SIEMENS EDA

Calibre® MDPAutoClassify™ User's Manual

Software Version 2021.2

SIEMENS

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction to Calibre MDPAutoClassify

Calibre® MDPAutoClassify™ is used for automatic classification of defects observed on a blank substrate before any patterning is performed on the substrate.

The following are the benefits of using defect classification:

- Knowledge of defect types and their properties provide a qualitative and quantitative estimate of the usability of the mask. The presence of large or serious defects can lead to the blank mask being discarded entirely.
- During the patterning process, knowledge of defect properties aids in optimal matching of a defective but usable blank substrate with patterns to be written on them.
- The defect types provide the means to monitor various processes involved in the mask preparation stages. Repeated occurrences of a specific type of defect may hint at a process-related problem which can then be rectified.

In the past, defect classification was typically a manual process, but due to newer technology nodes and the consequent increased subtlety of defects, consistency and effort problems crept into this process. This was the basis for automating the defect classification step.

Calibre MDPAutoClassify takes in multiple blank inspection reports performed at various stages of mask preparation and extracts various defect properties to identify the nature of the defect. As a result, the tool assigns a defect classification code and other defect properties such as defect area, and defect horizontal and vertical sizes are also calculated and reported.

Calibre MDPAutoClassify Prerequisites

There are several prerequisites prior to invoking Calibre MDPAutoClassify.

- **Platform support** — Calibre MDPAutoClassify 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 Calibre MDPAutoClassify, you must have a license for Calibre® MDPAutoClassify and Calibre® DefectReview. For more information on licensing, refer to the *Calibre Administrator's Guide.*

- **Required files** — Blank inspection files must be loaded in Calibre® DefectReview™. Calibre MDPAutoClassify is to be run on these blank inspection files. Refer to the *Calibre DefectReview User's Manual* for detailed instructions on loading defect and inspection files).

For further information on all products related to Calibre MDPAutoClassify, refer to the following table.

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*

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

Related Products	Documentation
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*
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 MDPAutoClassify Modes of Operation

Calibre MDPAutoClassify can be invoked using two different modes, GUI mode and through the command line interface.

Using Calibre MDPAutoClassify GUI Mode

You can invoke Calibre MDPAutoClassify directly from Calibre DefectReview.

Prerequisites

- Calibre DefectReview must be installed and invoked.
- A blank inspection file must be loaded into Calibre DefectReview.

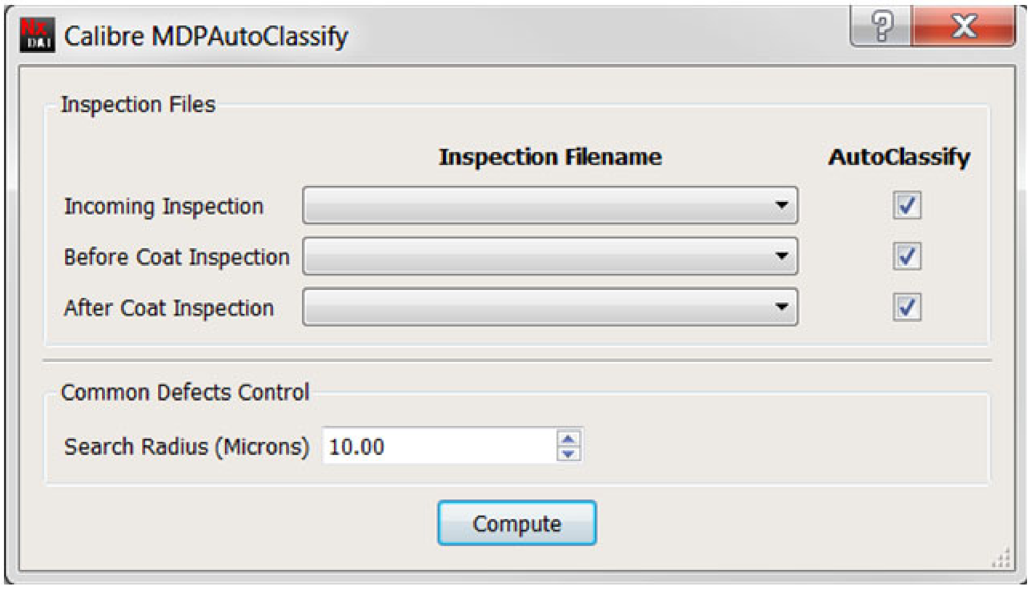
Procedure

In Calibre DefectReview, click **Utilities** > **MDPAutoClassify**, or click the Calibre MDPAutoClassify icon in the toolbar at the top of the Calibre DefectReview window.

Results

The Calibre MDPAutoClassify dialog box appears.

Figure 1-1. Calibre MDPAutoClassify Dialog Box


Calibre MDPAutoClassify

Inspection Files

	Inspection Filename	AutoClassify
Incoming Inspection		✓
Before Coat Inspection		✓
After Coat Inspection		✓

Common Defects Control

Search Radius (Microns) 10.00

Calibre MDPAutoClassify Command Line Mode

To run Calibre MDPAutoClassify using a command line interface, you must first prepare a TCL file, then use the Calibre MDPAutoClassify options on the Calibre DefectReview command line.

blankautoclassify Tcl File

Calibre MDPAutoClassify Tcl file

Before running Calibre MDPAutoClassify from the command line, you must first prepare a Tcl file using the blankautoclassify command.

Format

blankautoclassify *incoming_insp_id autoclassify_incoming_insp beforecoat_insp_id autoclassify_beforecoat_insp aftercoat_insp_id autoclassify_aftercoat_insp common_defects_radius*

Parameters

- ***incoming_insp_id*, *beforecoat_insp_id*, *aftercoat_insp_id***

  A required set of arguments that specifies the IDs for the following types of inspections:

  - ***incoming_insp_id*** (incoming inspection) — Refers to the inspection performed on the blank as it is received from the blank mask vendor.
  - ***beforecoat_insp_id*** (before coat inspection) — Also called "after clean inspection", refers to the inspection performed after the cleaning stage.
  - ***aftercoat_insp_id*** (after coat inspection) — Refers to the inspection following the application of resist coat on the blank mask.

  In the following example:

  ```
  blankautoclassify 1 1 2 1 3 1 10
  ```

  the ***incoming_insp_id*** is set to 1, ***beforecoat_insp_id*** is set to 2, and ***aftercoat_insp_id*** is set to 3. All three arguments should correspond to valid inspections (.ldf files), loaded into Calibre DefectReview using the following command line:

  ```
  NxDAT.exe -c <tcl_file> -i <ldf_file_1> <ldf_file_2> <ldf_file_3>
  (Windows)
  nxdat -c <tcl_file> -i <ldf_file_1> <ldf_file_2> <ldf_file_3>
  (Linux)
  ```

  Refer to "Invoking Calibre MDPAutoClassify from the Command Line" on page 16 for more information on the Calibre DefectReview invocation. If a specific inspection is to remain empty, specify 0 as the ID. The following are two example scenarios:

  - If only the before coat and after coat inspections are available and these two inspections are already loaded as inspection number 1 and 2, respectively, then the command is:

    ```
    blankautoclassify 0 1 1 1 2 1 10
    ```

- If only the incoming and after coat inspections are available and these two inspections are already loaded as inspection number 2 and 4, respectively, then the command is:

  ```
  blankautoclassify 2 1 0 1 4 1 10
  ```

- ***autoclassify_incoming_insp***, ***autoclassify_beforecoat_insp***, ***autoclassify_aftercoat_insp***

  A required argument that specifies the option to activate or deactivate automatic defect classification of respective inspections.

  - 1 — Activates automatic defect classification on the specified inspection.
  - 0 — Deactivates automatic defect classification for the specified inspection.

  The value of ***autoclassify_incoming_insp*** determines if automatic defect classification is required for incoming inspection (specified by incoming_insp_id). Similarly, ***autoclassify_beforecoat_insp*** determines if automatic defect classification is required for before a coat inspection (specified by ***beforecoat_insp_id***), and ***autoclassify_aftercoat_insp*** decides if automatic defect classification is required for after coat inspection (specified by ***aftercoat_insp_id***)

  When one or more of the inspection IDs are set to 0, the value of the corresponding ***autoclassify_*_insp*** options are ignored and internally set to 0. For example:

  ```
  blankautoclassify 1 1 0 1 2 1 10
  ```

  Here, ***beforecoat_insp_id*** is not specified (the ID is set to 0). This means that the corresponding ***autoclassify_beforecoat_insp*** value of 1 is ignored and internally set to 0.

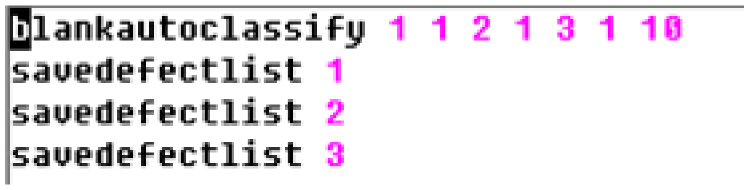
- ***common_defects_radius***

  A required argument that specifies the search radius (measured in microns) to be used for adder analysis on specified inspections.

Examples

The following is an example of the input Tcl file.

Figure 1-2. Example Tcl File



```
blankautoclassify 1 1 2 1 3 1 10
savedefectlist 1
savedefectlist 2
savedefectlist 3
```

Invoking Calibre MDPAutoClassify from the Command Line

Once the Tcl input file is prepared, you can invoke Calibre MDPAutoClassify from the Calibre DefectReview command line.

Prerequisites

- An input Tcl file must be prepared. See "blankautoclassify Tcl File" on page 14 for complete information.

Procedure

At a command line prompt, use the following invocation syntax:

NxDAT.exe -c *tcl_file* -i *ldf_file_1 ldf_file_2 ldf_file_3* (Windows)

$NXDAT_MGC_HOME/bin/nxdat -c *tcl_file* -i *ldf_file_1 ldf_file_2 ldf_file_3* (Linux®[1])

where:

- ***tcl_file*** refers to the name of the input Tcl file.
- ***ldf_file_1***, ***ldf_file_2***, and ***ldf_file_3*** correspond to blank inspections to be loaded into Calibre DefectReview.
- **$NXDAT_MGC_HOME** points to the directory where Calibre DefectReview is installed.

Inspection IDs used by the Tcl files are assigned in the order of .ldf files specified in the command line. For example, the inspection ID corresponding to ***ldf_file_1*** is 1, ***ldf_file_2*** is 2, and ***ldf_file_3*** is 3.

The number of .ldf files to be loaded may vary for each case. For example, if only the cleaned and coated inspections are available, then the two corresponding .ldf files can be loaded as ***ldf_file_1*** and ***ldf_file_2***, skipping ***ldf_file_3***.

1. Linux® is a registered trademark of Linus Torvalds in the U.S. and other countries.

Chapter 2
Using Calibre MDPAutoClassify

Calibre MDPAutoClassify enables you to automate classification of defects observed on a blank substrate before patterning is performed.

To run Calibre MDPAutoClassify, you must enter values for inspection files and common defects in the Calibre MDPAutoClassify dialog box.

Setting Up Inspection Files

A blank mask passes through the cleaning and coating mask preparation stages. For continuous monitoring of the state of the mask blank, inspections are performed corresponding to each of the preparation stages. On the whole, three inspections are performed on a mask blank.

The types of inspections are as follows:

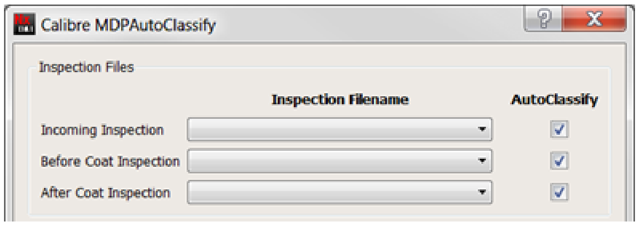
- **Incoming inspection** — The inspection performed on the blank as it is received from the blank mask vendor.
- **Before Coat inspection** — The inspection performed after the cleaning stage. This is also called “After Clean inspection”.
- **After Coat inspection** — The inspection following the application of a resist coat on the blank mask.

Calibre MDPAutoClassify has been designed to run together or selectively on the three inspections and interpret the defect classification accordingly. Further information can be found in “Defect Source Type Classification” on page 36.

Procedure

1. After invoking Calibre MDPAutoClassify, in the Inspection Files section, for each of the mask preparation steps (Incoming, Before Coat, and After Coat), select a corresponding Inspection Filename and click the MDPAutoClassify check box to enable auto-classification.

Figure 2-1. Inspection Files



The inspection file name is selected from a drop-down menu of all the loaded inspections in Calibre DefectReview. However, if the inspection data corresponding to a specific mask preparation stage is not available, the Inspection Filename field can be left blank. For example, if, for a blank mask, only the cleaned and coated inspections are available, then the field corresponding to Incoming inspection can be left blank. By default, inspection numbers 1, 2 and 3 (the first, second, and third inspections in the order of their loading) are respectively considered as Incoming, Before Coat, and After Coat inspections.

The AutoClassify check box provides an option for to skip automatic defect classification for the unchecked inspection. An example scenario is when an Incoming inspection is to be used as a reference for defect classification of a Before Coat inspection, but it does not need go through classification again since that step has previously been completed. Only the results from an earlier run are required; the defect classification process is redundant. If no inspection filename has been specified for any of the inspections, the corresponding check box status is ignored

2. Proceed to the steps detailed in "Setting Up Common Defects Controls" on page 18. However, if the requisite inspection has not been loaded:

   a. Close the Calibre MDPAutoClassify dialog box.

   b. Load the required inspections through the Add Inspection(s) option in Calibre DefectReview. Detailed instructions regarding this option can be found in the *Calibre DefectReview User's Manual*.

   c. Re-launch the Calibre MDPAutoClassify dialog box.

Setting Up Common Defects Controls

After setting up the inspection file parameters, you must set up the controls for common defects.

Prerequisites

- You have performed the steps detailed in "Setting Up Inspection Files" on page 17.

Procedure

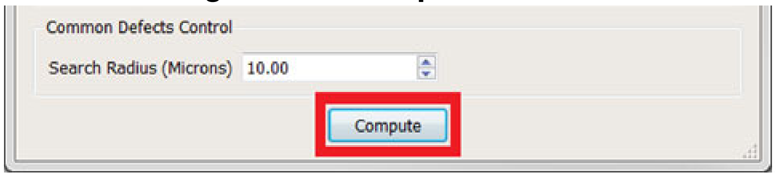
1. In the Common Defects Control section of the Calibre MDPAutoClassify dialog box, enter a value for the Search Radius (Microns) field.

Search Radius (Microns) is a parameter whose value determines the radius to be used to determine which defects are common across inspections based on their location on the mask. A sample value of 3 um means that if defects from different inspections have their locations within 3 um of each other, then they are essentially the same defect. More on repeatability analysis can be found in the *Calibre DefectReview User's Manual*. A default value of 10 microns is used by Calibre MDPAutoClassify.

The importance of finding out whether a defect is common or an adder is explained in the "Defect Source Type Classification" on page 36.

2. Click the **Compute** button to initiate a Calibre MDPAutoClassify run on specified inspections.

Figure 2-2. Compute Button



Search Radius (Microns) 10.00

A progress bar dialog appears to indicate the progress until the run completes.

Results

Once **Compute** in the Calibre MDPAutoClassify dialog box is pressed, a progress bar appears for each of the specified inspections. Upon completion of execution on all the inspections, the defect list is updated with defect classification and other information as detailed in "Calibre MDPAutoClassify Results" on page 24.

Calibre MDPAutoClassify Configuration File

Configuration file for Calibre MDPAutoClassify

In addition to Calibre MDPAutoClassify dialog box parameters, Calibre MDPAutoClassify's performance can be further tuned by altering various parameters listed in the configuration file, *adc-ini.xml*. On a Windows environment, this file is in the folder *C:\DefectAnalysisTool\bin*; for a Linux installation, the path is *$HOME/.calibrenx_workspace/adc-ini.xml*, where $HOME is the user's home directory.

Format

An XML format file, similar to the Calibre DefectReview *dat-ini.xml* file.

Parameters

- blankReflImgName

  Specifies the name of the folder for reflected defect images. The default value is "Image". The value of this parameter should currently match the value of the node ldfReflImgName in the *dat-ini.xml* file.

- blankTransImgName

  Specifies the name of the folder for transmitted defect images. The default value is "Transmission". The value of this parameter should currently match the value of the node ldfTransImgName in the *dat-ini.xml* file.

- blankCropTopLeftColumn

  Specifies the X-coordinate (column) of the top left corner of the defect image where signals are to be identified. (Indices start from 0).

- blankCropTopLeftRow

  Specifies the Y-coordinate (row) of the top left corner of the defect image where signals are to be identified. (Indices start from 0).

- blankCropWidth

  Specifies the width of the area (in image pixels) within the defect image inside where signals are to be identified. The default value is 320.

- blankCropHeight

  Specifies the height of the area (in image pixels) within the defect image inside where signals are to be identified. The default value is 240.

  Note

  The default cropped area is the center quarter of the defect image, assuming the size of the defect image to be 640 x 480 image pixels

- blankImageCornerSize

  Specifies the width of corner square areas (in image pixels) used for calculating statistics of defect image background. The default value is 100 (four squares of size 100 x 100 pixels are considered to obtain defect image background characteristics).

- blankIgnoreTopBottomRows

  Specifies the number of rows ignored on top and at the bottom of defect image. This is to avoid processing the part of image with overlaid text. The default value is 50.

- blankDiffThresholdRefl

  Specifies the minimum absolute strength for a strong valid defect signal in a reflected defect image. Strength is measured as the difference of pixel intensity and average image intensity. The default value is 100.

- blankRatioThresholdRefl2

  Specifies the minimum relative strength for a weak valid defect signal in a reflected defect image. This is used in conjunction with the parameter blankDiffThresholdRefl2. The default value is 2.5.

- blankDiffThresholdRefl2

  Specifies the minimum absolute strength for a weak valid defect signal in a reflected defect image. This is used in conjunction with the parameter blankRatioThresholdRefl2. Default value is 50.

- blankDiffThresholdTrns

  Specifies the minimum absolute strength for a strong valid defect signal in a transmitted defect image. The default value is 100.

- blankRatioThresholdTrns2

  Specifies the minimum relative strength for a weak valid defect signal in a transmitted defect image. Used in conjunction with the parameter blankDiffThresholdTrns2. Default value is 2.5.

- blankDiffThresholdTrns2

  Specifies the minimum absolute strength for a weak valid defect signal in a transmitted defect image. Used in conjunction with the parameter blankRatioThresholdTrns2. Default value is 50.

- blankDefectSizeTuning

  A defect size correction parameter that limits the maximum size of the defect. It defines the pixel intensities which are to be considered a part of the defect contour. Valid values range from 0 to 1. A value of 0 implies the defect contour includes pixels with intensities ranging from the darkest pixel inside the contour to average image intensity; 1 implies the defect contour only includes the pixels with intensities equal to that of the darkest pixel. For example, a value of 0.4 implies that darkest (brightest) pixels with intensities within 40% of the pixel intensity range constitute the defect contour. The default value is 0.4.

- blankNegligibleAreaSignal

  If the defect area (in image pixels) is below this parameter's value, then the detected signal is considered as nuisance or false alarm. The default value is 70.

- blankSecSignalsStrengthPercent

  Specifies the minimum expected strength (in percentage) of the secondary signals relative to the strongest signal. The default value is 80.

- blankAreaThreshold

  If a defect has an area (in image pixels) above this parameter’s value, then it is classified as a Large defect. The default value is 500.

- blankEBRSAreaThreshold

  If a defect’s pixel size, as measured by the inspection machine is above this parameter's value, then the defect is classified as an EBR (Edge Bead Removal) Splash case. The default value is 500.

- blankFuzzyGradientLowerThreshold

  Specifies the minimum change in pixel intensity across the defect signal's boundary. The default value is 70.

- blankFuzzyGradientHigherThreshold

  Specifies the maximum change in pixel intensity across the defect signal's boundary. The default value is 140.

- blankFuzzyGradientCount

  Specifies the minimum number of image pixels with acceptable change in pixel intensity, defined by previous two parameters, blankFuzzyGradientLowerThreshold and blankFuzzyGradientHigherThreshold. The default value is 80.

- blankFuzzyArea

  Specifies the minimum area (in image pixels) of a fuzzy spot defect. The default value is 240.

- blankLiquidDropMinRadius

  Specifies the minimum radius (in image pixels) of a liquid drop defect. The default value is 20.

- blankLiquidDropMaxRadius

  Specifies the maximum radius (in image pixels) of a liquid drop defect. The default value is 140.

- blankMaximumAreaFalse

  If a defect's pixel size, as measured by the inspection machine, is above this parameter's value, then the defect cannot be classified as a false defect. The default value is 20.

- blankDefectSizeClusterRadius

  If the distance (in image pixels) between two defect signals simultaneously identified in an image is less than this parameter's value, then they are considered as originating from a single defect. The size computation is done by combining the two signals. The default value is 100.

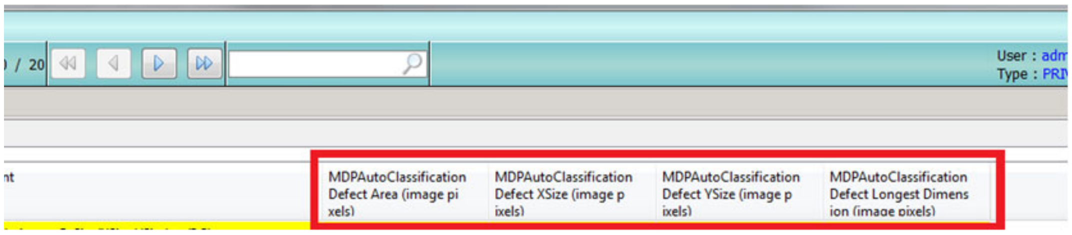
Calibre MDPAutoClassify Results

After running the Calibre MDPAutoClassify tool, a number of results are output for analysis.

Defect List Updates

Several columns in the Defect List window of Calibre DefectReview are updated upon completion of a Calibre MDPAutoClassify run.

Figure 2-3. Updated Defect List Columns


/ 20
User : adm
Type : PRI
nt
MDPAutoClassification Defect Area (image pixels)
MDPAutoClassification Defect XSize (image pixels)
MDPAutoClassification Defect YSize (image pixels)
MDPAutoClassification Defect Longest Dimension (image pixels)

The following table lists all of the Defect List columns that are populated by an MDPAutoClassify run.

Table 2-1. Updated Defect List Columns

Column	Description
MDPAutoClassification	This is the disposition code assigned to a defect. This code is obtained from a user-defined combination of classification codes from other tiers (for example, Defect Type and Source Type). The section "Defect Disposition" on page 27 describes how to define the combinations.
MDPAutoClassification Comment	If the classification code is obtained from a custom combination of classification codes from other tiers, then this column displays the combination used to obtain this tier's classification code.
MDPAutoClassification Defect Type	This is the classification code assigned to each of the blank defects. The section "Calibre MDPAutoClassify Supported Defect Types" on page 29 provides details the classification codes used by the tool.
MDPAutoClassification Defect Type Comment	This consists of additional information associated with the blank defect, extracted by Calibre MDPAutoClassify. The section "Defect Classification Comments" on page 35 provides details on the content this column.

Table 2-1. Updated Defect List Columns (cont.)

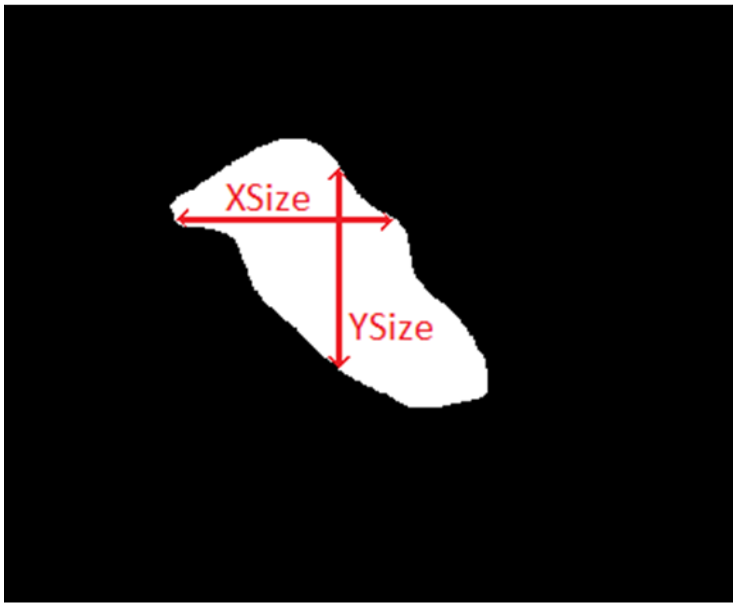
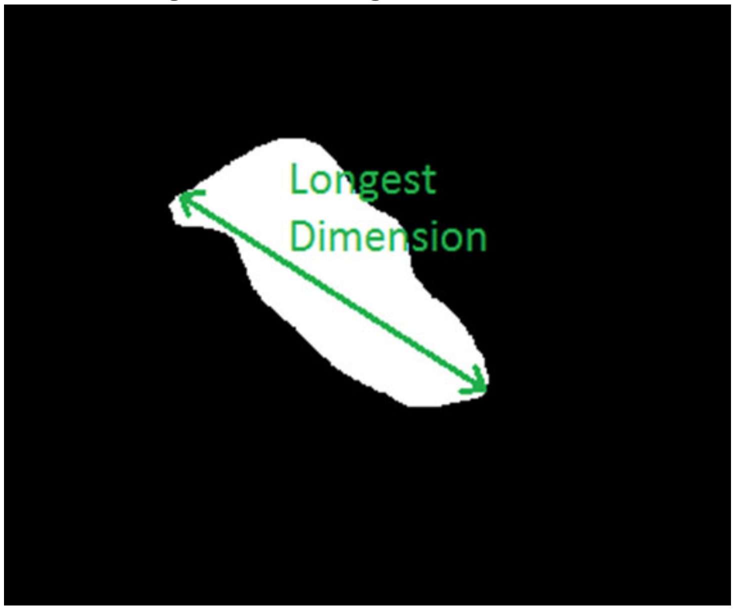
Column	Description
MDPAutoClassification Source Type	This is the defect source type classification code assigned to each of the blank defects. The section "Defect Source Type Classification" on page 36 describes the supported source types.
MDPAutoClassification Source Type Comment	For each defect, this column consists of additional information (primarily the defect ID) from previous inspections or occurrences, if any.
MDPAutoClassification Defect Area (image pixels)	This is the total area of the defect, as determined by Calibre MDPAutoClassify. It is measured in terms of image pixels over which the defect is spread.
MDPAutoClassification Defect XSize (image pixels)	This is the widest part of the extracted defect contour across all the horizontal cross-sections of the blank defect. It is measured in terms of image pixels. An example of Defect XSize measurement for an extracted defect contour is shown in the following figure. **Figure 2-4. Defect X-Size and Defect Y-Size** 
MDPAutoClassification Defect YSize (image pixels)	This is the longest part of the extracted defect contour across all the vertical cross-sections of the blank defect. It is measured in terms of image pixels. Refer to Figure 2-4 for an example of Defect YSize measurement.

Table 2-1. Updated Defect List Columns (cont.)

Column	Description
MDPAutoClassification Defect Longest Dimension (image pixels)	This is the longest part of the extracted defect contour across all non-orthogonal cross-sections of the blank defect. It is measured in terms of image pixels. Refer to the following figure for an example of Longest Dimension measurement. Figure 2-5. Longest Dimension 

Defect Disposition

Classification rule mapping in the *dat-ini.xml* file

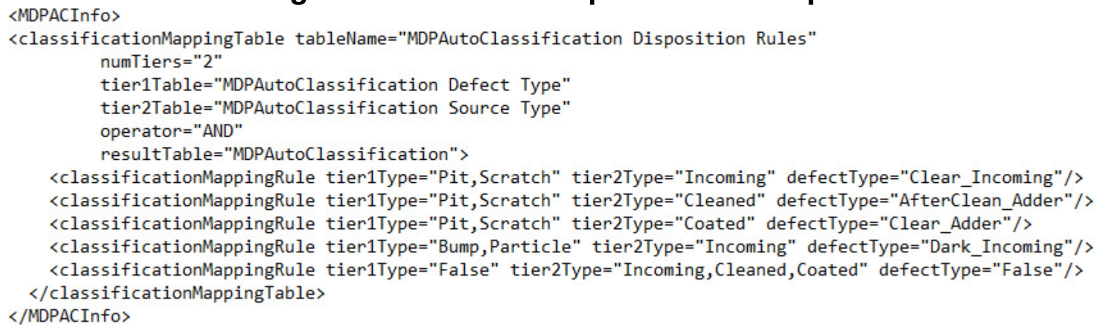
Calibre MDPAutoClassify enables you to define defect dispositions, customized combinations of classification codes from other tiers.

This disposition, classified under the column MDPAutoClassification, can be defined as combinations of classifications from Defect Type and Source Type. For example, a Pit defect observed first in a Cleaned inspection can be assigned a disposition of AfterClean_Adder. Calibre MDPAutoClassify can identify the defect type and source stage, but the disposition is defined using specific nodes in the *dat-ini.xml* file.

Format

Uses the same XML-based format as the *dat-ini.xml* file.

Figure 2-6. Defect Disposition Example



```
<MDPACInfo>
<classificationMappingTable tableName="MDPAutoClassification Disposition Rules"
         numTiers="2"
         tier1Table="MDPAutoClassification Defect Type"
         tier2Table="MDPAutoClassification Source Type"
         operator="AND"
         resultTable="MDPAutoClassification">
    <classificationMappingRule tier1Type="Pit,Scratch" tier2Type="Incoming" defectType="Clear_Incoming"/>
    <classificationMappingRule tier1Type="Pit,Scratch" tier2Type="Cleaned" defectType="AfterClean_Adder"/>
    <classificationMappingRule tier1Type="Pit,Scratch" tier2Type="Coated" defectType="Clear_Adder"/>
    <classificationMappingRule tier1Type="Bump,Particle" tier2Type="Incoming" defectType="Dark_Incoming"/>
    <classificationMappingRule tier1Type="False" tier2Type="Incoming,Cleaned,Coated" defectType="False"/>
  </classificationMappingTable>
</MDPACInfo>
```

Parameters

ClassificationMappingTable Parameters

To define defect disposition, the *dat-ini.xml* uses the ClassificationMappingTable node containing several user-defined parameters.

- tableName

  An optional parameter that specifies the name of the table that defines the disposition mapping.

- numTiers

  An optional parameter that specifies the number of tiers whose classification categories are combined to get the classification categories in the resulting tier. In the example shown in Figure 2-6, the classification categories from two tiers, MDPAutoClassification Defect Type and MDPAutoClassification Source Type are combined.

- tierXTable

  An optional parameter that specifies the secondary classification table name (defined separately in the *dat-ini.xml* file), from which classification categories are selected to define combinations specified using the classificationMappingRule parameters. The number of these attributes must be the same as the value specified in numTiers. Any additional attributes exceeding numTiers are ignored.

- operator

  An optional parameter that specifies the logical combination between tier categories. Currently, only AND is supported.

- resultTable

  An optional parameter that specifies the secondary classification table name (defined separately in *dat-ini.xml*), from which classification categories are selected as the output of combinations specified under classificationMappingRule parameters.

ClassificationMappingRule Parameters

The ClassificationMappingRule parameters define the combinations between input classification categories from secondary classification tables specified by tierXTable, combined by a logical operator specified by the operation parameter, resulting in a classification category from the secondary classification table specified by resultTable.

- tierXType

  An optional parameter that contains input classification categories from their respective secondary classification tables. For example, as illustrated in Figure 2-6, the attribute tier1Type contains classification categories from the secondary classification table specified under the attribute tier1Table, and so on.

- defectType

  An optional parameter that contains the resulting classification categories output from the secondary classification table as specified by resultTable.

Calibre MDPAutoClassify and Defect Classification

The column “MDPAutoClassification Defect Type” contains the classification code for each of the blank defects as determined by Calibre MDPAutoClassify.

Calibre MDPAutoClassify Supported Defect Types

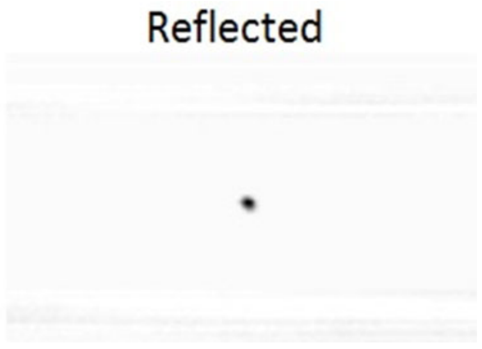
The following are the defect types into which blank defects are classified by Calibre MDPAutoClassify by applying certain rules on defect properties corresponding to nature of the defect.

The following are the supported defect types.

- Dark

  This type refers to dark signals observed on reflected mask blank defect image. There is no accompanying transmitted image.

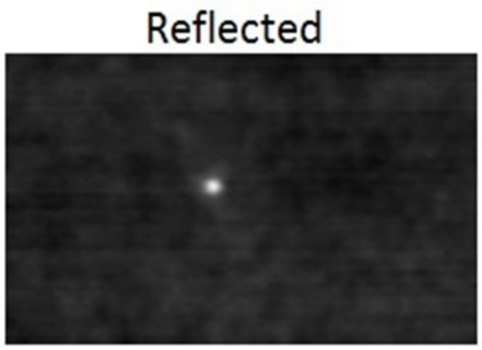
Figure 2-7. Dark Defect



- Clear

  This type refers to clear signals observed on reflected mask blank defect image. There is no accompanying transmitted image.

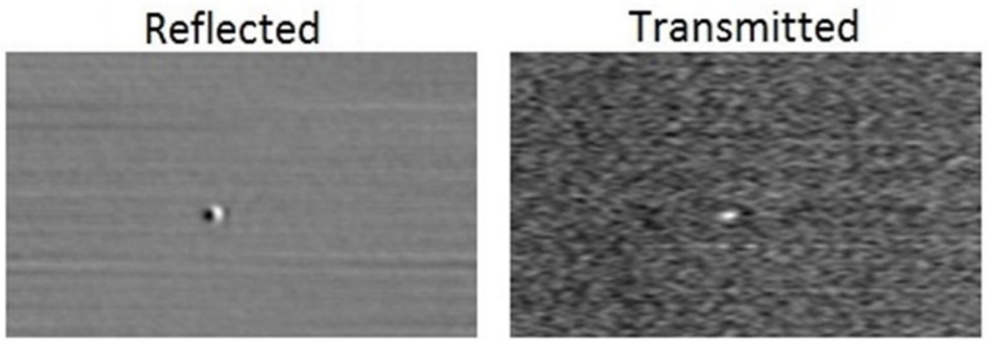
Figure 2-8. Clear Defect



- Pit

  This type refers to small pits observed on the mask blanks.

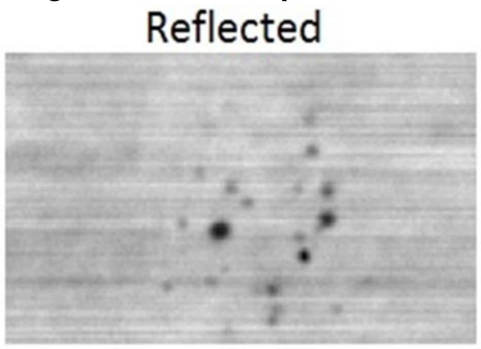
Figure 2-9. Pit or Pinhole Defect



- Multiple Signals

  This type is reported when multiple defect signals are identified in the reflected mask blank defect image.

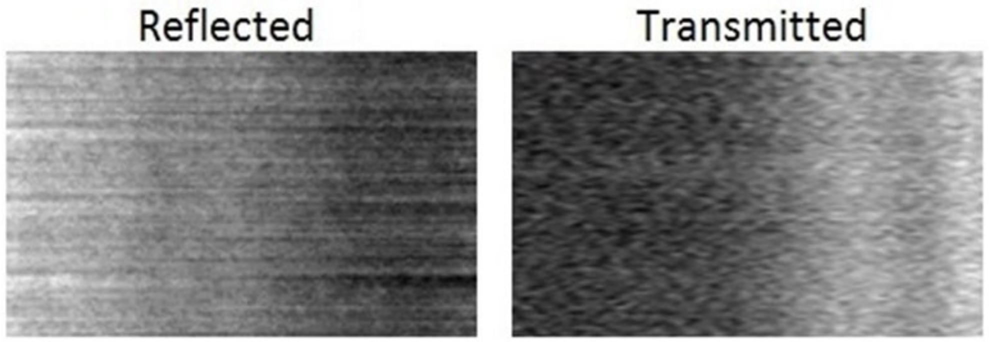
Figure 2-10. Multiple Defects



- EBR Splashes

  This type refers to back splashes of cleaning liquid from the resist EBR process. They are much larger than the scope of the defect image captured by the inspection machine.

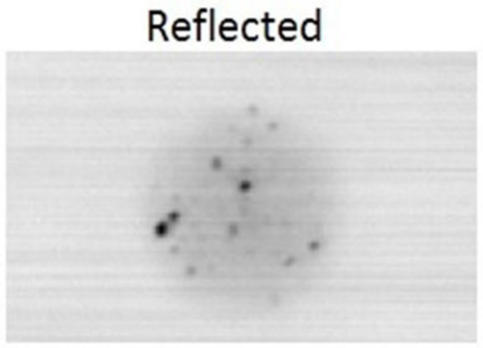
Figure 2-11. EBR Splash Defect



- Liquid Drop

  This type refers to dark signals seen in reflected defect image. These signals have circular shapes, fuzzy boundaries and are usually at least 40 pixels in diameter.

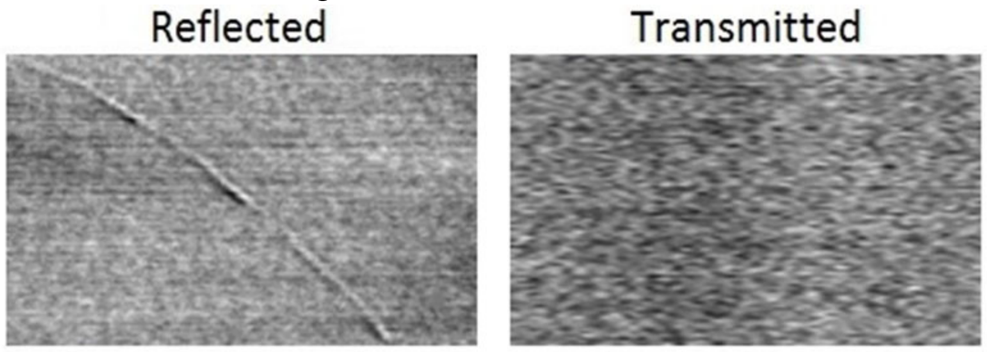
Figure 2-12. Liquid Drop Defect



- Scratches

  This type refers to long marks running across a defect image. Their considerable size makes them serious, usually rendering the mask blank unusable.

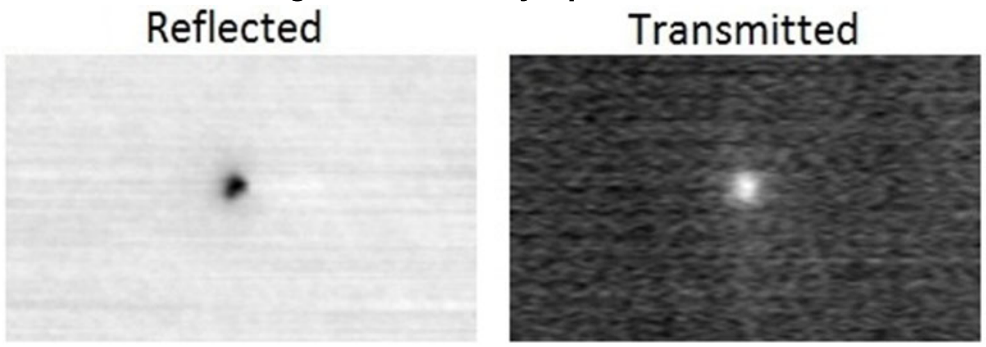
Figure 2-13. Scratch Defect



- Fuzzy Spots

This type refers to defects whose appearance is similar to particles, but can be distinguished by their fuzzy nature (for instance, blurred edges). These defects are frequently small in size, but can cause large defects on the final pattern.

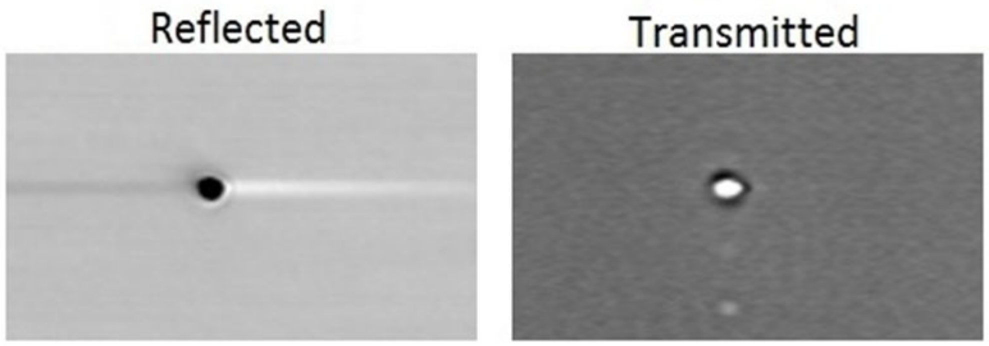
Figure 2-14. Fuzzy Spot Defect



- Resist Voids

  This type refers to air bubbles trapped in the resist layer during the coating process. If identified, the resist layer can be re-worked to get rid of resist void defects.

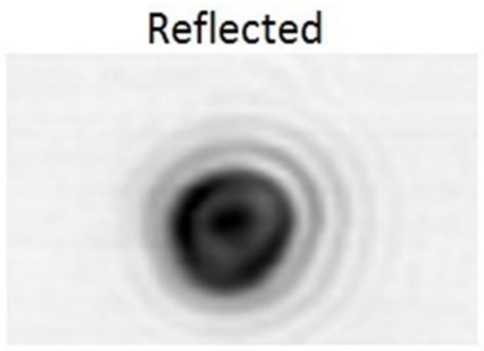
Figure 2-15. Resist Void Defect



- Large Defect

  This type refers to dark or clear defects having a certain minimum area.

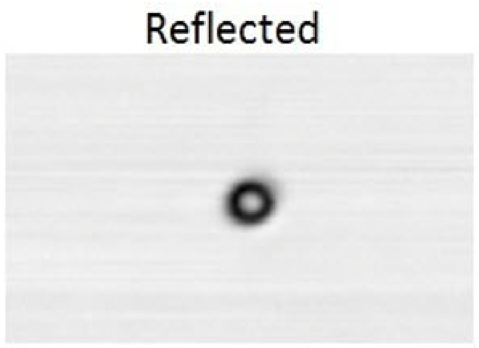
Figure 2-16. Large Defect



- Donut

This type refers to medium-sized defects with the appearance of a donut, as seen in the following figure.

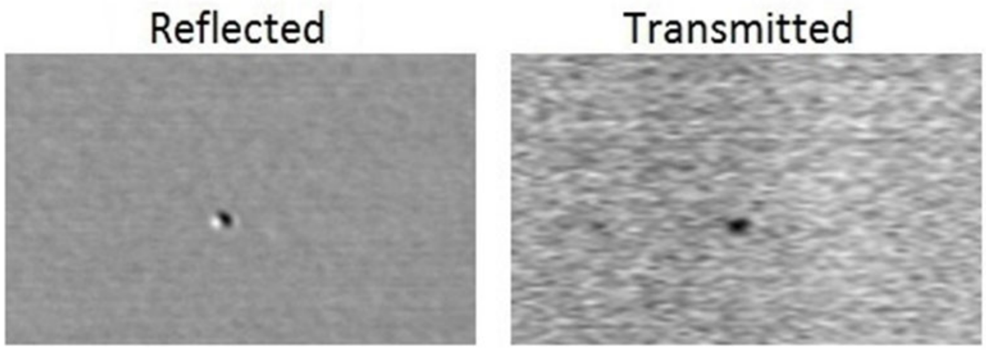
Figure 2-17. Donut Defect



- Bump

  This type refers to small bumps present on the surface of a mask blank. They appear as small dark signals in the reflected image, with or without an accompanying dark signal in the transmitted image.

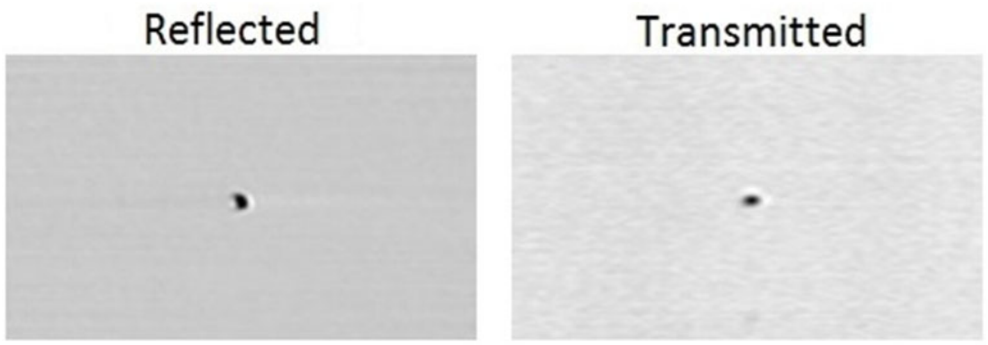
Figure 2-18. Bump Defect



- Particles

  This refers to particles observed on the surface of mask blanks. If these are contamination, they can be repaired by re-doing the cleaning step.

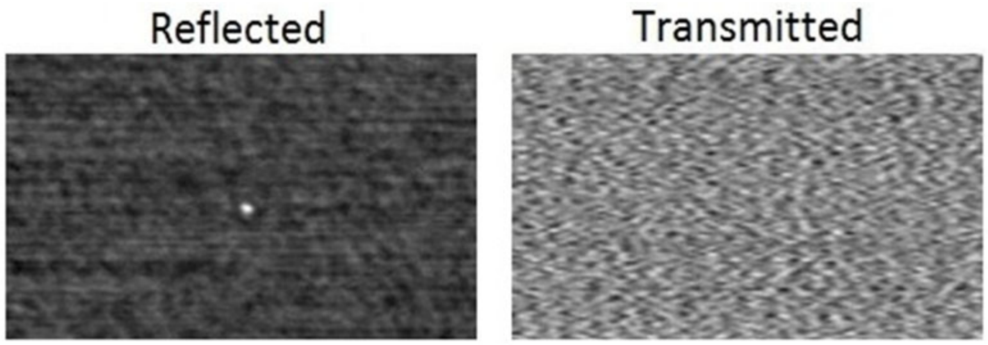
Figure 2-19. Particle Defect



- Missing ARC

  This type refers to the case where bright signals are observed in reflected mask blank defect image due to the absence of Anti-Reflective Coating (ARC) on the mask blank. They can only be seen in Incoming or Cleaned inspections, never in Coated inspections.

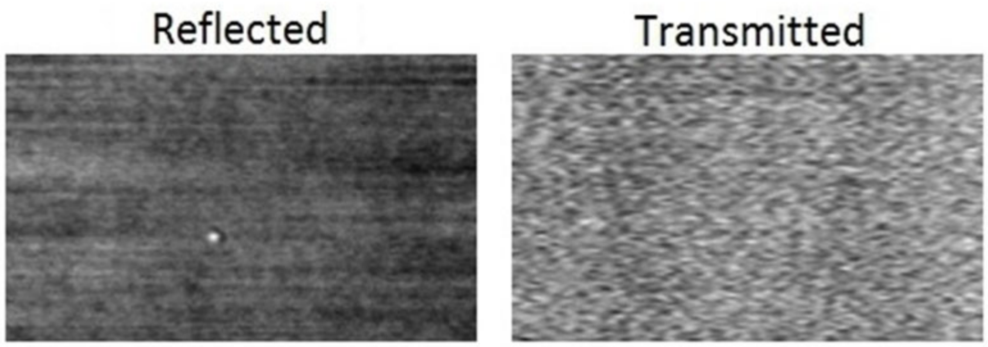
Figure 2-20. Missing ARC Defect



- Bright Adders

  This type refers to adder defects seen in coated inspections. They look like non-concerning defects observed when the Anti-Reflective Coat (ARC) is missing. They are identified based on the process step at which they are observed.

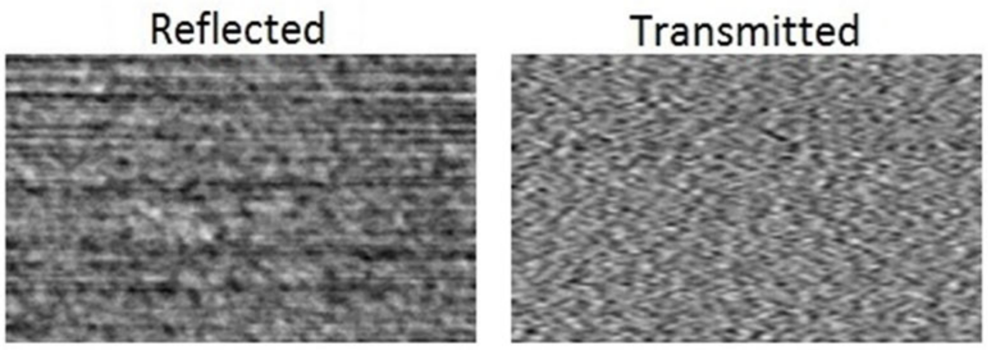
Figure 2-21. Bright Adder Defect



- False

  This type refers to false defects where the machine assumes the presence of a defect but in reality there is none.

Figure 2-22. False Defect



- Engineering Review

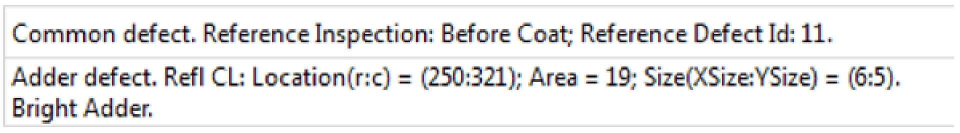
  These are blank defects that could not be classified by Calibre MDPAutoClassify, and hence are to be manually reviewed by the operator. The tool reports the statistics gathered and the analysis results to aid manual review.

Defect Classification Comments

Upon processing each blank defect, Calibre MDPAutoClassify populates certain details on the properties of the defect.

The following figure illustrates sample comments for blank defects.

Figure 2-23. Sample Calibre MDPAutoClassify Comments



Common defect. Reference Inspection: Before Coat; Reference Defect Id: 11.

Adder defect. Refl CL: Location(r:c) = (250:321); Area = 19; Size(XSize:YSize) = (6:5). Bright Adder.

The following is a list and description of the content of the MDPAutoClassification Comment column:

Table 2-2. Calibre MDPAutoClassify Comment Types

Comment Type	Description
Common or Adder status	Except for the first incoming inspection, specifies the repeatability status of all the defects reported for process monitoring purposes.
Defect location	Specifies accurate locations of each of the identified defect signals reported in the format (row index, column index).
Defect Area	Specifies the number of pixels spanned by the defect.
Defect size (in image pixels)	Specifies the largest pair of horizontal and vertical cross-sections of the defect. They are, at most, equal to the width and height of the boundary.

Table 2-2. Calibre MDPAutoClassify Comment Types (cont.)

Comment Type	Description
Defect type	Based on the defect properties, Calibre MDPAutoClassify attempts to identify the nature of the defect (a scratch, pinhole, and so on). For example, in Figure 2-23, the adder defect has been identified as a Bright Adder type by Calibre MDPAutoClassify. The section "Calibre MDPAutoClassify Supported Defect Types" on page 29 describes the various defect types that are identified by Calibre MDPAutoClassify.

Note

All the image pixel location values used and reported are with respect to a coordinate system where the top-left pixel of the image has the coordinates as (0,0), unless stated otherwise.

Defect Source Type Classification

Calibre MDPAutoClassify supports classification of a defect based on the source inspection type (the first occurrence of a defect). The tool uses repeatability analysis of blank inspections across multiple mask preparation stages to track a defect and provide insight into its possible origin. This information can be used to monitor process-related information.

For example, a large number of resist void defects are detected, indicating a potential problem with the resist coating process. Using repeatability analysis, it can be determined conclusively that these defects were introduced during the coating process and not in earlier stages.

The source inspection type classification is a secondary classification in addition to the defect type classification. The following are the supported defect source type classification categories:

- **Incoming —** Indicates when the defect initially occurs during the first inspection, as specified in the Calibre MDPAutoClassify dialog box.
- **Cleaned —** Indicates when the defect initially occurs during the second inspection, as specified in the Calibre MDPAutoClassify dialog box.
- **Coated —** Indicates when the defect is found only in the coated (or the third inspection), as specified in the Calibre MDPAutoClassify dialog box.

Based on the repeatability or adder analysis, Calibre MDPAutoClassify also classifies a blank defect as one of the following:

- **Blank or Common Defect —** These are blank defects that either exist on the incoming blank mask substrate or have been observed in a previous mask preparation step. In other words, the defects were transferred from one preparation step to another (such as from the cleaned stage to coated stage).

- **Adder Defect —** These are blank defects that are initially detected during the cleaned or coated inspections. They were not resident on the incoming blank substrate and have been introduced during one of the mask preparation steps.

The Calibre MDPAutoClassify tool has the additional capability of automatically determining the correct angle of rotation when the blank mask under inspection is rotated across different inspection stages.

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.