SIEMENS EDA

DFM Data Analysis User's and Reference Manual

Software Version 2021.2

SIEMENS

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

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction to DFM Data Analysis

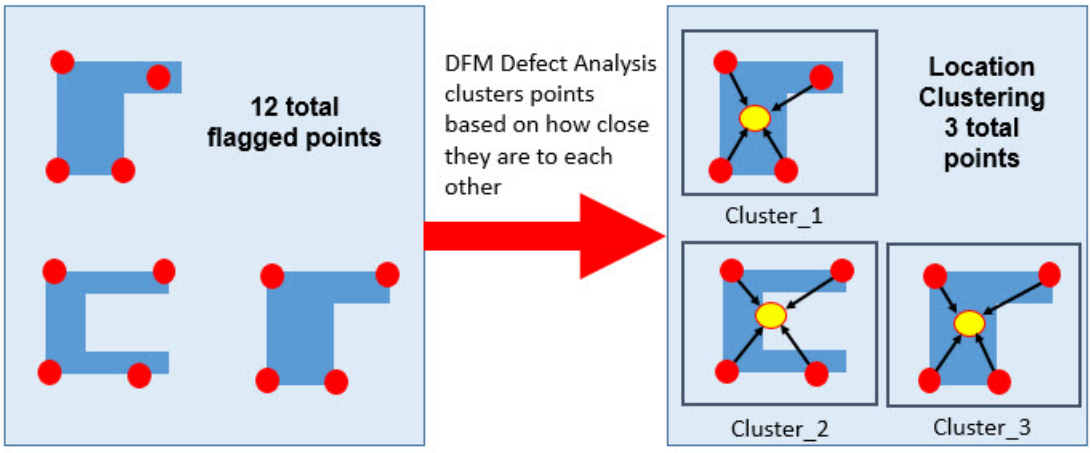
DFM Data Analysis (DFMDA) is a utility used to assist wafer defect analysis by applying various proximity clustering and geometry classification techniques as well as certain layout analysis operations to defect information obtained from wafer inspections.

A typical wafer defect inspection compares the image of circuit patterns between inspected and reference, die to die, or die to database in order to detect defects. Due to the size and complexity of wafer layouts, an inspection can potentially produce a large quantity of defect information points. When performing defect analysis to identify and correct problematic areas on the layout, certain critical information must be extracted from the volume of data produced from an inspection, including location, defect type, and dimension.

DFM Data Analysis helps organize defect information from an input inspection file (a CSV spreadsheet), extracting and organizing information most relevant for defect analysis. The information categories include:

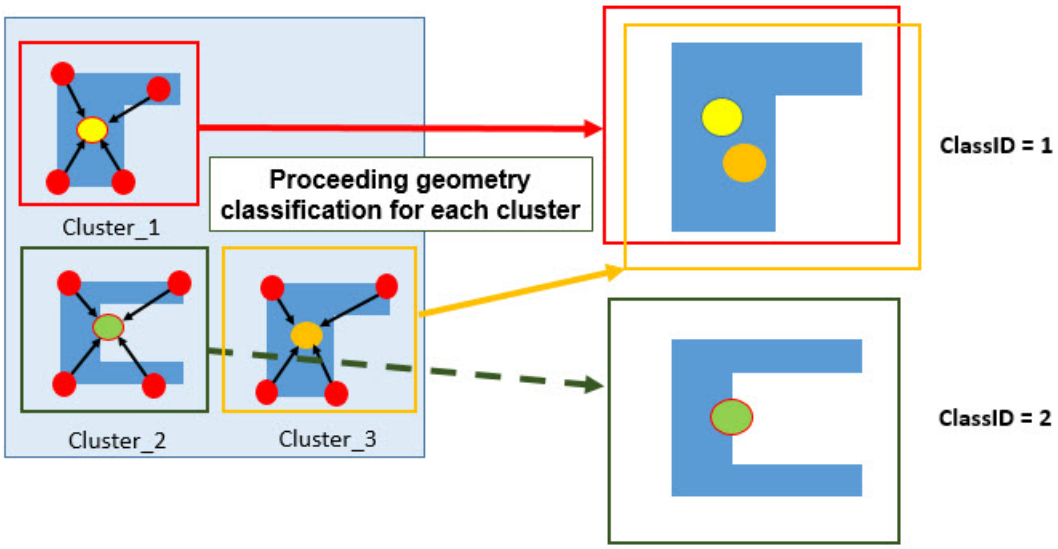
- **Proximity Clustering** — DFMDA extracts the location information (the x- and y-coordinates on a wafer or die) and clusters defect points based on how close they are to each other. All input coordinates are clustered according to user-specified criteria, and assigned a cluster ID. DFMDA utilizes the most optimal clustering and indicates the location of the cluster center, representing a potentially problematic location.

Figure 1-1. Location Clustering



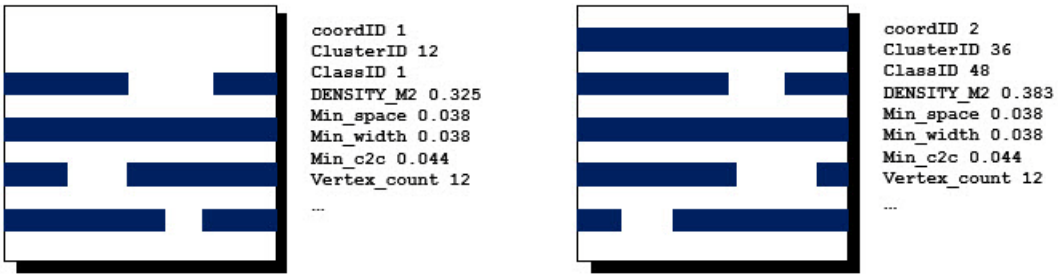
- **Geometry Classification** — DFMDA analyzes the geometry around each proximity cluster location and assigns a classID based on the geometry. Proximity clusters with the same geometry have the same classID. You can optionally save a pattern library with each unique geometry that was identified. Displacements of the cluster center are considered in the classification process.

Figure 1-2. Geometry Classification



- **Feature Extraction** — DFMDA extracts the minimum dimensions of matching defects detected in the input CSV. Density, vertex count, minimum space, minimum width, and minimum corner-to-corner values are provided for each input coordinate.

Figure 1-3. Feature Extraction



DFM Data Analysis Prerequisites

There are several prerequisites for using DFM Data Analysis.

- **Platform support —** DFM Data Analysis 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.

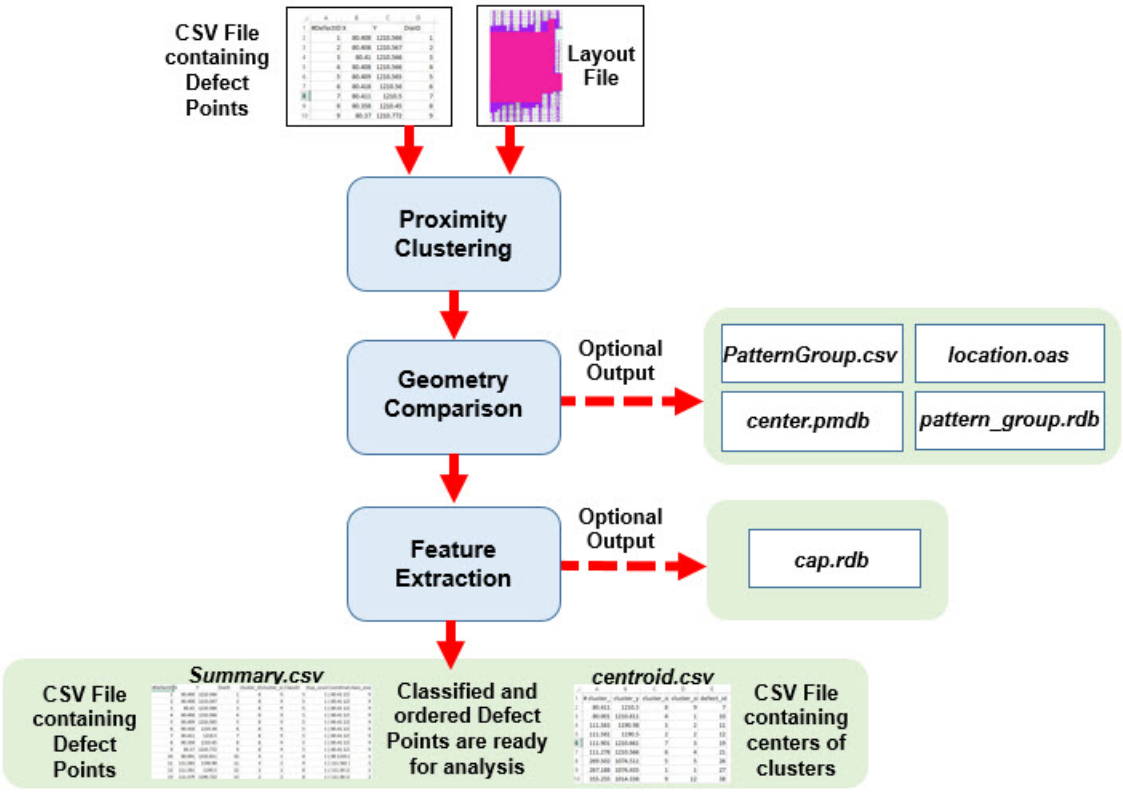
- **Licensing —** To run the complete DFM Data Analysis flow, you must have a license for Calibre Pattern Matching Manufacturing, Calibre® nmDRC™, Calibre® nmDRC-H™, and the Wafer Defect Management utility launched from Calibre® DefectReview™ (documented in the *Calibre DefectReview User's Manual*). For more information on licensing, refer to the *Calibre Administrator's Guide.*
- **Environment Variables —** The CALIBRE_HOME environment variable is required to specify the path of the Calibre software tree. Refer to "CALIBRE_HOME Environment Variable" in the *Calibre Administrator's Guide*.
- **Required Files —** A CSV file produced from a wafer inspection with the chip, layout, or design origin-based defect x- and y- coordinates (an offset correction may be required based on the defect inspection origin to the chip, layout, or design origin difference), and an OASIS or GDSII layout file.

DFM Data Analysis Workflow

The DFM Data Analysis workflow utilizes the results of a wafer inspection to organize defect data to assist in defect analysis.

DFM Data Analysis takes a CSV file from a wafer defect inspection report and the design layout as input. Using the settings in a user-specified configuration file, the utility then performs three different operations: proximity clustering, geometry classification, and feature extraction (computing dimension measurements for all input data points).

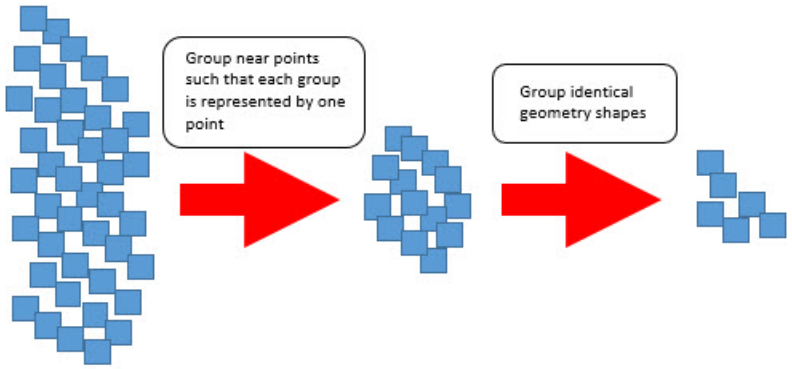
Figure 1-4. DFM Data Analysis Workflow



Once the DFM Data Analysis run is complete, a CSV file is produced containing the original defect data points, but now with additional columns containing the proximity clustering, geometry classification, and feature extraction data, depending on the settings in the run configuration file. The newly-organized and categorized CSV file is then used for defect analysis. Additional relevant output files can be produced based on the settings in the configuration file.

The purpose of this flow is to decrease the number of defects by grouping them by proximity and geometric shapes.

Figure 1-5. Grouping Defects to Decrease Overall Number



Syntax Conventions

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

Table 1-1. Syntax Conventions

Convention	Description
Bold	Bold fonts indicate a required item.
Italic	Italic fonts indicate a user-supplied argument.
Monospace	Monospace fonts indicate a shell command, line of code, or URL. A bold monospace font identifies text you enter.
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.
Example: **DEVice** {***element_name*** [‘(’*model_name*‘)’]} ***device_layer*** {***pin_layer*** [‘(’*pin_name*‘)’] …} [‘<’*auxiliary_layer*‘>’ …] [‘(’*swap_list*‘)’ …] [BY NET | BY SHAPE]	

Chapter 2
Using DFM Data Analysis

DFM Data Analysis takes a CSV file from a wafer defect inspection report. The utility performs several different sorting algorithms to categorize and organize the defect data.

Running DFM Data Analysis From the Command Line

A DFM Data Analysis run can be initiated from a command line.

Prerequisites

- You have fulfilled the prerequisites as described in "DFM Data Analysis Prerequisites" on page 10.
- A CSV file from a wafer inspection (with offset correction). The input CSV file must only contain the following columns:
  - #DefectID — The defect ID header is the first column in the CSV file.
  - x-coordinate column header — The column header name is referenced by the DFMDA configuration file when a run is initiated.
  - y-coordinate column header — The column header name is referenced by the DFMDA configuration file when a run is initiated.
  - DieID — An optional column that indicates the Die ID for each defect point.

Procedure

1. Using a text editor, create a DFMDA configuration text file (for example, *config.txt*) and place it in your working directory. The following table summarizes the configuration file parameters, organized by functional categories.

Table 2-1. DFMDA Configuration File Parameter Summary

Parameter	Description	Examples
General Input and Output		
output_folder	Specifies the output folder for the DFMDA run.	*/home/user/my_output_folder*

Table 2-1. DFMDA Configuration File Parameter Summary (cont.)

Parameter	Description	Examples
output_intermediates	Used in conjunction with other configuration file parameters, outputs intermediate files produced during the DFMDA flow. Refer to the Results section for a description of the files.	1 or 0
generate_center_pattern	If this parameter and geometry_comparison are set to 1, outputs a pattern matching database (PMDB) named *center.pmdb* that contains a unique pattern from the results of geometry classification.	1 or 0
geometry_comparision	Enables geometry classification and outputs a CSV file called *PatternGroup.csv*. If output_intermediates is set to 1, also generates a results database, *pattern_group.rdb* and an OASIS file, *location.oas*, containing polygons used for classification.	1 or 0
Input Defect Sheet (Input CSV)		
defect_sheet	Specifies the input defect sheet (the input CSV file containing all defect data points).	*/home/user/input_defect.csv*
defect_sheet_x_col	References the x-oordinate column header in the input defect sheet.	X
defect_sheet_y_col	References the y-coordinate column header in the input defect sheet.	Y
Input Layout File		
layout_file	Specifies the input layout file.	*/home/user/layout.oas*

Table 2-1. DFMDA Configuration File Parameter Summary (cont.)

Parameter	Description	Examples
layout_system	Specifies the layout system format.	OASIS or GDSII
layout_precision	Specifies the precision of the input layout file.	1000
layer_number	Specifies the target layer under inspection.	17
layer_datatype	Specifies the layer datatype under inspection.	0
Process Parameters		
pattern_shift	Specifies the pattern shift (in user units) used for auto-centering of the pattern location for proximity and classification searches.	0.3
pattern_size	Specifies the size (in user units) a square-shaped region to analyze.	0.4
array_layer	Specifies a layer number and datatype in the input layout. When array_layer is defined, the DFMDA utility identifies whether proximity clusters are inside or outside the geometries on the specified layer.	186.0
real_cluster_coordinate	Controls what location is used as the representative location for a defect cluster. By default (value 0), the center of gravity of all original defect points in the same proximity cluster is used as the representative location. When set to 1, one of the original defects points is selected as the representative location.	1 or 0
feature_extraction	Computes features (density, min_w, min_s, min_c2c, vertex_count) for all defects detected in the input CSV.	1 or 0

Table 2-1. DFMDA Configuration File Parameter Summary (cont.)

Parameter	Description	Examples
measure_range	Defines the measurement region in user units for the feature extraction process to compute defect features. Enabled only if feature_extraction=1.	0.2

The parameters for the configuration file are documented in "DFM Data Analysis Configuration File" on page 23.

Using the configuration file, you can specify different types of output containing proximity, geometry comparison, and feature extraction information. For example:

```
#general input and output
output_folder=./output
output_intermediates=1
geometry_comparison=1

#input defect sheet
defect_sheet=./input/def_ir.csv
defect_sheet_x_col=X
defect_sheet_y_col=Y

#input layout file
layout_file=./input/test_M2.oas
layout_system=OASIS
layout_precision=1000
layer_number=17
layer_datatype=8

#process parameters
pattern_shift=0.2
pattern_size=0.4
array_layer=186.8
feature_extraction=1
measure_range=8.4
```

Example configuration files are also located in "DFM Data Analysis Configuration File" on page 23.

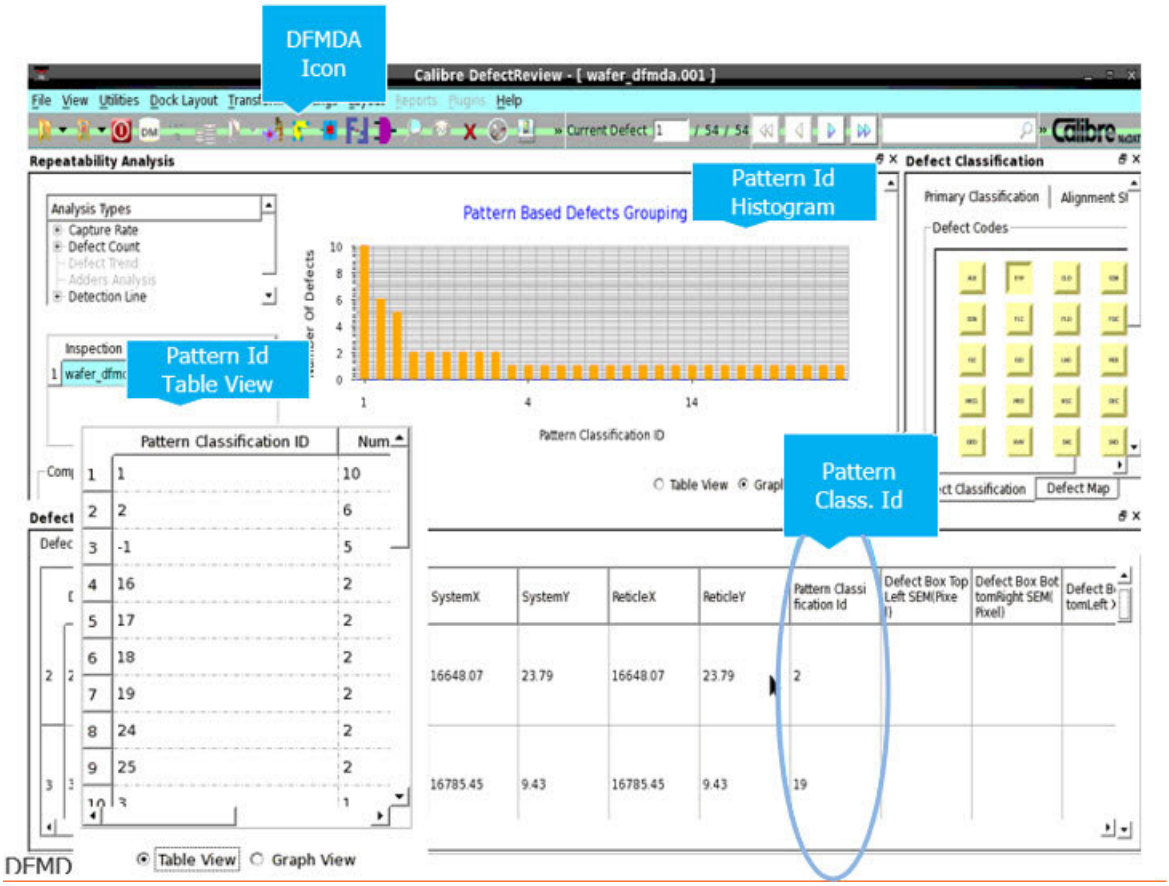
2. At a command prompt, use the following syntax and press Enter:

```
$CALIBRE_HOME/bin/pdl_lib_mgr -turbo DFMDA config config.txt
```

The DFMDA utility processes the defect information contained in the input CSV file.

3. The DFMDA run produces a number of files depending on how the configuration file was set. These files are described in the Results section. You can load the *Summary.csv* file produced from the DFMDA run into the Wafer Defect Management tool in Calibre DefectReview for further analysis. Refer to the *Calibre DefectReview User's Manual* for details.

Figure 2-1. Wafer Defect Management With DFMDA Results



Results

The following files are produced after the DFMDA run, depending on the settings in the DFMDA configuration file:

- *Summary.csv* — The primary output file of a DFMDA operation. If geometry_comparision is set to 1, the CSV contains all points from the original defect sheet, plus cluster size, class ID, duplicate count, coordinates, and class size. If geometry_comparison is set to 0, it contains all points from the original defect sheet, plus cluster ID and cluster size.
- *centroid.csv* — This output CSV file identifies the center of a cluster group from the proximity grouping process. This CSV contains one defect location per defect cluster, as well as the cluster's x- and y-coordinates, cluster ID, cluster size (number of defects with the same clusterID), and defect ID. This is used as an indicator for potentially problematic areas in the layout.
- *center.pmdb* — Produced if generate_center_pattern and geometry_comparision are set to 1. This output file is a pattern matching database (PMDB) that contains one unique pattern from the results of geometry classification.

- *PatternGroup.csv* — Produced if geometry_comparision is set to 1. This output file contains one line for each proximity cluster with the cluster ID, geometry class ID, duplicate count, and class size.
- *pattern_group.rdb* — Produced if output_intermediates and geometry_comparison are set to 1. This file is the DRC results database (RDB) in ASCII format containing a marker at the location of all input defects with the attached properties (cluster ID, geometry class ID, duplicate count, and class size) produced by the geometry classification process.
- *location.oas* — Produced if output_intermediates and geometry_comparison are set to 1. The first layer of this OASIS layout file contains polygons used for geometry classification while the other remaining layers represent the original input CSV defect points.
- *cap.rdb* — Produced if output_intermediates and feature_extraction are set to 1. This results database contains all matching defects found in the input CSV with their features (minimum width, minimum space, minimum corner to corner distance, and vertex count) calculated as the output of the feature extraction process.

The output files are described in further detail in “DFMDA Output Files” on page 28.

Chapter 3
DFM Data Analysis Reference

The DFM Data Analysis flow uses a text-based file to configure a run and can be launched from the command line. You can specify different types of output results by setting arguments in the configuration file.

DFMDA Command

The DFM Data Analysis utility can be launched using a command line syntax.

Usage

pdl_lib_mgr -turbo [*num_proc*] **DFMDA config *config_file_path***

Arguments

- **-turbo** [*num_proc*]

  A required argument that specifies to use multithreaded parallel processing. The argument set must be specified as shown in the syntax line, before the cluster argument. The optional value *num_proc* is a positive integer that specifies the number of CPUs to use for multithreaded processing. If *num_proc* is not specified, Calibre runs on the maximum number of CPUs available for which you have licenses. For best performance, it is recommended that you avoid specifying *num_proc*.

- **config *config_file_path***

  A required argument that specifies the path to the configuration text file used to run the DFMDA process.

Description

DFM Data Analysis can be invoked from the command line, using a text file to configure the run (refer to "DFM Data Analysis Configuration File" on page 23 for a description). The utility uses the results of a wafer inspection to organize defect data to assist in defect analysis.

Examples

Example 1

```
pdl_lib_mgr -turbo DFMDA config config.txt
```

DFM Data Analysis Configuration File

DFM Data Analysis run settings file

This configuration file contains all settings to configure a run of the DFM Data Analysis utility.

Format

The DFMDA utility uses a text format file for configuration. The arguments in this file are categorized into four groups: general input and output, input defect sheet, input layout file, and process. For example:

```
#general input and output
output_folder=./output
output_intermediates=1
geometry_comparison=1

#input defect sheet
defect_sheet=./input/def_ir.csv
defect_sheet_x_col=X
defect_sheet_y_col=Y

#input layout file
layout_file=./input/test_M2.oas
layout_system=OASIS
layout_precision=1000
layer_number=17
layer_datatype=8

#process parameters
pattern_shift=0.2
pattern_size=0.4
array_layer=186.8
feature_extraction=1
measure_range=8.4
```

Note

The configuration file ignores empty lines, is white space insensitive, and treats any line starting with a pound sign (#) or double backslashes (//) as comment lines.

Parameters

General Input and Output

The following arguments are used to specify input and output parameters for the DFMDA run.

- **output_folder** = ***pathname***

  A required argument that specifies the output folder for the DFMDA run.

- output_intermediates = {0 | 1}

  An optional argument set used in conjunction with other configuration arguments to enable the generation of intermediate files used during the DFMDA process, including geometry comparison and feature extraction.

- generate_center_pattern = {0 | 1}

  If this optional argument and geometry_comparison are set to 1, a pattern matching database (PMDB) named *center.pmdb* is output that contains one unique pattern from the results of geometry classification. The default setting is 0 (not enabled).

- geometry_comparision = {0 | 1}

  An optional argument that performs geometry classification. It produces the following files:

  - *PatternGroup.csv* — A CSV file containing one line for each proximity cluster with the cluster ID, geometry class ID, duplicate count, and class size.
  - *pattern_group.rdb* — Produced if output_intermediates is also set to 1, a DRC results database (RDB) in ASCII format containing a marker at the location of all input defects with the attached properties (cluster ID, geometry class ID, duplicate count, and class size) produced by the geometry classification process.
  - *location.oas* — Produced if output_intermediates is also set to 1, an OASIS file with a first layer containing polygons used for geometry classification, and all other layers representing the original input CSV defect points.

  The default setting is 0 (not enabled).

Input Defect Sheet

The following arguments are used to specify parameters for the input CSV spreadsheet containing defect information.

- **defect_sheet** = ***path***

  A required argument that specifies the input defect sheet (the input CSV file containing all defect data points).

- **defect_sheet_x_col** = ***header***

  A required argument that identifies the x-coordinate column header inside the defect sheet (for example, X).

- **defect_sheet_y_col** = ***header***

  A required argument that identifies the y-coordinate column header inside the defect sheet (for example, Y).

Input Layout File

The following arguments are used to specify parameters for the input layout file.

- **layout_file** = ***path***

  A required argument that specifies the input layout file.

- **layout_system** = {**GDSII** | **OASIS**}

  A required argument that specifies the layout system format, GDSII or OASIS.

- **layout_precision** = ***num***

  A required argument that specifies the precision of layout. Refer to the SVRF statement Precision in the *Standard Verification Rule Format (SVRF) Manual* for a complete description.

- **layer_number** = ***num***

  A required argument that specifies the layer under inspection.

- layer_datatype = *type*

  An optional argument that specifies the layer datatype under inspection. The default datatype is 0 (not defined).

Process Parameters

The following arguments are used to specify parameters for the DFMDA process.

- **pattern_shift** = ***val***

  A required argument that specifies the allowed pattern shift (in user units) used for auto-centering of the pattern location for proximity clustering and geometry classification.

- **pattern_size** = ***val***

  A required argument that specifies the size (in user units) size of the region to analyze. The region is a square with sides of length 2*pattern_size.

- array_layer = *layernum.datatype*

  An optional argument that specifies a layer number and datatype in the input layout. When array_layer is defined, the DFMDA utility identifies whether proximity clusters are inside or outside the geometries on the specified layer. See the columns InSRAM and OutSRAM in the file *PatternGroup.csv*, and the properties In_Array and Out_Array in the file *pattern_group.rdb*. See “DFMDA Output Files” on page 28.

- real_cluster_coordinate ={0 | 1}

  An optional argument that controls what location is used as the representative location for a defect cluster. By default (value 0), the center of gravity of all original defect points in the same proximity cluster is used as the representative location. When set to 1, one of the original defects points is selected as the representative location.

- feature_extraction = {0 | 1}

  An optional argument that computes features for all defects detected in the input CSV, including density, minimum width (min_w), minimum space (min_s), minimum corner to corner distance (min_c2c), and vertex count (vertex_count). When this option and output_intermediates are set to 1, the DFMDA run produces a DRC Results Database (RDB) file, *cap.rdb*, that contains the defects with their calculated features. The default is 0 (not enabled).

- measure_range = *val*

  An optional argument that defines the measurement range in user units for the feature extraction process to compute defect features (density, min_w, min_s, min_c2c, and vertex_count). This is enabled when feature_extraction is set to 1.

Examples

You can configure different outputs from a DFM Data Analysis run by using different argument settings.

Example 1 — Complete Flow

The following configuration file runs the entire DFMDA flow (proximity clustering, geometry classification, and feature extraction), generating a *Summary.csv* file containing the original defect sheet appended with additional columns cluster_id, cluster_size, class_id, duplicate_count, class_size, min_w, min_s, min_c2c, and Density.

```
#General Input and Output
output_folder=./output
geometry_comparision=1

#Input Defect Sheet
defect_sheet=./input/input.csv
defect_sheet_x_col=X
defect_sheet_y_col=Y

#Input Layout File
layout_file ./input/test_M2.oas
layout_precision=1000
layer_number=17
layer_datatype=0

#Process Parameters
pattern_shift=0.2
pattern_size=0.4
real_cluster_coordinate=1
feature_extraction=1
measure_range=0.3
```

Example 2 — Proximity Clustering Only

The following configuration file runs the proximity clustering flow, generating the *Summary.csv* file with the original defect sheet data and the additional columns cluster_id and cluster_size.

```
#General Input and Output
output_folder=./output

#Input Defect Sheet
defect_sheet=./input/input.csv
defect_sheet_x_col=X
defect_sheet_y_col=Y

#Input Layout File
layout_file=./input/test_M2.oas
layer_number=17
layer_dataype=0
layout_precision=1000

#Process Parameters
pattern_shift=0.2
pattern_size=0.4
real_cluster_coordinate=1
```

Example 3 — Proximity Clustering and Geometry Classification

The following configuration file runs the proximity clustering and geometry classification flow, generating the *Summary.csv* file with original defect sheet data and the additional columns cluster_id, cluster_size, class_id, duplicate_count, and class_size.

```
#General Input and Output
output_folder=./output
geometry_comparision=1

#Input Defect Sheet
defect_sheet=./input/input.csv
defect_sheet_x_col=X
defect_sheet_y_col=Y

#Input Layout File
layout_file=./input/test_M2.oas
layout_system=OASIS
layout_precision=1000
layer_number=17
layer_datatype=0

#Process Parameters
pattern_shift=0.2
pattern_size=0.4
measure_range=0.3
```

Example 4 — Proximity Clustering and Feature Extraction

The following configuration file runs the proximity clustering flow and feature extraction, generating the *Summary.csv* file with the original defect sheet data and the additional columns cluster_id, cluster_size, min_w, min_s, min_c2c, and Density.

```
#General Input and Output
output_folder=./output
output_intermediates=1

#Input Defect Sheet
defect_sheet=./input/input.csv
defect_sheet_x_col=X
defect_sheet_y_col=Y

#Input Layout File
layout_file=./input/test_M2.oas
layout_system=OASIS
layout_precision=1000
layer_number=17
layer_datatype=0

#Process Parameters
pattern_shift=0.2
pattern_size=0.4
feature_extraction=1
measure_range=0.3
```

DFMDA Output Files

Several output files are created by the DFMDA utility.

Table 3-1. DFMDA Output File Summary

File	Requirements for Generation
Summary.csv	None, generated by default
centroid.csv	None, generated by default
center.pmdb	geometry_comparision=1 and generate_center_pattern=1
PatternGroup.csv	geometry_comparision=1
location.oas	output_intermediates=1 and geometry_comparision=1
cap.rdb	output_intermediates=1 and feature_extraction=1
pattern_group.rdb	output_intermediates=1 and geometry_comparision=1

Summary.csv

- Always generated.
- Format: CSV
- Each row corresponds to a defect location in the input defect sheet.

Table 3-2. Summary.csv File

Column heading	Definition
#DefectID	The defect ID, copied from the defect sheet.
X	The x-coordinate of the defect, copied from the defect sheet.
Y	The y-coordinate of the defect, copied from the defect sheet.
DieID	The die ID of the defect, copied from the defect sheet.
cluster_ID	The ID of the proximity cluster that this defect belongs to.
cluster_size	The number of defects that belong to the proximity cluster.
Included when geometry_comparision=1	
ClassID	The ID of the geometry class that the defect belongs to.
Dup_count	The number of proximity clusters that belong to this geometry class (have the same geometry).
Coordinates	The x- and y-coordinates of the four corners of a 2 dbu square marker at the center of the proximity cluster. The center location is affected by real_cluster_coordinate.
class_size	The number of defects that belong to this geometry class.

centroid.csv

- Always generated
- Format: CSV
- Each row corresponds to a proximity cluster. The center location is affected by real_cluster_coordinate.

Table 3-3. centroid.csv File

Column heading	Definition
cluster_x	The x-coordinate of the center of the proximity cluster.
cluster_y	The y-coordinate of the center of the proximity cluster.
cluster_id	The ID of the proximity cluster.
cluster_size	The number of defects that belong to the proximity cluster.
defect_id	When real_cluster_coordinate=1, this is the defect ID selected as the center of the proximity cluster.

center.pmdb

- Generated if geometry_comparision=1 and generate_center_pattern=1

- Format: Pattern Library (PMDB)
- Has one pattern for each unique geometry class. The pattern has the target layer geometries in the region of the associated proximity cluster. Each pattern has attached properties.

Table 3-4. center.pmdb File

Pattern Property	Definition
ClassID	The ID of the geometry class.
PatDensity	The density of the target layer within the region.
VexCount	The number of vertices on the target layer within region.
cluster_ID	The ID of a proximity cluster associated with the geometry class.
dup	The number of proximity clusters that belong to this geometry class (have the same geometry).

PatternGroup.csv

- Generated if geometry_comparision=1
- Format: CSV
- Has one row for each proximity cluster.

Table 3-5. PatternGroup.csv File

Column heading	Definition
Cell	The name of the layout cell the proximity cluster is located in.
cluster_ID	The ID of the proximity cluster.
ClassID	The ID of the geometry class that the proximity cluster belongs to.
Dup_count	The number of proximity clusters that belong to this geometry class (have the same geometry).
Density	The density of the target layer within the proximity cluster region.
Vertex_count	The number of vertices on the target layer within the proximity cluster region.
InSRAM	Used only when array_layer is defined, and set to 0 otherwise. When array_layer is defined, this value is set to 1 if the proximity cluster is *inside* a geometry on the layer specified by array_layer.
OutSRAM	Used only when array_layer is defined, and set to 0 otherwise. When array_layer is defined, this value is set to 1 if the proximity cluster is *outside* the geometries on the layer specified by array_layer.

Table 3-5. PatternGroup.csv File (cont.)

Column heading	Definition
Coordinates	The x- and y-coordinates of the four corners of a 2 dbu square marker at the center of the proximity cluster. The center location is affected by real_cluster_coordinate.

location.oas

- Generated if output_intermediates=1 and geometry_comparision=1
- Format: OASIS database
- Layer 0 has a 2 dbu square marker at the center location of each proximity cluster. The other layers have 2dbu square markers at the location of each defect.

cap.rdb

- Generated if output_intermediates=1 and feature_extraction=1
- Format: ASCII format results database (RDB)
- Has one result for each input defect; the result geometry is a 2 dbu square. The property values are calculated in a square region centered around the defect, with the region size defined by measure_range.

Table 3-6. cap.rdb File

Property	Definition
cluster_ID	The ID of the proximity cluster that this defect belongs to.
Density	The density of the target layer in the region of the defect.
min_c	The minimum corner to corner spacing for target layer shapes in region of the defect.
min_s	The minimum spacing between target layer shapes in region of the defect.
min_w	The minimum width of target layer shapes in region of the defect.
Vertex	The number of vertices on the target layer in region of the defect.
Coordinates	The x- and y-coordinates of the four corners of the result marker.

pattern_group.rdb

- Generated if output_intermediates=1 and geometry_comparision=1
- Format: ASCII format results database (RDB)
- Has one result for each proximity cluster; the result geometry is a 2 dbu square.

Table 3-7. pattern_group.rdb

Properties	Definition
ClassID	The ID of the geometry class that the proximity cluster belongs to.
cluster_id	The ID of the proximity cluster.
Density	The density of the target layer within the proximity cluster region.
Dup_count	The number of proximity clusters that belong to this geometry class (have the same geometry).
In_Array	Used only when array_layer is defined, and set to 0 otherwise. When array_layer is defined, this value is set to 1 if the proximity cluster is *inside* a geometry on the layer specified by array_layer.
Out_Array	Used only when array_layer is defined, and set to 0 otherwise. When array_layer is defined, this value is set to 1 if the proximity cluster is *outside* a geometry on the layer specified by array_layer.
Vertex	The number of vertices on the target layer within the proximity cluster region.
Coordinates	The x- and y-coordinates of the four corners of the result marker.

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.