputs:[xy][12]......The search extent in database units.
    #
    # Output:A two-element list containing the point, or "false".
    #
    # Purpose :Returns the 2-D point around which the alignment region
should be
    # centered, or "false" if no such point could be found.
    #

######################################################################
    public method FindPoint { x1 y1 x2 y2 }
}
```

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.