SIEMENS EDA

Calibre® DefectClassify™ User's 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 Calibre DefectClassify

Calibre® DefectClassify™ reduces of time spent in manual analysis and classification of defects as observed across a variety of inspection steps during the printing phase of a circuit.

Calibre DefectClassify consists of three major functional tools:

- **Mask Pattern Classification —** This tool facilitates classification and characterization of defects observed on patterned masks when inspected in different modes (for example, aerial, highres, or reflected) using different detection modes (such as Die-to-die and Single die).
- **Image-to-Layout Alignment —** This tool aligns patterns that appear on SEM images with corresponding patterns on the design layout to improve defect analysis. This tool is required when the location of a defect reported by an inspection machine does not point to a corresponding location on the layout. In addition to alignment, this tool calculates and reports this offset in location.
- **SEM Automatic Defect Classification (ADC) —** This tool automatically classifies defects using images captured by a SEM inspection machine. It also automatically characterizes the defect, such as measuring the defect size and the number of patterns affected.

Calibre DefectClassify Prerequisites

Before using Calibre DefectClassify, there are several requirements that must be met first.

Calibre DefectClassify is distributed as part of the Calibre software package and utilizes the same installation procedures.

The following are the prerequisites required for Calibre DefectClassify:

- **Platform Support** — The main tools of Calibre DefectClassify are supported on AOI Linux hosts. The SEM contour extraction tool is supported on the AOI and AOJ platforms.

  Refer to the *Calibre Administrator's Guide* for instructions on how to install Calibre software.

- **Licensing** — To run Calibre DefectClassify, you must have the license file for both Calibre DefectClassify and Calibre© DefectReview™.

  For more information on licensing, refer to the *Calibre Administrator's Guide*.

- **Memory** — To run Calibre DefectClassify, each core has a memory requirement of 2 GB.

- **Environment Variables** — In addition to the changes made when installing Calibre DefectReview, the following code must be added to the *.cshrc* file:

  ```
  setenv PATH ${MGC_HOME}/bin:${PATH}
  ```

  This assumes that MGC_HOME has already been set to the Calibre package. Similar changes can also be done in other command shells such as bash and so on.

- **Configuration Files and Workspace** — When Calibre DefectReview is launched, the Calibre DefectClassify configuration file for patterned masks, *adc-ini.xml*, is copied into a workspace if it doesn't already exist. However, if the file already exists, then it is not copied or overwritten. Whenever the configuration file is updated from release to release and if the workspace exists, you must copy the latest configuration file. Otherwise, a new workspace can be defined or the existing workspace can be cleared for the latest configuration file to be automatically copied.

  Note

  As import settings are not backed up for the Calibre DefectClassify configuration file, use caution when deleting or overwriting existing configuration files.

The following table lists the documents associated with the 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*

Licensing for Multiple Cores

Calibre DefectClassify supports parallel runs utilizing multiple cores of a machine.

There are two configuration files utilized for Calibre DefectClassify:

- The Mask Pattern Classification tool configuration file is called *adc-ini.xml*. For further information, refer to "Mask Pattern Classification Configuration File" on page 72 under "Common Parameters"
- The Image-to-Layout Alignment and SEM ADC tools use the same user-specified configuration file (with a *.cfg* suffix). For further information, refer to "Image-to-Layout Alignment Run Configuration File" on page 101.

In the Mask Pattern Classification tool, parallel runs can be enabled using the parallelRun parameter in the *adc-ini.xml* configuration file. You specify the number of cores to be utilized for the parallel run using the parameter noOfCoresForParallelRun in the *adc-ini.xml* configuration files.

In addition, the Image-to-Layout Alignment and SEM ADC tools support both Multithreaded (MT) and Calibre MTflex modes.

- MT mode — By default, the tools are enabled to work in multithreaded mode. In this case, you can specify the number of cores to utilize. Otherwise, the tool starts running with all available cores of a system.
- MTflex mode — You can specify the number of cores to be utilized for a multithreaded run. This can be specified in a remote host file (see "Specifying Run Configuration Settings for Image-to-Layout Alignment" on page 92 for details). If the remote host file is not specified, the tool runs in MT mode.

Specifying multiple cores requires availability of multiple licenses. The following table lists the maximum number of cores you can request based on the number of Calibre DefectClassify licenses you have.

Table 1-2. Licenses and Cores Requested

Licenses Available	Maximum Number of Cores
1	1
2	4
3	8
4	12
5	16
6	12

With only a single license, Calibre DefectClassify cannot run in parallel mode. Availability of *n* licenses enables the use of 4*(*n*-1) cores, where *n* is greater than 1. A single core requires a single license, otherwise *n* cores require the licenses based on the following calculation, where *n* greater than 1:

```
(closest integral multiple of 4 >= n)/4 + 1
```

Licenses are checked out when Calibre DefectClassify execution starts, depending on number of cores specified in the *adc-ini.xml* configuration file.

Note

Calibre DefectClassify always calculates and checks out the number of licenses based on the number of cores requested, as per the rules described previously. If this number is greater than the number of licenses available, then the tool quits. You must always specify the correct number of cores based on the number of licenses available.

Chapter 2
Mask Pattern Classification

Calibre DefectClassify enables you to automate classification of defects observed on a pattern mask.

Defect characterization and classification is primarily done to better understand the nature of the defect and its properties. This helps determine the right strategies and procedures used to repair or mitigate the effect of a defect. Some types of defects provide process or tool-related information as well. For example, the bright spot defect, typically seen across multiple defect images, has the same frame location which implies that the camera sensor has a defective pixel.

The variety of data (patterns) that are written on a mask, such as line-spaces, contacts, sub-resolution assist features (SRAFs), fill, and so on result in a variety of defects arising from the processes involved. Apart from the real defects, a variety of false defects are also identified by the inspection machines. In fact, false defects constitute the majority of defects identified by an inspection machine. Reliable separation of false defects from real ones provides the largest improvements in classification throughput. This is because, following defect classification, more time can be dedicated to analyzing real defects. As false defects are not physically present on the mask, but are instead a result of incorrect image acquisition or processing of the inspection tool, considerable analysis effort is reduced. The Calibre DefectClassify tool further automates the entire defect classification process. Due to newer technology nodes and the increased subtlety of defects, consistency and effort problems creep into the classification process. Automation addresses these manual-error related issues.

Calibre DefectClassify works on pattern inspection defect files that contain defect images, the corresponding layout information (for example, a MEBES job deck or OASIS file), and optical parameters associated with the mask inspection machine and scanner. By applying advanced rule-based image algorithms to each of the defects, various defect characteristics are considered which are then used to classify the defect. A multitier classification infrastructure is supported, providing a detailed description of a defect. These tiers include the defect's type, its printability, and disposition. User-mapped defect classification codes are then assigned to each of the tiers.

Mask pattern classification for Calibre DefectClassify is currently supported for Klarf 1.2 and 1.8, XML (.def), Nuflare (.xml), and Lasertec (.lrf) inspection file formats. The Lasertec (.lrf) format includes inspections performed using the X700 and X800 series of inspection machines.

Mask Pattern Classification Key Concepts

Several key concepts apply to the mask pattern classification capability of Calibre DefectClassify that will be referenced throughout this document.

Defect Detection Mode

There are a number of detection mechanisms employed by mask inspection machines:

- Die-to-Die (D2D)

  When same die is repeated multiple times on a mask, corresponding regions from across dies are analyzed for differences. A die that differs from others is then considered defective.

- Die-to-Model (D2M)

  When there is no second die on the mask to compare against, a reference is generated using layout and optical conditions against which the comparison is performed.

- Non-Die-to-Die (NonD2D)

  This is typically applied to identify defects inside an active area but outside the device area. In this case, there is no reference to compare against. The transmitted and reflected images together are used to identify defects in this mode.

Of these modes, Calibre DefectClassify currently supports the D2D and NonD2D defect detection modes.

Defect Inspection Mode

Calibre DefectClassify supports classification of defects captured using a number of inspection modes:

- Aerial inspection (Transmitted)

  In this mode, the camera senses the light after it has passed through scanner optics. As this image is on the wafer-plane, it is the closest to the actual optics involved while printing a wafer.

- Reflected inspection

In this mode, the camera senses the light reflected off the top of the surface of a semi-tilted mask. This is typically used to identify surface defects on the opaque mask regions which may cause an issue in later stages of processing.

- High-resolution (Highres) inspection

  In this mode, the camera takes a high resolution of the mask. This is the simplest inspection method, but without the involvement of scanner optics, it becomes difficult to assess the impact of a mask defect at wafer-level. Calibre DefectClassify provides simulation capability, which recovers mask polygons from highres defect images using highres optical conditions, and then simulates the wafer-level output from these recovered polygons using scanner conditions, thus providing a more usable set of images to work on.

Currently, the tool is capable of processing inspections with square-shaped images only (where the defect and reference images have equal width and height). When images are rectangular in dimension, they are cropped to a square shape containing the largest central square part of the original image. These cropped images are then used as input for processing.

Classification Tiers

Calibre DefectClassify processes the defect image associated with each defect and classifies a defect into different categories based on different aspects. These aspects are called tiers. The following is a list of these tiers:

- Defect type classification

  This classification tier organizes defects into types based on categories.

- Printability classification

  This classification tier applies to aerial inspection and simulated highres inspections only, since wafer-level information is only available in these cases.

- Defect size classification

  Defect size refers to the actual size of the defect on the mask. It is applicable only to highres and reflected inspections. The defects are classified into multiple categories based on their nature and measured size.

- SEM classification

  This classification tier applies to SEM classification of a defect. This classification is currently not implemented. A default code of "Review" is assigned for all the defects.

- Defect disposition

  This classification tier refers to the final disposition that is applied to each defect. The disposition is based on either of two schemes, CD variation-based or printability and progress classifications based. The configuration file parameter dispositionBasedOnDefectProgressAndPrintability is used to choose between these two schemes.

- CD Variation Based — The disposition is the combined outcome of defect classification and maximum CD variation. Under this scheme, the following defect disposition categories are applied to defects defined by Calibre DefectClassify:
    - Acceptable — This disposition specifies whether mask quality is acceptable for use. It is applied to false defects and other defects with maximum CD variation less than the value of the parameter dispositionAims.
    - AIMSReview — This disposition specifies that AIMS review is recommended for the defect considering the significant CD variation measured. It is applied for real defects with a maximum CD variation between the values of parameters dispositionAims and dispositionSem.
    - SemReview — This disposition specifies that the mask should be subjected to SEM review because of a very high CD variation. It is applied when the maximum CD variation is higher than the value of parameter dispositionSem.
    - Review — This is the default disposition assigned to each defect.
- Printability and Progress Based — This disposition is the combined outcome of defect type, printability, and progress classification values. These are the supported schemes under the supported categories:
    - Fail — For real defects, when printability classification, using simulation, is either Warning or Fail or, when defect progress classification is Review, the defect disposition is categorized as a Fail.
    - Pass — For false or nuisance defects disposition is Pass. For real defects, if conditions for Fail do not satisfy, then it is Pass. As a special case, when neither database is configured nor simulation is run, the disposition is always Pass.

- Defect Progress Classification

  Progress classification refers to tracking of a defect's properties across inspections. A functional database is a prerequisite for this tier, as it works on the correlation of defects across inspections of same mask. Further details are described in the *Calibre DefectReview User's Manual*. When a database is not configured, this tier returns the default value Unclassified.

  For a recurring or repeating defect, properties such as residue, and residue size are measured and compared against previous occurrences. The following are the categories defined by Calibre DefectClassify:

    - Adder — When a defect is observed for the first time, it is called an Adder defect.
    - Repeater — When a defect has been previously observed, then it is called a Repeater defect.

- Review — These are a special case of repeater defects. Defect properties such as residue (transmitted and reflected), size, and transmittance are measured and compared against their base values (from the inspection where the defect was first observed). If any of the four properties has a deviation greater than the user-specified threshold, the defect is classified as Review.

The exact threshold values for each of the properties are obtained from the user through the Calibre DefectReview configuration file (the Thresholds node in the *dat-ini.xml* file). The following is an example of the Thresholds node:

```
<Thresholds>
  <DefectAttributes>
 <!-- Unit of Size is Square Microns such as 0.05 square microns -->
    <Size>
     <UpperControlLimit>0.05</UpperControlLimit>
     <LowerControlLimit>0</LowerControlLimit>
    </Size>
    <!-- Unit of TransmittedResidue is GL such as 10 GL -->
    <TransmittedResidue>
     <UpperControlLimit>10</UpperControlLimit>
     <LowerControlLimit>0</LowerControlLimit>
    </TransmittedResidue>
    <!-- Unit of ReflectedResidue is GL such as 10 GL -->
    <ReflectedResidue>
     <UpperControlLimit>10</UpperControlLimit>
     <LowerControlLimit>0</LowerControlLimit>
    </ReflectedResidue>
    <!-- Unit of MaxTransmittance is % such as 10% -->
    <MaxTransmittance>
     <UpperControlLimit>10</UpperControlLimit>
     <LowerControlLimit>0</LowerControlLimit>
    </MaxTransmittance>
  </DefectAttributes>
</Thresholds>
```

The four sub-nodes correspond to defect size, transmitted residue, reflected residue, and transmittance, respectively. The attribute UpperControlLimit (UCL) is used to specify the difference threshold. LowerControlLimit is currently not used. For example, with UCL for size specified as 0.05, if the size of a defect changes by more than 0.05 square um with reference to a base (first observed) value, then progress classification is specified as Review.

Refer to "Calibre DefectClassify Classification Tiers" on page 19 for complete descriptions of each tier.

Calibre DefectClassify Classification Tiers

Calibre DefectClassify processes the defect image associated with each defect and classifies the defects into different categories under five different categories.

Defect Type Classification

Calibre DefectClassify identifies a number of different types of defects. Defects can be classified into one of the two categories:

- Real — Defects that are physically present on the mask. These defects must be characterized, and then repaired or removed accordingly. Real defects include general dark and clear defects, surface contaminations, large defects, assist feature defects and fill defects.
- False — Defects that are not on the mask, but are instead tool or process related. Examples include nuisance defects and camera defects.

Combinations of real and false defects are frequent. When a false signal is identified, it is compensated for where possible. For example, if an alignment issue arises for two images, the two images are aligned during the process. After compensation, they are re-analyzed for the presence of a secondary real defect before discarding the defect as a false defect.

Classification of defects as real or false is an important part of improving the efficiency of a defect review process. This is because real defects, being actually present on the mask, require more detailed and precise characterization, which impacts the effectiveness of the repair processes employed. Higher false defects require more time for their review, decreasing the review throughput.

Incorrectly classifying real defects as false is also an issue as that leads to a physical defect being ignored on the mask. Combined with the fact that a significant number of false defects are caught by the inspection machines, reliable separation of false from real defects makes considerable difference to classification efficiency.

The following is complete list of defect types that are identified by Calibre DefectClassify (these are the actual classification names assigned to defects):

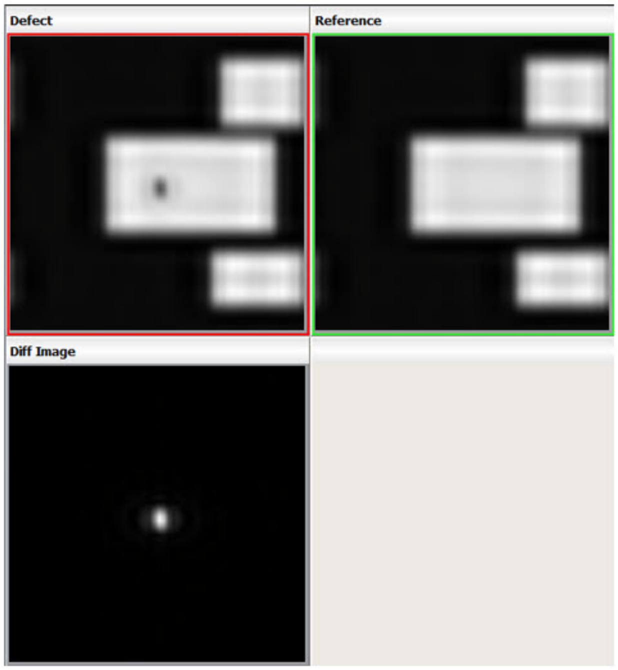
- Dark Defects

  These are defects for which the pixel intensities in the defect area of the defect image are less than those in the corresponding region of reference image. These are further classified into the following sub-categories:

  - IsolatedDark

    These defects do not impact any pattern edge on the mask. The parameters commonPatternDistanceFromDefectSignal (for aerial and reflected inspections) highresCommonPatternDistanceFromDefectSignal (for highres inspections) store the threshold values that determine the distance from an edge and are enough for the defect to be treated as isolated. Calibre DefectClassify detects this type of defect in D2D and NonD2D inspection modes. These parameters are described in "Mask Pattern Classification Configuration File" on page 72.

Figure 2-1. IsolatedDark Defect



- IsolatedDarkLowTrans

  These defects are similar to IsolatedDark except the transmittance value is lower than the value provided using the parameter transmissionThresholdForIsolatedOpaque (described in "Mask Pattern Classification Configuration File" on page 72). The transmittance value of an isolated blob is calculated as a ratio:

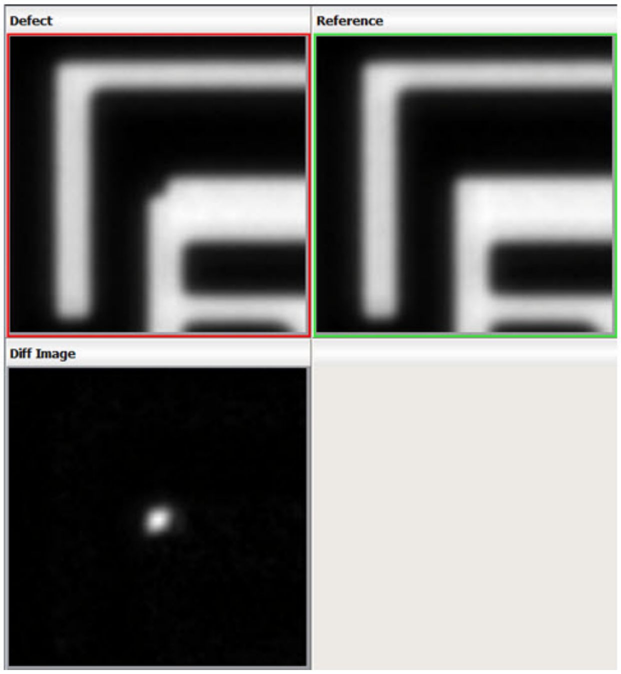
  (GL value of the darkest pixel in the blob) / (maximum GL value assigned to pixels in large clear regions by the inspection machine)

  For example, if the GL value of the darkest pixel in the blob is 170 and the maximum GL value of pixel in the larger clear region is 255, the transmittance value is 170 / 255 = 0.66. If the value is less than the value set by transmissionThresholdForIsolatedOpaque, the defect is classified as IsolatedDarkLowTrans; otherwise, it is classified as IsolatedDark. Calibre DefectClassify detects this type of defect in D2D and NonD2D inspection modes.

- OnEdgeDark

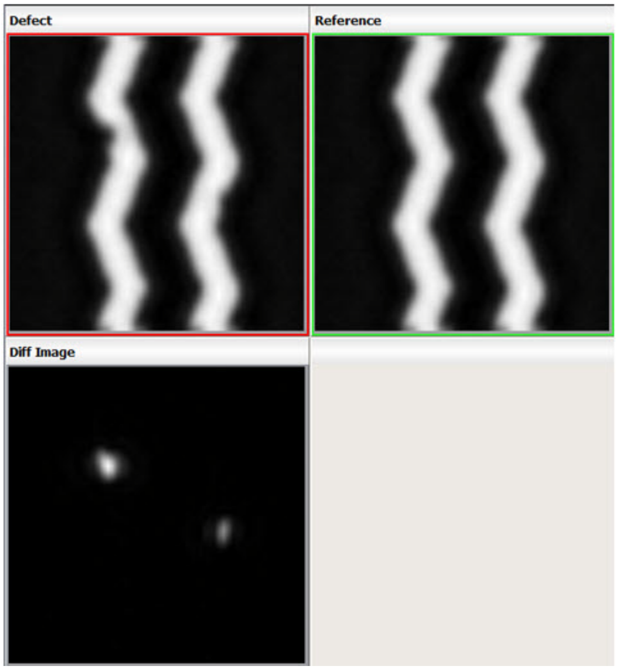
  These are dark defects where their distances to a pattern edge are less than the threshold specified by the parameters commonPatternDistanceFromDefectSignal and highresCommonPatternDistanceFromDefectSignal (see "Mask Pattern Classification Configuration File" on page 72), and affect a single pattern edge. Calibre DefectClassify detects this type of defect in D2D and NonD2D inspection modes.

Figure 2-2. OnEdgeDark Defect



- MultipleEdgesDark — These defects are similar to OnEdge defects except they affect multiple pattern edges. Calibre DefectClassify detects this type of defect in D2D inspection mode only.

Figure 2-3. MultipleEdgesDark



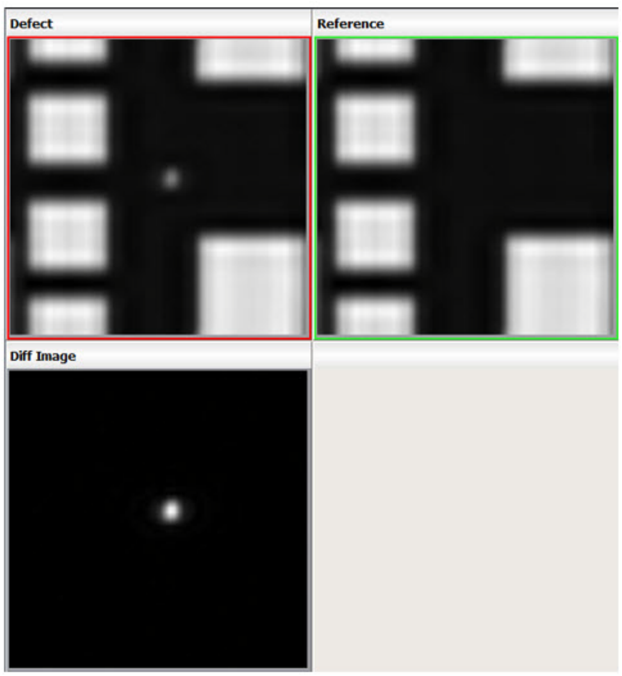
- Clear Defects

These are defects for which the pixel intensities in the defect area of the defect image are higher than those in the corresponding region of reference image. These are further classified into:

- IsolatedClear

  These defects are similar to their darker counterparts. They do not impact any pattern edge on the mask. The parameters commonPatternDistanceFromDefectSignal (for aerial and reflected inspections) and highresCommonPatternDistanceFromDefectSignal (for highres inspections) store the threshold valuse that determine if the distance from an edge is far enough for the defect to be treated as isolated. These parameters are described in "Mask Pattern Classification Configuration File" on page 72. Calibre DefectClassify detects this type of defect in D2D and NonD2D inspection modes.

Figure 2-4. IsolatedClear Defect



- IsolatedClearLowTrans

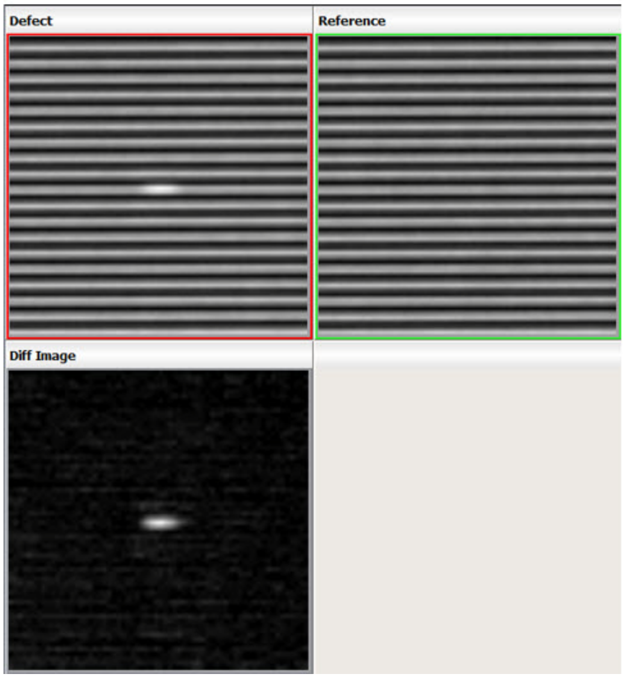
  These defects are similar to IsolatedClear defects, except their transmittance values are lower than 0.1. Calculation of the transmittance value of an isolated blob is the same as the defect type IsolatedDarkLowTrans. Calibre DefectClassify detects this type of defect in D2D and NonD2D inspection modes.

- OnEdgeClear

  These are clear defects similar to OnEdgeDark defects; their distance to a pattern edge is less than the threshold specified by the parameters commonPatternDistanceFromDefectSignal and highresCommonPatternDistanceFromDefectSignal (see "Mask Pattern

Classification Configuration File” on page 72), and they affect a single pattern edge. Calibre DefectClassify detects this type of defect in D2D inspection mode only.

Figure 2-5. OnEdgeClear Defect

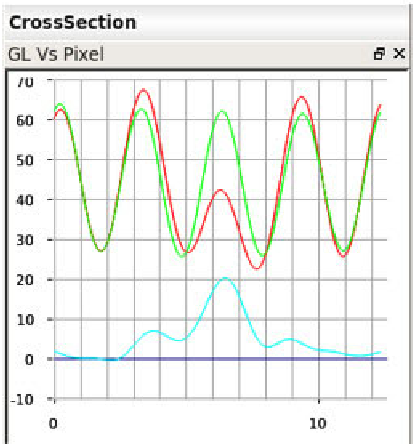


- MultipleEdgesClear — These defects are similar to OnEdge defects except they affect multiple pattern edges. Calibre DefectClassify detects this type of defect in D2D inspection mode only.

- ClearOpaque

  These defects have the characteristics of both opaque and clear defects. This means that they have an opaque region (where the defect pixel intensity is lower than the reference) and a clear region (where the defect pixel intensity is higher than the reference), and both regions exist adjacent to each other. Figure 2-6 shows the cross-section of this type of defect on a line-space pattern. The pixel intensities are mapped in a direction perpendicular to the line patterns. Defect pixel GL values (red) are both higher and lower than reference GL values (green) for adjacent line-space patterns (peaks correspond to line patterns and valleys are the space between patterns).

Figure 2-6. Cross-Section of a ClearOpaque Defect



- Contamination

  These are a group of defects that flag contamination that occurs in the mask:

  - Contamination

    These defects are found on the top surface of the mask lining up with the chrome regions. Due to the opacity of chrome regions, they are not visible on transmitted images. Therefore, reflected images are used to identify these defects. Calibre DefectClassify detects this type of defect in D2DRef and NonD2D inspection modes.

  - OnDarkEdgeContamination

    These are contamination defects that are closer to an edge and visible only in the reflected image. For aerial inspections, the parameter commonPatternDistanceFromDefectSignal (see "Mask Pattern Classification Configuration File" on page 72) stores the threshold value that decides when the distance from an edge is close enough for the defect to be termed as a defect on edge.

  - OnClearEdgeContamination

    These are contamination defects that are closer to an edge and visible in the transmitted image as well. They can be identified only if both transmitted and reflected images are present. If significant difference are present on a pattern edge in the transmitted image, the defect is classified as OnClearEdgeContamination.

- Repair Mark

  This is a marking made on the mask indicating the location of a previously existing defect. The mask may currently have stray or residual signals from the correction process. The markings typically form a rectangle or trapezoid with each of the four corners appearing similar to Contamination defects. Any residual signals from the correction process can be observed within the boundary formed by the corner points.

Because these defects are an extension of Contamination defects, Repair Marks can be seen in D2DRef and NonD2D inspection modes only.

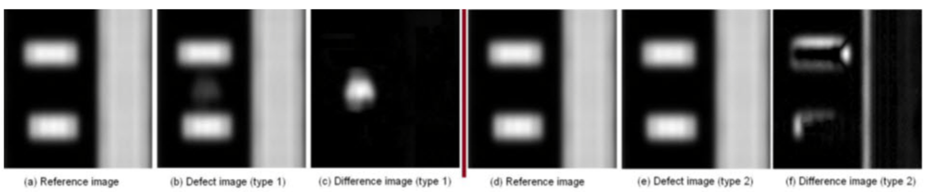
The Repair Mark defect type has two subcategories that can be detected:

- Patch — A dark rectangular area in the quartz region of the reflected image. The gray-level of the repair mark area is slightly different from the rest of the reflective white region.
- Corner points — Multiple contamination-like defects forming a certain regular shape. The corner points of the repair mark area are marked on the reflective white region. The geometry between these points is typically rectangular.

- MissingChrome

  For this type of defect, the molybdenum silicide (MoSi) layer is exposed because of a patch of chrome is missing or was etched. The defect typically appears as a faint isolated signal or as a strong signal along feature edges. They are identified using reflected images. Chrome layer information is required to identify this type of defect. The following figure illustrates shows the two types of missing chrome defects.

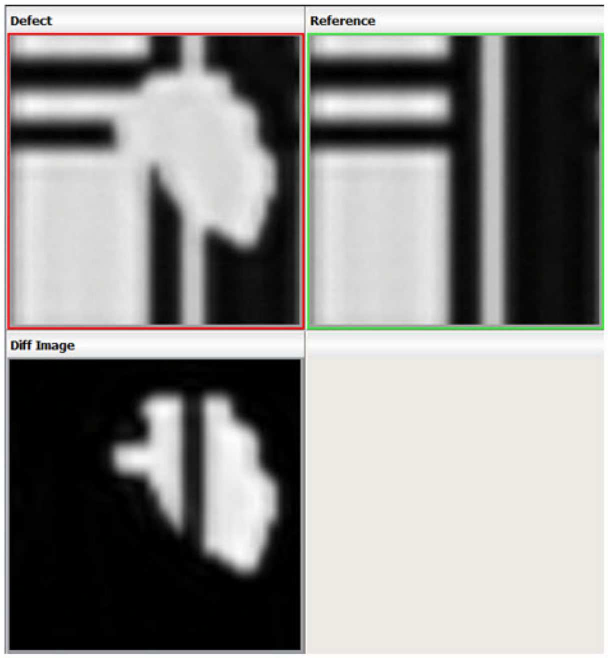
Figure 2-7. MissingChrome Defect



- LargeDefect

  These defects are large in size and are classified as non-repairable.

Figure 2-8. LargeDefect



The following user-configurable parameters are related to this category:

- nonrepairMaxDiffGLLowerLimit
- nonrepairDefectArea
- nonrepairNumPatternAffected
- highresNonrepairMaxDiffGLLowerLimit
- highresNonrepairDefectArea
- highresNonrepairNumPatternAffected

Refer to "Mask Pattern Classification Configuration File" on page 72 for a description of these parameters.

- Assist Feature Defects

  These defects are found on mask regions containing sub-resolution assist features (SRAFs). Accurate identification of SRAF regions requires layout information. As these regions do not print, it becomes impossible to identify a defect as overlaying on an SRAF region by looking at wafer-level images alone. Based on the pixel intensities of the defect relative to corresponding region in the reference image, these defects are classified as:

  - SRAFDark
  - SRAFClear

- Fill Defects

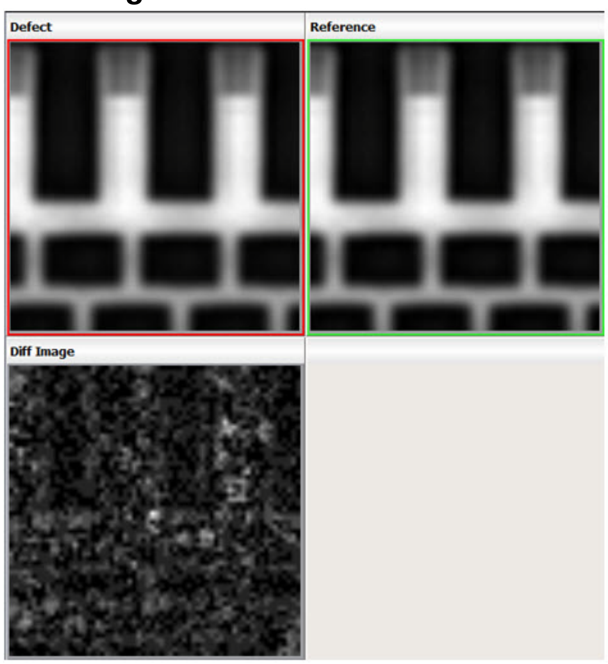
These defects are found on fill regions on the mask. Similar to SRAF defects, these defects also do not print, and thus additional layer data from layout file is required to correctly classify them. Based on relative pixel intensities again, these defects are classified as:

- FillDark
- FillClear

- Nuisance

  These are a mix of insignificant small defects and defect signals resulting due to inherent electrical noise. They are referred to as "nuisance" because the defect signal intensities are not significant and are unlikely to impact mask usability.

Figure 2-9. Nuisance Defect



- Real Nuisance

  These are small real defects on large patterns, corner, or assist features. They can only be detected if transmitted and reflected images are both present. Defects are differentiated with clear defects, dark defects, and SRAF defects based on residue and other defect properties. The distinguishing boundaries are defined using the following configurable thresholds set in the Calibre DefectClassify tuning file (see "Mask Pattern Classification Configuration File" on page 72):
  minDiffForHardDefectsOnSmoothEdgesInTransmittedImage,
  minDiffForHardDefectsOnSmoothEdgesInReflectedImage,
  minDiffForHardDefectsWithEdgeRoughnessInTransmittedImage,
  minDiffForHardDefectsWithEdgeRoughnessInReflectedImage,
  minDiffForHardDefectsOnJogsAndCornersInTransmittedImage,

minDiffForHardDefectsOnJogsAndCornersInReflectedImage, and minDiffForSrafDefects.

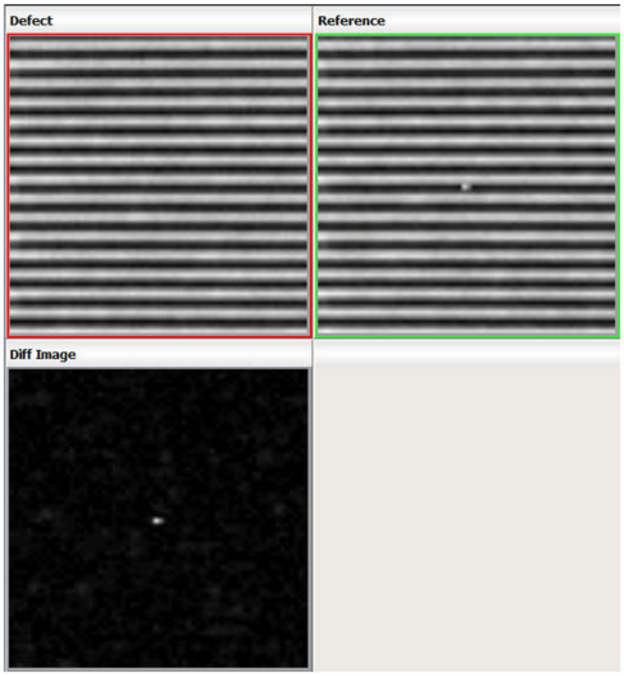
- Edge Roughness

  These are small defects on large patterns and considered as False if the width of the defect blob or intensity in the difference image (the difference between the defect and reference images) is smaller. The following parameters in the Calibre DefectClassify tuning file are used to differentiate Edge Roughness from Real Nuisance (see "Mask Pattern Classification Configuration File" on page 72): maxDiffIntensityToConsiderEdgeRoughness, maxDiffIntensityToConsiderAllEdgeRoughnessAsFalse and maxDefectSizeToConsiderRestEdgeRoughnessAsFalse.

- Bright Spot

  These defect signals originate due to erratic camera pixels during image acquisition, leading to unnaturally strong signals without any relation to the surrounding intensities. Moreover, owing to camera issues, the defect signals from multiple defects are seen at the same frame location, which is highly unlikely for real defects.

Figure 2-10. Bright Spot Defect



- Tap

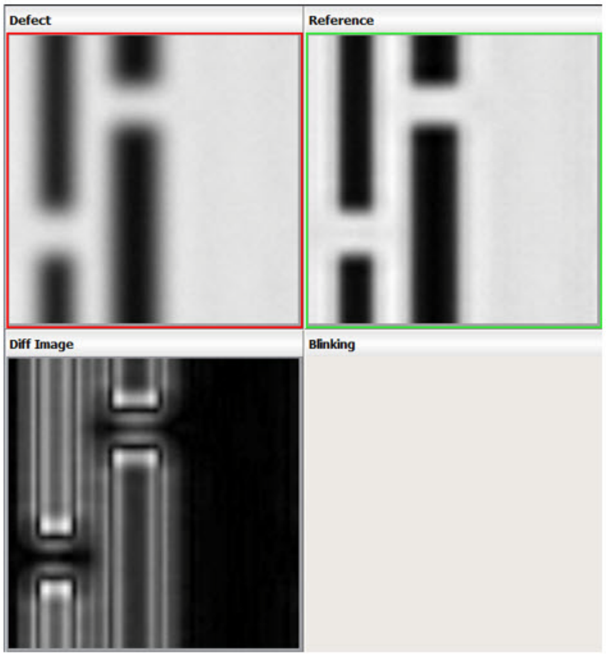
  These weak defect signals are a result of a corrupted data writing operation which itself can be due to different reasons. They are usually observed at boundaries of memory segment's sizes. For example, if a segment of size 128x128 is written each time, errors are observed at pixels with coordinates around multiples of 128.

Figure 2-11. Tap Defect

- Focus

  These defect signals are a result of an incorrect camera focus setting during image acquisition. Typically one of the two images, defect or reference, is out of focus.

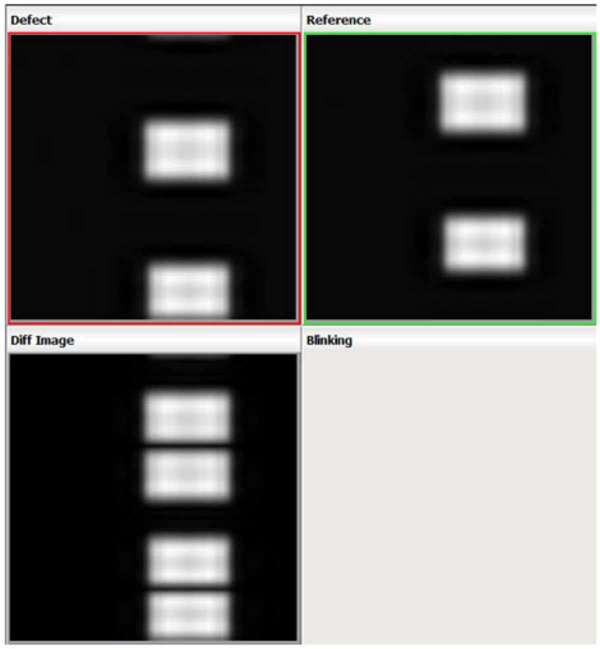
Figure 2-12. Focus Defect



- Alignment

The defect signals in this case are due to misalignment between the defect and reference images.

Figure 2-13. Alignment Defect



Printability Classification

Printability classification refers to the impact of a defect. For aerial and simulated highres inspections, the classification correlates to the wafer-level impact of the defect. However, for reflected inspection printability, the classification refers to the severity of the defect. For aerial and highres inspections, defects are placed into multiple bins based on maximum CD variation measured in the defect image. The following are the bins that are defined by Calibre DefectClassify:

- Nuisance

  This bin corresponds to defects with negligible CD variation. If the maximum CD variation in an image is below the threshold defined by the user-configurable parameter printabilityNuisance, the defect is assigned the printability classification of Nuisance.

- Pass

  This bin corresponds to defects with some CD variation, but not significant enough to deem the defect serious. For this bin, the maximum CD variation is between the values of parameters printabilityNuisance and printabilityPass.

- Warning

  This bin corresponds to defects with a little higher CD variation and acceptable levels. For this bin, the maximum CD variation is between the values of parameters printabilityPass and printabilityWarning.

- Fail

  This bin corresponds to defects with significantly high CD variation. For this bin, the maximum CD variation is higher than the value of the parameter printabilityWarning.

For reflected inspections, classification is based on size of the defect. The following are the printability classification categories:

- Pass

  This classification is applied when the defect is smaller or equal to a certain size, specified by the “sizeThreshRefl” parameter in the *adc-ini.xml* configuration file.

- Fail

  This classification is applied when the defect is larger than a certain size, specified by the “sizeThreshRefl” parameter in the *adc-ini.xml* configuration file.

When both transmitted and reflected images are present, printability classification depends on CD Variation (%) calculated using simulated transmitted images. If the CD Variation is equal to or more than 10%, then the tool reports both defect type classification and printability classification using transmitted images, even if the defect type from the reflected image has a higher priority than the transmitted image.

However, if the following conditions are true:

- CD Variation is less than 10%.
- The defect is classified as false using a transmitted image.
- The defect is classified as real using a reflected image.

the tool reports defect type classification based on the reflected image, and printability classification is based on CD Variation calculated using simulated transmitted images.

Defect Size Classification

Defect size refers to the actual size of the defect on the mask, not at the wafer level. Because aerial inspections report images close to wafer-level, defect size classification is applied only to highres inspections and to defects seen only in reflected images. The defects are classified into multiple categories based on their nature and measured size. These are the various classification categories currently supported by Calibre DefectClassify:

- False

  This category applies to false defects. Since these defects are not actually on the mask, size measurement does not apply in this case.

- Bin1 ... Bin9

These nine bins contain defects of increasing measured sizes. Eight threshold values define the boundaries of these bins. These threshold values can be customized through the configurable file *adc-ini.xml*. The parameter names corresponding to the eight thresholds are *sizeThresh1* through *sizeThresh8*. The following list details the range of defect sizes corresponding to each bin.

- Bin1: Measured Size <= sizeThresh1 (0, sizeThresh1)
- Bin2: *sizeThresh1* < Measured Size <= sizeThresh2 (*sizeThresh1*, sizeThresh2)
- Bin3: *sizeThresh2* < Measured Size <= *sizeThresh3* (*sizeThresh2*, *sizeThresh3*)
- Bin4: *sizeThresh3* < Measured Size <= *sizeThresh4* (*sizeThresh3*, *sizeThresh4*)
- Bin5: *sizeThresh4* < Measured Size <= *sizeThresh5* (*sizeThresh4*, *sizeThresh5*)
- Bin6: *sizeThresh5* < Measured Size <= *sizeThresh6* (*sizeThresh5*, *sizeThresh6*)
- Bin7: *sizeThresh7* < Measured Size <= *sizeThresh7* (*sizeThresh6*, sizeThresh7)
- Bin8: *sizeThresh8* < Measured Size <= *sizeThresh8* (*sizeThresh7*, *sizeThresh8*)
- Bin9: Measured Size > *sizeThresh8* (*sizeThresh8*, ∞)

- NotApplicable

  This classification is applied to defects that are identified in aerial transmitted mode, where mask level images are not available for size measurement.

SEM Classification

This classification tier deals with SEM classification of a defect. This classification is currently not implemented. A default code of Review is assigned for all the defects.

Defect Disposition

Defect disposition refers to the final disposition that is applied to each defect. Disposition is the combined outcome of defect classification and maximum CD variation. The following are the disposition categories applied to defects by Calibre DefectClassify:

- Acceptable

  This disposition means that mask quality is acceptable for use. It is applied to false defects and other defects with maximum CD variation less than the value of the parameter dispositionAims.

- AIMSReview

  This disposition means that AIMS review is recommended for the defect considering the significant CD variation measured. It is applied for real defects with maximum CD variation between the values of parameters dispositionAims and dispositionSem.

- SemReview

  This disposition means that the mask should be subjected to SEM review because of the very high CD variation. It is applied when the maximum CD variation is higher than the value of parameter dispositionSem.

- Review

  This is the default disposition assigned to each defect.

Calibre DefectClassify Defect Classification Rules

Calibre DefectClassify uses specific algorithms to determine the type of defects based on signals detected in transmitted or reflected images (or both), captured in D2D, D2DRef or NonD2D inspection modes. .

D2D and D2DRef Defect Classification

Sometimes inspection files have both transmitted (D2D) and reflected (D2DRef) images for a defect. Classification rules are different if an inspection is performed in contamination mode and otherwise.

For Contamination mode, defect type classification is performed using D2D reflected images; whereas CD Variation calculation and defect printability classification is performed using D2D transmitted images. Defect size classification is always done based on D2D Reflected images.

For inspections performed in any other mode, defect type classification is attempted using transmitted images. If Calibre DefectClassify does not find any real defect in D2D transmitted images, the defect type classification is done using reflected images. Defect printability classification is always done on D2D transmitted images. Defect size classification uses the same images that are used for defect type classification.

NonD2D Isolated Defect Classification

For NonD2D Isolated defects, the following algorithms are used for defects classification.

- IsolatedDark and IsolatedDarkLowTrans

  a. Calibre DefectClassify checks for the presence of a defect signal in the white region of a transmitted image.

  b. Calibre DefectClassify checks if a detected signal is locally dark.

  c. If the signals are present, then transmittance is calculated as follows:

     Defect transmittance = (darkest defect GL) / (max transmittance of inspection machine (listed in the inspection report))

  d. If transmittance is less than the value of transmissionThresholdForIsolatedOpaque (described in the "Mask Pattern Classification Configuration File" on page 72), then the defect is classified as IsolatedDarkLowTrans. If the transmittance is greater than transmissionThresholdForIsolatedOpaque, the defect is classified as Isolated Dark.

- IsolatedClear and IsolatedClearLowTrans

  a. Calibre DefectClassify checks for the presence of a defect signal in the dark region of transmitted image.

  b. Calibre DefectClassify checks if a detected signal is locally bright.

  c. If the signals are present, then transmittance is calculated as follows:

  Defect transmittance = (brightest defect GL) / (max transmittance of inspection machine (listed in the inspection report))

  d. If the transmittance is greater than 0.1, then the defect is classified as IsolatedClear. If it is less than 0.1, then the defect is classified as IsolatedClearLowTrans.

- Isolated Contamination

  a. Calibre DefectClassify checks for the presence of signal in the white region in reflected image and no defect signal exists at the corresponding location in the transmitted image.

  b. If a detected signal is dark, the defect is classified as an isolated Contamination defect.

- Bright Spots

  a. Calibre DefectClassify checks for the presence of a defect signal in with the following conditions:

  - Located in the white region of the transmitted image and no other signal was identified on the transmitted black region and reflected images.

  - Located in the reflected image and no other signal was identified in transmitted image.

  b. If a detected signal is locally bright, then the defect is defined as a Bright Spot.

Note

Only isolated Bright Spot defects can be identified using this algorithm.

If a defect is detected by multiple classifiers, then final classification is based on defect category's priority as specified in *adc-ini.xml* configuration file.

NonD2D On-Edge Defect Classification

For NonD2D On-Edge defects, the following algorithms are used for classification.

- OnDarkEdgeContamination

a. Calibre DefectClassify checks for the presence of a defect signal in the white region of a reflected image.

b. Calibre DefectClassify checks if a detected signal is locally dark.

c. If the signal is along or close to a pattern edge, the defect is defined as OnDarkEdgeContamination.

- Bright Spot

a. Calibre DefectClassify checks for the presence of a defect signal in the transmitted or reflected image.

b. Calibre DefectClassify checks if a detected signal is locally bright.

c. If the signal is along or close to a pattern edge, the defect is defined as a Bright Spot.

Mask Pattern Classification Modes of Operation

There are two modes of operation for the Mask Pattern Classification feature of Calibre DefectClassify: a command line interface (CLI) or the Calibre DefectReview graphical user interface (GUI).

Using Mask Pattern Classification in CLI Mode

The Mask Pattern Classification tool is can be executed from a command line where multiple inspections can be loaded and classified without manual intervention. After each inspection has completed classification, the defect file can be updated with the results. These updated files containing classification results can then be loaded in the Calibre DefectReview for further review and analysis. Calibre DefectClassify can be invoked from the Calibre DefectReview command line through a Tcl file.

To run the Mask Pattern Classification tool using command line interface (CLI), you must first prepare a Tcl file, then use the Calibre DefectClassify options on the Calibre DefectReview command line.

Procedure

1. In a text editor, create a Calibre DefectClassify Mask Pattern Classification Tcl file. There are several commands that can be used to create this file.

    - flipsource — This command flips the input source map in X- or Y- directions (or both). The output source map then can be used to generate an optical model using the opticsgen command. This command is useful only if optical model generation is performed using a source map file, which is generally the output of Source Mask Optimization done for a particular node mask. This command is only supported in the CLI mode of Calibre DefectReview.

    - patternautoclassify — Performs automatic defect classification of patterned mask defects.

    - opticsgen — Generates models from optical conditions. Both highres and aerial optical models are critical for reliable classification of defects. The basic syntax is as follows:

    ```
    opticsgen <optical_parameters_text_file>
    <output_model_folder_name>
    ```

This command reads the text file ***optical_parameters_text_file*** (containing highres or scanner optical conditions) and passes the opticsgen parameters to Calibre WORKbench (the tool that actually runs the opticsgen command). The generated optical model data is stored in the folder specified by ***output_model_folder_name***. This folder's path is later specified as a highres or scanner optical model directory (either through the CLI or the GUI).

For a full description of the opticsgen command, refer to *Calibre WORKbench User's and Reference Manual*. The following are two examples:

```
opticsgen hrsmodel.txt highres_model
opticsgen scannermodel.txt scanner_model
```

- saveas — Saves the input defect file along with classification results into a new file. This is a Calibre DefectReview command. For information on the saveas command, refer to the *Calibre DefectReview User's Manual*.

The following is an example Tcl file:

```
flipsource --in input.src --out output.src --mirrorx 0 --mirrory 1
opticsgen hrsmodel.txt highres_model
opticsgen scannermodel.txt scanner_model
patternautoclassify -p cli_params.xml
saveas classified 1 ./
```

2. Use the following syntax at a command line prompt:

```
$MGC_HOME/bin/nxdat -c <tcl_file> -i <insp_file>
```

where:

- ***tcl_file*** refers to the name of the input Mask Pattern Classification Tcl file.
- ***insp_file*** refers to the patterned mask inspection to be loaded into Calibre DefectReview.

Results

Refer to "Mask Pattern Classification Results" on page 48 for a description of the results.

Mask Pattern Classification in GUI Mode

You can invoke the Calibre DefectClassify Mask Pattern Classification tool directly from Calibre DefectReview. Selected defects can be filtered and run again through the GUI for further analysis.

Invoking Mask Pattern Classification in GUI Mode

You invoke the Calibre DefectClassify Mask Pattern Classification tool from the **Utilities** menu of Calibre DefectReview.

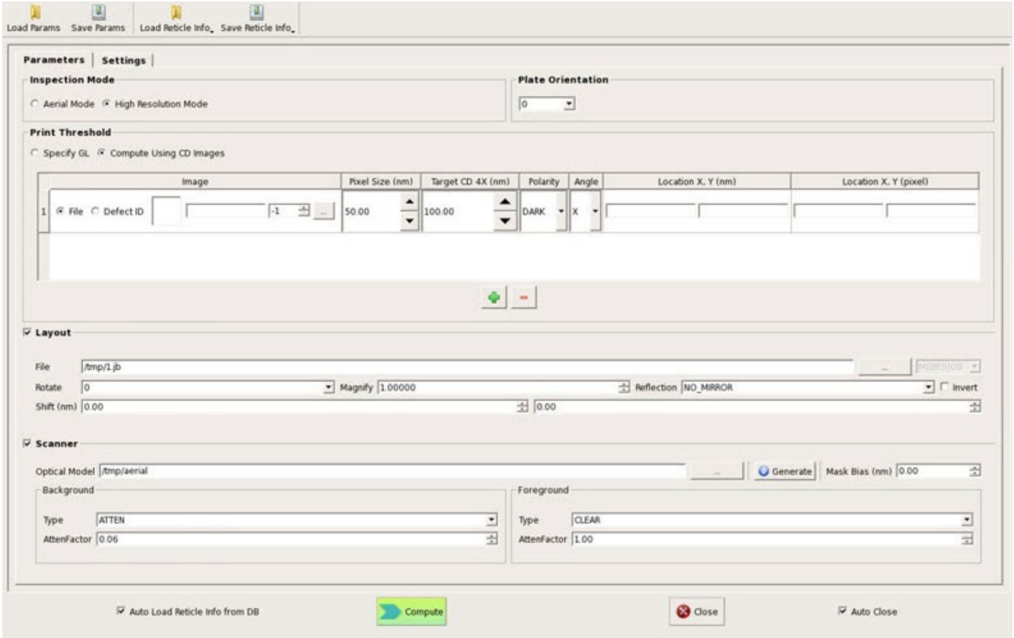
Prerequisites

- Calibre DefectReview must be invoked.
- A pattern inspection file must be loaded into Calibre DefectReview. This can be done through the **Add Inspection(s)** option in Calibre DefectReview. Detailed instructions regarding this option can be found in the *Calibre DefectReview User's Manual*.

Procedure

1. To launch Calibre DefectClassify from Calibre DefectReview, click **Utilities** > **DefectClassify** or click the Calibre DefectClassify icon. 

Figure 2-14. Calibre DefectClassify Mask Pattern Classification Window



The following operations are available to manage defect classification:

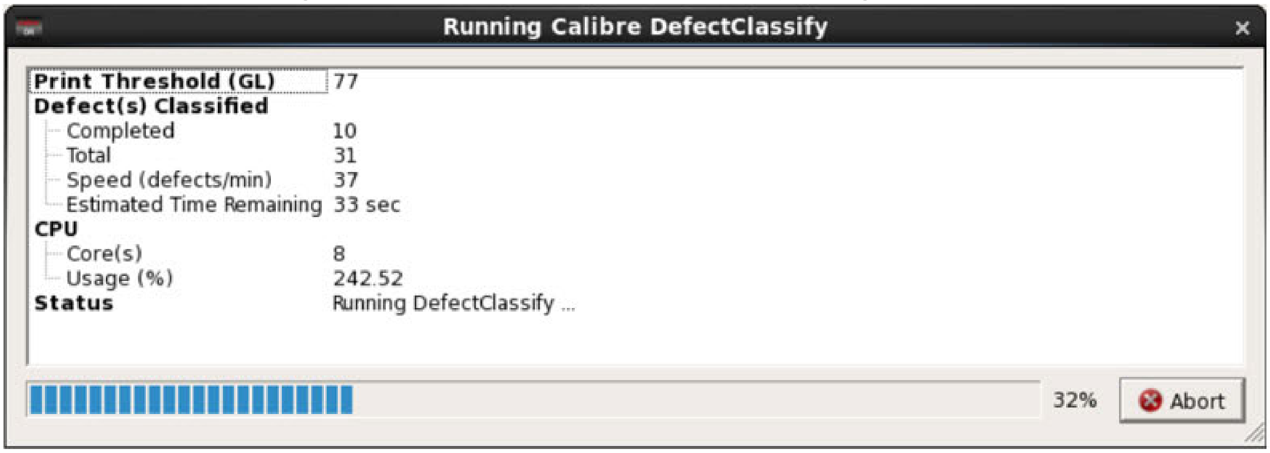
- All parameters set or entered in the GUI can be saved to an XML file by clicking the **Save Params** button. You can later load the XML file using the **Open Params** button.
- The **Auto Close** check box closes the GUI when a run is completed, otherwise the GUI remains open.
- Reticle Information Support — In addition to the parameters file, you can use reticle information either from a Reticle Information File (RIF) or from a defect database as input to a Calibre DefectClassify run. Click the **Load RIF Info** button to load this information into the GUI. Once the reticle information is loaded, you can edit and save changes back to the RIF file or defect database.

  The **Auto Load Reticle Info from DB** check box automatically enables loading reticle information associated with a defect file from the defect database. Calibre DefectClassify assumes that the reticle information is already stored in the DB and linked with the defect file.

  Note
  Not all reticle information is loaded into the GUI. Only information relevant to Calibre DefectClassify is loaded.

2. Click the **Compute** button at the bottom of the Calibre DefectClassify window to start automatic classification. A progress bar is displayed, showing the progress in terms of number of defects processed.

Figure 2-15. Calibre DefectClassify Progress Bar



Note

Upon launch, the Calibre DefectClassify window is populated with previously-used settings and parameter values. After each run, the Calibre DefectClassify window stays open to facilitate launching the next run.

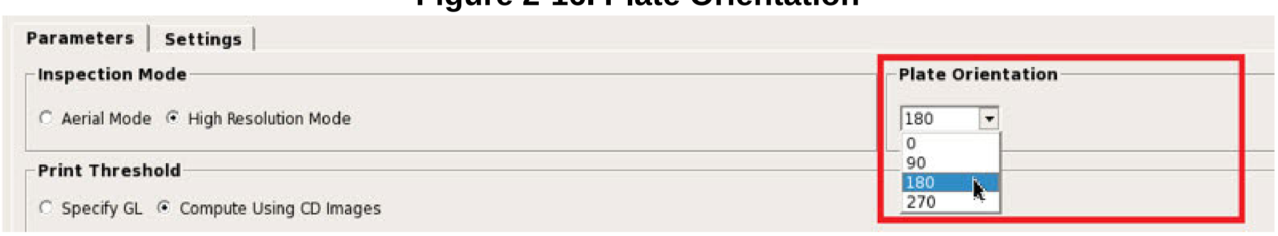
Setting Plate Orientation

To run the Mask Pattern Classification tool, you must select a plate orientation value.

Procedure

In the **Parameters** tab in the Calibre DefectClassify window, select a value from the Plate Orientation pull-down menu (0, 90, 180, 270).

Figure 2-16. Plate Orientation



The value selected here is used in the following operations:

- The value is added to the Rotation value specified for layers of the layout before using it in the Calibre DefectClassify flow.
- The Scanner model is rotated by the angle chosen for Plate Orientation before doing aerial simulation.

Setting Print Threshold Parameters

To run the Mask Pattern Classification tool, you must provide a print threshold.

Procedure

To set the print threshold, in the Print Threshold pane of the **Parameters** tab in Calibre DefectClassify, select one of the following:

- **User Specified GL** — Provide a Grey Level (GL) value between 0 and 255.

Figure 2-17. Specify GL

- **Compute Using CD Images** — Provide a CD image and auxiliary information from which the print threshold GL is calculated.

Figure 2-18. Compute Using CD Images

Fill in the following information:

- **File** or **Defect ID** — Specify the path of the target CD image containing the feature using which printability threshold is calculated. You can specify the CD Image either by referring to an external image (**File**) or by choosing an image corresponding to one of the defects in the loaded inspection (**Defect ID**). The CD image file type can be PNG, BMP, or TIF.
- **Pixel Size (nm)** — Specify the size of each CD image pixel in nm. The default pixel size is populated from currently loaded inspection result.
- **TargetCD 4X (nm)** — Specify the expected feature dimension at mask level (for example, 4X).
- **Polarity** — Specify the tone at the location specified above. Permitted values are Clear and Dark.
- **Angle** — Specify the angle at which cross-sectional CD measurement is performed. 0 represents vertical, with pixels considered starting from bottom

along the length of cross-sectional bar; 90 represents horizontal, from left to right; continuing in a clockwise direction. Possible values are 0, 90, 180 and 270.

- **Location X, Y (nm)** — This maps to the LocationInLayout parameter in the patternautoclassify file. Refer to the LocationInLayout description in "patternautoclassify Parameter File" on page 60.
- **Location X, Y (pixel)** — This maps to the LocationinImage parameter in the patternautoclassify file. Refer to the LocationInImage description in "patternautoclassify Parameter File" on page 60.

More than one CD image can be used for print threshold calculation. The print threshold used by Calibre DefectClassify, in this case, is the average of print thresholds calculated from each CD image.

The **+** and **-** buttons are used to add and remove CD images. To remove a CD image, select a row and the press the **-** button.

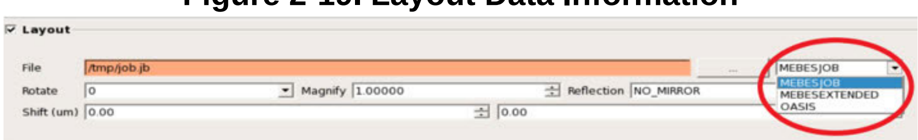
Setting Layout Parameters

You must provide the layout file and transformation information to align the layout file with the mask inspection result. The layout data formats currently supported are MEBES job decks, extended MEBES job decks, and OASIS files.

Procedure

1. In the **Parameters** tab of Calibre DefectClassify, click the **Layout** radio box.
2. Select the input type in the pulldown menu on the right side of the pane, either MEBESJOB, MEBESEXTENDED, or OASIS.
3. Enter the layout file and transformation parameters in the Layout pane.

Figure 2-19. Layout Data Information



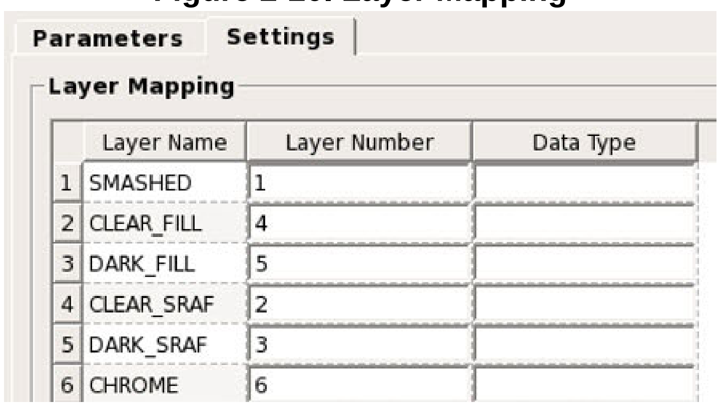
- **File** — Specify the layout file path.
- **Rotate** — Specify any rotation to be applied to layout data to align with mask data. The values can be 0, 90, 280 or 270. Rotation is performed around point (76,200,000 nm, 76,200,000 nm).
- **Magnify** — Specify the magnification level of layout data. Currently, this is 1 and cannot be changed.

- **Reflection** — Specify optional mirroring to be applied to layout data. Possible values are NO_MIRROR, X_MIRROR, and Y_MIRROR. The default is NO_MIRROR.
- **Invert** — Specify layout tones be inverted to synchronize with mask data. Click the check box if the mask is of negative tone.
- **Shift** — Specify the amount of translation to be applied to the layout.

4. Calibre DefectClassify can detect defects in SRAF (dark and clear) or Fill (dark and clear) regions only if the layout data file contains the corresponding polygons in separate layers. If this information is available, click the **Settings** tab in Calibre DefectClassify and provide the mapping for these Layer Name(s) to Layer Number(s) and Data Type(s) (see Figure 2-20). If the layout data is not split out, then only map the SMASHED Layer Name.

Note
At minimum, the SMASHED Layer Name must be mapped.

Figure 2-20. Layer Mapping



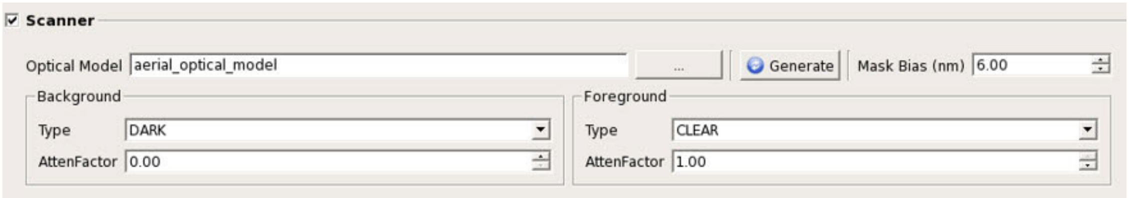
Setting Scanner Parameters

You can add scanner model parameters to the Mask Pattern Classification tool if simulation of high resolution defects is required. Scanner model parameters apply only to high resolution inspections.

Procedure

1. In the **Parameters** tab of Calibre DefectClassify, click the **Scanner** radio button.
2. Set the scanner parameters.

Figure 2-21. Scanner Parameters



- **Optical Model** — Specify the path where the generated optical model of the scanner is kept. Click **Generate** to create a new model (full details are in described in "Generating Optical Models for Mask Pattern Classification" on page 46).
- **Mask Bias** — Specify the mask bias (in nm) to be applied to mask polygons prior to forward simulation. The default value is 0. If the mask is inspected after chrome removal, the recommended value is 0.
- **Background** — Specify the background transmission of the mask.
    - Type — Specify the type of mask background area. It can be Dark, Atten, or Clear. For a Binary masks, the Type is Dark and the AttenFactor is ignored. For other masks, the Type is either Dark or Atten based on the background material of the mask, such as Chrome or MoSi.
    - AttenFactor — Specify the attenuation factor of the background region. It ranges from 0.00 to 1.00. For example, for a 6% attenuation, the value 0.06 should be specified.
- **Foreground** — Specify the foreground value. This must be set to CLEAR and AttenFactor = 1.

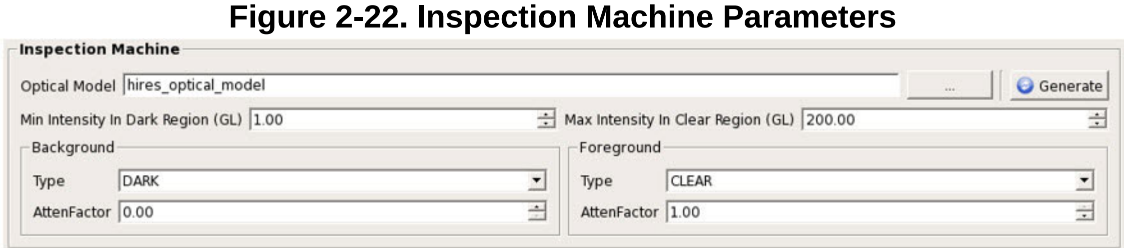
Setting Inspection Machine Parameters

You can add optical model parameters to the Mask Pattern Classification tool if simulation of high resolution defects is required. Optical model parameters apply only to high resolution inspections.

Procedure

In the Inspection Machine pane of the **Parameters** tab in Calibre DefectClassify, specify the following parameters:

Figure 2-22. Inspection Machine Parameters



- **Optical Model** — Specify the path where the generated optical model of the mask inspection machine is kept. Click **Generate** to create a new model (full details are described in "Generating Optical Models for Mask Pattern Classification" on page 46).
- **Min Intensity In Dark Region** — Specify the minimum intensity of an image in the darkest region of the mask captured by the inspection machine. The default value is 1.
- **Max Intensity In Clear Region** — Specify the maximum intensity of an image in the brightest region of the mask captured by the inspection machine. The default value is 200.
- **Background** — Specify the background transmission of the mask.
    - Type — Specify the type of mask background area. It can be Dark, Atten, or Clear. For a Binary masks, the Type is Dark and the AttenFactor is ignored. For other masks, the type is Dark when the mask is partially developed. For example, if the mask is an attenuated PSM, then after the first write, the chrome is not etched off; therefore, the background is Dark rather than Atten. On a fully written mask, the Type is either Dark or Atten based on the background material, such as Chrome or MoSi.
    - AttenFactor — Specify the attenuation factor of the background region. It ranges from 0.00 to 1.00. For example, for a 6% attenuation, the value 0.06 should be specified.
- **Foreground** — Specify parameters related to the mask foreground region. This must be set to CLEAR and AttenFactor set to 1.

Generating Optical Models for Mask Pattern Classification

You can optionally generate a new optical model that is used by the Mask Pattern Classification tool.

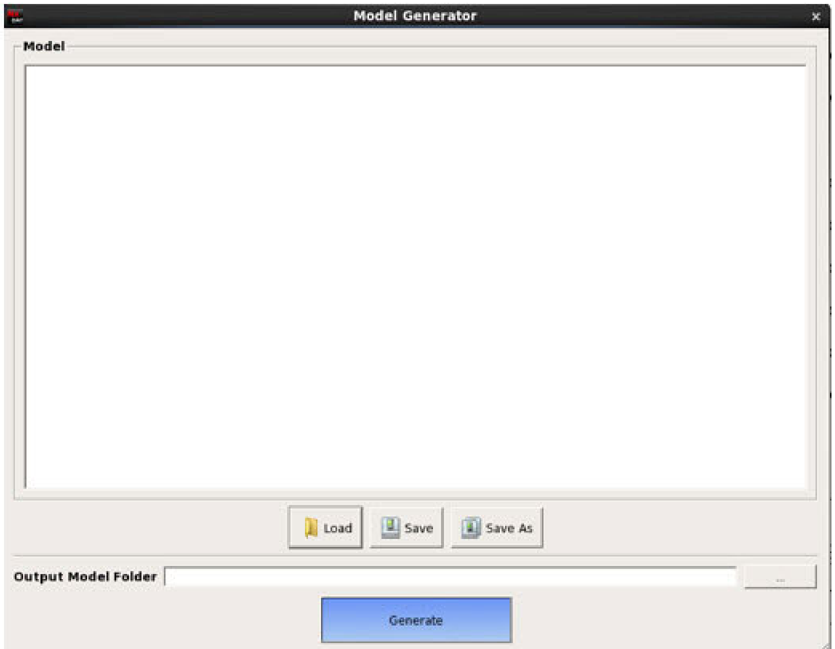
Prerequisites

- You performed the steps specified in "Setting Scanner Parameters" on page 44 or "Setting Inspection Machine Parameters" on page 45.

Procedure

1. In the Scanner pane of the **Parameters** tab in Calibre DefectClassify, click the **Generate** button. A Model Generator dialog box appears.

Figure 2-23. Model Generator



2. In the Model Generator dialog box, you can either enter the model text directory, or load a preexisting model.

3. Click **Generate** on the Model Generator dialog box to generate the model files. The generated model is kept in the path specified in Output Model Folder.

Filtering False Defects

Performing aerial simulation for false defects is optional during a Mask Pattern Classification run.

Procedure

In the Miscellaneous pane of the Calibre DefectClassify window, click **Filter out False Defects before simulation**.

Note

This option is effective only if Inspection mode is set to High Resolution Mode and the **Scanner** option is enabled.

Figure 2-24. Filter False Defects

If this option is enabled, Calibre DefectClassify first determines whether the defect classification is Real or False and performs aerial simulation only if the defect is Real. If this option is not enabled, Calibre DefectClassify runs aerial simulation for all defects. The list of False defect categories are specified under the FalseClassifications node for their respective inspection types in the *dat-ini.xml* file for Calibre DefectReview.

Copying Calibre DefectClassify Codes

After Calibre DefectClassify completes its run, you can copy the classification codes to their respective manual classification columns in the Defect List of Calibre DefectReview.

Procedure

In the Miscellaneous pane of the Calibre DefectClassify window, click **Copy MDPPattern Classifications**.

Figure 2-25. Copy MDPPattern Classifications

This option copies the generated classification codes to Calibre DefectReview into mapped columns in the Defect List. Refer to the *Calibre DefectReview User's Manual* for further information.

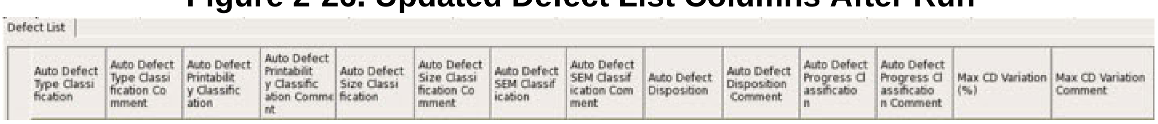
Mask Pattern Classification Results

After running the Mask Pattern Classification 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 DefectClassify run. The following figure illustrates all the columns updated.

Figure 2-26. Updated Defect List Columns After Run



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

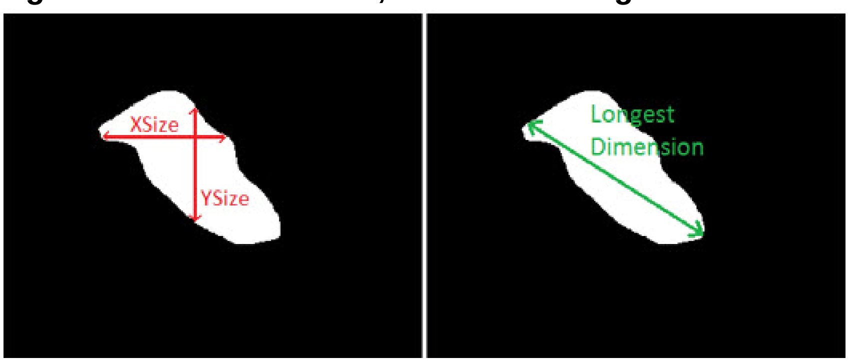
Table 2-1. Defect List Column Updates

Column	Description
Auto Defect Classification	Contains the defect type classification codes for each of defect. The section "Defect Type Classification" on page 20 provides details on the default defect types identified by the tool.
Auto Defect Classification Comment	Contains additional information extracted during classification step. It includes the list of multiple types of defects, if any, present in the defect image, suggestions on potential defects, number of edges affected, and so on.
Auto Defect Printability Classification	Contains the printability classification codes for each defect. The section "Printability Classification" on page 31 provides details on the default printability categories assigned that are used by the tool.
Auto Defect Printability Classification Comment	Contains printability classification-related comments. It includes a comment if the defect is a false defect; if not, it shows the maximum CD variation the location, and angle of measurement where the maximum CD is encountered.
Auto Defect Size Classification	Contains size classification code for each of the defects. The section "Defect Size Classification" on page 32 provides details on the default size categories assigned by the tool.
Auto Defect Size Classification Comment	Contains size classification related comments. It includes the size of defects measured along different orientations: XSize, YSize, and Longest Dimension (see Figure 2-27 for an illustration). This also reports the coordinates of the center of mass of the contour (for which the measurements were performed).
Auto Defect SEM Classification	Contains the SEM classification codes for each of defect. Currently, no categories have been defined; the default is "Review".
Auto Defect SEM Classification Comment	Contains comments that accompany SEM classification (will appear when SEM classification is supported).
Auto Defect Disposition	Contains the disposition of each defect. The section "Defect Disposition" on page 33 provides details on the default dispositions suggested by the tool.

Table 2-1. Defect List Column Updates (cont.)

Column	Description
Auto Defect Disposition Comment	Contains any associated comment with a defect disposition. Currently, false defects are listed, if found.
Auto Defect Progress Classification	Contains progress classifications for each of the defects. The Defect Progress Classification description in "Mask Pattern Classification Key Concepts" on page 16 provides details on the default defect progress categories assigned by the tool.
Auto Defect Progress Classification Comment	Contains defect progress classification-related comments. Currently, the comment lists the change to defect size, transmitted and reflected residue, and transmittance of the current defect with respect to the base inspection, if applicable. This comment also lists information on false defects, if found.
Max CD Variation (%)	Contains the maximum CD variation value encountered over the region of interest (ROI) in the defect image. ROI is specified using the parameter printabilityROI in the *adc-ini.xml* configuration file.
Max CD Variation Comment	Contains details of maximum clear and dark variations, if any, measured over the ROI in the image, along with respective locations and angles of measurement.

Figure 2-27. Defect XSize, YSize and Longest Dimension

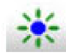


Save and Load Classification Results

As described in "Using Mask Pattern Classification in CLI Mode" on page 37, once classification is complete, results can be saved into an inspection file using the saveas command. File formats currently supported are XML and KLARF. These results include all of the values under the fields described in Table 2-1. For further information regarding saving KLARF files, refer to the *Calibre DefectReview User's Manual*.

The saved defect file can then be reloaded in the Calibre DefectReview GUI along with all the saved results for further analysis. Making another Calibre DefectClassify run using the GUI overwrites any previously displayed classifications, either loaded from defect file or from previous runs.

Display Measurement

After a Calibre DefectClassify run, the Display Measurement window is updated. Clicking the Display Measurement icon  overlays the Display Measurement window on the defect image in the Image Measurement Unit (IMU).

Double-clicking the Display Measurement window opens a small window that lists defect properties such as the accurate location (coordinates) as well as CD variation related values such as CD variation, CD type (for example, dark and clear), and CD measurement angle.

Calibre DefectClassify Classification Table Name Mapping

Calibre DefectClassify uses default generic names for the support of various classification tiers. You can choose to map these generic names to specific names for each of the classification tiers. This mapping is defined in the configuration file *adc-ini.xml*, which can be found in *$MGC_HOME/pkgs/nxdat.aoi/pvt/*.

A secondary table associates each user-defined name with a short code and display name to be used in the Calibre DefectReview Defect List. This secondary mapping of user-defined names to a code and description is defined in the Calibre DefectReview configuration file, *dat-ini.xml*, which can be found at *$MGC_HOME/pkgs/nxdat.aoi/pvt/*.

For KLARF format (.klarf), the *dat-ini.xml* file is the configuration file to be referred to. This secondary table has already been added under the klarfConfigParams node in the *dat-ini.xml* file.

Mapping Between Calibre DefectClassify and Calibre DefectReview Configuration Files

The following is an example of a Calibre DefectClassify first tier classification table in an *adc-ini.xml* file.

Figure 2-28. Calibre DefectClassify (adc-ini.xml) First Tier Classification Table

```
<AutoDefectClassifyMapTable tableName="Auto Defect Type Classification Map Table" datIniTableName="Defect Type Classification"
defaultDefectType="Review">
  <ClassificationMapData toolDefectType="IsolatedDark"          defectType="IsolatedDark"/>
  <ClassificationMapData toolDefectType="IsolatedDarkLowTrans"  defectType="IsolatedDarkLowTrans"/>
  <ClassificationMapData toolDefectType="OnEdgeDark"            defectType="OnEdgeDark"/>
  <ClassificationMapData toolDefectType="MultipleEdgesDark"     defectType="MultipleEdgesDark"/>
  <ClassificationMapData toolDefectType="Contamination"         defectType="Contamination"/>
  <ClassificationMapData toolDefectType="OnDarkEdgeContamination"  defectType="OnDarkEdgeContamination"/>
  <ClassificationMapData toolDefectType="OnClearEdgeContamination" defectType="OnClearEdgeContamination"/>
  <ClassificationMapData toolDefectType="SRAFDark"              defectType="SRAFDark"/>
  <ClassificationMapData toolDefectType="FillDark"              defectType="FillDark"/>
  <ClassificationMapData toolDefectType="IsolatedClear"         defectType="IsolatedClear"/>
  <ClassificationMapData toolDefectType="IsolatedClearLowTrans" defectType="IsolatedClearLowTrans"/>
  <ClassificationMapData toolDefectType="OnEdgeClear"           defectType="OnEdgeClear"/>
  <ClassificationMapData toolDefectType="MultipleEdgesClear"    defectType="MultipleEdgesClear"/>
  <ClassificationMapData toolDefectType="MissingChrome"         defectType="MissingChrome"/>
  <ClassificationMapData toolDefectType="SRAFClear"             defectType="SRAFClear"/>
  <ClassificationMapData toolDefectType="FillClear"             defectType="FillClear"/>
  <ClassificationMapData toolDefectType="LargeDefect"           defectType="LargeDefect"/>
  <ClassificationMapData toolDefectType="RepairedMark"          defectType="RepairedMark"/>
  <ClassificationMapData toolDefectType="ClearOpaque"           defectType="ClearOpaque"/>
  <ClassificationMapData toolDefectType="Review"                defectType="Review"/>
  <ClassificationMapData toolDefectType="Unclassified"          defectType="Unclassified"/>
  <ClassificationMapData toolDefectType="RealNuisance"          defectType="RealNuisance"/>
  <ClassificationMapData toolDefectType="Nuisance"              defectType="Nuisance"/>
  <ClassificationMapData toolDefectType="Tap"                   defectType="Tap"/>
  <ClassificationMapData toolDefectType="EdgeRoughness"         defectType="EdgeRoughness"/>
  <ClassificationMapData toolDefectType="Focus"                 defectType="Focus"/>
  <ClassificationMapData toolDefectType="Alignment"             defectType="Alignment"/>
  <ClassificationMapData toolDefectType="BrightSpot"            defectType="BrightSpot"/>
</AutoDefectClassifyMapTable>
```

The value of the field datIniTableName in the first line of the AutoDefectMapTable node points to a corresponding table in the *dat-ini.xml* file. In the Calibre DefectClassify configuration file (*adc-ini.xml*), under each tier's mapping tables, the left column toolDefectType holds defect types identified by Calibre DefectClassify, and the defectType column maps Calibre DefectClassify's names to user-defined names. The values of the left column cannot be modified. However, entire rows can be deleted for which the mapping then takes the default value (Review) in this case.

The following code fragment from the Calibre DefectReview configuration file (*dat-ini.xml*), shows a Calibre DefectClassify first tier secondary classification table.

Figure 2-29. Calibre DefectReview (dat-ini.xml) Secondary Classification Table

```
<classificationTable tableName="Defect Type Classification" classCodeUnion="false" higherPriorityToDatIni="true">
    <classificationData defectName="IsolatedDark"        defectType="IsolatedDark"        defectCode="ISD"/>
    <classificationData defectName="OnEdgeDark"          defectType="OnEdgeDark"          defectCode="OED"/>
    <classificationData defectName="MultipleEdgesDark"   defectType="MultipleEdgesDark"   defectCode="MED"/>
    <classificationData defectName="Contamination"       defectType="Contamination"       defectCode="CON"/>
    <classificationData defectName="SRAFDark"            defectType="SRAFDark"            defectCode="SRD"/>
    <classificationData defectName="FillDark"            defectType="FillDark"            defectCode="FLD"/>
    <classificationData defectName="IsolatedClear"       defectType="IsolatedClear"       defectCode="ISC"/>
    <classificationData defectName="OnEdgeClear"         defectType="OnEdgeClear"         defectCode="OEC"/>
    <classificationData defectName="MultipleEdgesClear"  defectType="MultipleEdgesClear"  defectCode="MEC"/>
    <classificationData defectName="MissingChrome"       defectType="MissingChrome"       defectCode="MCR"/>
    <classificationData defectName="SRAFClear"           defectType="SRAFClear"           defectCode="SRC"/>
    <classificationData defectName="FillClear"           defectType="FillClear"           defectCode="FLC"/>
    <classificationData defectName="LargeDefect"         defectType="LargeDefect"         defectCode="LRG"/>
    <classificationData defectName="ClearOpaque"         defectType="ClearOpaque"         defectCode="CLO"/>
    <classificationData defectName="Review"              defectType="Review"              defectCode="RVW"/>
    <classificationData defectName="Unclassified"        defectType="Unclassified"        defectCode="UNC"/>
    <classificationData defectName="Nuisance"            defectType="Nuisance"            defectCode="NSC"/>
    <classificationData defectName="Tap"                 defectType="Tap"                 defectCode="TAP"/>
    <classificationData defectName="EdgeRoughness"       defectType="EdgeRoughness"       defectCode="EDR"/>
    <classificationData defectName="Focus"               defectType="Focus"               defectCode="FOC"/>
    <classificationData defectName="Alignment"           defectType="Alignment"           defectCode="ALN"/>
    <classificationData defectName="BrightSpot"          defectType="BrightSpot"          defectCode="BSP"/>
</classificationTable>
```

The corresponding table "Defect Type Classification" in the *dat-ini.xml* file contains three fields:

- defectName contains user-defined names listed under the defectType column of the *adc-ini.xml* file.
- defectType should currently be the same as the corresponding defectName.

- defectCode contains codes for classification categories to be shown on buttons in Defect Classification widget.

The following table maps each of the classification tier names with secondary classification table names in Calibre DefectClassify configuration file (*adc-ini.xml*) and Calibre DefectReview configuration files (*dat-ini.xml*).

Table 2-2. Classification Table Name Mapping

Classification Tier	Calibre DefectClassify Configuration File Mapping Table Name	Calibre DefectReview Configuration File Secondary Classification Table Name
Defect Type Classification	Auto Defect Classification Map Table	Defect Type Classification
Printability Classification	Auto Defect Printability Classification Map Table	Defect Printability Classification
Size Classification	Auto Defect Size Classification Map Table	Defect Size Classification
SEM Classification	Auto Defect SEM Classification Map Table	SEM Classification
Defect Disposition	Auto Defect Disposition Classification Map Table	Defect Disposition
Defect Progress Classification	Auto Defect Progress Classification Map Table	Defect Progress Classification

Mask Pattern Classification Command and File Reference

Calibre DefectClassify utilizes several different commands and file types to perform operations.

flipsource

Calibre DefectClassify Tcl command

Flips the input source map before using it for optical model generation.

Usage

flipsource --in *input_source_map* --out *output_source_map* --mirrorx [**0** | **1**] **--mirrory** [**0** | **1**]

Arguments

- **--in *input_source_map***

  A required argument that specifies the input source map to be flipped.

- **--out *output_source_map***

  A required argument that specifies the output source map, the flipped version of the input source map.

- **--mirrorx** [**0** | **1**]

  A required argument that, if set to 1, flips the image of the input source map along the x-axis.

- **--mirrory** [**0** | **1**]

  A required argument that, if set to 1, flips the image of the input source map along the y-axis.

Description

The flipsource Tcl command reads the input source map and flips it around x-axis or y-axis (or both) depending on whether mirrorx or mirrory (or both) are set to 1, respectively. It then generates the output source map, which is the flipped version of input source map. If you want to mirror around the x-axis, the mirrorx option should be set to 1 and mirror be set to 0. To mirror around the y-axis, set mirrory to 1 and mirrorx to 0. This command is used along with the following other commands in a Calibre DefectClassify Tcl configuration file:

- patternautoclassify — Performs automatic defect classification of patterned mask defects.
- opticsgen — Generates optical models from optical conditions. This command is identical to the opticsgen command in Calibre WORKbench. See the *Calibre WORKbench User's and Reference Manual* for full information.
- saveas — Saves the input defect file along with classification results into a new file. This is a Calibre DefectReview command. Refer to the *Calibre DefectReview User's Manual* for information.

This command typically used only if optical model generation is performed using a source map file, which is generally the output of Source Mask Optimization performed for a particular node mask. The full syntax to generate optical models using a source map file is further explained in the *Calibre WORKbench User's and Reference Manual*.

Note

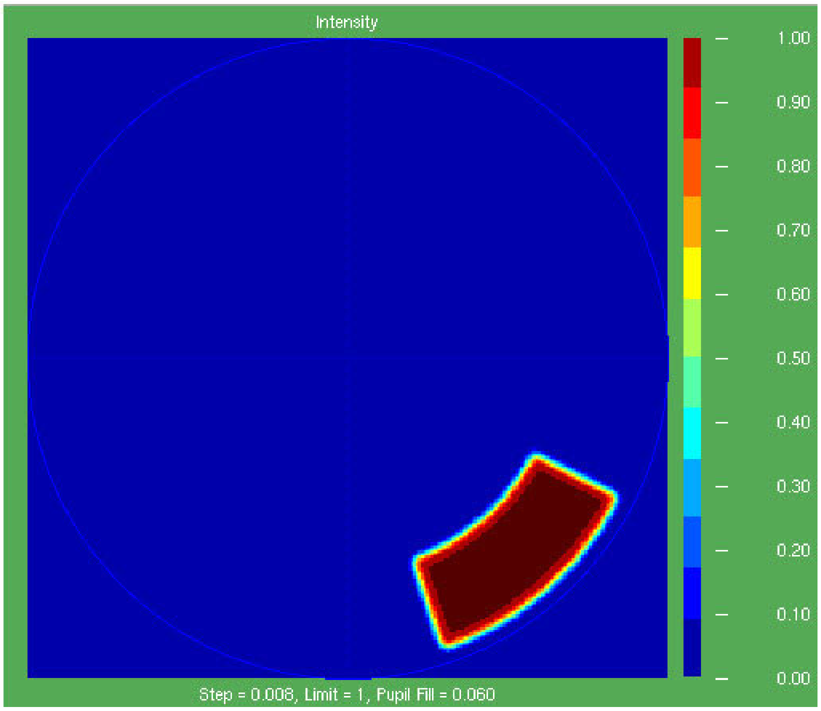
Flipping the optical source map is supported only in the CLI mode of Calibre DefectReview.

Examples

The following illustrates a flipsource example, an input source map, and the resulting flipped output map:

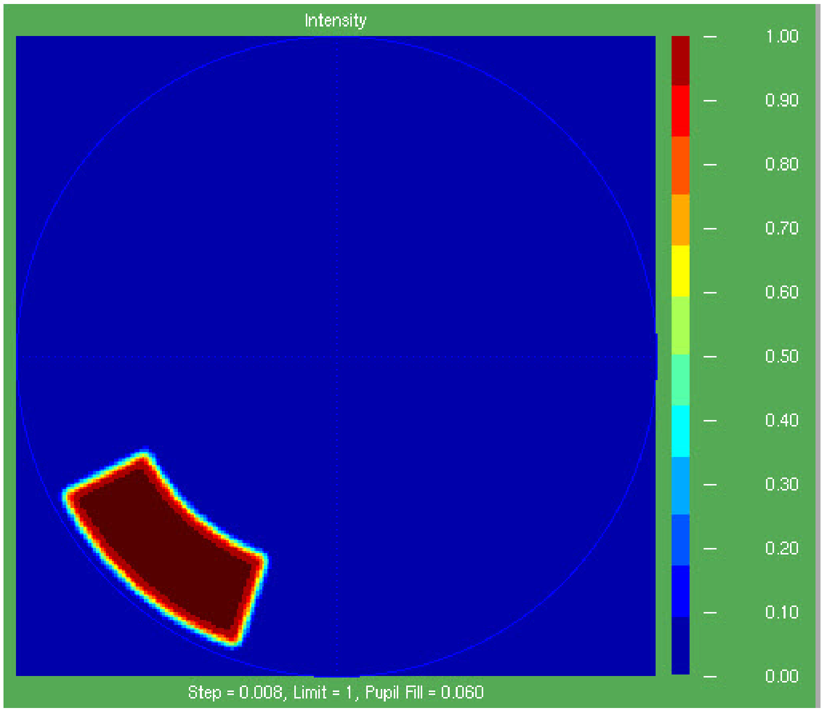
```
flipsource --in input_source_map --out output_source_map --mirrorx 0 --
mirrory 1
```

Figure 2-30. Input Source Map



The output map is flipped along the y-axis as shown in the following figure.

Figure 2-31. Output Source Map



patternautoclassify

Calibre DefectClassify Tcl command

Performs automatic defect classification of patterned mask defects.

Usage

There are three different invocation modes.

patternautoclassify -p *parameterfile*

patternautoclassify -i *inspection_id* -f *reticle_info_file* [-u]

patternautoclassify -i *inspection_id* -m *mask_id* [-u]

Arguments

- **-p *parameterfile***

  All inputs are read from a user-specified XML file containing arguments to configure Calibre DefectClassify for automatic defect classification of patterned mask defects. Refer to "patternautoclassify Parameter File" on page 60 for complete information.

- **-i *inspection_id* -f *reticle_info_file*** [-u]

  All inputs are read from a user-specified Reticle Information File (reticle_info_file). As the RIF may contain information from multiple inspections of a reticle, the inspection_id input clarifies which is to be used. The optional -u keyword updates the RIF with Calibre DefectClassify outputs such as print threshold (DCPrintabilityThreshold) used in the run.

- **-i *inspection_id* -m *mask_id*** [-u]

  All inputs are read from the defect database. The defect database should have already been populated using the savereticleinfotodb command in Calibre DefectReview. The mask_id is used to identify the entry in the defect database to be used for this run. As the information corresponding to the mask_id may contain information from more than one inspection, the inspection_id clarifies which to use. The optional -u keyword updates the defect database with Calibre DefectClassify outputs such as print threshold (DCPrintabilityThreshold) used in the run.

Description

The patternautoclassify command is one of the main Tcl commands used by Calibre DefectClassify to perform automatic defect classification of patterned mask defects. This command is used in conjunction with the following commands to configure a Calibre DefectClassify run:

- flipsource — This command flips the input source map in x or y directions (or both). The output source map then can be used to generate an optical model using the opticsgen command.

- opticsgen — Generates optical models from optical conditions. This command is identical to the opticsgen command in Calibre WORKbench. See the *Calibre WORKbench User's and Reference Manual* for complete information.
- saveas — Saves the input defect file along with classification results into a new file. In the following example:

  ```
  saveas classified 1 ./
  ```

  If the input file is *test.def*, the output name is automatically determined as *classified_test.def* and saved in the current working directory. This is a Calibre DefectReview command. Refer to the *Calibre DefectReview User's Manual* for information.

Refer to "Using Mask Pattern Classification in CLI Mode" on page 37 for information.

Examples

The following is an example of patternautoclassify in a sample Tcl file for a highres inspection.

```
flipsource --in input.src --out output.src --mirrorx 0 --mirrory 1
opticsgen hrsmodel.txt highres_model
opticsgen scannermodel.txt scanner_model
patternautoclassify cli_params.xml
saveas classified 1 ./
```

patternautoclassify Parameter File

Configuration file used by the patternautoclassify command

The patternautoclassify command uses a single XML format file to specify parameters that are provided as arguments to the Tcl-based command.

Format

The parameter file is in XML format and the functional parameters are categorized into sections with further customizable parameters as sub-nodes. All distance units in the parameter file are in nanometers (nm).

Parameters

- version

  Specifies the version of the XML file that is supported by Calibre DefectClassify. The value must be 1.0.

- IsHighResInspection

  Specifies if the inspection mode is aerial/reflected or highres. Supported values are true or false.

- PlateOrientation *rotation*

  Specifies any rotation to be applied to the scanner model and layout data to align with mask data. The values can be 0, 90, 180, or 270.

- FilterFalseDefectsBeforeSimulation

  An optional node that specifies whether or not aerial simulation is to be performed for False defects. Valid values are true and false. If set to false, Calibre DefectClassify performs simulation for all defects. If value is set to true, simulation is skipped for false defects. This parameter is used only if the value for the parameter IsHighResInspection is set to true and values for node SimulationParams are specified. The default value is false.

  Defect categories that are to be treated as false defects must be listed under the FalseClassifications node of respective inspection types in the *dat-ini.xml* configuration file for Calibre DefectReview.

- PrintThresholdParams

  Specifies parameters to determine the wafer-level printability threshold. The sub-node useImages specifies if a target CD image is to be used to calculate the printability threshold. Possible values for this field are true and false. The sub-nodes for useImages include:

  - UserSpecified

    If the value of the parameter useImages is “false”, then this value is used; otherwise, CDImages is used.

  - CDImages

The pixelSize of this sub-node refers to the actual size of each image pixel in nm. This parameter is optional and defaults to the value taken from the inspection file. The CDImages sub-nodes include:

- Image — This node consists of a set of data required to calculate printability threshold. Multiple such nodes can be defined performing independent calculations and their average is reported as the print threshold. The following table lists the Image sub-nodes:

Table 2-3. Image Sub-Nodes

Node	Description
File	Contains the path of the target CD image containing the feature using which printability threshold is calculated.
ImageIndex	This node must be specified only if the target CD image file (**File** is selected) is a TIF file. It specifies the index of the TIF image in the target CD image file.
DefectId	Specifies the defect ID associated with the reference image to be used as the target CD image on which the printability threshold is calculated. **Note:** File or DefectId are mutually exclusve; only one should be present in the parameter file.
TargetCD	Specifies the expected feature dimension at mask level (for example, 4X).
Polarity	Specifies the tone at the location specified above. Permitted values are “clear” and “dark”.
Angle	Specifies the angle at which cross-sectional CD measurement is performed. 0 represents vertical, with pixels considered starting from bottom along the length of cross-sectional bar; 90 represents horizontal, from left to right; and so on in a clockwise direction. Possible values are 0, 90, 180 and 270.
LocationInLayout	Specifies the location in the layout data (for example, in a MEBES job deck, extended MEBES job deck, or OASIS file) where the CD feature is present. The location must be in the center of the CD feature. The print threshold is computed from this parameter.

Table 2-3. Image Sub-Nodes (cont.)

Node	Description
LocationInImage	This parameter should be used only if the exact position of the CD feature in the layout data (specified by the LocationInLayout parameter) is unknown. Use this parameter to specify the location in the CD image where the CD feature is present. • The top left corner of the image is (0, 0). The X-coordinate value increases from the left to right of the image and the Y-coordinate value increases from the top to bottom of the image. The behavior of the LocationInLayout parameter changes when this parameter is specified. • For high resolution inspections, the print threshold is calculated from image location specified by this parameter. The LocationInLayout parameter should specify an approximate location where the CD feature is present in the layout, as the print threshold is performed on the simulated image and the layout data must be converted. • For aerial inspections, the LocationInLayout parameter is ignored.

- LayoutDataParams

  Contains parameters related to layout data. Sub-nodes include the following:

  - Type

    Contains the format(s) of layout data that are supported. This sub-node supports the following value: MEBESJOB for MEBES job deck format, MEBESEXTENDED for extended MEBES job deck format, or OASIS for OASIS format.

  - File

    Contains the path of the job deck file associated with the inspection on which Calibre DefectClassify is run.

  - Layers

    Can contain multiple sub-nodes, each named “Layer”, corresponding to each of the layers in layout data (for example, smashed layer, Dark SRAF layer, Clear Fill layer, and so on).

    - Layer — Contains parameters specific to each layer, including the following:

      Name — Specifies the layer name. The name can take values from amongst the following: “SMASHED” (composite of all layers), “DARK_SRAF” (contains dark SRAF features), “CLEAR_SRAF” (contains clear SRAF features), “DARK_FILL” (contains dark fill regions), “CLEAR_FILL” (contains clear fill regions), and “CHROME” (contains chrome areas).

LayerDataType — Specifies the layer number and datatype assigned to the data layer. The format is *layer*:*datatype* (for example, 1:1). The datatype is an optional input; the layer number is required as input.

CorrelationParams — Corresponds to the **Correlation**” tab in the Calibre DefectClassify dialog box. It contains parameters required to correlate layout data with mask data. There are several sub-nodes to CorrelationParams that are summarized in the following table.

Table 2-4. CorrelationParams Nodes

Node	Description
Rotation	Represents any rotation to be applied to layout data to align with mask data. The values can be 0, 90, 280 or 270.
Magnification	Represents the magnification level of layout data. Currently, this is 1 and cannot be changed.
Reflection	Represents any optional mirroring to be applied to layout data. Possible values are “NO_MIRROR”, “X_MIRROR”, and “Y_MIRROR”. The default is “NO_MIRROR”.
Invert	Inverts the layout tones to synchronize with mask data. Possible values are “true” and “false”.
Translate	Specifies alignment mismatch between layout and mask data.
Origin	Specifies that x- and y-coordinates currently remain at (0,0).

- SimulationParams

  Contains parameters related to aerial simulation (for example, highres and scanner optical conditions, mask transmittance, and so on).

  - InspectionMachineParams

    Contains parameters related to mask pattern recovery (inverse lithography).

    - OpticalModelDirectory

      Specifies the path where the highres optical model is present.

    - Background

      Contains parameters related to mask background region.

- Type — Specifies the type of mask background area. Values are “Dark”, “Atten”, or “Clear”.
- AttenFactor — Specifies the attenuation factor of the background region. It ranges from 0.00 to 1.00.

- Foreground

  Contains parameters related to mask foreground region.

  - Type — Specifies the type of mask background area. It can be “Dark”, “Atten”, or “Clear”.
  - AttenFactor — Specifies the attenuation factor of the background region. It ranges from 0.00 to 1.00.

- IntensityRange

  Specifies the pixel intensity range captured by the inspection machine.

  - Min — Specifies the minimum intensity of image in darkest region of the mask captured by inspection machine. The default value is 1.
  - Max — Specifies the maximum intensity of image in brightest region of the mask captured by inspection machine. The default value is 200.

- ScannerParams

  Contains parameters related to forward aerial simulation.

  - OpticalModelDirectory

    Specifies the path from where scanner optical model is present.

  - Background

    This node contains parameters related to mask background region.

    - Type — Specifies the type of mask background area. It can be “Dark”, “Atten”, or “Clear”.
    - AttenFactor — Specifies the attenuation factor of the background region. It ranges from 0.00 to 1.00.

  - Foreground

    This node contains parameters related to mask foreground region.

    - Type — Specifies the type of mask background area. It can be “Dark”, “Atten”, or “Clear”.
    - AttenFactor — Specifies the attenuation factor of the background region. It ranges from 0.00 to 1.00.

  - MaskBias

Specifies the mask bias (in nm) to be applied to mask polygons prior to forward simulation. The default value is 0. If the mask is inspected after chrome removal, the recommended value is 0.

Inspection ID

The RIF file can contain information for more than one inspection. The corresponding inspection IDs are specified against the node InspectionUniqueId of the RIF file. This input from the user specifies which of the Inspection IDs should be used as input to the tool.

Reticle Information File

The Reticle Information File (RIF) contains all information related to a reticle or mask, multiple inspections of the reticle, job deck, and CD information related to the reticle. Refer to the *Calibre DefectReview User's Manual* for details on the RIF.

Mask ID

The defect database contains RIF information stored in it using the Calibre DefectReview savereticleinfotodb command (refer to the *Calibre DefectReview User's Manual* for details). RIF information corresponding to a reticle is uniquely identified by a mask ID (MaskUniqueId). With this input, you can specify which RIF information (entry) from the defect database should be used for the current run of the tool.

Mapping of the RIF and Parameter File Inputs

The RIF file contains most of the inputs required for a Calibre DefectClassify run and several additional inputs for use by other tools such as Calibre DefectReview. The following table describes the mapping between entries in the RIF and the patternautoclassify parameter file. Certain parameters not present in the RIF that are collected from other locations are described in the Other column.

Table 2-5. Parameter File and RIF Mapping

patternautoclassify Parameter File	Reticle Information File Entry	Other
IsHighResInspection	InspectionMode = DIE_TO_DIE_REFLECTED or DIE_TO_MODEL or DIE_TO_DIE_HIRES or STARLIGHT or DIE_TO_DIE_WITH_STARLIGHT then true InspectionMode = DIE_TO_DIE_AERIAL or SINGLE_DIE_CTM or INTENCD or DIE_TO_DIE_AERIAL_WITH_CTM then false	Not applicable
PlateOrientation	PlateOrientation	Not applicable

Table 2-5. Parameter File and RIF Mapping (cont.)

patternautoclassify Parameter File	Reticle Information File Entry	Other
FilterFalseDefectsBeforeSimulation	FilterFalseDefectsBeforeSimulation	Not applicable
PrintThresholdParams		
useImages	Not applicable	Always set to true
UserSpecified	DCPrintabilityThreshold	Not applicable
CDImages		
pixelSize	Not applicable	Pixel size is set to the pixel size of defect images associated with inspection report (defect file)
File	TargetCdImageFilename If the file is a TIF file, then the image is extracted from the TIF file, stored as a temporary file, and this file name is used.	Not applicable
TargetCD	CdMaskTarget Entry with CdStructure= “CD” is used.	Not applicable
Polarity	CdPolarity Entry with CdStructure= “CD” is used.	Not applicable
Angle	CdAngle (X = 90 and Y = 0) Entry with CdStructure= “CD” is used.	Not applicable
LocationInLayout to X, Y	CdLocX, CdLocY Entry with CdStructure= “CD” is used.	Not applicable

Table 2-5. Parameter File and RIF Mapping (cont.)

patternautoclassify Parameter File	Reticle Information File Entry	Other
LocationInImage	Not applicable	Not set or used.
LayoutDataParams		
Type	JobdeckType	Not applicable
File	JobdeckFilename	Not applicable
Layers		
Name	Not applicable	Picked from *adc-ini.xml* **AutoDefectClassifyAuxParams** > **LayerMapping**
LayerDataType	Not applicable	
CorrelationParams		
Rotation	(JobdeckRotation + PlateOrientation) % 360 degrees	Not applicable
Magnification	JobdeckMagnification	Not applicable
Reflection	JobdeckReflection	Not applicable
Invert	ReticleResistTone = “N” then true else false.	Not applicable
Translate to X, Y	JobdeckXTranslation, JobdeckYTranslation	Not applicable
Origin to X, Y	JobdeckXOrigin, JobdeckYOrigin	Not applicable
SimulationParams to InspectionMachineParams		
OpticalModelDirectory	InspectionModelFilename	Not applicable

Table 2-5. Parameter File and RIF Mapping (cont.)

patternautoclassify Parameter File	Reticle Information File Entry	Other
Background to (Type, AttenFactor)	MaskType = BIN then (DARK, 0) MaskType = OMOG then (ATTEN, TargetMosiTransmittance) (MaskType = ATTPSM or ALTPSM or HIT or EUV) and (InspectionStep = PATTERN_AFTER_FIRST_WRITE) then (DARK, 0) (MaskType = ATTPSM or ALTPSM or HIT or EUV) and (InspectionStep ? PATTERN_AFTER_FIRST_WRITE) then (ATTEN, TargetMosiTransmittance)	Not applicable
Foreground to Type, AttenFactor	Not applicable	CLEAR, 1
IntensityRange to (Min, Max)	InspectionMinGl, InspectionMaxGl	Not applicable
SimulationParams to ScannerParams		
OpticalModelDirectory	ScannerModelFilename	Not applicable
Background to (Type, AttenFactor)	MaskType = BIN then (DARK, 0)MaskType = OMOG or ATTPSM or ALTPSM or HIT then (ATTEN, TargetMosiTransmittance)	Not applicable
Foreground to (Type, AttenFactor)	Not applicable	CLEAR, 1
MaskBias	ScannerMaskBias	Not applicable

Examples

The following is an example XML parameter file.

```
<!-- All units are in nanometers unless otherwise specified -->
<PatternAutoClassifyParams version="1.0">
  <!-- true | false : If true then SimulationParams should be -->
  <!-- specified else should not be specified. -->
  <IsHighResInspection>true</IsHighResInspection>
  <!-- Value could be 0, 90 , 180, 270  -->
  <PlateOrientation>0</PlateOrientation>
  <!-- true | false : If true then aerial simulation is -->
  <!-- skipped for False defects. -->
  <FilterFalseDefectsBeforeSimulation>false
    </FilterFalseDefectsBeforeSimulation>
  <!-- true | false : If false then UserSpecified print threshold -->
  <!-- will be used. -->
  <PrintThresholdParams useImages="true">
    <!-- Value (GL) between 0 and 255 -->
    <UserSpecified>40</UserSpecified>
    <CDImages pixelSize="83.4">
      <Image>
        <File>/dummy/cdimg.png</File>
   <ImageIndex>-1<ImageIndex>
        <!-- Defect id can also be used for print threshold calculation -->
        <!-- <DefectId>1</DefectId> -->
        <!-- The target is at mask level (like 4X and not wafer level) -->
        <TargetCD>1560</TargetCD>
        <!-- clear | dark -->
        <Polarity>clear</Polarity>
        <!-- Angle could be 0 = ^, 90 = ->, 180 = v, 270 = <- direction -->
        <Angle>0</Angle>
        <LocationInImage>
          <X>30</X>
          <Y>25</Y>
        </LocationInImage>
        <LocationInLayout>
          <X>109995520</X>
          <Y>39815690</Y>
        </LocationInLayout>
      </Image>
    </CDImages>
  </PrintThresholdParams>
  <LayoutDataParams>
    <!-- MEBESJOB, MEBESEXTENDED, OASIS are supported-->
    <Type>MEBESJOB</Type> <!-- or OASIS -->
    <File>/dummy/dummy.jb</File>
    <Layers>
      <Layer>
        <!-- SMASHED, DARK_SRAF, CLEAR_SRAF, DARK_FILL, -->
        <!-- CLEAR_FILL,CHROME -->
        <Name>SMASHED</Name>
        <LayerDataType>1</LayerDataType> <!-- datatype is optional and -->
        <!-- can be added in the format layer:datatype. For example 1:1 -->
     <CorrelationParams>
          <!-- 0, 90, 180, 270 -->
          <Rotation>0</Rotation>
          <Magnification>1</Magnification>
          <!-- NO_MIRROR, X_MIRROR, Y_MIRROR -->
          <Reflection>NO_MIRROR</Reflection>
          <!-- true, false -->
          <Invert>true</Invert>
```

```
        <Translate>
          <X>0</X>
          <Y>0</Y>
        </Translate>
        <Origin>
          <X>0</X>
          <Y>0</Y>
        </Origin>
      </CorrelationParams>
    </Layer>
    <Layer>
      <Name>DARK_SRAF</Name>
      <LayerDataType>3</LayerDataType>
      <CorrelationParams>
        <Rotation>0</Rotation>
        <Magnification>1</Magnification>
        <Reflection>NO_MIRROR</Reflection>
        <Invert>true</Invert>
        <Translate>
          <X>0</X>
          <Y>0</Y>
        </Translate>
        <Origin>
          <X>0</X>
          <Y>0</Y>
        </Origin>
      </CorrelationParams>
    </Layer>
  </Layers>
</LayoutDataParams>
<SimulationParams>
  <InspectionMachineParams>
    <OpticalModelDirectory>/highres_model/</OpticalModelDirectory>
      <Background>
       <!-- dark, clear, atten -->
       <Type>DARK</Type>
       <!-- real number between 0 to 1 -->
       <AttenFactor>0</AttenFactor>
     </Background>
     <Foreground>
       <Type>CLEAR</Type>
       <AttenFactor>1</AttenFactor>
     </Foreground>
     <IntensityRange>
       <!-- GL value between 0 to 255 -->
       <Min>1</Min>
       <Max>200</Max>
     </IntensityRange>
  </InspectionMachineParams>
  <ScannerParams>
      <OpticalModelDirectory>/scanner_model/</OpticalModelDirectory>
    <Background>
      <Type>ATTEN</Type>
      <AttenFactor>0.06</AttenFactor>
    </Background>
    <Foreground>
      <Type>CLEAR</Type>
      <AttenFactor>1</AttenFactor>
```

```
      </Foreground>
      <MaskBias>0</MaskBias>
    </ScannerParams>
  </SimulationParams>
</PatternAutoClassifyParams>
```

Mask Pattern Classification Configuration File

XML file used to configure Calibre DefectClassify

The Calibre DefectClassify Mask Pattern Classification tool configuration file contains additional parameters that can be altered as required.

This configurable file, named *adc-ini.xml*, can be found in the folder *$MGC_HOME/pkgs/nxdat.aoi/pvt/* on Linux®[1]. Once the tool has been launched for the first time, the configuration file is copied to *$HOME/.calibrenx_workspace/*. The configuration file can then be edited for each Mask Pattern Classification tool run.

Format

An XML format file named *adc-ini.xml*, similar to *dat-ini.xml* file used by Calibre DefectReview.

Parameters

Common Parameters

- parallelRun

  Activates or deactivates parallel processing of defects utilizing multiple cores. Possible values are "yes" or "no". The default value is "no".

- noOfCoresForParallelRun

  Specifies the number of cores to be used during a parallel run. Calibre DefectClassify scales well with an increase in the number of cores used. The default value is 1.

- displayOtherImages

  Activates additional images, if any, for viewing. The default value is false.

- dispositionBasedOnDefectProgressAndPrintability

  Selects between two disposition schemes: CD variation-based and defect printability and progress-based. Specifying "no" selects CD variation-based and "yes" selects defect printability and progress based. The default value is "no".

- dispositionAims

  Specifies the CD variation (%) threshold below which the disposition for the defect is considered "Acceptable" (the defect does not need further review). The default value is 10.

- dispositionSem

  Specifies the CD variation (%) threshold above which the disposition for the defect is set to "SEM Review" (the defect is preferred to undergo a SEM review). The default value is 18.

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

- alignmentResizeFactor

  Specifies the resize or interpolation factor to be used for the alignment of defect and reference images. Defect and reference images must always be aligned before proceeding with defect detection. If images have a sub pixel alignment error, they must be interpolated for proper alignment at a finer resolution. The tool uses the value specified by the parameter for interpolating defect and reference images which are then aligned. The default value is 4.

- maxAlignmentError

  Specifies the maximum alignment error between defect and reference images. The tool uses the value specified by the parameter as the maximum shift required during the alignment of the defect and reference image. The default value is 4.

- commonNuisanceThreshold

  Specifies the GL intensity threshold above which pixels are considered for processing. The default value is 3.

- commonMinimumContourSize

  Specifies the minimum contour size (in square um) above which a defect can be classified as contamination in a reflected D2D inspection. The default value is 0.015 square um.

- maxDiffIntensityToConsiderEdgeRoughness

  If the maximum difference between defect and reference images is more than the value of this parameter, then the defect cannot be classified as Edge Roughness. The default value is 0, which means that, typically, edge roughness defects are not search or they are effectively deactivated. The parameter value can be increased to activate this category.

- maxDiffIntensityToConsiderAllEdgeRoughnessAsFalse

  If the maximum difference between defect and reference images is less than the value of this parameter, then a defect with edge roughness is classified as False. If the maximum intensity difference is larger than the value specified by the parameter, then defect is considered as False or Real Nuisance depending on the defect width. The default value is 0.

- maxDefectSizeToConsiderRestEdgeRoughnessAsFalse

  Specifies a threshold for the maximum width of a defect with edge roughness, below which the defect is classified as False (or else Real Nuisance). This parameter applies only if the maximum intensity between a defect and reference image is larger than the value specified by maxDiffIntensityToConsiderAllEdgeRoughnessAsFalse. The default value is 2.

- minDiffForContamination

  This parameter works as a contamination classifier. Calibre DefectClassify compares the maximum pixel intensity difference between defect and reference signals with the value specified by minDiffForContamination. If the intensity difference is larger than the threshold specified, then the defect is considered to be dominant enough to be classified as a contamination defect (that typically have stronger difference signals). The default value is 8.

- minDiffForHardDefectsOnSmoothEdgesInTransmittedImage

  Differentiates between real nuisance and hard defects, applicable when a defect is on a smooth edge and its corresponding pattern width is smaller. Calibre DefectClassify compares the maximum pixel intensity difference between the transmitted defect and reference signal using this parameter's value. If the intensity difference is larger than the threshold specified, then the defect is considered to not be dominant enough to be classified as a real nuisance defect (that typically have weaker difference signals). The default value is 8.

- minDiffForHardDefectsOnSmoothEdgesInReflectedImage

  Differentiates between real nuisance and hard defects, applicable when a defect is on smooth edge and its corresponding pattern width is smaller. Calibre DefectClassify compares the maximum pixel intensity difference between the reflected defect and reference signal using this parameter's value. If the intensity difference is larger than the threshold specified, then the defect is considered to not be dominant enough to be classified as real nuisance defects (that typically have weaker difference signals). The default value is 8.

- minDiffForHardDefectsWithEdgeRoughnessInTransmittedImage

  Differentiates between real nuisance and hard defects, applicable when the defect is on a relatively large pattern. Calibre DefectClassify compares the maximum pixel intensity difference between transmitted defect and reference signals with this parameter's value. If the intensity difference is larger than the threshold specified by this parameter, then the defect is considered to not be dominant enough to be classified as real nuisance defects (that typically have weaker difference signals). The default value is 8.

- minDiffForHardDefectsWithEdgeRoughnessInReflectedImage

  Differentiates between real nuisance and hard defects, applicable when the defect is on a relatively large pattern. Calibre DefectClassify compares the maximum pixel intensity difference between reflected defect and reference signals with this parameter's value. If the intensity difference is larger than the threshold specified by this parameter, then the defect is considered to not be dominant enough to be classified as real nuisance defects (that typically have weaker difference signals). The default value is 8.

- minDefectSizeForHardDefectsWithEdgeRoughness

  Differentiates between real nuisance and hard defects, applicable when the defect is on a relatively large pattern. Calibre DefectClassify compares the width of the defect blob with this parameter's value. If the width is larger than the threshold specified by this parameter, then the defect is considered to not be dominant enough to be classified as a real nuisance defect (that typically have weaker difference signals). The default value is 1.

- minDiffForHardDefectsOnJogsAndCornersInTransmittedImage

  Differentiates between real nuisance and hard defects, applicable when the defect is on jogs or corners. Calibre DefectClassify compares the maximum pixel intensity difference between transmitted defect and reference signals with this parameter's value. If the intensity difference is larger than the threshold specified by this parameter, then the defect is considered to not be dominant enough to be classified as real nuisance defects (that typically have weaker difference signals). The default value is 8.

- minDiffForHardDefectsOnJogsAndCornersInReflectedImage

  Differentiates between real nuisance and hard defects, applicable when the defect is on jogs or corners. Calibre DefectClassify compares the maximum pixel intensity difference between reflected defect and reference signals with this parameter’s value. If the intensity difference is larger than the threshold specified by this parameter, then the defect is considered to not be dominant enough to be classified as real nuisance defects (that typically have weaker difference signals). The default value is 8.

- minDiffForSrafDefectsLongPart

  Differentiates between real nuisance and SRAF defects, applicable when the defect is on long edges of SRAF patterns and not on SRAF pattern ends. Calibre DefectClassify compares the maximum pixel intensity difference between defect and reference signals with this parameter’s value. If the intensity difference is larger than the threshold specified by this parameter, then the defect is considered to not be dominant enough to be classified as a real nuisance defect (that typically have weaker difference signals). The default value is 25.

- minDiffForSrafDefectsCurvedPart

  Differentiates between real nuisance and SRAF defects, applicable when the defect is on SRAF pattern ends and not on long edges of SRAF patterns. Calibre DefectClassify compares the maximum pixel intensity difference between defect and reference signals with this parameter’s value. If the intensity difference is larger than the threshold specified by this parameter, then the defect is considered to not be dominant enough to be classified as a real nuisance defect (that typically have weaker difference signals). The default value is 30

- minimumPatternWidthToConsiderPatternAsLarge

  Specifies the minimum width of a pattern to consider it large. Calibre DefectClassify calculates the pattern width and compares it with this parameter’s value. If the calculated width is larger than the threshold specified by this parameter, the pattern is considered as large. The default value is 3.

- minRatioOfPatternToDefectWidth

  Specifies the threshold used to classify an on-edge defect as Contamination or RealNuisance in the case of a defect caught in D2D mode. When the pattern-to-defect width ratio is less than the value of this parameter, minDiffForContamination is used as the threshold to differentiate between RealNuisance and Contamination. Otherwise, minDiffForHardDefectsWithEdgeRoughnessInReflectedImage is used as the threshold for differentiation. The default value is 3.

- minRatioOfPatternToDefectWidthNonD2D

  Specifies the threshold used to classify a NonD2D on-edge dark defect as OnDarkEdgeContamination or RealNuisance. An on-edge dark defect having a residue greater than the value of the parameter minResidueForOnEdgeDarkNonD2D is be considered as OnDarkEdgeContamination, but only if the pattern width-to-defect width ratio is smaller than the value specified for this parameter. Otherwise, the defect is classified as RealNuisance. Figure 2-32 illustrates the flow to determine whether an on-edge defect is OnDarkEdgeContamination, RealNuisance, or False. The default value is 3.

- transmissionThresholdLowerForIsolatedDark

  Specifies the transmission threshold to distinguish IsolatedDark defects into two categories. If the transmission calculated is less than the threshold specified by this parameter, the IsolatedDark defect is classified as IsolatedDarkLowTrans. The default value is 0.7.

- transmissionThresholdUpperForIsolatedDark

  Specifies the transmission threshold to distinguish IsolatedDark defects from False defects. If the transmission calculated is more than the threshold specified by this parameter, the IsolatedDark defect is classified as False due to the weak nature of the defect. The default value is 1.0 (for example, no weak defects are classified as False).

Aerial Inspection Parameters

- firstTierFalseDefectPriority

  Specifies the priority of false defect types when a defect image has multiple defect types in it. The parameter lists the defect types supported by Calibre DefectClassify in decreasing order of priority (one listed earlier in the list has a higher priority). The default order in decreasing priority is: Bright Spot, Tap, Focus, EdgeRoughness, Alignment, and Nuisance.

- firstTierRealDefectPriority

  Specifies the priority of real defect types when a defect image contains multiple defect types. The parameter lists the defect types supported by Calibre DefectClassify in decreasing order of priority (a type listed earlier in the list has a higher priority). The default order (in decreasing priority) is RepairedMark, MissingChrome, SRAFClear, SRAFDark, FillDark, FillClear, LargeDefect, ClearOpaque, MultipleEdgesDark, OnEdgeDark, IsolatedDark, IsolatedDarkLowTrans, MultipleEdgesClear, OnEdgeClear, IsolatedClear, IsolatedClearLowTrans, Contamination, OnDarkEdgeContamination, OnClearEdgeContamination, Review, and RealNuisance.

- srafLayoutImageMargin

  Specifies the margin (in um) around the central point corresponding to the defect image. This is required for effective performance near image boundaries. The default value is 2um.

- commonPatternDistanceFromDefectSignal

  This parameter relates to the closest distance of a defect to a feature edge. It specifies the threshold distance value (in um) below which the defect is considered as impacting an edge, and is thus classified as an On Edge or Multiple Edges-type defect. For distances larger than the threshold, the defect is considered as an isolated defect. The default value is 0.5um.

- focusMaxDiffGL

  This parameter relates to a focus classifier. Calibre DefectClassify compares the maximum pixel intensity difference between defect and reference signals with this parameter's value. If the intensity difference is larger than the threshold specified by this parameter, then the defect is considered to be dominant enough to not be classified as a focus defect, which typically has weaker difference signals. The default value is 19.

- nonrepairMaxDiffGLLowerLimit

  Similar to the focusMaxDiffGL parameter, if the difference is less than the threshold specified by this parameter, then the defect is considered too weak to be classified as non-repairable. The default value is 15.

- nonrepairDefectArea

  Refers to the area of a non-repairable defect (in sq.um). A defect with an area less than the threshold specified by this parameter is not considered strong enough to be classified as non-repairable. The default value is 2 square um.

- nonrepairNumPatternAffected

  Refers to the number of patterns affected by a non-repairable defect. A defect affecting less number of patterns than the number specified by this parameter is not considered serious enough to be classified as non-repairable. The default value is 5.

- printabilityNuisance

  Specifies the first CD variation (%) threshold below which the printability classification of the defect is Nuisance. The default value is 4.

- printabilityPass

  Specifies the second CD variation (%) threshold below which, and above the printabilityNuisance threshold, the printability classification of the defect is Pass. The default value is 10.

- printabilityWarning

  Specifies the third CD variation (%) threshold below which, and above the printabilityPass threshold, the printability classification of the defect is Warning. The default value is 12. Above this value, the printability classification is Fail.

- printabilityROI

  This is the fraction of the image that is considered for CD variation calculation. The default value is 0.75 (center 3/4ths of the image).

- srafMaxCDVariationForSRAFDefect *val*

  This is the CD variation (%) threshold above which a defect, even if on an SRAF region, is preferred to be referenced as a real defect. The default value is 10.

- smashed

  Applicable to the Calibre DefectClassify GUI only, specifies the layer number for a smashed layer. The default value is 1.

- clearSraf

  Applicable to the Calibre DefectClassify GUI only, specifies the layer number for a clear SRAF layer. The default value is 2.

- darkSraf

  Applicable to the Calibre DefectClassify GUI only, specifies the layer number for a dark SRAF layer. The default value is 3.

- clearFill

  Applicable to the Calibre DefectClassify GUI only, specifies the layer number for a clear fill layer. The default value is 4.

- darkFill

  Applicable to the Calibre DefectClassify GUI only, specifies the layer number for a dark fill layer. The default value is 5.

- chrome

  Applicable to the Calibre DefectClassify GUI only, specifies the layer number for a chrome layer. The default value is 6.

- sizeThreshRefl

  Specifies the defect size (in um) upon which a reflected defect's printability classification depends. Defects larger than this size are classified as Pass; otherwise they are classified as Fail. The default value is 0.5 um.

- enableSizeBasedPrintabilityUpdate

  Specifies if the Printability Classification of a reflected defect is updated based on its size. The default value is no.

- enableOnEdgeContamDetection

  Specifies whether a contamination defect on an edge should be classified as OnDarkEdgeContamination or OnEdgeDark. If the value specified is true and the defect blob is closer to the edge, Calibre DefectClassify classifies the contamination defect as OnDarkEdgeContamination. If value specified for the parameter is false, the defect is classified as OnEdgeDark. The default value is false.

- nuisanceD2DRefGlLowerCutOff

  Specifies the GL intensity threshold above which pixels in reflected D2D images are considered for processing. The default value is 5.

Highres Inspection Parameters

- firstTierFalseDefectPriority

  Specifies the priority of false defect types when a defect image contains multiple defects. The parameter lists the defect types supported by Calibre DefectClassify in decreasing order of priority (a type listed earlier in the list has higher priority). The default order (in decreasing priority) is BrightSpot, Tap, Focus, EdgeRoughness, Alignment, and Nuisance.

- firstTierRealDefectPriority

  Specifies the priority of real defect types when a defect image contains multiple defect types. The parameter lists the defect types supported by Calibre DefectClassify in decreasing order of priority (a type listed earlier in the list has higher priority). The default order in decreasing priority is: RepairedMark, MissingChrome, SRAFClear, SRAFDark, LargeDefect, FillDark, FillClear, ClearOpaque, MultipleEdgesDark, OnEdgeDark, IsolatedDark, IsolatedDarkLowTrans, MultipleEdgesClear, OnEdgeClear, IsolatedClear, IsolatedClearLowTrans, Contamination, OnDarkEdgeContamination, OnClearEdgeContamination, Review, and RealNuisance.

- classifySrafWithoutLayout

  Activates and deactivates identifications of defects on SRAF patterns in the absence of layout information. The default value is false (defect identification is deactivated without layout information).

- srafLayoutImageMargin

  Specifies the margin (in um) around the central point corresponding to the defect image. This is required for effective performance near image boundaries. The default value is 2 um.

- layoutMarginForAlignment

  Specifies the margin (in um) around the central point which is used for alignment of layout data with mask patterns. The default value is 1.8um.

- deleteTempFiles

  Activates or deactivates the deletion of temporary files created during simulation runs. The default value is “true”.

- highresBrightSpotMaxDiffThreshold

  Specifies the minimum GL intensity difference between defect and reference images for the defect to even be considered by a BrightSpot classifier. The default value is 0 (any difference signal is processed for bright spot).

- highresCheckSecondarySignal

  Specifies, after a bright spot is found, whether the image is to be processed for the presence of other defects. The default value is yes (other defects are searched after a bright spot is found).

- highresCommonPatternDistanceFromDefectSignal

  Relates to the closest distance of a defect to a feature edge. This parameter specifies the threshold distance value (in um) below which the defect are considered as impacting an edge, and are classified as On Edge or Multiple Edges-type defects. For distances larger than the threshold, the defect is considered as an isolated defect. The default value is 0.5um.

- highresFocusMaxDiffGL

  Relates to a focus classifier. Calibre DefectClassify compares the maximum pixel intensity difference between defect and reference signals with this parameter’s value. If the intensity difference is larger than the threshold specified by this parameter, then the defect is considered to be dominant enough to not be classified as a focus defect, which usually has weaker difference signals. The default value is 30.

- minResidueForOnEdgeDarkNonD2D

  Specifies the threshold used to classify a NonD2D on-edge dark defect as OnDarkEdgeContamination or RealNuisance. An on-edge dark defect having a residue greater than the value of this parameter is classified as OnDarkEdgeContamination, but only if the pattern width-to-defect width ratio is smaller than the value of the parameter minRatioOfPatternToDefectWidthNonD2D. If the residue is less than the value specified for this parameter, the defect will never be classified as OnDarkEdgeContamination; instead it is classified as RealNuisance or False depending on whether the residue is greater or smaller than the value specified for the parameter minResidueForRealNuisanceNonD2D. Figure 2-32 illustrates the flow to determine whether an on-edge defect is classified as OnDarkEdgeContamination, RealNuisance, or False. The default value is 8.

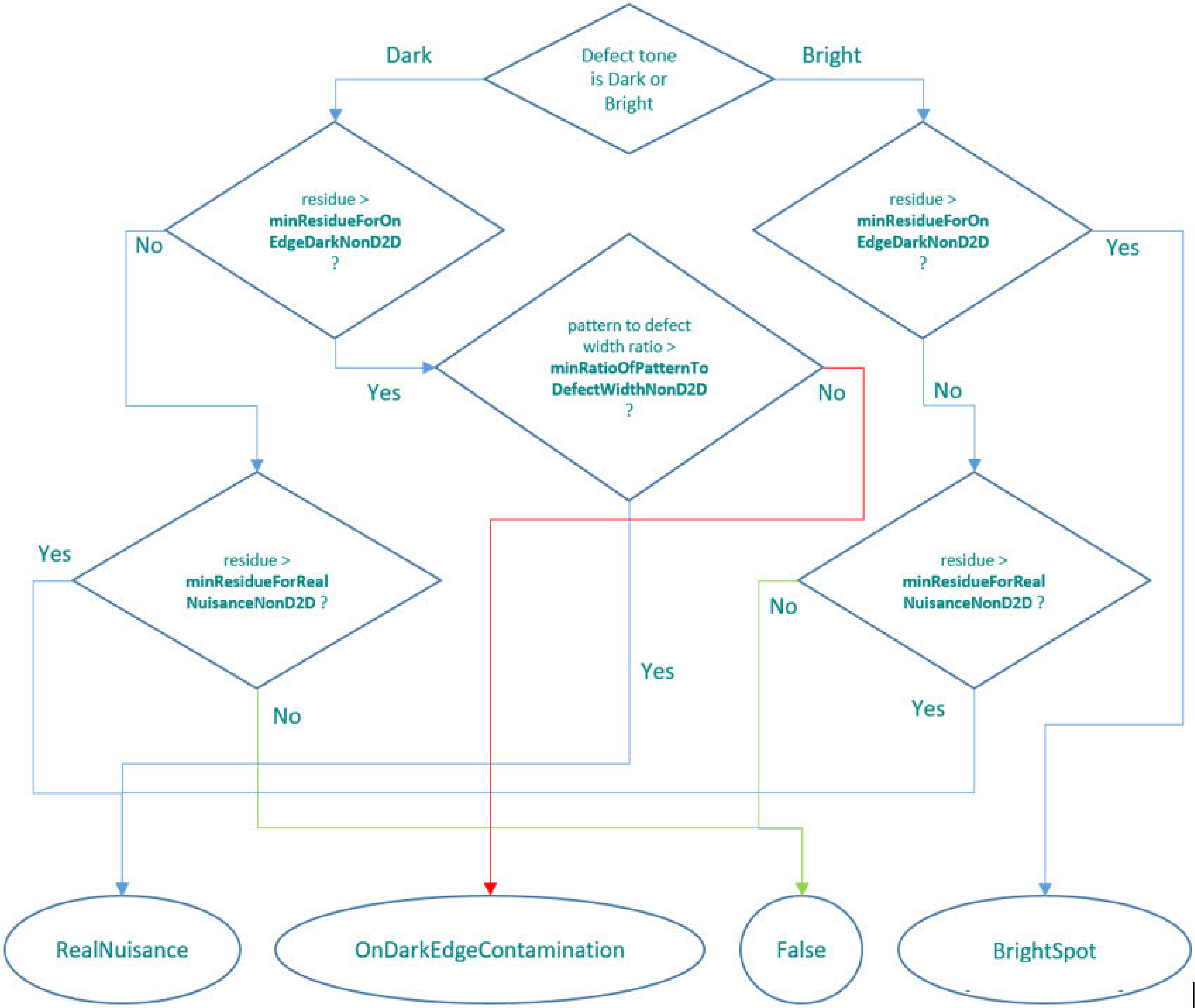
- minResidueForRealNuisanceNonD2D

  Specifies the threshold used to classify a NonD2D on-edge (Dark or Bright) defect as RealNuisance or False. An on-edge defect having a residue less than the value of this parameter is considered as False. However, if residue for the on-edge defect is greater than the value of this parameter and less than the value of the parameter minResidueForOnEdgeDarkNonD2D, then the defect is classified as RealNuisance. Figure 2-32 illustrates the flow to determine whether an on-edge defect is to be classified as OnDarkEdgeContamination, RealNuisance, Bright Spot, or False. The default value is 4.

- maxBrightResidueForRealNuisanceNonD2D

  Specifies the threshold used to call a locally bright on-edge NonD2D defect as False or RealNuisance. An on-edge bright defect having residue less than the value specified by this parameter will be classified as RealNuisance. If the residue is greater than the value specified for this parameter, the defect is classified as a Bright Spot. Figure 2-32 illustrates the flow to determine whether an on-edge defect is to be classified as a RealNuisance, Bright Spot, or False defect. The default value is 8.

Figure 2-32. Flow to Classify a NonD2D On-Edge Defect



- transDiffGLThresholdForRealOnEdge {true | <u>false</u>}

  This parameter applies to RealNuisance NonD2D defects. Generally, if defect residue is present in both transmitted and reflected images with opposite tones, the defect is classified as false. If the defect is also present only in the reflected image on the edge of larger pattern, it is typically classified as RealNuisance. This parameter specifies how much residue should be ignored when evaluating whether the residue should be considered a defect in the transmitted image as well.

  If the value specified is smaller (for example, 8), any residue above 8 in the transmitted image is considered as a defect in the transmitted image as well. Because the defect is present in both transmitted and reflected image, Calibre DefectClassify classifies the defects as false.

If the value specified is larger (greater than 8), the residue present in the transmitted image is ignored. In this case, because the defect is present only in the reflected image, Calibre DefectClassify classifies the defect as RealNuisance. The default value is 8.

- sensitivityThresholdLowerForIsolatedNonD2D *val*

  This parameter applies to isolated NonD2D defects. If defect residue is less than the value of this parameter, then the defect is ignored since it is a weak or faint defect based on the residue. If the residue is greater than the parameter value, then further processing depends on the value of the parameter sensitivityThresholdUpperForIsolatedNonD2D. The default value is 4.

- sensitivityThresholdUpperForIsolatedNonD2D *val*

  This parameter applies to isolated NonD2D defects. If the defect residue is greater than the value of this parameter, then the defect cannot be ignored because it is considered a strong defect based on the residue. If the residue is less than the parameter value and greater than sensitivityThresholdLowerForIsolatedNonD2D, the defect is processed based on its properties. The default value is 14.

- sensitivityScaleFactorForNonD2D *val*

  This parameter applies to isolated NonD2D defects and can be tuned to alter defect detection sensitivity. Decreasing the value (for example, to 3 from the default value of 4) increases the sensitivity of the algorithm, making it suitable for catching weaker or fainter signals. However, this also increases the chances of identifying weak, noisy signals as defects. Increasing the value (for example, to 5), decreases the sensitivity, catching only the more obvious defect signals. The default value is 4.

- highresNonrepairMaxDiffGLLowerLimit *val*

  If the transmitted residue is less than the threshold specified by this parameter, then the defect is considered too weak to be classified as non-repairable. The default value is 15.

- highresNonrepairDefectArea *val*

  Refers to the area of a non-repairable defect (in sq.um). A defect with an area less than the threshold specified by this parameter is not considered strong enough to be classified as non-repairable. The default value is 2.5 square um.

- highresNonrepairNumPatternAffected *val*

  Refers to the number of patterns affected by a non-repairable defect. A defect affecting less number of patterns than the number specified by this parameter is not considered serious enough to be classified as non-repairable. The default value is 5.

- minResidueForRealIsolatedNonD2D *val*

  Specifies the minimal strength and residue of an isolated NonD2D defect relative to its neighborhood. Defects above the value of this parameter are not classified as a Nuisance defect and are instead classified as Review.

The default value is 15. Decreasing this value leads to fewer under-classifications while considerably increasing classification of False defects to review. This usually causes a decline in overall accuracy.

- highresRepairMarkMinGLThreshold *val*

  Specifies the minimum reflected residue that is required to consider a difference signal for repair mark classifier. The default value is 18.

- sizeThresh1 *val*

  Specifies the upper threshold (in um) for the first defect size bin. Defects with sizes smaller than this parameter's value are placed the first bin, corresponding to smallest sized defects. The default value is 0.05 um.

- sizeThresh2 *val*

  Specifies the upper threshold (in um) for the second defect size bin. Defects with sizes larger than the value sizeThresh1 but smaller than this parameter's value are placed in the second bin. The default value is 0.075 um.

- sizeThresh3 *val*

  Specifies the upper threshold (in um) for the third defect size bin. Defects with sizes larger than the value of sizeThresh2 but smaller than this parameter's value are placed in the third bin. The default value is 0.1 um.

- sizeThresh4 *val*

  Specifies the upper threshold (in um) for the fourth defect size bin. Defects with sizes larger than the value of sizeThresh3 but smaller than this parameter's value are placed in the fourth bin. The default value is 0.25 um.

- SizetThresh5 *val*

  Specifies the upper threshold (in um) for the fifth defect size bin. Defects with size larger than the value sizeThresh4 and smaller than this parameter's value are placed in the fifth bin. The default value is 0.5 um.

- sizeThresh6 *val*

  Specifies the upper threshold (in um) for the sixth defect size bin. Defects with sizes larger than the value of sizeThresh5 but smaller than this parameter's value are placed in the sixth bin. The default value is 0.75 um.

- sizeThresh7 *val*

  Specifies the upper threshold (in um) for the seventh defect size bin. Defects with sizes larger than the value of sizeThresh6 but smaller than this parameter's value are placed in the seventh bin. The default value is 1.0um.

- sizeThresh8

  Specifies the upper threshold (in um) for the eighth defect size bin. Defects with sizes larger than the value of sizeThresh7 but smaller than this parameter's value are placed in the eighth bin. The default value is 2.5 um.

- ND_MAXGL_CUTOFF *val*

  Specifies the GL intensity threshold above which pixels in highres images are considered for processing. The default value is 8.

Calibre DefectClassify Auxilliary Parameters

- LayerMapping

  Specifies the mapping of a layer name to a layer number (map layer) and datatype. For MEBES job decks, the datatype value must be set to -1. For example:

```
<LayerMapping>
 <Map layer="1" datatype="-1" name="SMASHED"/>
 <Map layer="2" datatype="-1" name="BLOCK_FILL"/>
 <Map layer="3" datatype="-1" name="DARK_SRAF"/>
 <Map layer="4" datatype="-1" name="CLEAR_SRAF"/>
 <Map layer="5" datatype="-1" name="DARK_FILL"/>
 <Map layer="6" datatype="-1" name="CLEAR_FILL"/>
 <Map layer="7" datatype="-1" name="CHROME"/>
</LayerMapping>
```

  For OASIS, the appropriate values for datatype must be set.

- LayoutFolders

  Specifies the default folder(s) to search for a layout file. For example, a layout file *test.jb* is in the folder */home/defectclassify/*. If you specify *file test.jb* as input in the Calibre DefectClassify GUI or command line without a folder path, but have the following specification in the *adc-ini.xml* file, Calibre DefectClassify searches and finds the file in the */home/defectclassify/* folder:

```
<LayoutFolders>
 <Folder value="./"/>
 <Folder value="/home/defectclassify/"/>
</LayoutFolders>
```

- CDImageFolders, InspectionModelFolders, ScannerModelFolders

  These parameters behave the same as LayoutFolders parameter. They only apply to CD image file, inspection model folder, and scanner model folder.

Chapter 3
Image-to-Layout Alignment

Calibre DefectClassify analyzes SEM images captured at various defect locations on the wafer and correlates them to layout patterns in order to classify defects. For correlation, layout coordinates are provided as input. However, the patterns around layout coordinates do not always correspond to patterns found in SEM images. This requires accurate alignment of SEM and layout patterns as a prerequisite to defect classification.

The purpose of the Image-to-Layout Alignment tool is to find the actual coordinate location in the layout corresponding to the SEM image. This is referred to as SEM image to layout alignment. After alignment, the tool outputs the amount of shift as well as the corrected coordinate locations. Calibre DefectClassify takes the input layout coordinate, checks the area near that location, and works to find the correct location in the layout. The size of the target search area is specified through a configurable parameter.

As obtaining SEM images for each and every defect is typically time consuming, some defects do not have associated SEM images. Only coordinate locations (layout and wafer) are available. To calculate the shift for these defects, known shifts from other defects that have SEM images are used. The underlying assumption is that, during the inspection and SEM image capturing process, a certain coordinate offset is introduced, known as global offset for the dataset. If the global offset can be correctly calculated for a set of defects, any defect coordinate can be shifted by that offset, with or without the SEM image, to align it properly. This process also helps to eliminate any false alignment of defects with SEM images. For each SEM image, the layout is typically queried with a 4.5 micron margin in all directions. The best match for SEM image in the query region is then made.

Image-to-Layout Alignment Workflow

The Calibre DefectClassify uses a simple workflow to perform an analysis of defects detected on the wafer.

Prerequisites

- Calibre DefectReview must be installed and invoked.
- Calibre DefectClassify must follow the prerequisites as described in "Calibre DefectClassify Prerequisites" on page 9.

Procedure

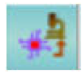
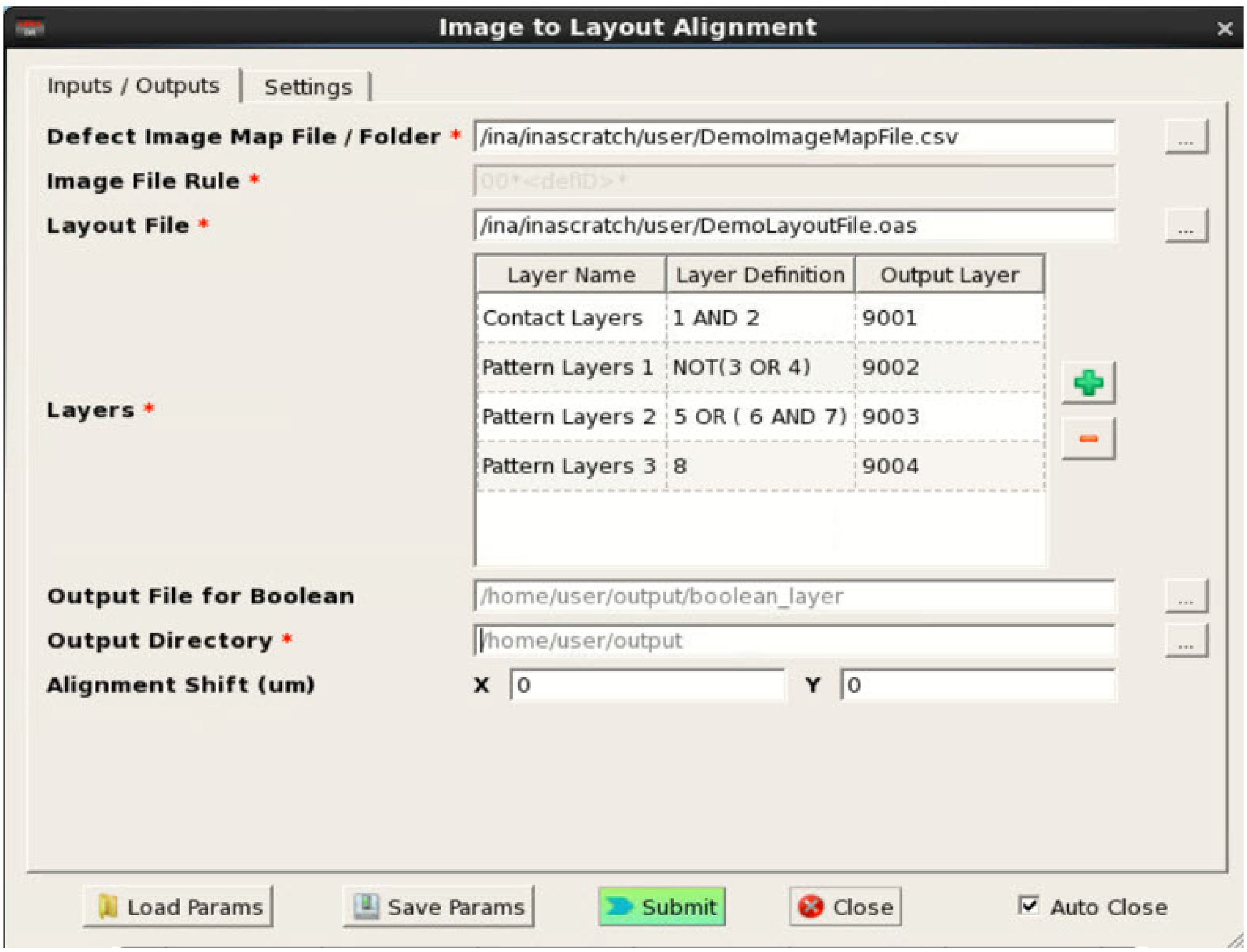
1. In Calibre DefectReview, click **Utilities > Image to Layout Alignment**, or click the Image to Layout Alignment icon  at the top of the Calibre DefectReview window. The Image to Layout Alignment dialog box appears.

Figure 3-1. Image to Alignment Dialog Box



2. Click the **Inputs/Outputs** tab and specify input and output parameters for an alignment run. This is described in detail in "Specifying Inputs and Outputs for Image-to-Layout Alignment" on page 87.

3. Click the **Settings** tab to enter parameters to configure an alignment run. This is described in detail in "Specifying Run Configuration Settings for Image-to-Layout Alignment" on page 92.

4. Click **Submit** to initiate the run. The results of the run are described in "Image-to-Layout Alignment Results Analysis" on page 96.

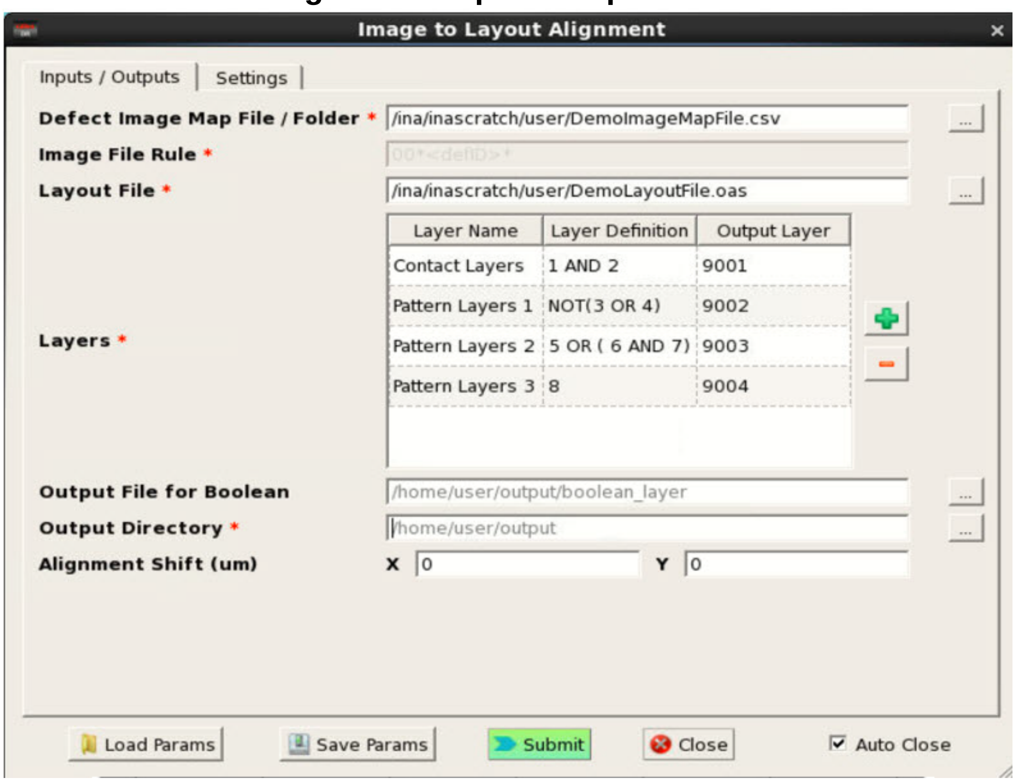
Specifying Inputs and Outputs for Image-to-Layout Alignment

After invoking the Image to Layout Alignment tool, you specify inputs and outputs for a run.

Procedure

1. Invoke the Image to Layout Alignment dialog box as described in "Image-to-Layout Alignment Workflow" on page 85

Figure 3-2. Inputs/Outputs Tab



2. In the **Inputs/Outputs** tab of the Image to Layout Alignment dialog box, specify the following parameters:

   - **Defect Image Map File/Folder —** Specifies either a defect image map file or a SEM images folder. When a defect ID has more than one defect image corresponding to it, use the SEM image folder. This requires analysis of image file names to identify which images correspond to the same defect ID. In this case, use the **Image File Rule** option.

- Defect Image Map File: This file contains the mapping of SEM images with some of the defects present in the wafer file. An example defect image map file is shown in the following figure:

Figure 3-3. Defect Image Map File

Defect ID	imagepath	pixelSize (um)
140	/home/user/demoDefectImage_1	0.002083333
144	/home/user/demoDefectImage_2	0.00294117
131	/home/user/demoDefectImage_3	0.00294117

The map file contains the image path and pixel size for each defect. This file can be created in CSV or TXT format. A TXT file should be formatted like a CSV file.

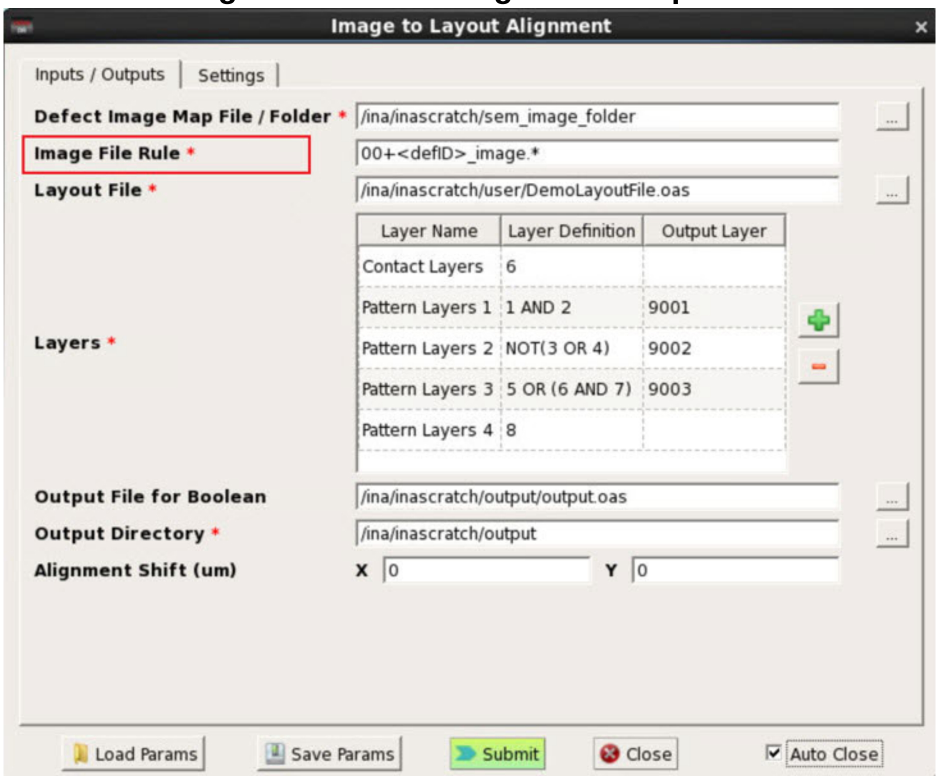
Note

In Klarf 1.8 format, if the FOV field is defined in the inspection file, then the corresponding FOV value is used to calculate the pixel size for the respective defect image during the Image to Layout Alignment process. If the FOV field is not defined in the inspection or set to 0, the pixelSize field from the defect image map file is considered if it is populated.

In the defect image map file, you can keep the pixel size field empty if you are not certain of its value. In that case, you must specify a list of possible pixel size values in the pixel_size_list parameter in the Image-to-Layout Alignment configuration file. The tool automatically determines the most suitable pixel size from the list for a defect.

- o SEM Image Folder: You can specify a SEM image folder as shown in the following figure:

Figure 3-4. SEM Image Folder Option



The Image File Rule option is required when an image folder is specified.

- **Image File Rule —** This is a regular expression to identify SEM images corresponding to one defect ID when this option is selected (as illustrated in Figure 3-4). This option is deactivated when a defect image map file is selected. The following is an example image file rule:

```
image_file_regex   ->   00+<defID>_image.*
```

The tool automatically picks up all the SEM images corresponding to a defect, runs alignment on all of the defects, and displays the best result based on the confidence-metric value obtained with each SEM image.

- **Layout File —** Specifies the layout file with which the SEM image is to be aligned. Currently, the supported format is OASIS[1](valid extensions are *.oas*, *.oasis*, and *.OAS*).

- **Layers —** This field has three columns; Layer Name, Layer Definition, and Output Layer. The format is <*layer_number*: *datatype*> where the *datatype* is optional.

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.

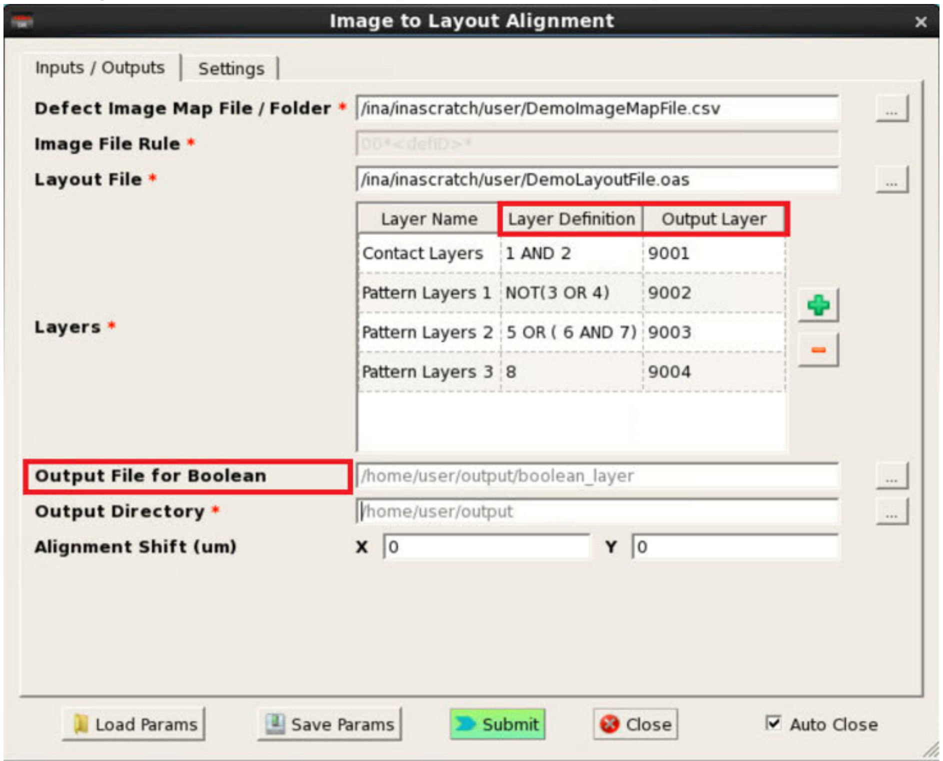
- Layer Name — Lists the type or group of layers at various levels of the mask. These include:
  - Pattern Layers 1, 2... <*N*> — Specifies layers to be combined using a particular pattern. Each Pattern Layers row can contain comma-separated layers, merged using a logical OR operation. Enter pattern layers according to their hierarchy on the mask (the bottom-most layer must be entered in Pattern Layers 1, followed by the next layer up in Pattern Layers 2, and so on).
  - Contact Layers — Specifies any contact layers. Multiple contact layers must be comma-separated. Unlike pattern layers, there is a single row for all contact layers present. You can specify single contact layers when the run configuration file parameter alignment_method is set to 1.
- Layer Definition — Defines Boolean operations on layers to combine them (see Figure 3-5). Valid Boolean operations are OR, AND, XOR, and NOT. You can use parenthesis (**()**) when necessary. The valid keywords for Boolean operations are listed in the following table:

Table 3-1. Supported Boolean Operations and Keywords

Operation	Valid Keywords
OR	OR, or, | (pipe), , (comma)
AND	AND, and, & (ampersand)
XOR	XOR, xor
NOT	NOT, not, ! (exclamation point)

- Output Layer — Saves the results of the Boolean expression specified in the Layer Definition column to an output OASIS file for future use. The output layer for each such expression and datatype are to be specified under this column, as shown in Figure 3-5.

Figure 3-5. Boolean Operation and Output Layout File



- **Output File for Boolean —** Specifies the file to store the output layout layers specified in the Output Layer column. This file can be loaded with precomputed layers.

Note

When **Output File for Boolean** is specified, the tool assigns default layer numbers to each row of the Layers pane, and the Output Layer fields are left unspecified.

- **Output Directory —** Specifies the directory containing all files output from the tool during a run. After a tool run, the folder contains:
  - Images — A directory containing all output images and bin files from the run.
  - *input.csv* — A file generated from the *klarf* and *defectmap* files.
  - *nxdat_image_to_layout_log.txt* — The tool run log file.
  - *sem_aligned_out.csv* — The output CSV file.
  - *setup.txt* — The setup file containing information on the layout file, layers, and output directory specified in the **Inputs/Outputs** tab.
- **Alignment Shift (um) —** An optional parameter used to align wafer defect coordinates in the wafer file. The default value for both X and Y is 0 microns.

- **Save Params —** Saves the current settings in Image to Layout Alignment dialog in a specified XML file.

Note

It is recommended that you save output files in a separate location after an Image to Layout Alignment run to prevent previous data from being lost.

- **Load Params —** Loads all the required inputs in this dialog from an existing settings file (*.xml*). This XML file must be in a specific format generated by clicking **Save Params**.
- **Auto Close —** Closes the Image to Layout Alignment dialog box when the run completes successfully. If there is any error in execution or if the process is aborted, the Image to Layout Alignment dialog box remains open. If this option is not checked, the transcript log window remains open for both successful completion and error conditions. Both the transcripts window and the Image to Layout Alignment dialog box must then be closed manually by clicking **Close**.

Results

The next step is to specify run configuration settings in the **Settings** tab. Proceed to "Specifying Run Configuration Settings for Image-to-Layout Alignment" on page 92 for complete instructions.

Specifying Run Configuration Settings for Image-to-Layout Alignment

After specifying inputs and outputs for the Image-to-Layout Alignment tool, specify configuration settings for the run.

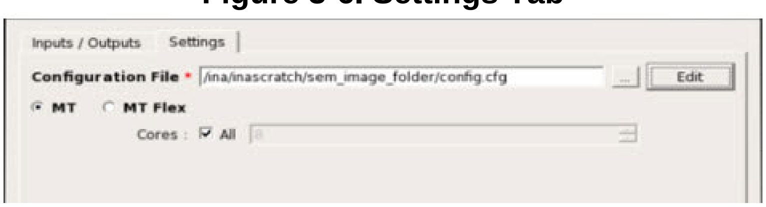
Prerequisites

- Inputs and outputs are specified as described in "Specifying Inputs and Outputs for Image-to-Layout Alignment" on page 87.

Procedure

1. In the Image to Layout Alignment dialog box, click the **Settings** tab.

Figure 3-6. Settings Tab



2. In the **Settings** tab, specify values for the following fields:

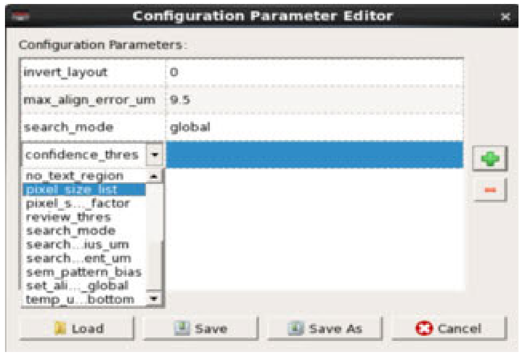
    - **Configuration File —** Specifies a text file (with a *.cfg* suffix) containing run configuration parameters. The parameters are summarized in the following table, with complete descriptions in "Image-to-Layout Alignment Run Configuration File" on page 101.

Table 3-2. Image-to-Layout Alignment Configuration File Parameters

Parameter	Description
max_align_error_um	Specifies the maximum alignment error in microns.
search_radius_um	Specifies the search radius in microns.
search_step_increment_um	Specifies the increments used in the search radius for each step.
search_mode	Specifies whether a local or global method is to be used to search for the global shift radius.
min_defects_in_search:	Specifies the minimum number of defects with a SEM that should be present to calculate the global offset using local method. This is not required if search_mode is set to global.
invert_layout	Specifies whether or not a layout image inversion is required.
pixel_size_scale_factor	Specifies the resolution scale factor for a layout and SEM image.
pixel_size_list	Specifies a comma-separated list of various pixel sizes of defect images. The tool automatically determines the most suitable pixel size for each defect image from values provided in this list. This field has lower priority than fov_list.
fov_list	Specifies a comma-separated list of FOV (Field of VIEW) values for defect images. The tool automatically determines the most suitable FOV value for each defect from the values provided in this list. This field has higher priority than pixel_size_list.

You can open or modify an existing configuration file by clicking the **Edit** button. This opens the Configuration Parameter Editor, as shown in the following figure:

Figure 3-7. Configuration Parameter Editor



Click the green plus button to add a new parameter. This adds a new row to the table. Double-clicking on the first column of the new row populates a combo box with all available configuration parameters. A default value is automatically assigned for that parameter that you can then edit by clicking on the entry.

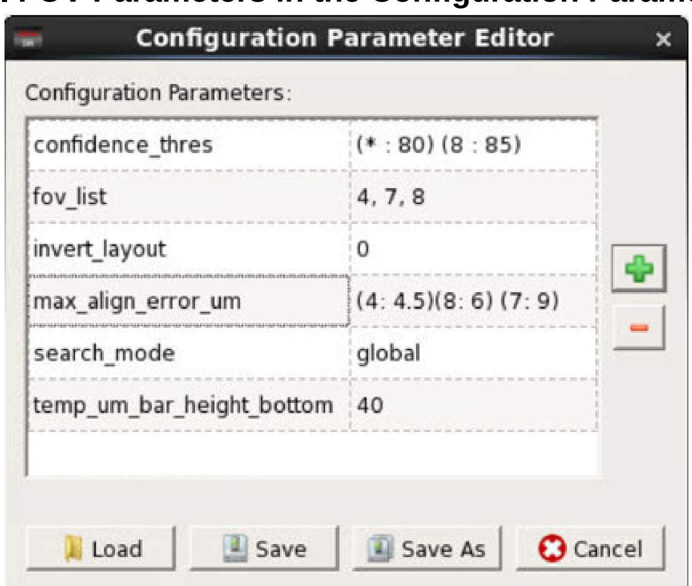
Click the red minus button to remove a parameter from the table.

To save any changes to the parameters, click **Save** or **Save As**.

To load an existing configuration file, click **Load**. Once a different configuration file is loaded, it becomes the current default configuration file.

- **Specifying FOV Parameters —** You can specify values for multiple FOV (Field of View)-dependent parameters for a defect SEM image as well as the corresponding FOV in the run configuration file. These parameters are displayed in the Configuration Parameter Editor.

Figure 3-8. FOV Parameters in the Configuration Parameter Editor



For example, if the fov_list contains the following values:

```
fov1, fov2, fov3
```

you can specify values for all parameters affected by each FOV value.

```
fov_dependent_param
 (fov1:paramVal1)(fov2:paramVal2)(fov3:paramVal3)
```

In this example, paramVal1 is applied to all defect images with an FOV of fov1, paramVal2 is applied to defect images with FOV of fov2, and so on. When one value applies to more than one FOV (or a default value is to be specified) then an asterisk (*) can be used in place of an FOV value. For example:

```
fov_dependent_param2 (fov1:paramVal1) (*:paramVal2)
```

In this example, paramVal1 applies to defect images with an FOV of fov1, and paramVal2 applies to all the defects that do not use fov1.

Note

Not all run configuration parameters are FOV-dependent. All FOV-dependent parameters are listed in “Run Configuration File Parameter Summary” on page 108.

- **MTflex (Distributed) —** Specifies a remote host in *.crp* format. The remote host file format is:

  ```
  REMOTE HOST <hostname> <Number_of_cores> MGC_HOME
  <Calibre_install_path> DIR <remote_directory>
  ```

  For example:

  ```
  REMOTE HOST localhost 8 MGC_HOME /user/work_area/Mgc_home/
     DIR /output_dir
  ```

- **MT (Multithreaded) —** By default, the tools are enabled to work in multithreaded mode. You can specify the number of cores to utilize. Otherwise, the tool starts running with all available cores of a system.

3. Click **Submit** to initiate an Image-to-Layout alignment run.

Results

A progress window appears, displaying a status log while the tool runs. Refer to “Image-to-Layout Alignment Results Analysis” on page 96 for a description of the results. The results can be saved by selecting **File** > **Save** or **File** > **Save As**. These results include all of the values under the columns ShiftX, ShiftY, Confidence Metric, and Alignment Status. In addition, all generated images and bin files are also saved. The bin files are required to show contour information for aligned SEM images.

The saved defect file can be reloaded in Calibre DefectClassify for further analysis. When you load a saved file, the columns SystemX and SystemY will show a layout shifted X- and Y-coordinates, respectively, from the previously saved run. A rerun of the Image to Layout Alignment tool overwrites any previously displayed data, either loaded from the defect file or from previous runs.

Image-to-Layout Alignment Results Analysis

After the Image-to-Layout Alignment tool completes its run, several actions are performed that enable you to analyze the results.

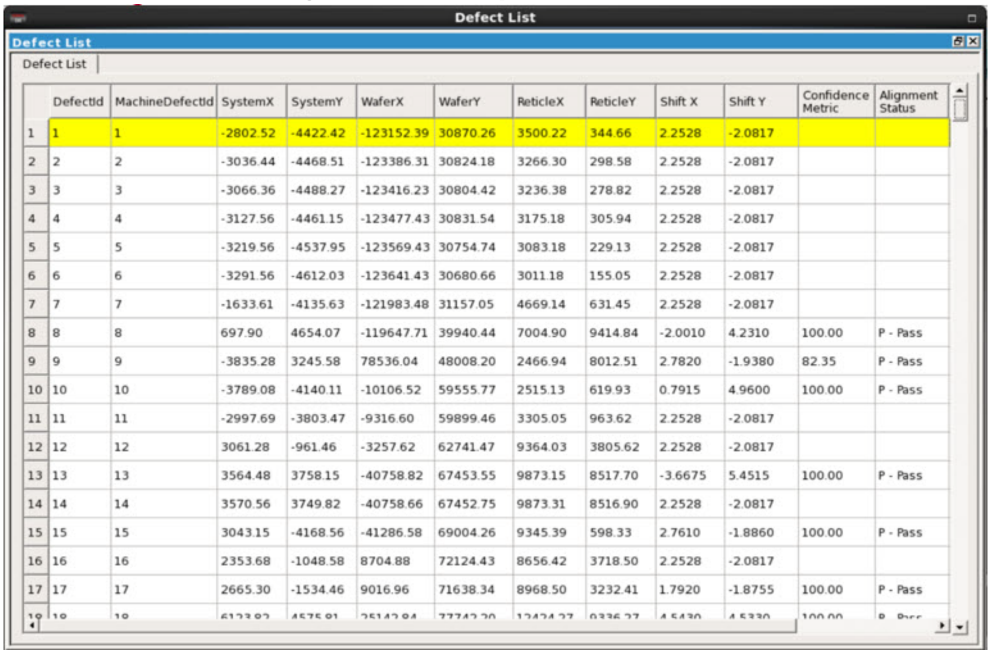
After entering inputs and run settings, specifying run configuration settings, and clicking **Submit**, a run is initiated. During the run, several actions are performed:

- Calculation of shiftX and shiftY required for layout alignment.
- Output of aligned images in output directory.
- The defect list is updated with shifted layout coordinates.
- The stacked wafer file is loaded into Calibre DefectReview with updated systemX and systemY values, and displays the following images in the Image Measurement Unit (IMU):
    - Sem — The original SEM image.
    - Aligned Sem — The original SEM image on which aligned layout contours are overlaid by clicking the **Display layout contour on SEM image** icon in the IMU tool bar. 
    - Layout — The layout image corresponding to the SEM image after alignment.
    - Layout Inverted — The inverted layout for cases when SEM images have inverted polarities for patterns and background regions (patterns are darker than background region).

Note
The images listed are not shown by default; you must configure the IMU display to be able to see these images. The images are listed under **Other Images** in the right-click menu found in any IMU.

After the run is completed, the Alignment status column is populated in the Defect List.

Figure 3-9. Output Defect List


Defect List

Defect List

Defect List

	DefectId	MachineDefectId	SystemX	SystemY	WaferX	WaferY	ReticleX	ReticleY	Shift X	Shift Y	Confidence Metric	Alignment Status
1	1	1	-2802.52	-4422.42	-123152.39	30870.26	3500.22	344.66	2.2528	-2.0817		
2	2	2	-3036.44	-4468.51	-123386.31	30824.18	3266.30	298.58	2.2528	-2.0817		
3	3	3	-3066.36	-4488.27	-123416.23	30804.42	3236.38	278.82	2.2528	-2.0817		
4	4	4	-3127.56	-4461.15	-123477.43	30831.54	3175.18	305.94	2.2528	-2.0817		
5	5	5	-3219.56	-4537.95	-123569.43	30754.74	3083.18	229.13	2.2528	-2.0817		
6	6	6	-3291.56	-4612.03	-123641.43	30680.66	3011.18	155.05	2.2528	-2.0817		
7	7	7	-1633.61	-4135.63	-121983.48	31157.05	4669.14	631.45	2.2528	-2.0817		
8	8	8	697.90	4654.07	-119647.71	39940.44	7004.90	9414.84	-2.0010	4.2310	100.00	P - Pass
9	9	9	-3835.28	3245.58	78536.04	48008.20	2466.94	8012.51	2.7820	-1.9380	82.35	P - Pass
10	10	10	-3789.08	-4140.11	-10106.52	59555.77	2515.13	619.93	0.7915	4.9600	100.00	P - Pass
11	11	11	-2997.69	-3803.47	-9316.60	59899.46	3305.05	963.62	2.2528	-2.0817		
12	12	12	3061.28	-961.46	-3257.62	62741.47	9364.03	3805.62	2.2528	-2.0817		
13	13	13	3564.48	3758.15	-40758.82	67453.55	9873.15	8517.70	-3.6675	5.4515	100.00	P - Pass
14	14	14	3570.56	3749.82	-40758.66	67452.75	9873.31	8516.90	2.2528	-2.0817		
15	15	15	3043.15	-4168.56	-41286.58	69004.26	9345.39	598.33	2.7610	-1.8860	100.00	P - Pass
16	16	16	2353.68	-1048.58	8704.88	72124.43	8656.42	3718.50	2.2528	-2.0817		
17	17	17	2665.30	-1534.46	9016.96	71638.34	8968.50	3232.41	1.7920	-1.8755	100.00	P - Pass
18	18	18	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

The following columns are updated:

Table 3-3. Defect List Columns Updated After a Image-to-Layout Alignment Run

Column	Description
Shift X (um)	Specifies the X-coordinate shift value after the run.
Shift Y (um)	Specifies the Y-coordinate shift value after the run.
Original Layout X (um)	Specifies the original X location of the image center in the layout. This value is calculated by adding the Layout X (um) value to the Shift X (um) value.
Original Layout Y (um)	Specifies the original Y location of the image center in the layout. This value is calculated by adding Layout Y (um) value to the Shift Y (um) value.
Confidence Metric	It is not always possible to review each image manually to check whether the alignment is correct or not. The confidence metric automates the process of reviewing results. By checking these values, you can determine how good or bad the alignment is. The values are expressed as a percentage.

Table 3-3. Defect List Columns Updated After a Image-to-Layout Alignment Run (cont.)

<table>
<tr><th>Column</th><th>Description</th></tr>
<tr><td>Alignment Status</td><td>
Provides a quick check of the alignment result. The possible values for this column are Pass, Fail and Review. The values are determined based on the Pass and Review thresholds applied to the confidence metric value. These two threshold values are taken as input in the configuration file.
The default threshold value is 75 for Pass, and 50 for Review. For example, a confidence metric of above 75 results in Pass, between 50 and 75 results in Review, and below 50 translates to Fail as the alignment result.
You can modify the Alignment Status for each defect based on a manual review using the Alignment Status tab in the Defect Classification window.
Figure 3-10. Alignment Status Review

Note: The Alignment Status value is present in the Defect List only for defects with a SEM image.
</td></tr>
</table>

Note

For defect locations that do not have SEM images, the tool populates the Shift X and Shift Y columns with an common offset calculated by the tool. The offset is calculated using one of two methods: global or local. These methods are described in the “Global Offset” and “Local Offset” sections in this topic.

Global Offset

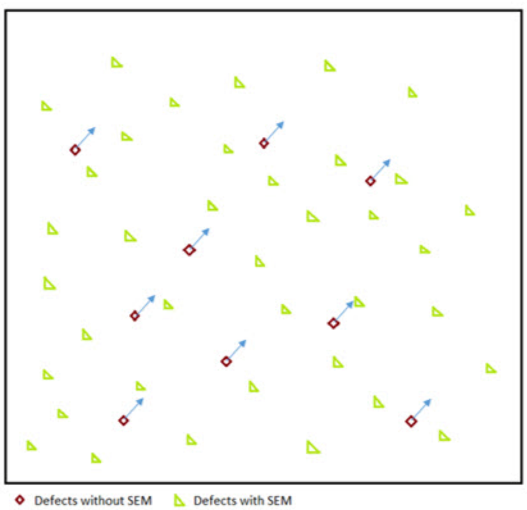
A global offset is a common shift value calculated by considering shift values obtained from high quality alignment of defects with SEM images and their respective design patterns. The quality of alignment is determined using a confidence metric. X and Y shift values can be considered as points in a 2-dimensional coordinate space (that are clustered) and the center of gravity (CoG) of the densest cluster is chosen as the global offset. For SEM images with repeated patterns, multiple shift values similar to the best shift value are considered for clustering.

- Global offsets can be used to assign Shift X and Shift Y values for defects without SEM images when the search_mode argument in the Image-to-Layout run configuration file is set as global. Figure 3-11 shows an example global offset.
- The global offset value is also used to fine tune the alignment location for defects with SEM images. In this case, the override_global_offset argument in the Image-to-Layout run configuration file is set as false (default).
- For defect IDs with SEM images that are not aligned correctly (with a low confidence metric), the global offset value is used as the alignment shift. This can be deactivated by setting the Image-to-Layout run configuration file argument set_align_fail_to_global to 0. The shift values are not replaced by a global offset.

The following issues apply to global offsets:

- Given the multiple applications of the global offset, the offset calculation is always enabled in the alignment flow.
- If the data set does not contain enough defects with SEM images that are correctly aligned, the tool does not calculate the global offset and X and Y are set to (0, 0).

Figure 3-11. Global Offset (All Defects Without SEM Have Same Shift)



Local Offset

A local offset is calculated only for defects that do not have SEM images. Unlike the global method, the local method calculates the shift for each entry separately.

For each defect without a SEM image, the local offset is calculated from the *n* number of nearby neighbors with SEM images that have good alignment scores (based on the confidence metric). The *n* value is specified using the min_defects_in_search argument in the Image-to-Layout run configuration file. Calibre DefectClassify uses the WaferX and WaferY coordinates to identify the neighbors. Figure 3-12 shows the shifts of defects without SEM images are dependent on the shifts of neighboring defects with SEM. In this case, the min_defects_in_search value is 3.

Figure 3-12. Local Offset (Defects Without SEM Images Have Different Shifts)

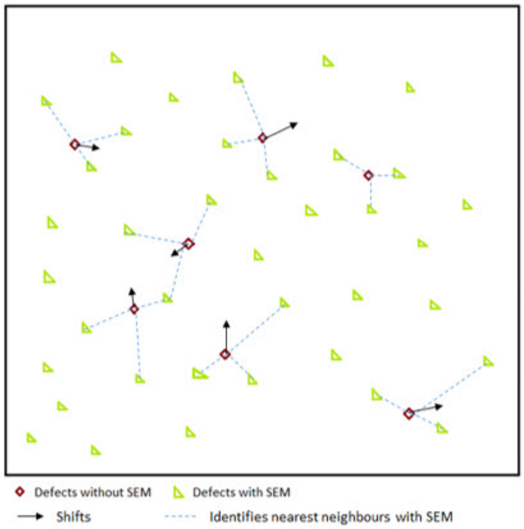


Image-to-Layout Alignment Run Configuration File

Configuration file for the Image-to-Layout Alignment tool

An Image-to-Layout Alignment tool run is configured by several parameters in a simple text file, specified in the Configuration File entry of the Image-to-Layout Alignment GUI.

Format

A simple text format file with a *.cfg* suffix (the path is specified in the Configuration File entry of the Image-to-Layout Alignment tool).

Parameters

- max_align_error_um *val*

  An optional parameter in microns that specifies the maximum possible misalignment between a SEM image center and the X or Y coordinate of a layout. The SEM image size is calculated in microns, and then the image is padded with this value in all 4 directions. A query is then performed for that region in the OASIS file. If the misalignment is greater than this value, no match occurs. The default value is 4.5.

- alignment_method *method_num*

  An optional parameter that selects the alignment algorithm for SEM images with both single and multi-layered layouts.

  For single-layered layout input:

  - The tool categorizes each SEM image into three image categories: sparse, semi-dense, and dense. This is done based on the pattern density of the image.

    For images categorized as semi-dense or desne, separate alignment methods are used. However, for sparse images, the tool uses five different alignment methods. The methods are selected using the *method_num* argument, including the following:

    - 1 — Edge based. This method is suitable for images with a strong contrast of gray-level intensities near pattern edges.

      Note

      For dense SEM images, when alignment_method 1 is specified, the tool switches to this sparse alignment method instead of the dense alignment algorithm.

    - 2 — Patch based. This method is suitable for images where the background texture is smooth, but the pattern has a texture consisting of small patches with different intensities

    - 3 (Default) — An automatic switch between methods 1 and 2 (edge-based and patch-based). This method is an automatic selection of either method 1 and 2, between edge-based and patch-based. The method selected is based on the properties of the defect image. When method 3 is specified, the tool detects the most suitable method for each defect and proceeds with the alignment. This means that for a single dataset, certain defects use the edge-based, and others the patch-based method.

Note

If a particular defect has multiple images, the tool can select different algorithms for different images based on the image properties.

- 4 — Automatic segmentation based. This method is suitable for defects where the pattern in the foreground has different gray-level intensities than the background.
- 5 — Adaptive edge based. This method is similar to method 1, but is required for defect images containing pattern edges with varying contrasts across the image. This may be related to the hierarchical level of the pattern (for example, the lower patterns have weaker contrasts), noise, or non-uniform illumination of the image (for example, containing a glare).
- 6 — Straight edge based. This method is similar to method 1, but suitable for images where oblique or slant edges are of equally strong contrast as pattern edges, thus making edge-based alignment (method 1) difficult. This method also works for images with non-uniform illumination.
- 7 — An automatic switch between method 1 and 6. This method (similar to method 3) is an automatic selection between method 1 & 6, between edge-based and straight-edge-based. The method chosen is based on the properties of the defect image. When method 7 is specified, the tool detects the most suitable method for each image and proceeds with the alignment.

Note

As with method 3, the auto-switching applies to each defect image, and can be different depending on the defect images.

For single layer input (determined based on the layer input in the setup file), if this parameter is not specified, algorithm 3 is selected by default.

- Multilayer input supports two methods:

Note

For merging different layers, it is assumed that the bottom layer is not visible where the top layer is present. Layers are stacked according to their hierarchy on the mask. For details, refer to Figure 3-5 on page 91 for information on layer hierarchy.

- 5 — Multilayer, contour based. This method is suitable for defects where the multilayer pattern in the foreground has different grey-level intensities than the background, similar to method 4. This is also suitable for multi-mask patterns where patterns of multiple layers are merged together without maintaining the hierarchy of the layer exposure.
- 6 — Multilayer, contour and edge based. This method is suitable for SEM images where noise is present on the pattern, the pattern feature is not smooth, or different pattern structures have different illuminations. Multiple layers are merged together without maintaining the hierarchy of the layer exposure.

If an invalid value is provided for multilayer input, an edge-based method similar to method 1 for single layer alignment is used by default. The difference in this case is that multiple layers are merged.

In cases of multilayer input determined based on the layer input in the setup file, if this parameter is not specified, algorithm 3 is selected by default.

Typically, single mask images can select algorithms 1 through 7, multi-mask images can select 5 and 6. In cases when this parameter value is within the range but there is no algorithm specific to that value, the tool always uses the default algorithm for respective cases based on whether the layout is single or multi-layered. Specifically, values 5 and 6 are then valid for both single layer and multilayer input. Once the layer data is input in the Layers field shown in Figure 3-5 on page 91, the tool determines which algorithm to use based on whether the input has single or multiple layers.

- search_mode {global | local}

  An optional parameter that specifies which search mode to enable, global or local. These are algorithms that calculate the alignment shift for entries without a SEM image.

  In global mode, Calibre DefectClassify uses a global offset as a shift value. In this case, all defects without a SEM image uses the same shift value. Refer to the "Global Offset" description in "Image-to-Layout Alignment Results Analysis" on page 96.

  Unlike the global method, the local method calculates the shift for each entry separately. It is calculated from the local neighboring defects with SEM images. Refer to the "Local Offset" description in "Image-to-Layout Alignment Results Analysis" on page 96.

  The default setting is global.

- min_defects_in_search *num*

  An optional parameter, used only in local mode, that specifies the number of neighbors with a SEM image that must be considered in order to calculate the alignment offset for a particular defect without a SEM. The default value is 3.

- invert_layout {1 | 0}

  An optional parameter used to internally invert the layout image before matching. It has two valid values: 1 or 0. 1 inverts the layout. 0 does not invert the layout. The default value is 0.

- invert_layout_indices *val*

  An optional parameter used to internally invert the one mask layer in the layout image before alignment. This is specifically used for multilayer SEM alignment. It assumes the index of the layer which requires inversion. The indexing starts from 0.

- confidence_thres *val*

  An optional parameter specifying that for each SEM image aligned, a confidence metric in terms of percentage accuracy is calculated to measure the accuracy of alignment.

  If the confidence metric value is greater than confidence_thres, then the alignment status is reported as Pass; otherwise, it is either Review or Fail depending on the review_thres value. The default value is 75.

- review_thres *val*

  An optional parameter that specifies the threshold to trigger a Review or Fail condition, as described in confidence_thresh. If the confidence metric value is greater than review_thres, then the alignment status is Review; otherwise, it is Fail. The default value is 50.

- pixel_size_scale_factor {<u>1</u> | 2}

  An optional parameter that controls the pixel size used for image processing. With a small pixel size, there are occasions where the querying of layout results in identifying the wrong polygon for smaller size patterns, such as a contact. This can affect the overall tool accuracy. In this case, the pixel size must be increased using pixel_size_scale_factor to get an accurate result. Two values are valid: 1 (the default) specifies that the pixel size does not change, and 2 increases the pixel size twice before processing the images. Any other value specified is set to the default.

- pixel_size_list *val1*,*val2*,...

  An optional parameter that enables you to specify a comma-separated list of potential pixel size values in a configuration file. Because calculation of pixel sizes for every defect from a corresponding defect can be a time-consuming process, you can instead analyze images selectively, then specify the list of potential files in the configuration file. The tool then automatically selects the pixel size that best corresponds to the defect image. For example, an entry in the configuration file might be written in this format:

  ```
  pixel_size_list 0.005, 0.0083
  ```

  There are two methods to specify pixel size:

  - Input CSV

    You can specify the pixel size for each defect in the input CSV in a separate column. If a pixel size value is listed in the CSV file, then that pixel size is used and the list of pixel sizes specified for the parameter pixel_size_list are ignored.

  - Configuration File

If pixel size field is left blank in the input CSV file, then the tool automatically chooses the best pixel size from the list of pixel sizes specified for the parameter pixel_size_list.

Note

If the pixel size for a defect does not exist in either of these methods, then no alignment attempt is made for that defect. A warning message is displayed and recorded in the output log.

Pixel size values specified with the pixel_size_list field must be between 0 and 1, inclusive.

- multi_mask_bias *val*

  An optional parameter used only for multimask alignment. Each of the mask layers images can be biased separately before merging them into a single layout image before alignment. This parameter specifies the bias values in a comma-separated list.

  The bias values must be integer numbers. The bias value list size should be at least equal to the number of mask layers. If you place more values than mask layers, and if the number of mask layers is *n*, then the first *n* values are considered for biasing while the rest are ignored. The default value is 0 (all layers).

- sem_pattern_bias *val*

  An optional parameter used to bias the thresholded SEM image. The bias values must be an integer number. Do not use this parameter unless a run requires it. The default value is 0.

- set_align_fail_to_global *val*

  An optional parameter that determines whether images that FAIL go to pass 2 to be output with the Global Offset, whether or not it was calculated. After calculating the Global Offset, each defect that FAILED in the process is shifted to align to a layout location with the SEM image by the Global Offset and the layout clip is queried in the new layout location. These defects are then output with the layout clipped to the size of the SEM images. The default value is 1.

- no_text_region *x1*, *y1*, *x2*, *y2*

  If text is present on the image, specify this parameter for better alignment results. The parameter is specified as following:

  ```
  no_text_region x1, y1, x2, y2
  ```

  where (*x1*, *y1*) is the top-left pixel coordinate and (*x2*, *y2*) is the bottom-right pixel coordinate in the image. It is assumed that the text printed region lies outside this area. The default region is whole image:

  ```
  no_text_region  0, 0, image_width, image_height
  ```

You must specify four valid positive integer values separated by commas. In other cases, a specific error message is displayed. If the user specified coordinates lie outside the image, the tool takes the default region described previously.

If the image does not contain text, then this parameter is not required.

- temp_um_bar_height_bottom *val*

  A micron bar is typically printed at the bottom of the SEM image. This optional parameter is used to specify the height of the micron bar position in the image in terms of the number of pixels. If the SEM image does not contain a micron bar, this option is not required. The default value is 0.

- preprocessing_steps *step1*, *step2*... *stepN*

  An optional parameter used to preprocess the SEM image to improve its quality, resulting in improved efficiency of the processing algorithms. This is identical to the preprocessing_steps parameter used in the SEM ADC contour extraction file as documented in "Contour Extraction Setup File" on page 122.

  Note

  The values specified using this parameter are applicable only to the alignment functionality of the tool.

- pixel_size_precision *val*

  An optional parameter that specifies the precision of the pixel size value when it is calculated using Field of View (FOV). It specifies how many decimal places are used to calculate the pixel size value. The range of this parameter is [3, 10]. The default value of this parameter is 7 (the default pixel size is calculated with a precision of 7 decimal places).

- match_threshold *threshold_percentage*

  An optional parameter that specifies a threshold value (in percentage) to determine a matching location for image pattern alignment. When a SEM image pattern is present in more than one location in the queried layout region, the SEM image may align to any of the multiple locations based on a match score. This threshold, expressed in a percentage, defines that fraction of a best match score that determines whether a location has the same pattern as the best matching location. If the location value is above the threshold, then the location is considered as a matching pattern. The default value of this parameter is 97 (if the best match score is 1, tool considers all the match locations which have a score above 0.97 or 97% as potential matches).

- override_global_offset {true | <u>false</u>}

  An optional parameter that specifies if defects are to be aligned using the global offset. By default, the tool attempts to find a global offset for a dataset, then aligns the defects within a short distance from global offset. To align all the defects near a user-defined layout location and not near the global offset, set this parameter to true. The default value is false.

- search_radius_from_centre_um *val*

  An optional parameter that specifies a radius around an input layout coordinate for alignment. When a SEM image pattern is present in more than one location in the queried layout region, the SEM image may align to any of these locations. You can specify a radius around the input layout coordinate using this parameter. When this parameter is specified, the tool attempts to find an alignment location within this search region and ignores other matches. If the tool is unable to find a suitable location within this specified area, it looks for other matches and reports the best one. This parameter is only considered when the confidence value of the alignment is greater than the review threshold, otherwise it is ignored. The default value of this parameter is the entire search region specified by max_align_error_um. This parameter is specified in microns.

- fov_list

  An optional parameter that contains a list of comma-separated FOV (Field of View) values corresponding to SEM images for that defect data set. The tool then automatically calculates and selects the best FOV for each defect image. Pixel size is then calculated from the FOV. The following is an example entry:

  ```
  fov_list 5 8 12
  ```

  You can select FOV values from either an input CSV or using fov_list.

  - Input CSV

    You can specify the FOV values for each defect in the input CSV file in separate columns using Klarf 1.8 format. If a FOV value is listed in the input CSV file, then that value is used. The list of FOV values specified using fov_list are then ignored.

  - Configuration File

    If the FOV field is left blank in the input CSV file, then the tool automatically chooses the best FOV from the list specified by fov_list.

  Note

  The following apply:

  - If the FOV value for a defect does not exist in the input CSV file, the tool looks for the pixel size value in the CSV file instead. If both are absent, then the tool checks the fov_list parameter in configuration file, followed by pixel_size_list. If nothing is present, then no attempt is made to align the defect. A warning message is displayed and it is recorded in the log.

  - Any of the FOV values entered in the fov_list field cannot be negative. A valid FOV value is greater than 0.

- dump_raw_layout {true | <u>false</u>}

  An optional parameter used to output the queried layout in BMP format in the *Images* directory of a user-specified output directory. If this parameter value is set to true, for each pattern or contact layer queried, the layout image is output with the name *clipped_layout_image_<def_ID>_<layer_number>.bmp*. The default value is false.

- alignment_roi {rect, *x1*, *y1*, *x2*, *y2* | center, *x1*, *y1*, *width*, *height*}

  An optional parameter used to specify a Region of Interest (ROI) on the SEM image. You can specify one of two possible syntax methods:

  - alignment_roi rect, *x1*, *y1*, *x2*, *y2*

    This syntax specifies a rectangle using top-left and bottom-right corners. The top-left corner is specified using *x1* and *y1* and the bottom-right corner uses *x2* and *y2*.

  - alignment_roi center, *x1*, *y1*, *width*, *height*

    This method specifies a center coordinate along with a width and height of the ROI image. The center coordinate is specified using *x1* and *y1*, *width* specifies the ROI image width, and *height* specifies the ROI image height.

  When using this keyword, the alignment and confidence metric are calculated only on the ROI image. However, the tool displays the alignment on the actual SEM image in the IMU, reporting the aligned coordinates relative to the actual image.

Examples

The following is an example configuration file with alignment-related parameters listed:

```
max_align_error_um 4.5
min_defects_in_search 5
invert_layout 1
alignment_method 5
search_mode local
invert_layout_indices 2
confidence_thres 75
review_thres 50
pixel_size_scale_factor 2
pixel_size_list 0.025, 0.00833
multi_mask_bias 0, -4, 2
no_text_region 0, 95, 480, 435
temp_um_bar_height_bottom 40
set_align_fail_to_global 0
preprocessing_steps NoiseRemoval_2(5) ContrastEnhancement_2
pixel_size_precision 6
```

Run Configuration File Parameter Summary

This section contains a summary of the behavior of the Image-to-Alignment tool when the run configuration file parameters are outside of the defined range.

Table 3-4. Tool Behavior Outside of Valid Range of Parameter

Parameter	Valid Range	Tool Behavior When Out of Range	Default Value	FOV Dependent?
max_align_error_um	Value > 0	Tool exits with error message.	N/A	Yes
alignment_method	Single layer: 1, 2, 3, 4, 5, 6, 7 Multiple layer: 5 or 6	Tool exits with error message.	3 (single-layer), separate algorithm (multiple layer)	Yes
search_mode	local, global	Tool exits with error message.	global	No
min_defects_in_search	integer value > 0	Tool exits with error message.	3	No
invert_layout	0, 1	The layer is inverted only when = 1.	0	No
invert_layout_indices	integer	Internally ignored if that layer number is not present.	N/A	No
confidence_thres	0 < value < 100	Tool exits with error message.	80	Yes
review_thres	0 < value < 100	Tool exits with error message.	50	Yes
pixel_size_scale_factor	0 < Double < 4	Tool exits with error message.	1	Yes
pixel_size_list	A list of comma separated values within the range > 0 and < 1 only	Tool exits with error message.	N/A	N/A

Table 3-4. Tool Behavior Outside of Valid Range of Parameter (cont.)

Parameter	Valid Range	Tool Behavior When Out of Range	Default Value	FOV Dependent?
multi_mask_bias	integer1, integer2...integer*n*	If any value other than integer is populated, that value is ignored. If the number of layers are *n* and more values are specified, only first *n* values are considered.	0 for all layers	Yes
sem_pattern_bias	integer	Default value is taken.	0	Yes
set_align_fail_to_global	0,1	If anything other than integer is given default value is chosen. If any integer value other than 1 is given, parameter is set to 0.	1	No
no_text_region	List of 4 integers. Cannot be negative.	x1 cannot be greater than x2. y1 cannot be greater than y2.	Image size	No
temp_um_bar_crop_bottom	integer value >= 0	Default value is taken if greater than image size.	0	No
preprocessing_steps	N/A	No preprocessing is performed.	N/A	No
pixel_size_precision	3<= Integer <= 10	Tool exits with an error for invalid values.	7	Yes
match_threshold	0 < value <= 100	Tool exits with an error for invalid values.	97	Yes
override_global_offset	true, false	Tool exits with error message.	false	No

Table 3-4. Tool Behavior Outside of Valid Range of Parameter (cont.)

Parameter	Valid Range	Tool Behavior When Out of Range	Default Value	FOV Dependent?
search_radius_from_centre_um	Double > 0	Tool exits with error message.	Same as max_align_error_um	Yes
fov_list	A list of comma separated values > 0	Tool exits with error message.	N/A	N/A
dump_raw_layout	true, false	Tool exits with error message.	false	No
alignment_roi	N/A	Tool exits with error message.	N/A	No

Chapter 4
SEM Automatic Defect Classification

The goal of the SEM Automatic Defect Classification (ADC) tool is to locate any defects, if present in a SEM image.

Using a SEM image and an aligned layout image as input, the SEM ADC tool attempts to match each layout edge with any significant gradient changes in the SEM image. The parts that do not have significant changes in the gradient are deemed to be potential defect locations. Such locations are further processed and, with the help of the gradient of the SEM, a defect contour is found.

Once all the potential defect locations are mapped to defect contours, they are analyzed for validity and based on location-specific rules and classified as different defects. An ordering of these defects (based on area of the defect contour and priority of the defect class) is generated. Finally, the highest priority defect with maximum coverage area is reported by the tool.

SEM ADC Defect Types

The SEM Automatic Defect Classification (ADC) tool can detect and classify a number of different defect types.

The following defect pattern types can be detected by a SEM ADC run:

- Missing Pattern

  A defect where the intended layout pattern is printed incorrectly or not at all printed.

- Pinch

  A missing pattern defect where the width of a pattern is reduced (the pattern becomes narrower).

- Line End Shortening

  A missing pattern defect that occurs at line-end types of patterns and the length of the pattern is reduced.

- Line End Extension

This is the opposite of Line End Shortening, where the length of the pattern is increased because of additional pattern material being printed at the short end of a line pattern.

- Bridge

  A defect that occurs when two patterns bridge (touch whereas they should not have).

- Added Pattern

  A defect that occurs when an unintended isolated additional pattern is printed.

- False

  This occurs where no defect is present in the field of view. In this case, the inspection machine may have erroneously considered a stray signal as a defect.

Defect Priority Configuration

Each defect has a default priority and only the defect with highest priority is reported when multiple defects are detected within a SEM image. The priority for each defect is configurable and can be changed in the *dat-ini-wafer.xml* file.

The classificationTable node SEM Defect Type Classification in *the dat-ini-wafer.xml* file contains details such as defect name, defect color, defect code, and defect priority for each defect. The following figure shows an example with default types and details listed.

Figure 4-1. SEM Defect Type Classification Node

```
▾<classificationTable tableName="SEM Defect Type Classification" higherPriorityToDatIni="true" classCodeUnion="false">
   <classificationData defectCode="B" defectType="Bridge" defectPriority="1" defectName="Bridge" defectColor="#0000ff"/>
   <classificationData defectCode="P" defectType="Pinch" defectPriority="2" defectName="Pinch" defectColor="#ffa500"/>
   <classificationData defectCode="M" defectType="MissingPattern" defectPriority="3" defectName="MissingPattern" defectColor="#ff0000"/>
   <classificationData defectCode="A" defectType="AddedPattern" defectPriority="4" defectName="AddedPattern" defectColor="#00ffff"/>
   <classificationData defectCode="S" defectType="LineEndShortening" defectPriority="5" defectName="LineEndShortening" defectColor="#ffff00"/>
   <classificationData defectCode="E" defectType="LineEndExtension" defectPriority="6" defectName="LineEndExtension" defectColor="#800080"/>
   <classificationData defectCode="R" defectType="Review" defectPriority="7" defectName="Review" defectColor="#008000"/>
   <classificationData defectCode="F" defectType="False" defectPriority="8" defectName="False" defectColor="#ee82ee"/>
   <classificationData defectCode="U" defectType="Unclassified" defectPriority="9" defectName="Unclassified" defectColor="#808080"/>
 </classificationTable>
```

The priority value of each defect can be interchanged while adhering to the following criteria:

- No two defect types can have same priority value.
- You cannot omit a predefined defect type nor can you specify a new defect type.

SEM ADC Workflow

The SEM ADC tool runs as an extension to the Image-to-Layout Alignment tool, operating following an alignment run.

Prerequisites

- Calibre DefectReview must be installed and invoked.

- Calibre DefectClassify must follow the prerequisites as described in "Calibre DefectClassify Prerequisites" on page 9.

Procedure

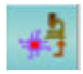
1. In Calibre DefectReview, click **Utilities > Image to Layout Alignment**, or click the Image to Layout Alignment icon  at the top of the Calibre DefectReview window. The Image to Layout Alignment dialog box appears.
2. Click the **Inputs/Outputs** tab and specify input and output parameters for a run as you would for the Image-to-Layout Alignment tool. This is described in detail in "Specifying Inputs and Outputs for Image-to-Layout Alignment" on page 87.
3. Click the **Settings** tab to enter parameters to configure an alignment run. This is described in detail in "Specifying Run Configuration Settings for Image-to-Layout Alignment" on page 92. However, there are configuration parameters that you must specify that are unique to a SEM ADC run. To enable a SEM ADC run following an Image-to-Layout Alignment run, you must set the parameter run_semadc to true in the configuration file.

Table 4-1. SEM ADC Configuration File Parameters

Parameter	Description
run_semadc	A required argument that specifies whether a SEM ADC run is to occur after an Image-to-Layout Alignment run. If set to true, SEM ADC is run. If set to false (the default), SEM ADC is deactivated.
semadc_minbias	Determines whether a negative bias is to be applied to the layout when SEM ADC attempts to bias layout edges to align to SEM edges.
semadc_maxbias	Determines whether a positive bias is to be applied to the layout when SEM ADC attempts to bias layout edges to align to SEM edges.
semadc_cdvar_thresh	Specifies the allowed variations of SEM dimensions in terms of percentage.
semadc_defect_thresh1	Specifies the minimum defect size in pixels. Any value below this threshold is ignored.
semadc_defect_thresh2	Specifies the maximum size of defect in pixels. For any value above this threshold, no variation is calculated.
semadc_output_all_defects	Specifies whether to report all defects detected in a SEM image (when set to 1) or not (when set to 0).

These parameters are described in detail in "SEM ADC Run Configuration File Extension" on page 120.

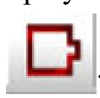
4. Click **Submit** to initiate the run.

Results

After the SEM ADC run completes, several images are generated and columns in the Defect List are updated.

The following images are displayed in the Image Measurement Unit pane of Calibre DefectReview:

- Sem — The original SEM image.
- Aligned Sem — A SEM image with the aligned layout overlaid on top. The color of the layout overlay is controlled by the LayerColor parameter of the SEMADCInfo node in the *dat-ini-wafer.xml* file.
- Layout — The binary image of aligned layout.
- Inverted Layout — The binary image of inverted version of aligned layout.
- Defect Bounding Boxes — A SEM image with defect bounding boxes overlaid on top.
- Defect Contours — A SEM image with accurate defect contours overlaid on top.
- Sem Contours — A SEM image with extracted contours overlaid on top. The color of the contour overlay is controlled by ContourColor parameter of the SEMADCInfo node in the *dat-ini-wafer.xml* file.

The bounding box and contours are displayed when you select the **Display defect contour on SEM Image** icon in the IMU toolbar .

Other images such as alignedLayoutOnSEM, ContoursOnSem, defectBBoxFinal, and ExtractedContours are additional images generated by internal algorithms. These serve the same purpose as the images previously described.

Note

The images listed are not shown by default; you must configure the IMU display to be able to see these images. The images are listed under **Other Images** in the right-click menu found in any IMU.

Note

If a single SEM image has multiple defects, then only the details of a defect with highest priority are populated in the Defect List.

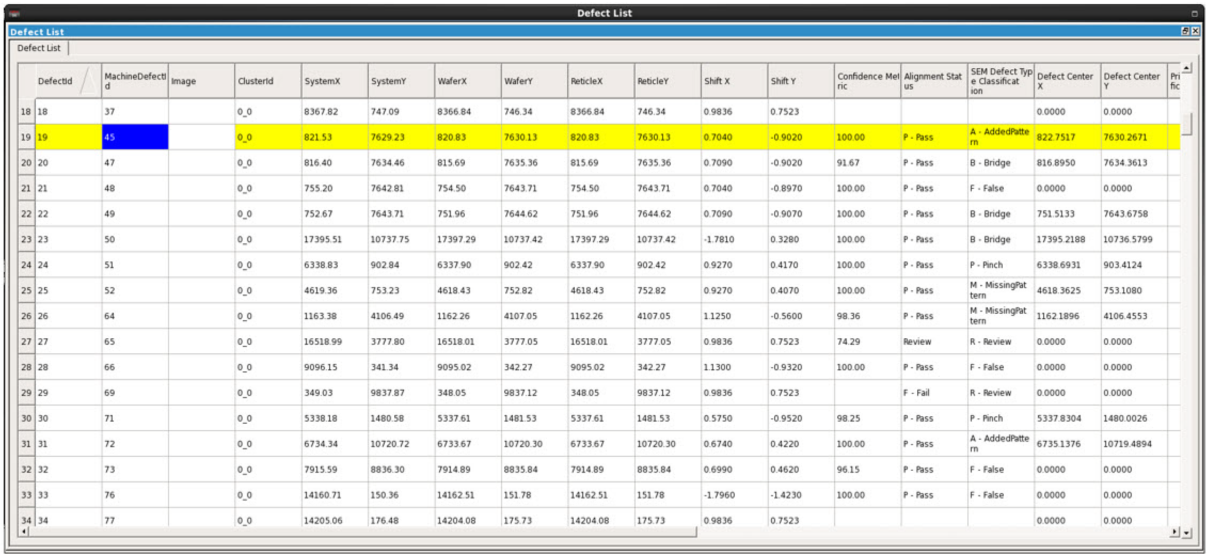
After the run is completed, the following columns in the Defect List are updated:

Table 4-2. Defect List Columns Updated After a Run

Column	Description
SEM Defect Type Classification	Displays the defect type detected (described in "SEM ADC Defect Types" on page 113). This column is present whether SEM ADC was run or not. If SEM ADC was not run, then the value Unclassified is populated for defects that have SEM images associated with them. The same value is also populated when SEM ADC is not run for a defect because of a failure in reading the image, lack of an input pixel size, or when the confidence metric value of a Image-to-Layout Alignment run is low.
Defect Center X	Specifies the X-coordinate of the center of the defect bounding box in microns relative to the layout coordinates. This column appears in the Defect List only when the *run_semadc* parameter in the configuration file is set to true. Only defects with associated SEM images have this field populated. Other defects have a default value of 0.
Defect Center Y	Specifies the Y-coordinate of the center of the defect bounding box in microns relative to the layout coordinates. This column appears in the Defect List only when the *run_semadc* parameter in the configuration file is set to true. Only defects with associated SEM images have this field populated. Other defacts have a default value of 0.

The updated defect list columns are shown in the following figure:

Figure 4-2. Defect List Column Update After a SEM ADC Run



	DefectId	MachineDefectId	Image	ClusterId	SystemX	SystemY	WaferX	WaferY	ReticleX	ReticleY	Shift X	Shift Y	Confidence Metric	Alignment Status	SEM Defect Type Classification	Defect Center X	Defect Center Y	Pri fic
18	18	37		0_0	8367.82	747.09	8366.84	746.34	8366.84	746.34	0.9836	0.7523				0.0000	0.0000	
19	19	45		0_0	821.53	7629.23	820.83	7630.13	820.83	7630.13	0.7040	-0.9020	100.00	P - Pass	A - AddedPattern	822.7517	7630.2671	
20	20	47		0_0	816.40	7634.46	815.69	7635.36	815.69	7635.36	0.7090	-0.9020	91.67	P - Pass	B - Bridge	816.8950	7634.3613	
21	21	48		0_0	755.20	7642.81	754.50	7643.71	754.50	7643.71	0.7040	-0.8970	100.00	P - Pass	F - False	0.0000	0.0000	
22	22	49		0_0	752.67	7643.71	751.96	7644.62	751.96	7644.62	0.7090	-0.9070	100.00	P - Pass	B - Bridge	751.5133	7643.6758	
23	23	50		0_0	17395.51	10737.75	17397.29	10737.42	17397.29	10737.42	-1.7810	0.3280	100.00	P - Pass	B - Bridge	17395.2188	10736.5799	
24	24	51		0_0	6338.83	902.84	6337.90	902.42	6337.90	902.42	0.9270	0.4170	100.00	P - Pass	P - Pinch	6338.6931	903.4124	
25	25	52		0_0	4619.36	753.23	4618.43	752.82	4618.43	752.82	0.9270	0.4070	100.00	P - Pass	M - MissingPattern	4618.3625	753.1080	
26	26	64		0_0	1163.38	4106.49	1162.26	4107.05	1162.26	4107.05	1.1250	-0.5600	98.36	P - Pass	M - MissingPattern	1162.1896	4106.4553	
27	27	65		0_0	16518.99	3777.80	16518.01	3777.05	16518.01	3777.05	0.9836	0.7523	74.29	Review	R - Review	0.0000	0.0000	
28	28	66		0_0	9096.15	341.34	9095.02	342.27	9095.02	342.27	1.1300	-0.9320	100.00	P - Pass	F - False	0.0000	0.0000	
29	29	69		0_0	349.03	9837.87	348.05	9837.12	348.05	9837.12	0.9836	0.7523		F - Fail	R - Review	0.0000	0.0000	
30	30	71		0_0	5338.18	1480.58	5337.61	1481.53	5337.61	1481.53	0.5750	-0.9520	98.25	P - Pass	P - Pinch	5337.8304	1480.0026	
31	31	72		0_0	6734.34	10720.72	6733.67	10720.30	6733.67	10720.30	0.6740	0.4220	100.00	P - Pass	A - AddedPattern	6735.1376	10719.4894	
32	32	73		0_0	7915.59	8836.30	7914.89	8835.84	7914.89	8835.84	0.6990	0.4620	96.15	P - Pass	F - False	0.0000	0.0000	
33	33	76		0_0	14160.71	150.36	14162.51	151.78	14162.51	151.78	-1.7960	-1.4230	100.00	P - Pass	F - False	0.0000	0.0000	
34	34	77		0_0	14205.06	176.48	14204.08	175.73	14204.08	175.73	0.9836	0.7523				0.0000	0.0000	

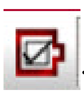
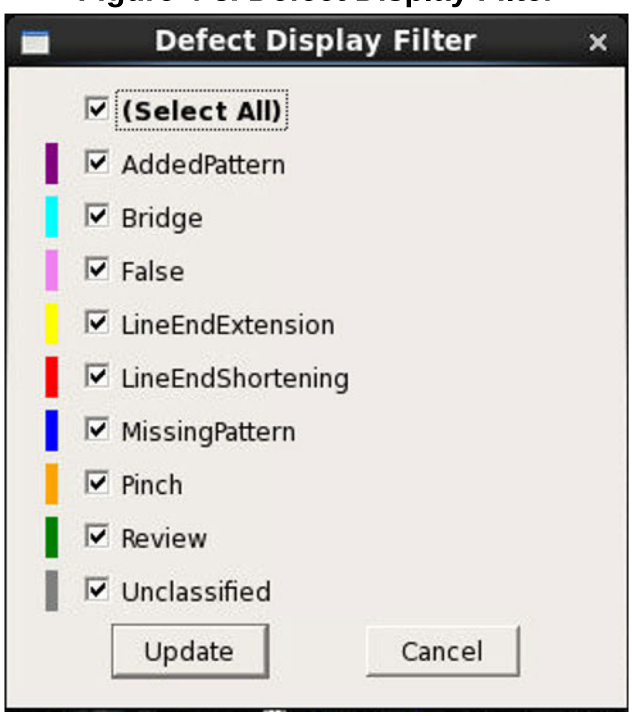
You can also display defect contours and boundary boxes for specific types by using the Defect Display Filter dialog box, invoked by clicking the **Defect Display Filter** icon in the IMU toolbar .

Figure 4-3. Defect Display Filter



This dialog box lists all supported defect types with a corresponding color to use as a visual indicator. To update the display of defect contours and defect bounding boxes, or de-select defect types (or click **Select All** to select all types), then click the **Update**.

SEM ADC Run Configuration File Extension

Configuration file extension used for Calibre DefectClassify SEM ADC

The SEM ADC tool runs as an extension to the Image-to-Layout Alignment tool and both use the same configuration file. However, the SEM ADC tool has its own unique set of parameters to set up a run.

Format

A simple text format file with a *.cfg* suffix (the path is specified in the Configuration File entry of the Image-to-Layout Alignment tool). Refer to "Image-to-Layout Alignment Run Configuration File" on page 101 for details on the Image-to-Layout Alignment tool configuration file.

Parameters

- **run_semadc** {**true** | **false**}

  A required argument that specifies whether a SEM ADC run is to occur after a Image-to-Layout Alignment run. If set to true, SEM ADC is run. If set to false (the default), SEM ADC is deactivated.

- semadc_minbias *val*

  The processing stage of SEM ADC attempts to bias the layout edges to align exactly to SEM edges so that defects can be caught correctly. This optional parameter determines whether a negative bias has to be applied to the layout. The default value is 0, which means no negative bias is applied.

- semadc_maxbias *val*

  The preprocessing stage of SEM ADC attempts to bias the layout edges to align exactly to SEM edges so that defects can be caught correctly. This optional parameter determines whether a positive bias is to be applied to the layout. The default value is 6, which means the tool attempts to bias the layout from 0 pixels to 6 pixels positively to align the edges correctly.

- semadc_cdvar_thresh *val*

  An optional parameter that specifies the allowed variations of SEM dimensions in terms of percentage. Any value below this threshold is considered a defect. The default value is 0.8 (80%).

- semadc_defect_thresh1 *val*

  An optional parameter that specifies the minimum defect size in pixels. Any value below this threshold is ignored. The default value is 2, which means all defects with a size less than 2 pixels are ignored and not reported.

- semadc_defect_thresh2 *val*

  An optional parameter that specifies the maximum size of defect in pixels. For any value above this threshold, no variation is calculated. The default value is 7, which means all defects with a size greater than 7 pixels are not subject to variation calculation and are reported as is.

- semadc_output_all_defects *val*

  An optional parameter that specifies whether to report all defects detected in a SEM image (when set to 1) or not (when set to 0). The default value is 0, which means only one defect of highest priority is reported in the output file. If the value is set to 1, then an additional output file is created that contains all defects detected for a SEM image populated in additional rows.

- semadc_method *method_num*

  An optional parameter that selects between two different processing methods used by the defect classification tool to detect defects. The following are the corresponding value for each of those methods (*method_num*):

  - 3 — Contour-based defect detection using adaptive segmentation. This is the default method.
  - 4 — Contour-based defect detection using an internal contour extraction method. This also requires a user-created setup file and setting the semadc_contour_extract_setup_file parameter.

- semadc_contour_extract_setup_file *path*

  When semadc_method is set to 4, this parameter specifies the path to a user-created setup file that configures internal contour extraction functionality enabled for contour-based defect detection. The setup file is described in "Contour Extraction Setup File" on page 122. The tool exits if this parameter points to either an invalid file path or if no file is specified when semadc_method is set to 4. It has no impact if semadc_method is set to 3.

Contour Extraction Setup File

Configuration file used by SEM ADC to configure contour extraction for contour-based defect detection.

When the semadc_method in the SEM ADC run configuration file is set to 4, it requires a user-created setup file that configures internal contour extraction functionality in order to perform contour-based defect detection. The semadc_contour_extract_setup_file parameter must also point to this file.

Format

A simple text format file (*setup.in*).

Parameters

Required Comparability Parameters

Note

The contour extraction setup file also requires several required parameters be present to be compatible with future contour extraction functionality external to SEM ADC. Though required for the SEM ADC run, their values are ignored.

- **version 1**

  This parameter is required for the contour extraction setup file syntax, but its value is ignored by SEM ADC.

- **alignment_error_in_nm 0**

  This parameter is required for the contour extraction setup file syntax, but its value is ignored by SEM ADC.

- **sem_pixel_size_nm** ***val***

  This parameter is required for the contour extraction setup file syntax, but its value is ignored by SEM ADC.

- **dump_processed_image false**

  This parameter is required for the contour extraction setup file syntax, but its value is ignored by SEM ADC.

- **debug_level 0**

  This parameter is required for the contour extraction setup file syntax, but its value is ignored by SEM ADC.

Contour Extraction Parameters

The following parameters are utilized by SEM ADC to configure the contour extraction algorithm used for defect detection.

- segmentation_method {1 | 2 | 5}

  An optional parameter specifying the segmentation method to be used. The default value of this parameter is 2.

  - Method 1 is used for SEM images with good contrast between pattern and background.
  - Method 2 is used for SEM images with strong edges. This method is slower than the method 1. This is the default value.
  - Method 5 is used for SEM images where a user-defined set of operations controlled by process_step extracts the contours.

- contours_to_extract {both | inner | outer}

  An optional parameter specifying the contours to extract.

  - both — Both inner and outer contours are extracted. This allows the nominal contour to be extracted. This is the default value.
  - inner — Only inner contour is extracted.
  - outer — Only outer contour is extracted.

- acs_points_maxlen *value_in_pixels*

  An optional parameter that specifies the maximum length between two points in the evolving contour. The default value of this parameter is 7. Use a lower value for this parameter for a smoother contour.

- preprocess_step *preprocessing_step1 preprocessing_step2... preprocessing_stepN*

  An optional parameter used to preprocess the SEM image to improve its quality, resulting in improved efficiency of the processing algorithms. This parameter specifies the processing steps that need to be applied before the contours are extracted from SEM image. The steps that can be combined are:

  - NoiseRemoval 1(5) — Median based noise removal with window size 5.
  - NoiseRemoval 2(1.7) — An edge-preserving noise removal algorithm. Increase the parameter to smooth the image.
  - NoiseRemoval 4(3,40) — Removes patches of flare and shadows. The first argument corresponds with the smaller patterns removed. The second argument corresponds with the larger patches removed. Increasing the first argument ignores the smaller patches. Increasing the second argument removes larger patches.
  - ContrastEnhancement_1 — Linear scaling of the image intensities.
  - ContrastEnhancement_2 — Non-linear scaling of the image intensities.

Note

The values specified using this parameter are applicable only to the alignment functionality of the tool.

- process_step *processing_step1 processing_step2 ... processing_stepN*

  This parameter is required if segmentation_method setup file parameter is set to 5. The contour extraction applies these transformations to the preprocessed SEM image and possibly apply postprocess_steps to the image. Processing steps that can be combined are as follows:

  - AutoThreshold(90, true) — Automatic threshold value is calculated based on the image and thresholding is performed at 90% of the calculated threshold value. Image is inverted if the second parameter is true.
  - Threshold(100, true) — Thresholding is performed at the intensity value (100) provided in the first parameter and the image is inverted if the second parameter is true.
  - Smooth_1(5) — The image is smoothed by performing the morphological operations. This removes small white blobs with a size smaller than 5 pixels. This must be applied after an AutoThreshold or Threshold operation. Increasing the parameter makes the contours smoother as well as ignores bigger white blobs.
  - Smooth_2(5) — The image is smoothed by performing the morphological operations. This step removes small dark blobs with a size smaller than 5 pixels. This should be applied after an AutoThreshold or Threshold operation. Increasing the parameter makes the contours smoother as well as ignores the bigger dark blobs.
  - Bias(5) — Biasing is applied to the contours. If the value is positive, the contours are positively resized. If the value is negative, the contours are negatively resized. This should be applied after an AutoThreshold or Threshold operation.

  This segmentation method is used if the SEM image has low noise and good contrast between the pattern and background.

- postprocess_step *processing_step1 processing_step2... processing_stepN*

  An option parameter that specifies the morphological operations to be applied on the extracted contours. Morphological operations include biasing and image smoothing. Postprocessing steps that can be combined are as follows:

  - Smooth_1(5) — The same as described in the process_step parameter.
  - Smooth_2(5) — The same as described in the process_step parameter.
  - Bias(5) — The same as described in the process_step parameter.
  - Rebuild — Smooths the extracted contour using curve fitting.
  - Invert — Inverts the extracted contours.

- contour_evolving_method {standard | custom | custom1}

  An optional parameter controlling the evolution of the contour. Default value of the parameter is "standard".

  - standard — Initial contour is based on the layout pattern itself and is evolved based on the gradient of the SEM image.
  - custom — Initial contour is located at the Center of Gravity of the layout pattern. The contour is evolved based on the gradient of the image.
  - custom1 — Initial contour is located at the Center of Gravity of the layout pattern. The contour is evolved based on the second-order gradient of the image.

- custom_contour_evolving_parameter4 *value*

  An optional parameter that specifies the window size around a pixel to find the maximum GL, which is then used to decide whether the current pixel's GL is sufficient to ignore for potential contour points. The default value of the parameter is 15.

- custom_contour_evolving_parameter5 *value*

  An optional parameter specifying the factor to max GL calculated within the window size provided for custom_contour_evolving_parameter4. This value is applied when considering if the current pixel is treated as noise or a potential contour pixel. This parameter value must be between 0 and 1 inclusive. The default value is 0.6.

- custom_contour_evolving_parameter9 {true | false}

  An optional parameter used to get correct contours if the pattern is a L or U shape and then initial contour is a small circle.

  If the value is set to true, the algorithm finds an automatic way to grow in the correct direction.

  If it is set to false, the direction is determined once and the contour grows accordingly. This is the default value.

- custom_contour_evolving_parameter10 {true | false}

  An optional parameter that is used with custom_contour_evolving_parameter4.

  If the value is set to true, the maximum GL is calculated along a line instead of a square of window size specified by custom_contour_evolving_parameter4. The line is in the direction of the maximum gradient at the pixel. The parameter is useful when corners are relatively weaker than horizontal and vertical edge pixels.

  The default value is false and uses a square for searching max GL in the neighborhood.

- custom_contour_evolving_parameter11 *value*

  An optional parameter used to ignore noise based on local statistics. This is an alternative for the noise handling method using custom_contour_evolving_parameter3 and custom_contour_evolving_parameter5, where statistics were global.

The value specified is used as a contrast threshold for noise. Any blob that has a contrast less than the threshold is considered as noise and is ignored during the contour detection algorithm. This algorithm is enabled only if value1 for custom_contour_evolving_parameter12 is set to true.

The method is best used for noisy and weaker-edged SEM images. The value of the parameter ranges from 0.02 to 0.06. The default value of this parameter is 0.

- custom_contour_evolving_parameter12 *value1*,*value2*,*value3*

  An optional parameter that specifies the shape of the initial contour to be used in segmentation method 2. It expects three comma-separated values (for example, true,7,5).

  If *value1* is true, the skeleton of the layout polygon is used as the initial contour, and if the *value1* is set to false, a circle with a 3-pixel radius is the initial contour. For a 2D pattern or noisy images, *value1* is recommended to be set to true. This is used to address leaks on the edge of the SEM image. If *value1* is true, set the value for custom_contour_evolving_parameter3 and custom_contour_evolving_parameter5 to 0. The value for parameter custom_contour_evolving_parameter11 must also be set between 0.02 to 0.06. The default value of *value1* is false.

  The *value2* option specifies the number of pixels to be eroded before getting the skeleton of the layout polygon.

  The *value3* option is used if the skeleton of the layout polygon is less than 3 pixels. In this case, the layout polygon is eroded by this amount and is used as an initial contour. This is primarily applies in the case of a contact pattern.

- acs_gamma *value*

  An optional parameter that specifies the weight given to a gradient during contour extraction. The recommended value is between 0.0 to 1.0. The default value of this parameter is 0.8.

  The following settings are recommended to use for 1D and good contrast images:

  ```
  custom_contour_evolving_parameter3 1.0
  custom_contour_evolving_parameter5 0.65
  ```

  Other custom parameters can be ignored. You must fine-tune the value for custom_contour_evolving_parameter5 to obtain a better result. The value can be between 0.06 to 1.0.

  The following settings are recommended to use for 2D and for relatively weaker edges:

  ```
  custom_contour_evolving_parameter3 0 // Fixed, no tuning
  custom_contour_evolving_parameter5 0 // Fixed, no tuning
  custom_contour_evolving_parameter11 0.02
  custom_contour_evolving_parameter12 true,5,7
  acs_gamma 0.2
  ```

  You must fine-tune the value for custom_contour_evolving_parameter11 and acs_gamma to get better results. The value can be between 0.01 to 0.07.

Examples

```
version 1
sem_pixel_size_nm 0.6592
alignment_error_in_nm 0
dump_processed_image false
debug level 0
segmentation_method 1
contours_to_extract inner
preprocess_step NoiseRemoval_5(7,10) ContrastEnhancement_2
postprocess_step Bias(3)
contour_evovling_method custom1
```

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.