SIEMENS EDA

Calibre® OPCsbar™ User's and Reference Manual

Software Version 2021.2

SIEMENS

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

Table of Contents

Chapter 1
Introduction to Calibre OPCsbar 13
Calibre OPCsbar Flow Overview 13
Calibre OPCsbar Requirements 13
Calibre OPCsbar Modes of Operation 14
Calibre Output Results 15
Syntax Conventions 18

Chapter 2
Calibre OPCsbar Usage 19
Scattering Bars for ISO Features 19
Gate-only Scattering Bars 22
Scattering Bars for Gates 27
Incremental Scattering Bars 30

Chapter 3
Calibre OPCsbar Syntax 33
SVRF Commands 34
OPCSBAR 35
Layer Arguments 38
in_layer 39
OFFSETLAYER 40
ref_layer 41
SBLAYER 42
Required Rule Arguments 44
SPACE 45
SPACELAYER 52
Manufacturing Constraint Arguments 54
LINEENDOFFSET 55
LINEENDSPACE 56
LINEENDTOLONGSPACE 57
MAXSBLENGTH 58
MAXSBWIDTH 60
MINEDGELENGTH 61
MINSBLENGTH 63
MINSBOFFSET 64
MINSBSPACE 66
MINSBWIDTH 67
Style Control Arguments 68
ANGLED 69
ANGLEEDGE 70
HORIZONTAL 71

INTERSECTION 72
NEGATIVE 74
OPPOSITEEXTENDED 75
STRICTCENTER 76
VERTICAL 77
WITHWIDTH 78
Cleanup Control Arguments 79
CENTERMERGE 80
COLINEARIZE 81
EXTENSION 83
JOG FILL 84
LINEENDMERGE 86
MINJOGWIDTH 88
NOCLEANUP 90
OPENT 91
OPTION CAREFUL_CLEANUP 92
OPTION CLEAN_RESEAT 93
OPTION FAST_MRC 94
OPTION INCLUSIVE_JOG_LENGTH 95
OPTION MINSBSPACE_CLEANUP_TIEBREAKER 97
OPTION PRIORITIZE_SPACE 99
PRIORITIZE BY LAYER 100
PRIORITYCENTER 101
SMOOTH 103
Processing Mode Arguments 104
OPTION NO_PUSH 105
OPTION PROCESSING_MODE 106
OPTION TILEMICRONS 107
OPTION VERIFICATION_MODE 108
SPACE and SPACELAYER Overview 108
Data Returned by Calibre OPCsbar 109

Chapter 4
Calibre OPCsbar Concepts 111

Measurement Concepts 111
Types of Scattering Bars 112
Scattering Bar Type and Arguments 113
Definition of Scattering Bar Treatment 115
SPACE Keyword 115
SPACELAYER Keyword 125
Scattering Bar Cleanup 128
How Priority Affects Cleanup 140
Scattering Bar Cleanup Priority 141

Chapter 5
Calibre OPCsbar Best Practices and Problem Resolution 147

Best Practices 147
Recommendations for Scattering Bar Generation 148

Problem Resolution 149

Appendix A
OPCSBAR Command Keywords 161
OPCSBAR Summary Table. 161

Appendix B
Calibre OPCsbar Rule File Template. 165
A Simple OPCSBAR Implementation. 165

Index

Third-Party Information

List of Figures

Figure 1-1. Merging Positive Scattering Bars with Original Data 17
Figure 1-2. Merging Negative Scattering Bars with Original Data 17
Figure 2-1. Rule File for ISO Feature Scattering Bars 21
Figure 2-2. Method to Obtain Gate-only Scattering Bars 23
Figure 2-3. Rule File for Gate-only Scattering Bars 24
Figure 2-4. Why Supply an Edge as the Input Layer (in_layer) 25
Figure 2-5. More Effects of Using a Reference Layer (ref_layer) 26
Figure 2-6. Method for Obtaining Gate Scattering Bars 27
Figure 2-7. Rule File for Gate Scattering Bars 28
Figure 2-8. PRIORITIZE BY LAYER Methodology 29
Figure 2-9. Method to Obtain Incremental Scattering Bars. 30
Figure 2-10. Rule File Excerpt for Incremental Scattering Bars 31
Figure 2-11. Using SBLAYER 32
Figure 3-1. Use of OFFSETLAYER as a Scattering Bar Keepout Area 40
Figure 3-2. Using CLEANSBLAYER. 43
Figure 3-3. WIDTH for Positive Scattering Bars 46
Figure 3-4. WIDTH Treatment by Edge 46
Figure 3-5. SBOFFSET Usage 47
Figure 3-6. Scattering Bar Generation with SBPITCH 47
Figure 3-7. Scattering Bars Generated Using CENTER 48
Figure 3-8. Scattering Bars Generated Using CENTERPITCH without Offset 49
Figure 3-9. Scattering Bars Generated Using CENTERPITCH with Offset 49
Figure 3-10. Bounded and Unbounded Spaces 50
Figure 3-11. SPACE Keyword Generating Positive Scattering Bars 51
Figure 3-12. SPACE Keyword Generating Negative Scattering Bars. 51
Figure 3-13. SPACELAYER General Usage 53
Figure 3-14. Where LINEENDOFFSET is Measured. 55
Figure 3-15. Where LINEENDSPACE is Measured. 56
Figure 3-16. LINEENDSPACE during CLEANUP 56
Figure 3-17. Where LINEENDTOLONGSPACE is Measured 57
Figure 3-18. MAXSBLENGTH Example 58
Figure 3-19. Target Contact too Close for SBAR Alignment 59
Figure 3-20. Scattering Bars Cut Due to Conflicting Constraints 59
Figure 3-21. Where MAXSBWIDTH is Measured 60
Figure 3-22. MINEDGELENGTH and Line-ends. 61
Figure 3-23. Impact of MINEDGELENGTH 62
Figure 3-24. Where MINSBOFFSET is Measured 64
Figure 3-25. MINSBOFFSET with Multiple Conditions 65
Figure 3-26. Where MINSBSPACE is Measured 66
Figure 3-27. Where MINSBWIDTH is Measured. 67

Figure 3-28. INTERSECTION Options 72
Figure 3-29. Partial-Cut Negative Scattering Bars 74
Figure 3-30. CENTERMERGE 80
Figure 3-31. COLINEARIZE Example 82
Figure 3-32. Extensions on Positive Scattering Bars 83
Figure 3-33. Styles of Jog Interaction 84
Figure 3-34. Merging at Line-ends 86
Figure 3-35. Resolving Merge Conflicts 87
Figure 3-36. Measuring Jog Widths 89
Figure 3-37. Using NOCLEANUP 90
Figure 3-38. Cleanup with and without OPENT using INTERSECTION N 91
Figure 3-39. INCLUSIVE_JOG_LENGTH 96
Figure 3-40. OPTION MINSBSPACE_CLEANUP_TIEBREAKER 98
Figure 3-41. PRIORITYCENTER 102
Figure 3-42. Typical SPACE Rule 109
Figure 3-43. Typical SPACELAYER Rule 109
Figure 4-1. Where OFFSET and SPACE are Measured 111
Figure 4-2. Two Types of Scattering Bars 112
Figure 4-3. Full- and Partial-Cut Negative Scattering Bars 113
Figure 4-4. Comparing MINSBOFFSET Keepout Areas 114
Figure 4-5. Measuring Positive and Negative SBOFFSET Values 114
Figure 4-6. How OPCSBAR Generates Scattering Bars using SPACE 116
Figure 4-7. Specifying Edges With SPACE 116
Figure 4-8. Defining Multiple Contiguous SPACE conditions 117
Figure 4-9. Breaking Edges During Classification 118
Figure 4-10. Scattering Bar Treatment by Classification 119
Figure 4-11. Facing Edges Classified by Space 119
Figure 4-12. Scattering Bars Generated by Classification 120
Figure 4-13. Scattering Bars Created for Two Boundary Edges on the in_layer 121
Figure 4-14. Scattering Bars Generated for a Single Edge 121
Figure 4-15. WIDTH Example for Positive Scattering Bars 122
Figure 4-16. WIDTH Example for Negative Scattering Bars 123
Figure 4-17. Comparing WIDTH Classification Options 124
Figure 4-18. SPACE Keyword and Negative Scattering Bars 125
Figure 4-19. Generation of Scattering Bars from SPACELAYER 126
Figure 4-20. Scattering Bars Generated using SPACELAYER 126
Figure 4-21. Cleaning up DRC and MRC Violations 128
Figure 4-22. Measurement Regions for OFFSET and SPACE Rules 131
Figure 4-23. Parameters Used to Define Edge Spacing and Length 132
Figure 4-24. Raw Scattering bars with Non-Orthogonal Angles 133
Figure 4-25. MINSBWIDTH and Jogs 134
Figure 4-26. Jog Interaction Styles 135
Figure 4-27. Jog Fill Between Non-Touching Scattering Bars 135
Figure 4-28. Jog Fill Between Non-Touching Scattering Bars (cont.) 136
Figure 4-29. Achieving an “N” Intersection 138

Figure 4-30. Achieving “L” Intersections 139
Figure 4-31. Resolving “L” Intersections that Result in “U” 140
Figure 4-32. Cleaning of Overlapping Scattering Bars 140
Figure 4-33. Cleanup Based on Prioritization 142
Figure 4-34. Island Layer Prioritization 143
Figure 4-35. Island Layer Prioritization (cont.) 143
Figure 4-36. Cleanup Based on Prioritization by Layer 144
Figure 4-37. Scattering Bar Prioritization Graph. 144
Figure 4-38. Cleanup Based on SBLAYER Prioritization 145
Figure 5-1. Calculating Minimum SPACE 149
Figure 5-2. Isolation of Incorrectly Placed Scattering Bars. 151
Figure 5-3. NOCLEANUP Usage 156

List of Tables

Table 1-1. Syntax Conventions 18
Table 3-1. SVRF-based OPCSBAR Command 34
Table 3-2. OPCSBAR Argument Groups 36
Table 3-3. Layer Arguments 38
Table 3-4. Required Rule Arguments 44
Table 3-5. Manufacturing Constraint Arguments 54
Table 3-6. Style Control Arguments 68
Table 3-7. Cleanup Control Arguments 79
Table 3-8. Processing Mode Arguments 104
Table 4-1. How Arguments Impact Scattering Bar Type 113
Table 4-2. Comparing SPACE and SPACELAYER Keywords 115
Table 4-3. Measuring Spaces for Edge Classification 117
Table 4-4. Cleanup Keywords 128
Table 4-5. Defining Spacing Constraints Between Different Types of Edges 132
Table 4-6. Internal Angles Remaining After Cleanup 136
Table 5-1. Recommendations for Scattering Bars 148
Table 5-2. Variables to Define 149
Table 5-3. Values for Low End of SPACE Constraint 149
Table 5-4. Resolving Cleanup Problems 152
Table 5-5. Resolving Design Problems 153
Table 5-6. Scattering Bars Returned Based on Priority 153
Table 5-7. Resolving Rules Problems 155
Table 5-8. Comparing Methods for Finding CD Features 158
Table A-1. OPCSBAR Keywords 161

Chapter 1
Introduction to Calibre OPCsbar

The Calibre® OPCsbar™ tool is a single SVRF operation that automates the creation of scattering bars.

Calibre OPCsbar Flow Overview 13
Calibre OPCsbar Requirements 13
Calibre OPCsbar Modes of Operation 14
Calibre Output Results 15
Syntax Conventions 18

Calibre OPCsbar Flow Overview

This single operation performs several tasks.

1. Identifies portions of the mask design that require scattering bars.
2. Creates scattering bars according to user-defined rules[1].
3. Checks for spacing and width violations caused by the newly-created scattering bars, and modifies the scattering bars as needed to eliminate those errors.
4. Checks for potential manufacturing and printing problems and modifies those scattering bars that would cause problems.

Calibre OPCsbar Requirements

Before invoking the Calibre hierarchical engine, you must set all required environment variables.

Calibre tools require the CALIBRE_HOME environment variable to be set. See "Setting the CALIBRE_HOME Environment Variable" in the *Calibre Administrator's Guide* for details.

1. The scattering bars are created on a separate layer which you can merge with the original layer using Boolean operations. For positive scattering bars you use the OR operation. For negative scattering bars you use the NOT operation.

Required Input Files

When you invoke the Calibre® nmDRC™ hierarchical engine you supply two forms of input:

- **One or more layout databases**

  GDS databases containing the design layer or layers to which you will add scattering bars. To process layer data using the OPCSBAR command, you must explicitly load them into Calibre using the LAYER statement.

- **One or more rule files**

  ASCII files composed of legal Standard Verification Rule Format (SVRF) statements. These statements control:

  - The design environment.
  - Loading of data from the GDS database(s).
  - Layer preprocessing (sizing, generating/isolating new layers, and so on).
  - Design and manufacturability checks.
  - Invoking Calibre OPCsbar.
  - Writing the final results to a results database.

Calibre OPCsbar Modes of Operation

Calibre OPCsbar is a Calibre-based application invoked by using the OPCSBAR command.

You control this operation two ways:

- Through the layer data you pass to the operation.
- Through the keywords and rules you use to configure the operation.

Calibre Command Syntax

To run Calibre OPCsbar, you invoke the Calibre nmDRC hierarchical engine and enter the calibre command in the command line of a shell window. The basic syntax is as follows:

```
calibre -drc [-hier] [-turbo [n]] [-turbo_litho [N]] [-turbo_all] rules
```

where:

-drc — A required switch specifying DRC checking.

-hier — An optional switch specifying hierarchical DRC checking (nmDRC-H).

-turbo *n* — An optional argument that selects multi-threaded parallel processing for DRC. The value of n specifies the number of CPUs to use for non-RET processes. If you do not specify n, the application uses the maximum number of CPUs on the machine. For more information on multi-threaded processing, refer to the *Calibre Administrator's Guide*.

-turbo_litho *N* — An optional argument that selects multi-threaded parallel processing for batch operations invoked by any of the RET operations, including Calibre OPCsbar. The value of N specifies the number of CPUs to use for these batch processes. If you do not specify N, the application uses the maximum number of CPUs on the machine. Note that the number specified for -turbo-litho does not need to agree with the number specified for -turbo. For more information on multi-threaded processing, refer to the *Calibre Verification User's Manual*.

-turbo_all — An optional argument used in conjunction with -turbo and/or -turbo_litho. When the number of licenses specified in -turbo and -turbo_litho are not available, the Calibre nmDRC hierarchical engine normally runs with fewer threads than requested. This command line option causes the Calibre nmDRC hierarchical engine to exit if it cannot get the number of licenses specified.

rules — A required user-supplied argument specifying the name of the SVRF rule file that contains the Calibre OPCsbar operation that executes the batch tool.

Calibre Output Results

The Calibre nmDRC hierarchical engine saves results and status information from each run described in various forms.

Note

Due to internal considerations involving hierarchy management, Calibre OPCsbar cannot guarantee identical output between a hierarchical run and a flattened run on the same data; nor can identical output from hierarchical runs be guaranteed across Calibre versions.

The Results Database

This database stores the results of the Calibre run. The Calibre nmDRC hierarchical engine writes results to the database file specified with the DRC RESULTS DATABASE statement in the SVRF rule file. If this is a GDS database, which is typical of Calibre OPCsbar runs, the Calibre nmDRC hierarchical engine writes all the data to layer 0. Using the DRC CHECK MAP statement, you can assign different types of data to different layers and/or instruct the application to write the results to different database files. These commands are detailed in the *Standard Verification Rule Format (SVRF) Manual*.

The Calibre Transcript

The Calibre nmDRC hierarchical engine automatically generates a text transcript at run-time and sends the transcript to stdout, which is typically the shell window from which you invoked

the application. During the Calibre run, the transcript displays event logs, warning messages, and summary information. This transcript documents the tasks the application performs: compiling the rule file, loading in the design data, and operating on that data.

The information the Calibre OPCsbar operation writes to the transcript includes some information about how long processing takes. The processing time it reports is the time OPCSBAR spends processing data while adding scattering bars. This information is designed to help you evaluate and optimize performance.

For example:

```
<HDBG/HPRF: OPCSBAR classification complete.
   CPU TIME = 4  REAL TIME = 3  LVHEAP = 2/5/5>
<HDBG/HPRF: OPCSBAR prepare prioritize by layer complete.
   CPU TIME = 0  REAL TIME = 1  LVHEAP = 2/5/5>
<HDBG/HPRF: OPCSBAR generation complete.

   CPU TIME = 1  REAL TIME = 1  LVHEAP = 1/8/8>
<HDBG/HPRF: OPCSBAR merge complete.
   CPU TIME = 0  REAL TIME = 0  LVHEAP = 1/8/8>
<HDBG/HPRF: OPCSBAR push after priority complete.
   CPU TIME = 0  REAL TIME = 0  LVHEAP = 1/8/8>
<HDBG/HPRF: OPCSBAR prioritize complete.

   CPU TIME = 62 REAL TIME = 63 LVHEAP = 1/8/8>
<HDBG/HPRF: OPCSBAR cleanup complete.
   CPU TIME = 15 REAL TIME = 15 LVHEAP = 1/8/8>
<HDBG/HPRF: OPCSBAR push after cleanup complete.
   CPU TIME = 1  REAL TIME = 0  LVHEAP = 1/8/8>
```

It is good practice to save the transcript by redirecting the output to a file, which you can do when you invoke the Calibre nmDRC hierarchical engine, by using the pipe or tee syntax, as follows:

```
calibre -drc -hier -turbo -turbo_litho opc.drc | tee opc.log
```

For a more complete description of the Calibre transcript, refer to the "Calibre nmDRC Results" chapter of the *Calibre Verification User's Manual.*

The DRC Results Summary

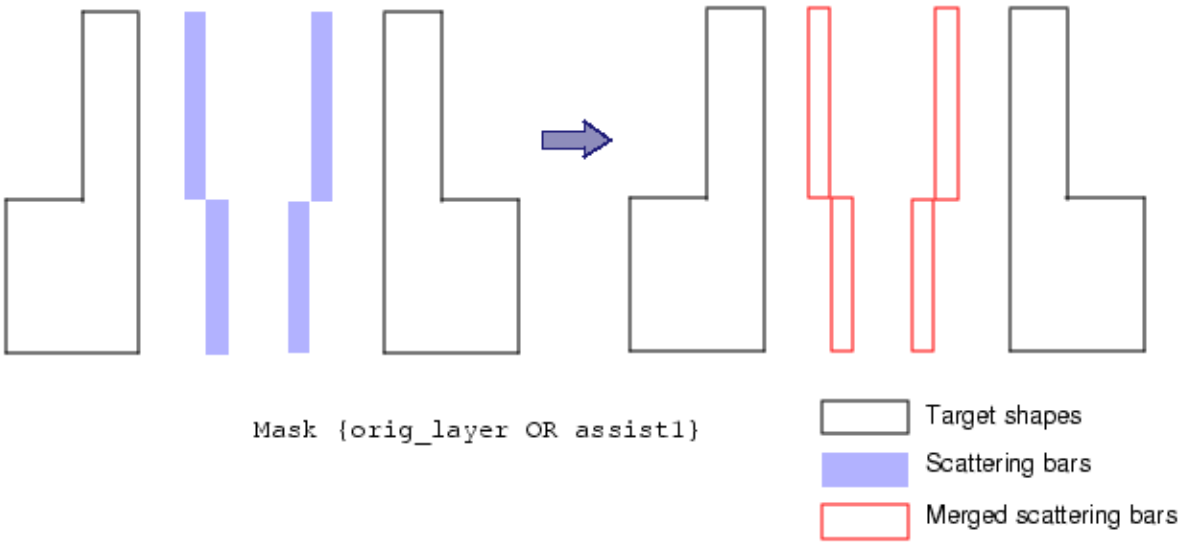
When you specify a pathname using the DRC SUMMARY REPORT statement (detailed in the *Standard Verification Rule Format (SVRF) Manual*), the Calibre nmDRC hierarchical engine generates a results summary text file and saves it to this file. The file contains heading information, describing the particulars of the run, runtime warnings list, statistics for the layers before and after processing, and a runtime summary.

Calibre OPCsbar Results

Calibre OPCsbar returns layer data containing the scattering bars it creates. These scattering bars are on a separate layer. To combine the scattering bars with the design data, you must merge the scattering bar layer with the original layer using Boolean operations.

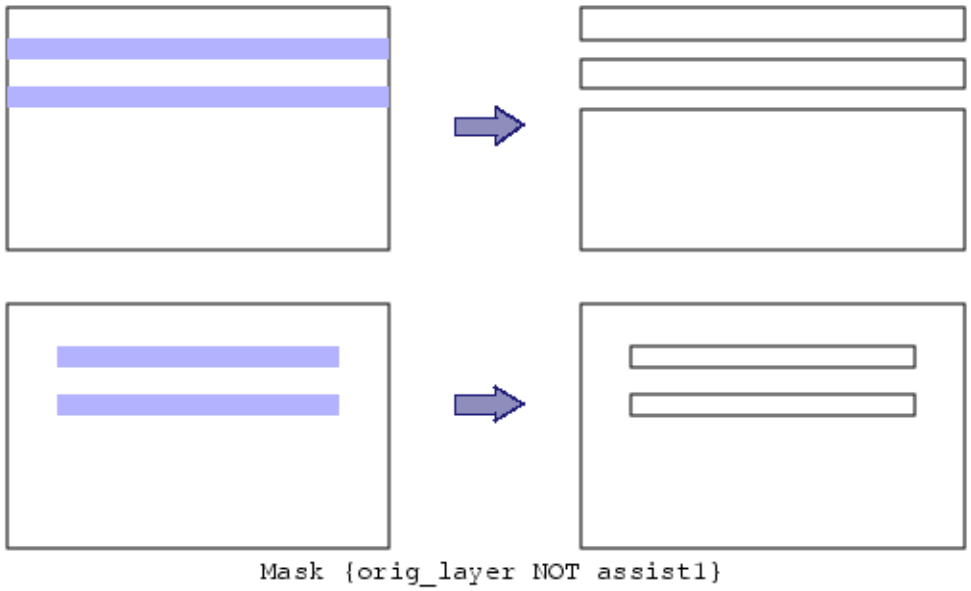
- For positive scattering bars you would use the OR operation.

Figure 1-1. Merging Positive Scattering Bars with Original Data



- For negative scattering bars you would use the NOT operation.

Figure 1-2. Merging Negative Scattering Bars with Original Data



Syntax Conventions

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

Table 1-1. Syntax Conventions

Convention	Description
Bold	Bold fonts indicate a required item.
Italic	Italic fonts indicate a user-supplied argument.
`Monospace`	Monospace fonts indicate a shell command, line of code, or URL. A bold monospace font identifies text you enter.
Underline	Underlining indicates either the default argument or the default value of an argument.
UPPercase	For certain case-insensitive commands, uppercase indicates the minimum keyword characters. In most cases, you may omit the lowercase letters and abbreviate the keyword.
[ ]	Brackets enclose optional arguments. Do not include the brackets when entering the command unless they are quoted.
{ }	Braces enclose arguments to show grouping. Do not include the braces when entering the command unless they are quoted.
‘ ’	Quotes enclose metacharacters that are to be entered literally. Do not include single quotes when entering braces or brackets in a command.
| or ||	Vertical bars indicate a choice between items. Do not include the bars when entering the command.
…	Three dots (an ellipsis) follows an argument or group of arguments that may appear more than once. Do not include the ellipsis when entering the command.
Example: **DEVice** {***element_name*** [‘(’*model_name*‘)’]} ***device_layer*** {***pin_layer*** [‘(’*pin_name*‘)’] …} [‘<’*auxiliary_layer*‘>’ …] [‘(’*swap_list*‘)’ …] [BY NET | BY SHAPE]	

Chapter 2
Calibre OPCsbar Usage

There are four usage scenarios provided to introduce you to the capabilities of Calibre OPCsbar.

Experiment with each scenario by copying the rule files and running them against your own data. The scenarios are:

- Scattering Bars for ISO Features
- Gate-only Scattering Bars
- Scattering Bars for Gates
- Incremental Scattering Bars

Once you understand how Calibre OPCsbar affects the data, modify the rules to see how small changes produce significant differences in results. In general, throughout each of these scenarios, ensure that:

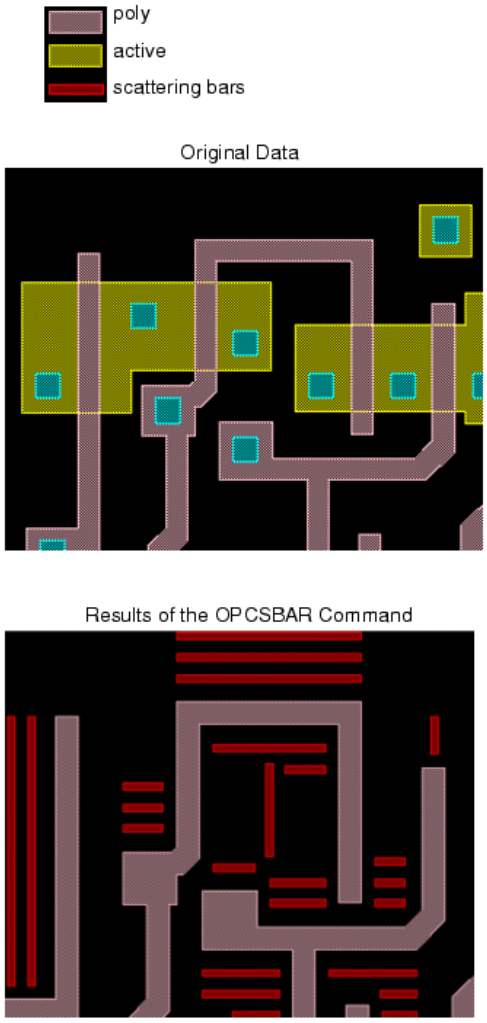
- You name a small layout *sbar_test.gds* (the rules file points to a design with this name). Using a small layout reduces run time.
- Poly is on layer 4 and Active is on layer 2.
- The rule file is an ASCII file named *sbar_test.drc*.
- The rule file and the layout file are in the directory from which you invoke Calibre OPCsbar.

Scattering Bars for ISO Features **19**
Gate-only Scattering Bars **22**
Scattering Bars for Gates **27**
Incremental Scattering Bars **30**

Scattering Bars for ISO Features

Rules classify edges based on distances to nearby features that generate scattering bars when the distance is greater than one micron.

Method to Obtain Scattering Bars for ISO Features



These methods are typically used to write a rule file for scattering bar treatment, and review the results:

1. Classify scattering bar treatment based on the distance to the nearest feature using the SPACE keyword.
2. Input all required layers to the OPCSBAR command.
3. Return the scattering bar data as a separate layer.
4. Run Calibre OPCSBAR using the rule file:

   ```
   calibre -drc -hier rules.svrf
   ```

5. To view the results, open the output design file. For example:

   ```
   calibrewb result.oas
   ```

Figure 2-1. Rule File for ISO Feature Scattering Bars

```
//*************************************************************************
// Mentor Graphics (c) 2014
//*************************************************************************

// ********************************************************Input Section
LAYOUT SYSTEM GDSII
LAYOUT PATH "sbar_test.gds"
LAYOUT PRIMARY "top"
FLAG SKEW YES
LAYOUT ERROR ON INPUT YES
PRECISION 1000
RESOLUTION 1
LAYER poly 4
// *******************************************************Output Section
DRC MAXIMUM RESULTS ALL
DRC RESULTS DATABASE "sbar_test1.gds" GDSII
DRC SUMMARY REPORT "/dev/null"
DRC MAXIMUM VERTEX 199
DRC KEEP EMPTY YES
DRC CHECK MAP poly 4
DRC CHECK MAP assist 3

// *******************************************************OUTPUT LAYERS
poly {COPY poly}

assist {OPCSBAR poly
   MINSBOFFSET  0.11
   MINSBSPACE   0.080
   MINSBWIDTH   0.059
   MINSBLENGTH  0.20
   INTERSECTION N
   SPACE >=1.0 <1.2 SBWIDTH 0.06  CENTER
   SPACE >=1.2 <1.4 SBWIDTH 0.06  SBOFFSET 0.16
   SPACE >=1.4 <1.6 SBWIDTH 0.06  SBOFFSET 0.16
   SPACE >=1.6 <1.8 SBWIDTH 0.06  SBOFFSET 0.16
      SBWIDTH 0.06  SBOFFSET 0.315
   SPACE >=1.8 <2.0 SBWIDTH 0.06  SBOFFSET 0.16
      SBWIDTH 0.06  SBOFFSET 0.315
      SBWIDTH 0.06  CENTER
   SPACE >=2.0 SBWIDTH 0.06  SBOFFSET 0.16
      SBWIDTH 0.06  SBOFFSET 0.315
      SBWIDTH 0.06  SBOFFSET 0.47
}
```

The DRC CHECK MAP statements define the layer assignment for the output of this run.

How OPCSBAR Applies Rules

During classification, OPCSBAR breaks the original edges into groups that meet one specific SPACE constraint.

For information regarding edge breaking, refer to the SPACE keyword section entitled Edges.

OPCSBAR generates scattering bars according to SBOFFSET, CNETER, and SBWIDTH commands and their parameters as specified in the rules.

OPCSBAR then cleans up the initial scattering bars and applies MRC rules to them by trimming or removing them.

Gate-only Scattering Bars

Isolate the gate poly and input it to Calibre OPCsbar for treatment.

Figure 2-2. Method to Obtain Gate-only Scattering Bars

Method

1. Write a rule file that defines the scattering bar treatment required (see Example 2-2).
2. In the rule file you must:

   — Derive the gate_poly layer:

   ```
   temp = poly AND active
   gate_poly = temp COIN EDGE poly
   ```

   — Input the gate_poly layer as the in_layer to OPCSBAR.

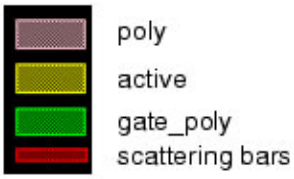
   — Input the poly layer as the ref_layer to OPCSBAR.

   ```
   assist {OPCSBAR gate_poly poly
      MINSBOFFSET  0.11
      <sbar recipe>
   ```

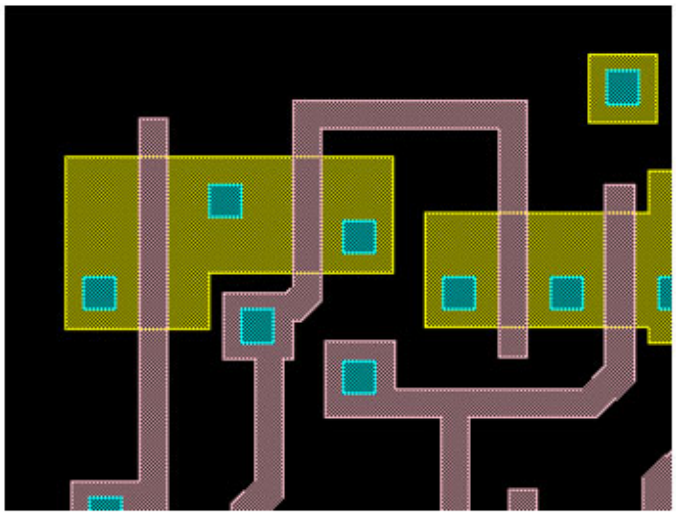
3. Run calibre using this rule file as input:

   ```
   % calibre -drc sbar_test.drc
   ```

4. To view the results, open sbar_test1.gds in Calibre WORKbench.



Original Data



Results of the OPCSBAR Command

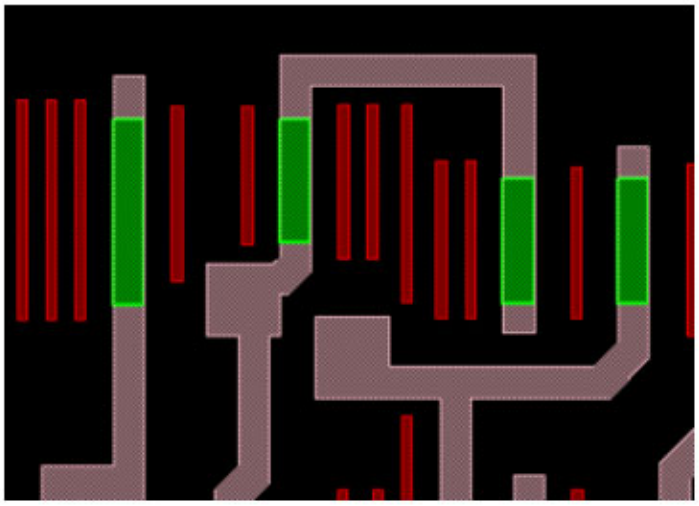


Figure 2-3. Rule File for Gate-only Scattering Bars

```
//*************************************************************************
// Mentor Graphics (c) 2014
//*************************************************************************

// ********************************************************Input Section
LAYOUT SYSTEM GDSII
LAYOUT PATH "public_html/gds_files/sbar_test.gds"
LAYOUT PRIMARY "top"
FLAG SKEW YES
LAYOUT ERROR ON INPUT YES
PRECISION 1000
RESOLUTION 1
LAYER poly  4
LAYER active 2
// ********************************************************Output Section
DRC MAXIMUM RESULTS ALL
DRC RESULTS DATABASE "public_html/gds_files/sbar_test2.gds" GDSII
DRC SUMMARY REPORT "/dev/null"
DRC MAXIMUM VERTEX 199
DRC KEEP EMPTY YES
DRC CHECK MAP poly 4
DRC CHECK MAP gate_poly 0
DRC CHECK MAP assist  3
// ****************************************The DRC RECIPE STARTS HERE
//
// Generate an edge layer containing edges on gate poly only
//
temp = poly AND active
gate_poly = temp coincident edge poly
// ********************************************************OUTPUT LAYERS
poly {COPY poly}
gate_poly {COPY gate_poly}
assist {OPCSBAR gate_poly poly
   MINSBOFFSET  0.11
   MINSBSPACE   0.080
   MINSBWIDTH   0.059
   MINSBLENGTH  0.130
   EXTENSION    0.10
   INTERSECTION N
   ANGLED
   SPACE >=0.32 <0.54 SBWIDTH 0.06 CENTER
   SPACE >=0.54 <0.70 SBWIDTH 0.06 SBOFFSET 0.16
   SPACE >=0.70 <0.96 SBWIDTH 0.06 SBOFFSET 0.16
   SPACE >=0.96 <1.12 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06 SBOFFSET 0.315
   SPACE >=1.12 <1.28 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06 SBOFFSET 0.315
      SBWIDTH 0.06 CENTER
   SPACE >=1.28 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06  SBOFFSET 0.315
      SBWIDTH 0.06  SBOFFSET 0.47
}
```

Inputting the gate_poly edges rather than polygons ensures that you do not generate scattering bars for false edges.

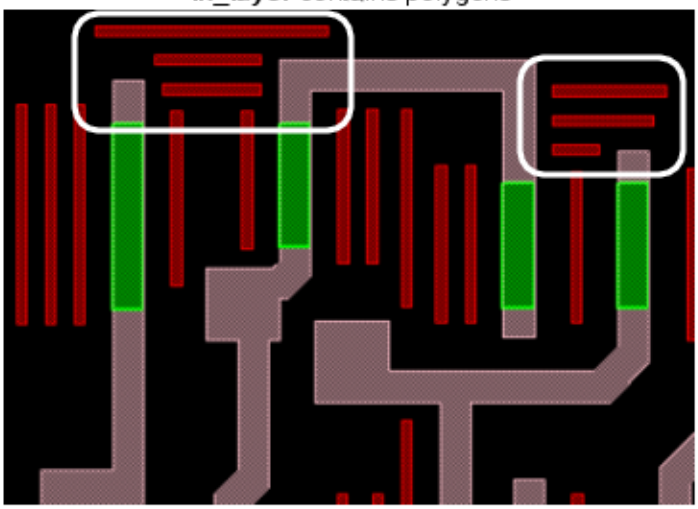
Figure 2-4. Why Supply an Edge as the Input Layer (in_layer)

Deriving the gate_poly using a simple boolean AND:

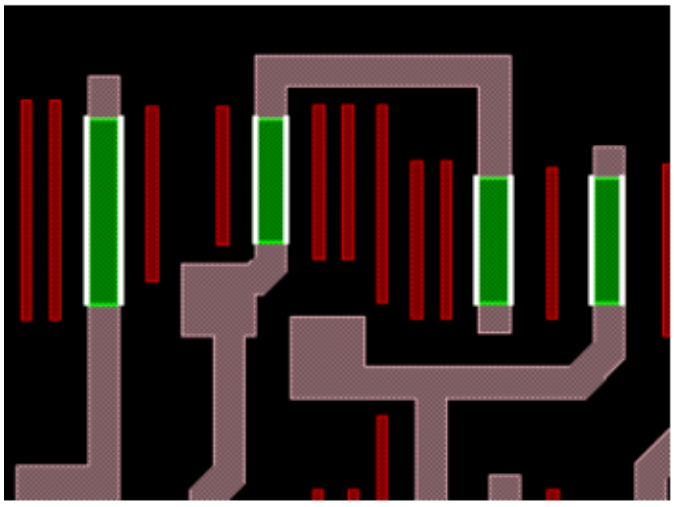
```
gate_poly = poly AND active
```

The ***in_layer*** then contains polygons. Calibre OPCsbar creates scattering bars for all edges on these polygons, including line-ends, which are false edges.

in_layer contains polygons



in_layer contains edges



To avoid creating scattering bars for false edges, you must input an edge layer as the ***in_layer***. Using the COINCIDENT EDGE operation, you can ensure that there are no false edges:

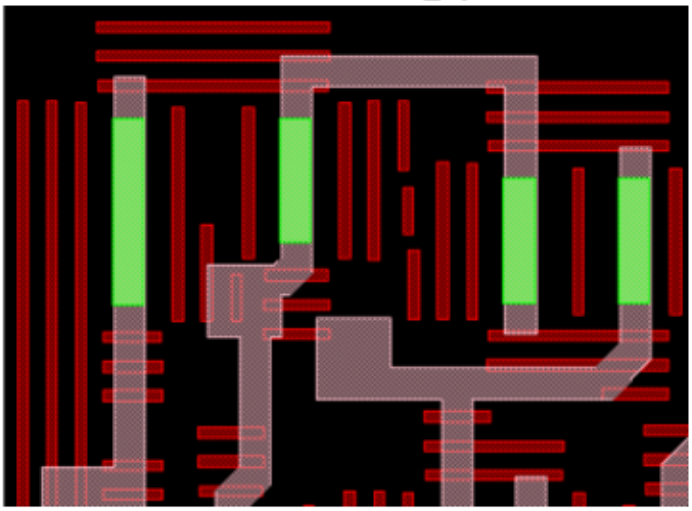
```
temp = poly AND active
gate_poly = temp COIN EDGE
poly
```

To use an edge layer as the ***in_layer***, you must supply a *ref_layer*.

Figure 2-5. More Effects of Using a Reference Layer (ref_layer)

In addition to making it possible for you to input edge data to Calibre OPCsbar as the ***in_layer***, *ref_layers* also ensure that no scattering bars are created over data on the same layer.

Results without *ref_layer*



poly
active
gate_poly
scattering bars

Code for results without *ref_layer*:

```
assist {OPCSBAR temp
   MINSBOFFSET  0.11
   MINSBSPACE   0.080
   MINSBWIDTH   0.059
   MINSBLENGTH  0.130
   EXTENSION    0.10
   INTERSECTION N
   ANGLED
   SPACE >=0.32 <0.54 SBWIDTH 0.06 CENTER
   SPACE >=0.54 <0.70 SBWIDTH 0.06 SBOFFSET 0.16
   SPACE >=0.70 <0.96 SBWIDTH 0.06 SBOFFSET 0.16
   SPACE >=0.96 <1.12 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06 SBOFFSET 0.315
   SPACE >=1.12 <1.28 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06 SBOFFSET 0.315
      SBWIDTH 0.06 CENTER
   SPACE >=1.28 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06  SBOFFSET 0.315
      SBWIDTH 0.06  SBOFFSET 0.47
}
```

Scattering Bars for Gates

You add scattering bars to the entire poly layer, using SPACE and WIDTH to generate scattering bars for poly. Make scattering bars a higher priority for gates.

Figure 2-6. Method for Obtaining Gate Scattering Bars

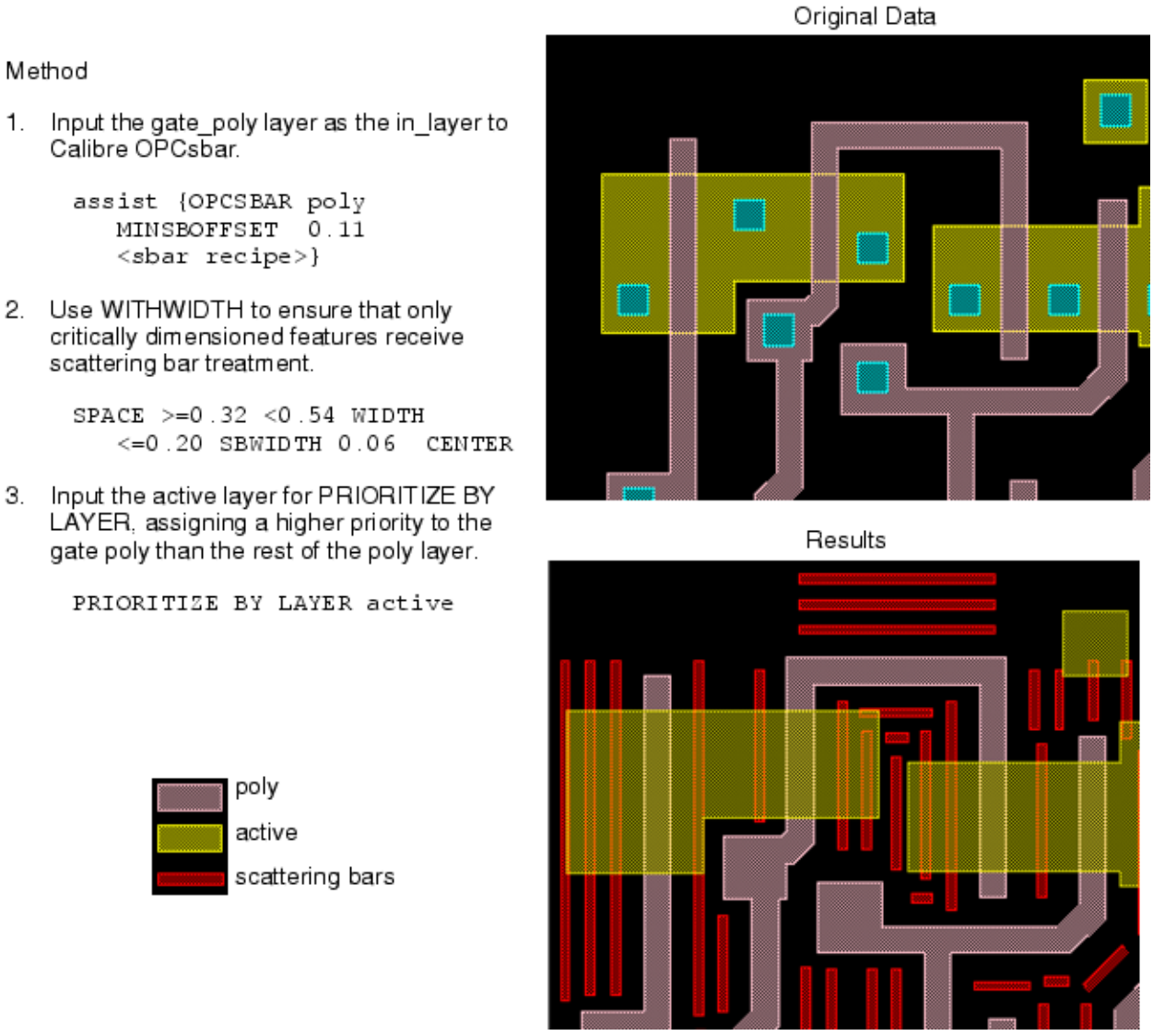


Figure 2-7. Rule File for Gate Scattering Bars

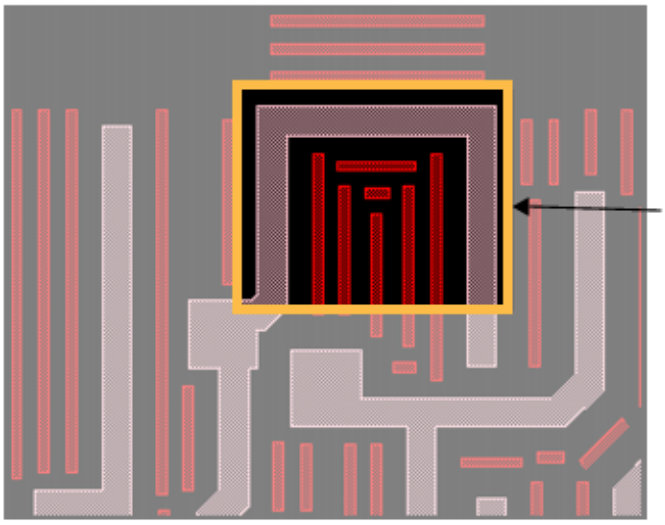
```
//*************************************************************************
// Mentor Graphics (c) 2014
//*************************************************************************

// ********************************************************Input Section
LAYOUT SYSTEM GDSII
LAYOUT PATH "public_html/gds_files/sbar_test.gds"
LAYOUT PRIMARY "top"
FLAG SKEW YES
LAYOUT ERROR ON INPUT YES
PRECISION 1000
RESOLUTION 1
LAYER poly  4
LAYER active 2
// ********************************************************Output Section
DRC MAXIMUM RESULTS ALL
DRC RESULTS DATABASE "public_html/gds_files/sbar_test3.gds" GDSII
DRC SUMMARY REPORT "/dev/null"
DRC MAXIMUM VERTEX 199
DRC KEEP EMPTY YES
DRC CHECK MAP poly 4
DRC CHECK MAP active 2
DRC CHECK MAP assist  3
// ***************************************The DRC RECIPE STARTS HERE
gate_poly = poly AND active
// ********************************************************OUTPUT LAYERS
poly {COPY poly}
active {COPY active}
assist {OPCSBAR poly
   PRIORITIZE BY LAYER active
   MINSBOFFSET  0.11
   MINSBSPACE   0.080
   MINSBWIDTH   0.059
   MINSBLENGTH  0.130
   EXTENSION    0.10
   INTERSECTION N
   ANGLED
   SPACE >=0.32 <0.54 WIDTH <=0.20 SBWIDTH 0.06  CENTER
   SPACE >=0.54 <0.70 WIDTH <=0.20 SBWIDTH 0.06  SBOFFSET 0.16
   SPACE >=0.70 <0.96 WIDTH <=0.20 SBWIDTH 0.06  SBOFFSET 0.16
   SPACE >=0.96 <1.12 WIDTH <=0.20 SBWIDTH 0.06  SBOFFSET 0.16
      SBWIDTH 0.06  SBOFFSET 0.315
   SPACE >=1.12 <1.28 WIDTH <=0.20 SBWIDTH 0.06  SBOFFSET 0.16
      SBWIDTH 0.06  SBOFFSET 0.315
      SBWIDTH 0.06  CENTER
   SPACE >=1.28 WIDTH <=0.20 SBWIDTH 0.06  SBOFFSET 0.16
      SBWIDTH 0.06  SBOFFSET 0.315
      SBWIDTH 0.06  SBOFFSET 0.47
}
```

Using the active layer with PRIORITIZE BY LAYER gives priority to the scattering bars for gate_poly.

Figure 2-8. PRIORITIZE BY LAYER Methodology

Results with Prioritize By Layer



poly
active
scattering bars

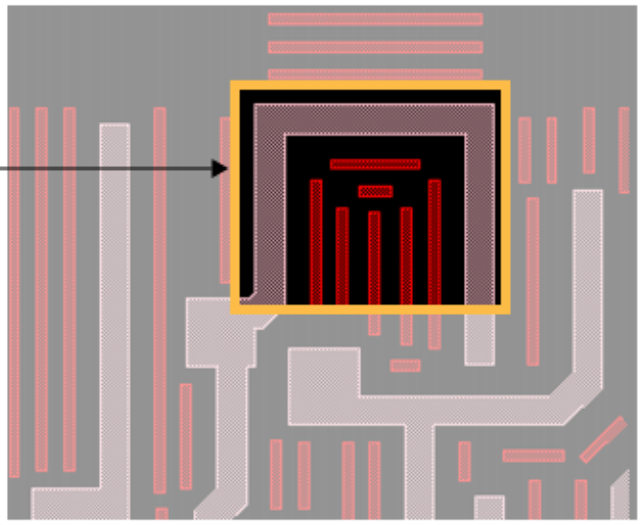
When using PRIORITIZE BY LAYER the gate poly, which overlaps the active, is assigned a higher priority.

During cleanup, Calibre OPCsbar preserves the scattering bars for the gate layer and trims away the other scattering bars.

Results without Prioritize By Layer

Without PRIORITIZE BY LAYER, scattering bars are prioritized based on proximity to the poly edge.

During cleanup, Calibre OPCsbar trims away similarly prioritized bars equally.



Code for results without PRIORITIZE BY LAYER:

```
assist {OPCSBAR poly
   MINSBOFFSET  0.11
   MINSBSPACE   0.080
   MINSBWIDTH   0.059
   MINSBLENGTH  0.130
   EXTENSION    0.10
   INTERSECTION N
   ANGLED
   SPACE >=0.32 <0.54 WIDTH <=0.20 SBWIDTH 0.06 CENTER
   SPACE >=0.54 <0.70 WIDTH <=0.20 SBWIDTH 0.06 SBOFFSET 0.16
   SPACE >=0.70 <0.96 WIDTH <=0.20 SBWIDTH 0.06 SBOFFSET 0.16
   SPACE >=0.96 <1.12 WIDTH <=0.20 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06  SBOFFSET 0.315
   SPACE >=1.12 <1.28 WIDTH <=0.20 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06  SBOFFSET 0.315
      SBWIDTH 0.06  CENTER
   SPACE >=1.28 WIDTH <=0.20 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06 SBOFFSET 0.315
      SBWIDTH 0.06 SBOFFSET 0.47
}
```

Incremental Scattering Bars

You input previously generated scattering bars using SBLAYER and additionally generate new scattering bars.

Figure 2-9. Method to Obtain Incremental Scattering Bars

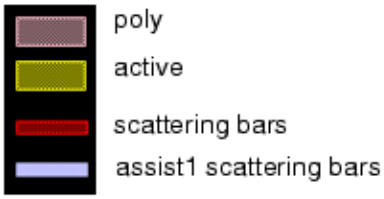
Method

1. Generate critical scattering bars using the first OPCSBAR command:

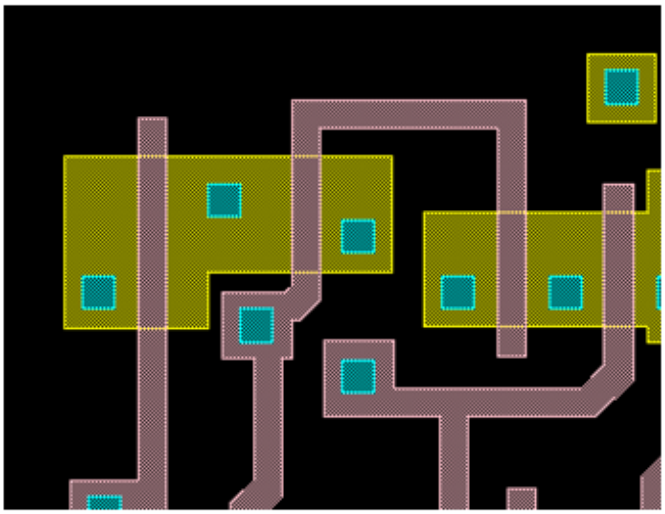
```
temp = poly AND active
gate_poly = temp COIN EDGE
poly
assist1 = OPCSBAR gate_poly
poly
   <first sbar recipe>
```

2. Input this layer to the second OPCSBAR command as an SBLAYER. The second operation creates scattering bars for the rest of the layer:

```
assist {OPCSBAR poly
   SBLAYER assist1
   PRIORITIZE BY LAYER active
   <second sbar recipe>
}
```



Original Data



Results of OPCSBAR

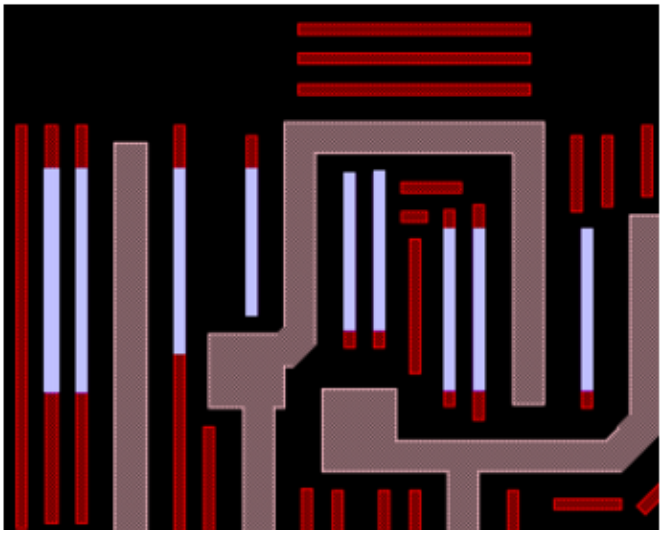


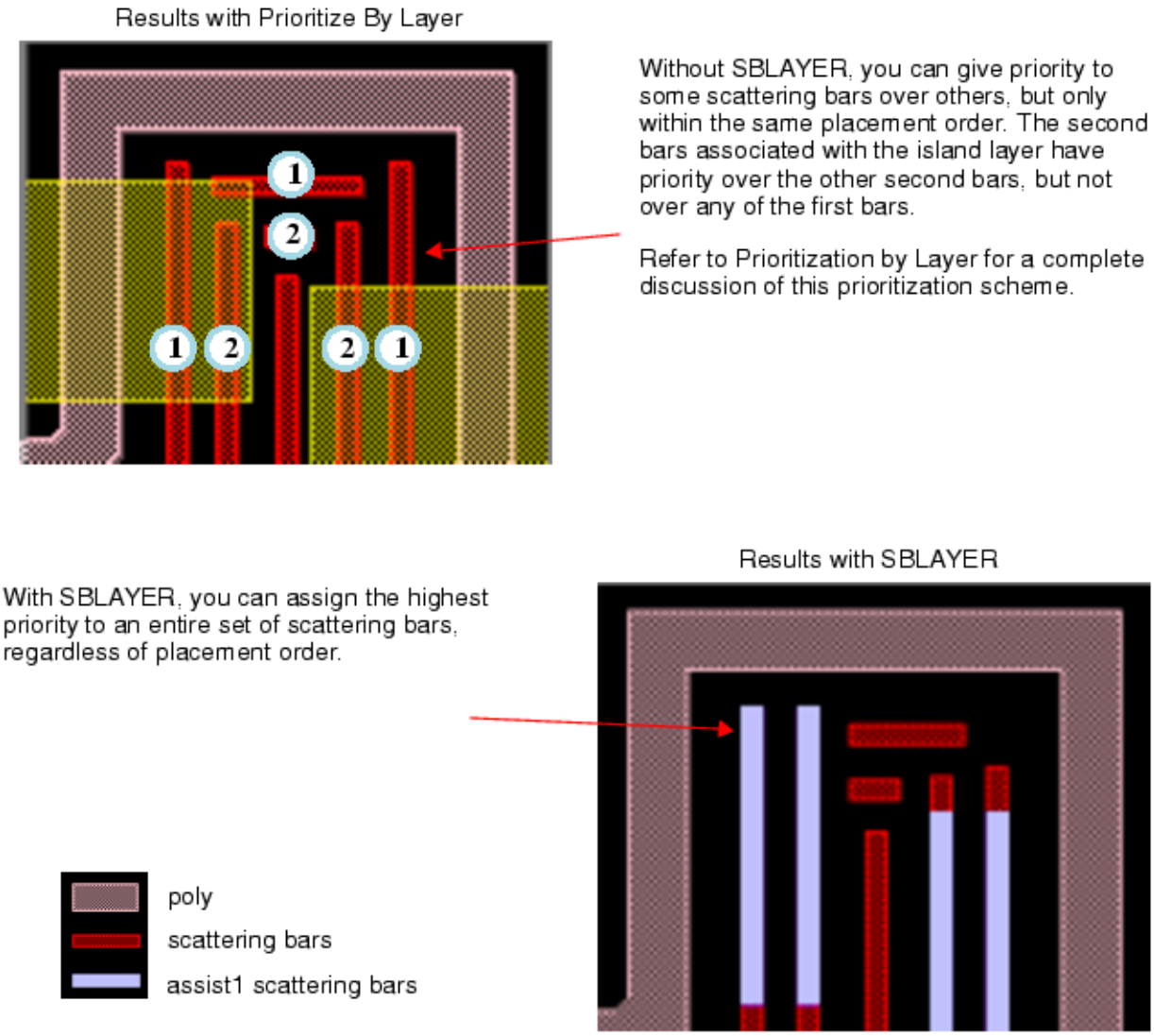
Figure 2-10. Rule File Excerpt for Incremental Scattering Bars

```
//This excerpt begins after all the required SVRF statements defining
//the inputs and outputs.
//
LAYER poly  4
LAYER active 2
DRC CHECK MAP poly 4
DRC CHECK MAP assist1 7
DRC CHECK MAP assist  3
// ****************************************The DRC RECIPE STARTS HERE
temp = poly AND active
gate_poly = temp COINCIDENT EDGE poly
// ********************************************************OUTPUT LAYERS
poly {COPY poly}
assist1 {COPY assist1}
// Generate a derived layer containing the gate_poly scattering bars,
// which are higher priority.
assist1 = OPCSBAR gate_poly poly
   MINSBOFFSET  0.11
   MINSBSPACE   0.080
   MINSBWIDTH   0.059
   MINSBLENGTH  0.130
   EXTENSION    0.10
   INTERSECTION N
   ANGLED
   SPACE >=0.32 <0.54 SBWIDTH 0.06 CENTER
   SPACE >=0.54 <0.70 SBWIDTH 0.06 SBOFFSET 0.16
   SPACE >=0.70 <0.96 SBWIDTH 0.06 SBOFFSET 0.16
   SPACE >=0.96 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06 SBOFFSET 0.315
// Input the gate_poly scattering bars to OPCSBAR as an SBLAYER
assist {OPCSBAR poly
   SBLAYER assist1
   PRIORITIZE BY LAYER active
   MINSBOFFSET  0.11
   MINSBSPACE   0.96
   MINSBWIDTH   0.059
   MINSBLENGTH  0.130
   EXTENSION    0.10
   INTERSECTION N
   ANGLED
   SPACE >=0.32 <0.54 WIDTH <=0.20 SBWIDTH 0.06 CENTER
   SPACE >=0.54 <0.70 WIDTH <=0.20 SBWIDTH 0.06 SBOFFSET 0.16
   SPACE >=0.70 <0.96 WIDTH <=0.20 SBWIDTH 0.06 SBOFFSET 0.16
   SPACE >=0.96 <1.12 WIDTH <=0.20 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06 SBOFFSET 0.315
   SPACE >=1.12 <1.28 WIDTH <=0.20 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06 SBOFFSET 0.315
      SBWIDTH 0.06 CENTER
   SPACE >=1.28 WIDTH <=0.20 SBWIDTH 0.06 SBOFFSET 0.16
      SBWIDTH 0.06 SBOFFSET 0.315
      SBWIDTH 0.06 SBOFFSET 0.47
}
```

This is a layer definition statement, not a rule check statement, because you need to input this data to the second OPCSBAR command.

This is a rule check statement to output the results. The operation merges the SBLAYER scattering bars with the scattering bars it creates, so both are output on the final scattering bar layer.

Figure 2-11. Using SBLAYER



Chapter 3
Calibre OPCsbar Syntax

Calibre OPCsbar commands and their syntax adhere to SVRF standards.

SVRF Commands **34**
- OPCSBAR 35
- Layer Arguments 38
- Required Rule Arguments 44
- Manufacturing Constraint Arguments 54
- Style Control Arguments 68
- Cleanup Control Arguments 79
- Processing Mode Arguments 104

SPACE and SPACELAYER Overview **108**

Data Returned by Calibre OPCsbar **109**

SVRF Commands

A typical SVRF rule file containing OPCSBAR may also contain other SVRF commands that specify input layers, perform preliminary layer manipulation, and save output layers.

These commands are documented in the *Standard Verification Rule Format (SVRF) Manual*.

Table 3-1. SVRF-based OPCSBAR Command

Command	Description
OPCSBAR	Generates scattering bars.
Layer Arguments	Calibre OPCsbar requires an input layer and allows various optional layers.
Required Rule Arguments	Calibre OPCsbar requires space and width rules to output scattering bars.
Manufacturing Constraint Arguments	Calibre OPCsbar adheres to process parameters to output MRC-clean scattering bars.
Style Control Arguments	Calibre OPCsbar provides detailed control to augment scattering bar placement.
Cleanup Control Arguments	Calibre OPCsbar provides post-placement cleanup to ensure MRC-clean scattering bar output.
Processing Mode Arguments	Calibre OPCsbar uses various run modes for optimal processing time.

OPCSBAR

SVRF Commands

Generates scattering bars.

Usage

OPCSBAR {

Layer Arguments

in_layer

[OFFSETLAYER *offsetlayer*]

[*ref_layer*]

[SBLAYER *sblayer* [CLEANSBLAYER]]

Required Rule Arguments (only one may be specified)

SPACE *condition* [WIDTH *condition*]
{**SBWIDTH *value***
{**SBOFFSET *value*** | **SBPITCH *value*** | **CENTER** [*offset*] | **CENTERPITCH** [*offset*]} …}

SPACELAYER *layer* {{**SBWIDTH *value* SBOFFSET *value***} …}

Manufacturing Constraint Arguments

[LINEENDOFFSET *value*]

[LINEENDSPACE *value*]

[LINEENDTOLONGSPACE *value*]

[MAXSBLENGTH *value*]

[MAXSBWIDTH *value*]

[MINEDGELENGTH *value*]

[MINSBLENGTH *value*]

[MINSBOFFSET [*condition*] *value* [OPPOSITE]]

[MINSBSPACE *value*]

[MINSBWIDTH *value*]

Style Control Arguments

[ANGLED]

[ANGLEEDGE]

[HORIZONTAL]

[INTERSECTION {L [*u_bottom_length*] | N | P}]

[NEGATIVE]

[OPPOSITEEXTENDED *value*]

[STRICTCENTER]

[VERTICAL]

[WITHWIDTH]

Cleanup Control Arguments

[CENTERMERGE *value* [*width*]]

[COLINEARIZE *length* [*width*]]

[EXTENSION *value* [BEFORECLEANUP]]

[JOG FILL *dist* [*width*]]

[LINEENDMERGE *value*]

[MINJOGWIDTH *value*]

[NOCLEANUP]

[OPENT]

[OPTION CAREFUL_CLEANUP {0 | 1}]

[OPTION FAST_MRC {NO | YES}]

[OPTION INCLUSIVE_JOG_LENGTH {NO | YES}]

[OPTION MINSBSPACE_CLEANUP_TIEBREAKER {NO | YES}][OPTION PRIORITIZE_SPACE {NO | YES}]

[PRIORITIZE BY LAYER *layer1* [*layerN* ...]]

[PRIORITYCENTER]

[SMOOTH *value*]

Processing Mode Arguments

[OPTION CLEAN_RESEAT {NO | YES}]

[OPTION PROCESSING_MODE {FLAT | HIERARCHICAL}]

[OPTION TILEMICRONS *value*]

[OPTION VERIFICATION_MODE {NO | YES}]

}

Arguments

Table 3-2. OPCSBAR Argument Groups

Argument Group	Description
Layer Arguments	Calibre OPCsbar requires an input layer and allows various optional layers.

Table 3-2. OPCSBAR Argument Groups (cont.)

Argument Group	Description
Required Rule Arguments	Calibre OPCsbar requires space and width rules to output scattering bars.
Manufacturing Constraint Arguments	Calibre OPCsbar adheres to process parameters to output MRC-clean scattering bars.
Style Control Arguments	Calibre OPCsbar provides detailed control to augment scattering bar placement.
Cleanup Control Arguments	Calibre OPCsbar provides post-placement cleanup to ensure MRC-clean scattering bar output.
Processing Mode Arguments	Calibre OPCsbar uses various run modes for optimal processing time.

Layer Arguments

Calibre OPCsbar requires an input layer and allows various optional layers.

For organizational keyword usage information, refer to “OPCSBAR” on page 35.

Table 3-3. Layer Arguments

Argument	Description
in_layer	Specifies an input layer name for which to generate scattering bars.
OFFSETLAYER	Enables shapes to act as keepout areas for scattering bar placement.
ref_layer	Reference layer name for measuring violations.
SBLAYER	Preexisting scattering bar layer name.

in_layer

OPCSBAR argument: Layer Arguments

Specifies an input layer name for which to generate scattering bars.

Usage

OPCSBAR *in_layer* {**…**}

Arguments

None.

Description

A required name of an input layer containing polygon or edge data. Edge data is valid input only if used with a ref_layer. There is no default layer name.

When used with the SPACE keyword, the target layer when:

- used without a *ref_layer* must be a drawn or derived polygon
- used with a *ref_layer* can be a drawn or derived polygon, or a derived edge

For a detailed discussion of how ***in_layer*** is used with the SPACE keyword, refer to “Boundary Edges and Layers” on page 120.

When used with the SPACELAYER keyword, the ***in_layer*** is used during cleanup to supply data from which spacing violations are performed. This layer serves the same purpose that *ref_layer* serves with the SPACE keyword. The input for SPACELAYER must be a polygon.

OFFSETLAYER

OPCSBAR keyword: Layer Arguments

Enables shapes to act as keepout areas for scattering bar placement.

Usage

OFFSETLAYER *offsetlayer*

Arguments

- *offsetlayer*

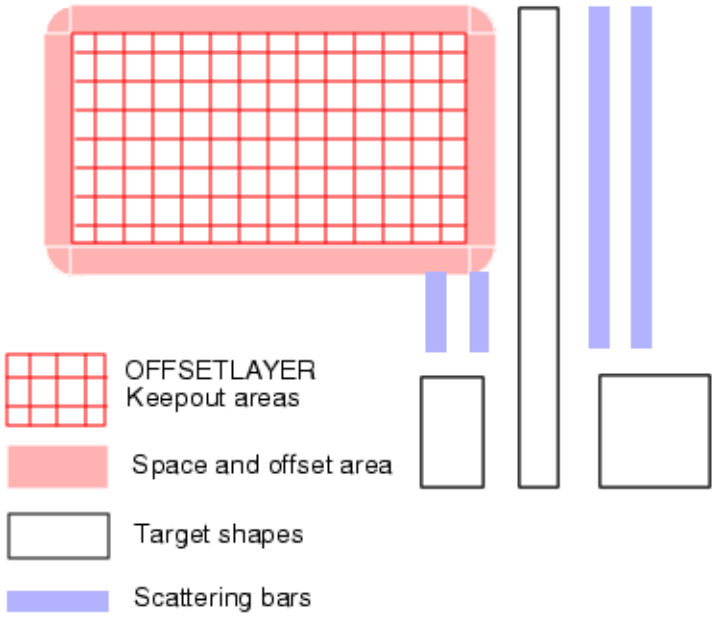
  A required polygonal layer name.

Description

An option enabling all shapes on the specified layer to act as areas where scattering bars cannot be generated. Additionally, MINSBOFFSET and LINEENDOFFSET MRC checks apply between the OFFSETLAYER and the generated scattering bars.

Figure 3-1 demonstrates the use of the OFFSETLAYER as a keepout area.

Figure 3-1. Use of OFFSETLAYER as a Scattering Bar Keepout Area



ref_layer

OPCSBAR argument: Layer Arguments

Reference layer name for measuring violations.

Usage

OPCSBAR {**…**} *ref_layer*

Arguments

None.

Description

The optional polygon layer against which spacing violations are performed during edge classification and cleanup of spacing and offset violations. When *ref_layer* is specified, spaces are measured between an edge on in_layer and edges on this layer. Otherwise, spaces are measured between edges on ***in_layer*** alone. There is no default for this layer name.

The *ref_layer* must be a superset of the ***in_layer***; it must contain all shapes on the ***in_layer***. You cannot specify *ref_layer* when using the SPACELAYER keyword.

SBLAYER

OPCSBAR keyword: Layer Arguments

Preexisting scattering bar layer name.

Usage

SBLAYER *sblayer* [CLEANSBLAYER]

Arguments

- *sblayer*

  A required name of a layer containing previously generated scattering bars.

- CLEANSBLAYER

  An optional command that performs cleanup operations on the sblayer with respect to the in_layer and ref_layer and other shapes on the sblayer. CLEANSBLAYER merges and enforces other MRC constraints on SBLAYER. If CLEANSBLAYER is not specified, no cleanup changes to the SBLAYER are performed. Cleanup is performed after all OPCSBAR-generated scattering bars have been cleaned up. It evaluates the shapes on sblayer with respect to all cleanup options, including COLINEARIZE, LINEENDMERGE, EXTENSION, JOG FILL, and correcting violations to the SPACE and SPACELAYER keywords. The CLEANSBLAYER argument is not compatible with OPTION VERIFICATION_MODE.

Description

An optional polygon layer containing preexisting scattering bars that are considered when placing new scattering bars. The LINEENDSPACE, LINEENDTOLONGSPACE and MINSBSPACE MRC checks apply between SBLAYER and OPCSBAR-generated scattering bars.

Note

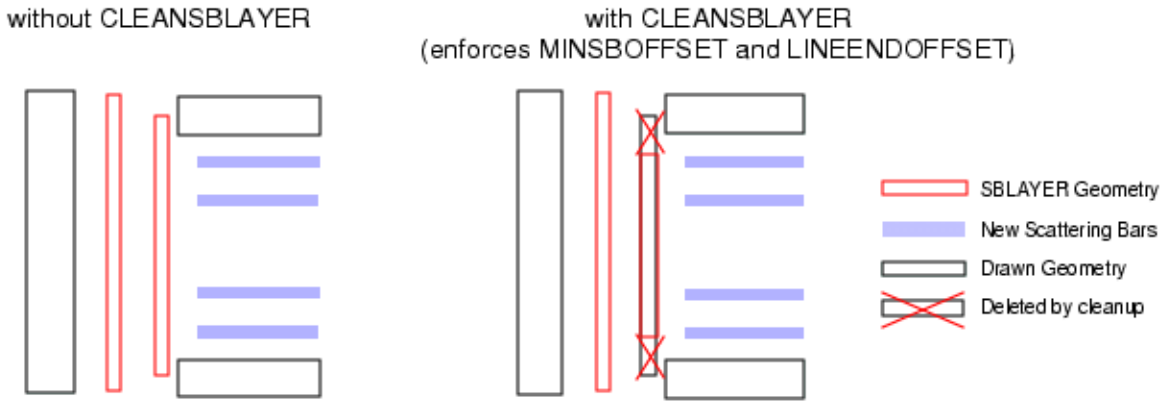
As of release 2016.4 and later, SBLAYER does not merge with any other SRAFs. Prior to 2016.4, SBLAYER is merged with other SRAFs generated with LINEENDMERGE.

SBLAYER is used in two ways:

- During cleanup, Calibre OPCsbar assigns scattering bars the highest priority on SBLAYER (see “How Priority Affects Cleanup” for more detail). Calibre OPCsbar also factors shapes on SBLAYER into all spacing checks.
- Calibre OPCsbar output shapes on SBLAYER with newly-generated scattering bars and outputs a single, merged layer.

Figure 3-2 shows the use of CLEANSBLAYER.

Figure 3-2. Using CLEANSBLAYER



Required Rule Arguments

Calibre OPCsbar requires space and width rules to output scattering bars.

For organizational keyword information, refer to “OPCSBAR” on page 35.

Table 3-4. Required Rule Arguments

Argument	Description
SPACE	Defines a constraining spatial range between edges to qualify and generate scattering bars.
SPACELAYER	Declares a layer name for generation of scattering bars.

SPACE

OPCSBAR keyword: Required Rule Arguments

Defines a constraining spatial range between edges to qualify and generate scattering bars.

Usage

SPACE *range*
{SBWIDTH *value*
{SBOFFSET *value* |
SBPITCH *value* |
CENTER [*offset*] |
CENTERPITCH [*offset*]}
…}
[WIDTH *range*]

Arguments

- **SPACE *range***

  range — A required range of non-negative floating point numbers specified in user units with no default value. For example:

  > a < b

  <= a < b

  > a

  A required argument as a range of one or two numbers qualifying a distance between two edges for which to generate scattering bars. Unbounded space ranges are interpreted as the complement of the bounded counterparts. For example, > a is interpreted as everything not in the <= a interval.

  All constraints used for boundary declarations are forced to match internally. Consequently, which ever boundary constraint (<, <=, >, or >=) is first used is applied to all subsequent SPACE keywords using the same value. If subsequent SPACE keywords specifying the same value do use different boundary constraints, a warning message is generated stating:

  ```
  "WARNING: Constraint > nn.nnn being replaced by >= nn.nnn"
  ```

  Space ranges must be mutually exclusive; that is, the range in one rule cannot overlap the range in another rule in the same operation. There is no default value for this keyword.

- WIDTH *range*

  range — An optional range of floating point numbers with no default value.

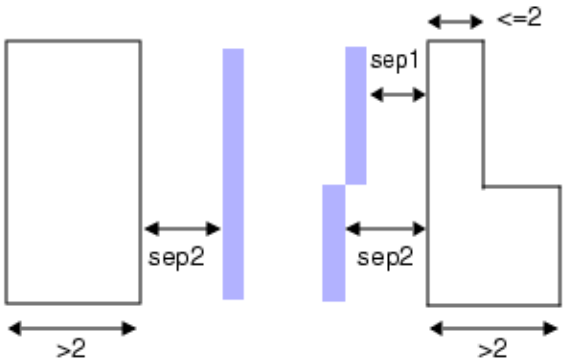
  An optional argument that defines the width condition of the target shape an associated edge must meet for SPACE to apply. In addition, the edges must meet both required condition and WIDTH condition for SPACE to apply.

  If you specify WIDTH and CENTER for positive scattering bars, edges must meet both SPACE and at least one WIDTH constraint to be treated with a centered scattering bar. If

STRICTCENTER is specified, then both SPACE and both WIDTH constraints must be met to place a centered scattering bar.

Figure 3-3 demonstrates the WIDTH argument for generation of positive scattering bars.

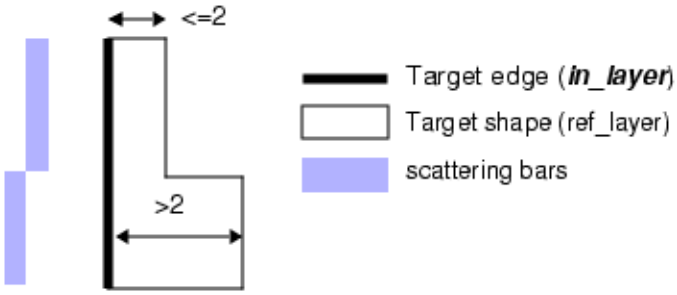
Figure 3-3. WIDTH for Positive Scattering Bars




```
SPACE <=space1 WIDTH <=2 SBWIDTH 0.08 SBOFFSET sep1
SPACE <=space1 WIDTH >2 SBWIDTH 0.08 SBOFFSET sep2
```

If in_layer is an edge, Calibre OPCsbar evaluates the width by measuring the polygon on the ref_layer that is associated with the edge as shown in Figure 3-4.

Figure 3-4. WIDTH Treatment by Edge



- **SBWIDTH *value***

  value— A floating point number that must be greater than or equal to MINSBWIDTH, less than MINSBLENGTH, and less than or equal to MAXSBWIDTH. If these conditions are not satisfied, Calibre OPCsbar confuses line-ends during cleanup. There is no default setting for this keyword.

  A required argument that specifies the width of scattering bars. One of its associated placement keywords, SBOFFSET, SBPITCH, CENTER, or CENTERPITCH, is required.

  You must specify SBWIDTH for each placement argument, even if all scattering bars are the same width. Depending on the rule and the placement argument, this keyword defines the width of either one or two scattering bars.

- **SBOFFSET *value***

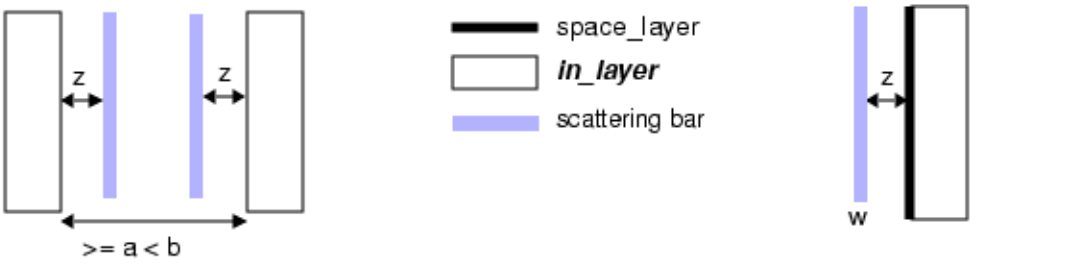
  value — A positive floating point number. There is no default for this setting.

A required argument that defines the placement for scattering bars. The nearest edge of a scattering bar is placed by the specified SBOFFSET ***value*** from any valid ***in_layer*** edge.

For positive scattering bars, the distance is measured between the scattering bar and the outside of the nearest valid ***in_layer*** edge. This is the default. For negative scattering bars, the distance is measured between the scattering bar and the inside of the nearest valid ***in_layer*** edge.

Depending on the rule and the data, each SBOFFSET triggers the generation of either one or two scattering bars as shown in Figure 3-5.

Figure 3-5. SBOFFSET Usage



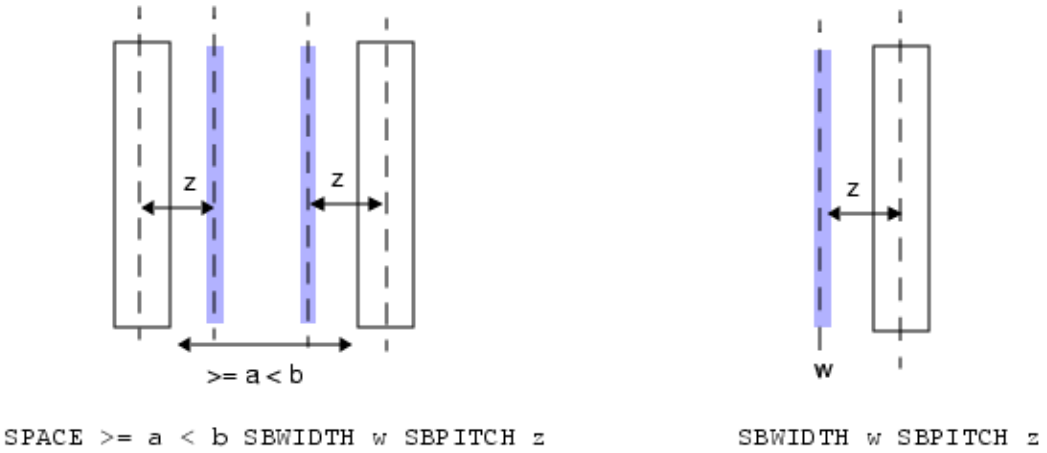
- **SBPITCH *value***

  value — A positive floating point number. There is no default for this setting.

  This argument defines the placement for scattering bars, expressed as the distance between the scattering bar center-line and the center of the feature at the nearest valid edge. SBPITCH requires a constrained WIDTH specification that must be satisfied in order for the rule to apply. Only positive scattering bars are supported with SBPITCH.

  Depending on the rule and the data, each SBPITCH triggers the generation of either one or two scattering bars. Figure 3-6 demonstrates SBPITCH usage.

Figure 3-6. Scattering Bar Generation with SBPITCH



- **CENTER** [*offset*]

  offset — A positive floating point number. There is no default for this setting.

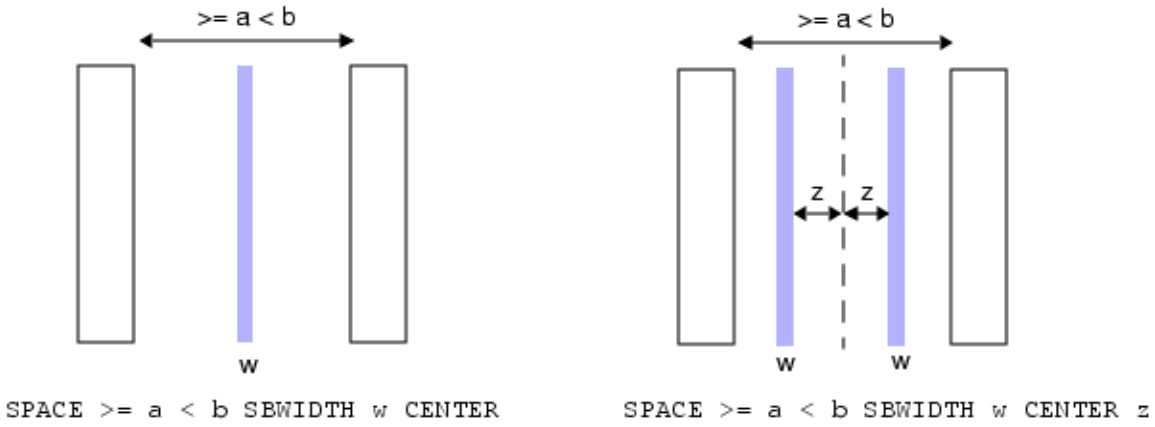
  This argument defines the placement for scattering bars generated by SPACE, relative to the center line of the space. Without *offset*, Calibre OPCsbar generates a single scattering bar an equal distance between the two valid edges that satisfy the SPACE condition. For spaces an even number of user units wide, scattering bars may be 1/2 user unit off center, due to snapping. With *offset*, Calibre OPCsbar generates two scattering bars offset from the center line by a distance of *offset*.

  You can only specify one CENTER placement argument for each SPACE condition; if you have several sets of scattering bars for a SPACE condition, only one can use the CENTER placement argument. If you specify WIDTH and CENTER for positive scattering bars, edges must meet both SPACE and at least one WIDTH constraint to be treated with a centered scattering bar. If STRICTCENTER is specified, then both SPACE and both WIDTH constraints must be met to place a centered scattering bar. CENTER supports NEGATIVE scattering bars if a WIDTH constraint is specified.

  For unbounded spaces, CENTER has no meaning. CENTER is supported only for two valid parallel edges.

  Figure 3-7 illustrates the behavior of CENTER.

Figure 3-7. Scattering Bars Generated Using CENTER



- CENTERPITCH [*offset*]

  offset — An optional positive floating point number. The default for this setting is 0, indicating there is no offset from the center pitch line, and therefore resulting in only one scattering bar generated between qualified shapes.

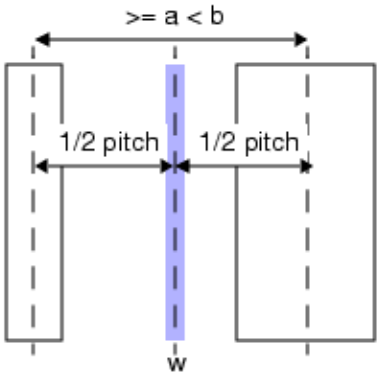
  This argument defines the placement of generated scattering bars, expressed relative to the center line of the pitch of two qualified target shapes. CENTERPITCH requires a constrained WIDTH specification that must be satisfied by both qualified target shapes in order for the rule to apply. Only positive scattering bars are generated by CENTERPITCH.

You can only specify one CENTERPITCH placement argument for each SPACE condition. If you specify WIDTH and CENTERPITCH for positive scattering bars, edges must meet both SPACE and WIDTH constraints to place the centered scattering bar.

CENTERPITCH does not support NEGATIVE scattering bars. For unbounded spaces, CENTERPITCH has no meaning. CENTERPITCH is supported only for two valid parallel edges.

If no *offset* is specified, Calibre OPCsbar creates a single scattering bar, equidistant between centers of the features containing two valid edges that satisfy the SPACE condition, as illustrated in Figure 3-8.

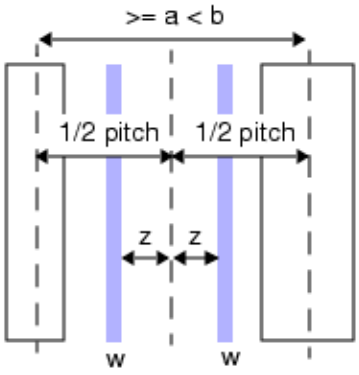
Figure 3-8. Scattering Bars Generated Using CENTERPITCH without Offset




```
SPACE >= a < b SBWIDTH w WIDTH > a < b CENTERPITCH
```

With an offset specified, Calibre OPCsbar creates two scattering bars, offset from the center line found equidistant between centers of the features containing two valid edges that satisfy the SPACE condition. This is illustrated in Figure 3-9.

Figure 3-9. Scattering Bars Generated Using CENTERPITCH with Offset




```
SPACE >= a < b SBWIDTH w WIDTH > a < b CENTERPITCH z
```

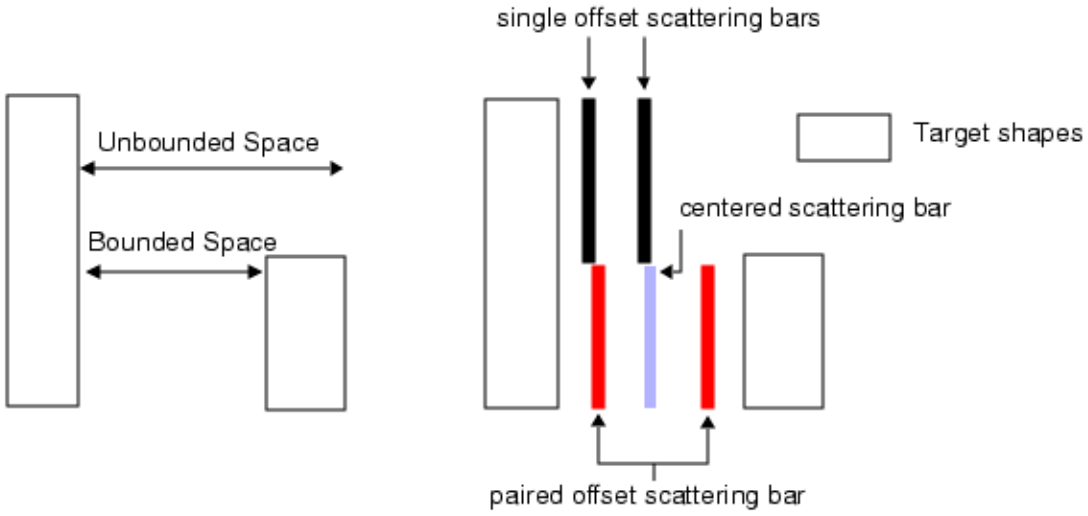
Description

A required keyword cluster that defines the space and, optionally, width conditions shapes must meet for the rule to apply. SPACE and SPACELAYER are mutually exclusive within a single OPCSBAR command.

You must specify one placement argument (SBOFFSET, SBPITCH, CENTER, or CENTERPITCH) for each SPACE keyword.

For a bounded space, Calibre OPCsbar generates scattering bars parallel to each valid boundary edge. For an unbounded space, Calibre OPCsbar generates scattering bars parallel to the single boundary edge, with the placements as specified by SBOFFSET. Placements specified by CENTER have no meaning for unbounded spaces. Because a space fits within an unbounded range does not mean it is an unbounded space. Any space having two boundary edges is bounded. For a complete discussion of spaces and how Calibre OPCsbar evaluates them, refer to "Understanding Spaces" on page 116. Figure 3-10 depicts bounding concepts.

Figure 3-10. Bounded and Unbounded Spaces

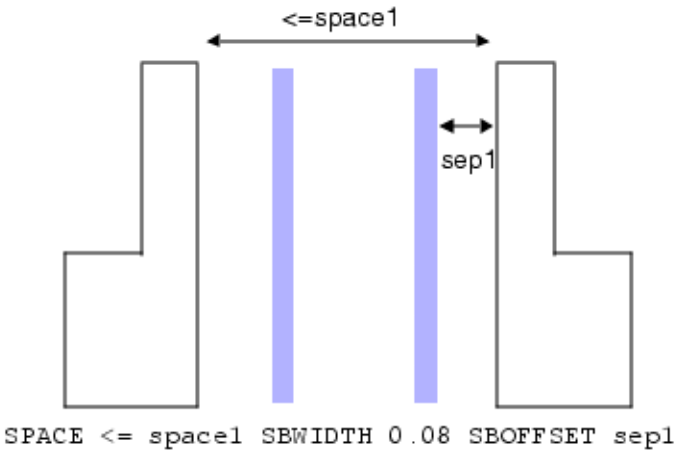


Examples

Example 1 — SPACE for positive scattering bars

Figure 3-11 demonstrates SPACE generation of positive scattering bars.

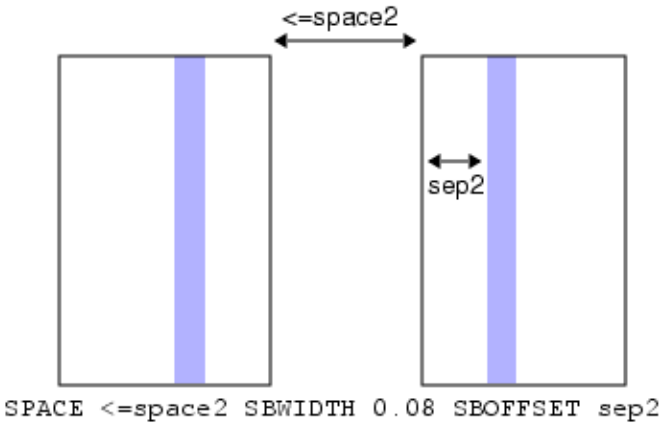
Figure 3-11. SPACE Keyword Generating Positive Scattering Bars



Example 2 — SPACE for negative scattering bars

Figure 3-12 demonstrates SPACE generation of negative scattering bars.

Figure 3-12. SPACE Keyword Generating Negative Scattering Bars



SPACELAYER

OPCSBAR keyword: Required Rule Arguments

Declares a layer name for generation of scattering bars.

Usage

SPACELAYER ***layer*** {**SBWIDTH** ***value*** **SBOFFSET** ***value***}…

Arguments

- ***layer***

  A required name that specifies a layer containing the edges needing scattering bars.

- **SBWIDTH** ***value***

  value — A floating point number that must be greater than or equal to MINSBWIDTH, less than MINSBLENGTH, and less than or equal to MAXSBWIDTH. If these conditions are not satisfied, Calibre OPCsbar confuses line-ends during cleanup. There is no default setting for this keyword.

  A required argument that specifies the width of scattering bars. One of its associated placement keywords, SBOFFSET, SBPITCH, CENTER, or CENTERPITCH, is required.

  You can only specify one CENTER placement argument for each SPACELAYER condition; if you have several sets of scattering bars for a SPACELAYER condition, only one can use the CENTER placement argument. If you specify WIDTH and CENTER for positive scattering bars, edges must meet both SPACE and at least one WIDTH constraint to be treated with a centered scattering bar. If STRICTCENTER is specified, then both SPACE and both WIDTH constraints must be met to place a centered scattering bar.

  You must specify SBWIDTH for each placement argument, even if all scattering bars are the same width. Depending on the rule and the placement argument, this keyword defines the width of either one or two scattering bars.

- **SBOFFSET** ***value***

  value — A positive floating point number. There is no default for this setting.

  A required argument that defines the placement for generated scattering bars. The nearest edge of a scattering bar is placed a distance equivalent to the specified SBOFFSET ***value*** from any valid ***in_layer*** edge.

  For positive scattering bars, the distance is measured between the scattering bar and the exterior side of the nearest valid ***in_layer*** edge. This is the default. For negative scattering bars, the distance is measured between the scattering bar and the interior side of the nearest valid ***in_layer*** edge.

  Depending on the rule and the data, each SBOFFSET triggers the creation of either one scattering bar or two, as shown in Figure 3-5.

Description

The SPACELAYER keyword specifies a layer containing the edges requiring scattering bars. This layer must be an original or derived edge layer. There is no default for this layer setting.

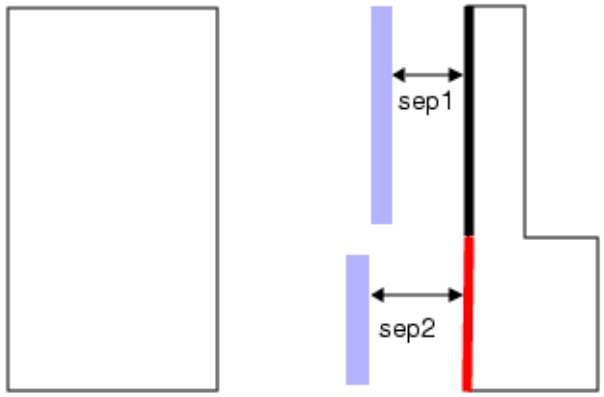
You must specify both SBWIDTH and SBOFFSET arguments for each SPACELAYER keyword. Every SPACELAYER keyword must have a unique layer.

SPACELAYER is mutually exclusive with SPACE.

Examples

Figure 3-13 demonstrates a general SPACELAYER usage.

Figure 3-13. SPACELAYER General Usage




```
SPACELAYER black SBWIDTH 0.08 SBOFFSET sep1
SPACELAYER red SBWIDTH 0.08 SBOFFSET sep2
```

Manufacturing Constraint Arguments

Calibre OPCsbar adheres to process parameters to output MRC-clean scattering bars.

For organizational keyword information, refer to “OPCSBAR” on page 35.

Table 3-5. Manufacturing Constraint Arguments

Argument	Description
LINEENDOFFSET	Defines minimum line end offset.
LINEENDSPACE	Defines minimum line end spacing.
LINEENDTOLONGSPACE	Defines minimum line end to side spacing.
MAXSBLENGTH	Defines maximum length for positive and negative scattering bars.
MAXSBWIDTH	Defines maximum width of scattering bars.
MINEDGELENGTH	Defines minimum edge lengths.
MINSBLENGTH	Defines minimum length of scattering bars.
MINSBOFFSET	Defines a minimum offset.
MINSBSPACE	Defines minimum scattering bar spacing.
MINSBWIDTH	Defines minimum scattering bar width.

LINEENDOFFSET

OPCSBAR keyword: Manufacturing Constraint Arguments

Defines minimum line end offset.

Usage

LINEENDOFFSET *value*

Arguments

- *value*

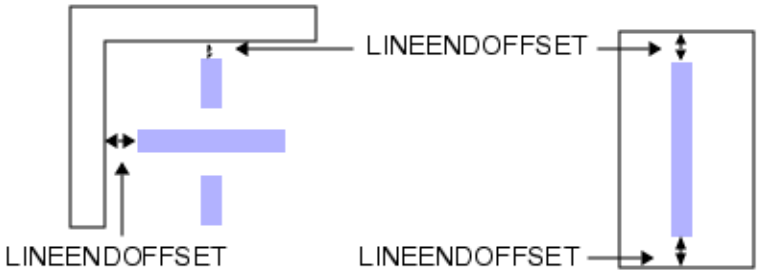
  A positive floating point number with a default value of MINSBOFFSET.

Description

An optional minimum offset requirement for all line-end edges of scattering bars created by Calibre OPCsbar. The value specifies the minimum distance allowed between any scattering bar line-end and original polygon edges. The *value* must be greater than or equal to 0, with 0 implying that a scattering bar may touch an original feature. If not specified, LINEENDOFFSET defaults to MINSBOFFSET.

A scattering bar line-end is defined as greater than or equal to MINSBWIDTH and less than or equal to MAXSBWIDTH. Figure 3-14 demonstrates LINEENDOFFSET.

Figure 3-14. Where LINEENDOFFSET is Measured



LINEENDSPACE

OPCSBAR keyword: Manufacturing Constraint Arguments

Defines minimum line end spacing.

Usage

LINEENDSPACE *value*

Arguments

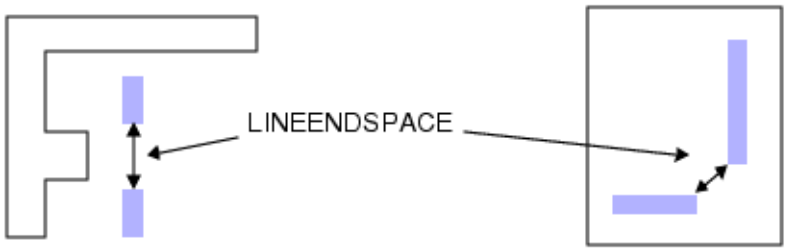
- *value*

  A floating point number with a default value of MINSBSPACE.

Description

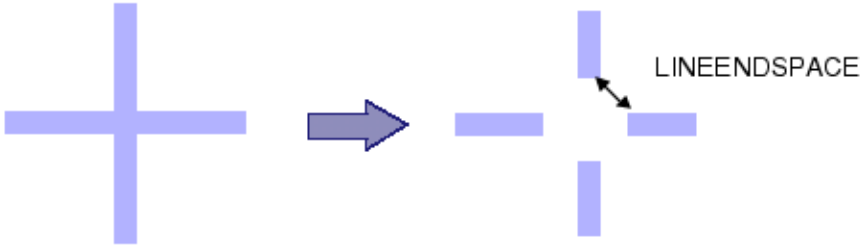
An optional minimum spacing requirement between the line-end edges of two scattering bars created by this operation. This operation affects all scattering bars created by Calibre OPCsbar. Figure 3-15 demonstrates LINEENDSPACE measurement.

Figure 3-15. Where LINEENDSPACE is Measured



LINEENDSPACE is a spacing rule Calibre OPCsbar uses when performing cleanup of violations from P to N or L intersections, as shown in Figure 3-16.

Figure 3-16. LINEENDSPACE during CLEANUP



LINEENDTOLONGSPACE

OPCSBAR keyword: Manufacturing Constraint Arguments

Defines minimum line end to side spacing.

Usage

LINEENDTOLONGSPACE *value*

Arguments

- *value*

  A floating point number with a default value of LINEENDSPACE.

Description

An optional minimum spacing requirement between the line-end edge of one scattering bar created by this operation and the long edge of another scattering bar. This operation affects all scattering bars created by Calibre OPCsbar.

Figure 3-17 demonstrates LINEENDTOLONGSPACE measurement.

Figure 3-17. Where LINEENDTOLONGSPACE is Measured

MAXSBLENGTH

OPCSBAR keyword: Manufacturing Constraint Arguments

Defines maximum length for positive and negative scattering bars.

Usage

MAXSBLENGTH *value*

Arguments

- *value*

  A floating point number with no default value.

Description

An optional argument specifying the maximum length allowed for all positive and negative scattering bars. The *value* specifies the maximum floating-point length. By default, MAXSBLENGTH is off until *value* is specified.

This operation aligns scattering bar cuts to the original layer breaks as shown in Figure 3-18.

Figure 3-18. MAXSBLENGTH Example

Scattering bars cannot be aligned to target contacts in the following two situations:

- Where neighboring target contacts are too close for alignment (see Figure 3-19). In this case, scattering bars are split into equal sizes while still obeying other MRC constraints.
- Where aligned scattering bars conflict with other constraints such as MINSBLENGTH and LINEENDSPACE in two ways (see Figure 3-20):
  - MAXSBLENGTH is greater than or equal to A
  - MINSBLENGTH is greater than A plus B less than LINEENDSPACE

  If such a conflict exists, then the scattering bars are cut to a uniform size.

  Figure 3-19 shows scattering bar cleanup when they are too close to align.

Figure 3-19. Target Contact too Close for SBAR Alignment

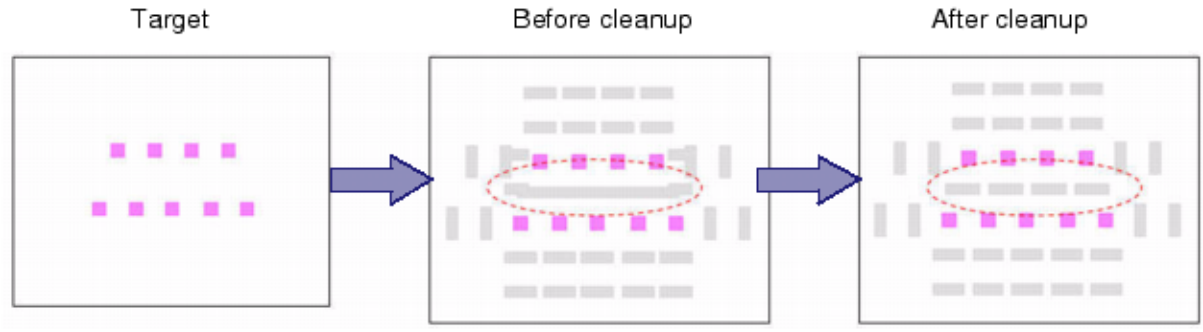
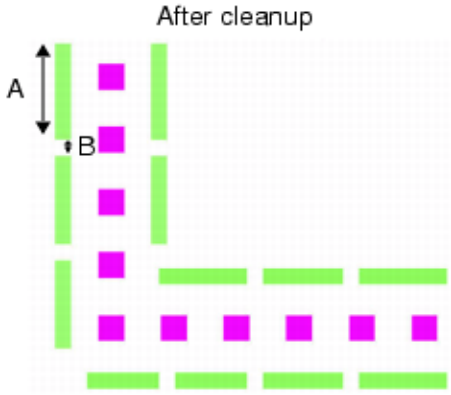


Figure 3-20 shows scattering bar cutting when constraints are in conflict.

Figure 3-20. Scattering Bars Cut Due to Conflicting Constraints



MAXSBWIDTH

OPCSBAR keyword: Manufacturing Constraint Arguments

Defines maximum width of scattering bars.

Usage

MAXSBWIDTH *value*

Arguments

- *value*

  A floating point number with a default value of the largest SBWIDTH in a SPACE or SPACELAYER keyword plus two database units.

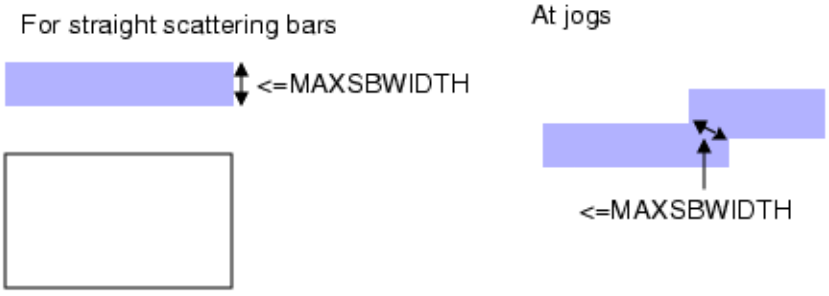
Description

An optional maximum width requirement that affects all scattering bars created by Calibre OPCsbar.

Set MAXSBWIDTH explicitly to prevent mismatching widths between your scattering bars and the SBWIDTH value. Calibre OPCsbar requires MAXSBWIDTH to be less than MINSBLENGTH. If MAXSBWIDTH is not set, it defaults to the largest value of SBWIDTH plus two database units. The two database units are used to account for angled scattering bars and also results in a width differential between your final scattering bars and SBWIDTH.

Figure 3-21 demonstrates MAXSBWIDTH measurement.

Figure 3-21. Where MAXSBWIDTH is Measured



MINEDGELENGTH

OPCSBAR keyword: Manufacturing Constraint Arguments

Defines minimum edge lengths.

Usage

MINEDGELENGTH *value*

Arguments

- *value*

  A floating point number with a default value of 0.

Description

An optional minimum length requirement that an edge must meet in order for it to receive scattering bars. The operation evaluates this parameter after classifying edges based on space conditions.

Use MINEDGELENGTH with caution, as the output produced when using it can vary significantly from output produced without it, as shown in Figure 3-22 and Figure 3-23. MINEDGELENGTH affects all scattering bars created by Calibre OPCsbar.

Figure 3-22 shows the impact of setting MINEDGELENGTH greater than MAXSBWIDTH. In this situation, scattering bars are not created for line-ends, so the vertical scattering bars would not be created by merging several small scattering bars.

Figure 3-22. MINEDGELENGTH and Line-ends

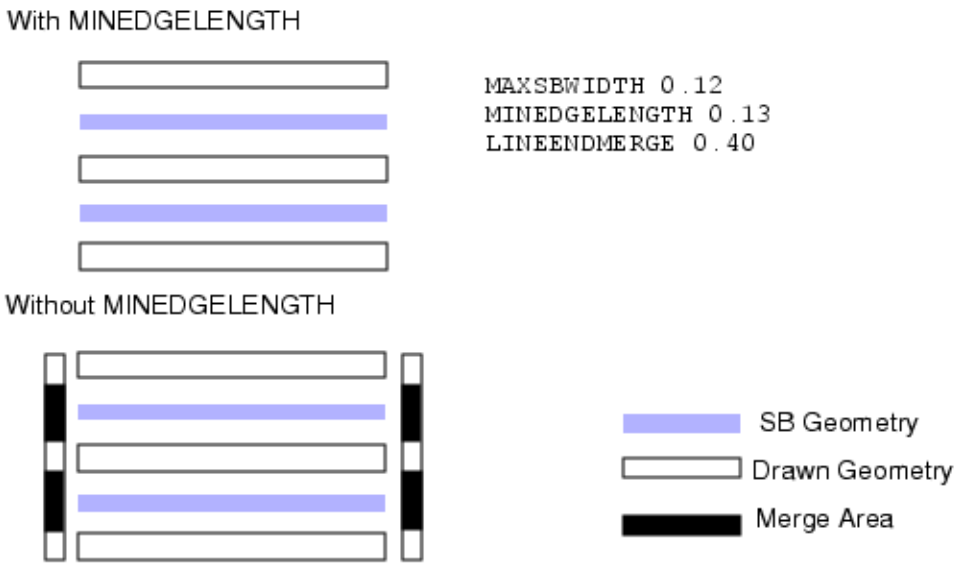
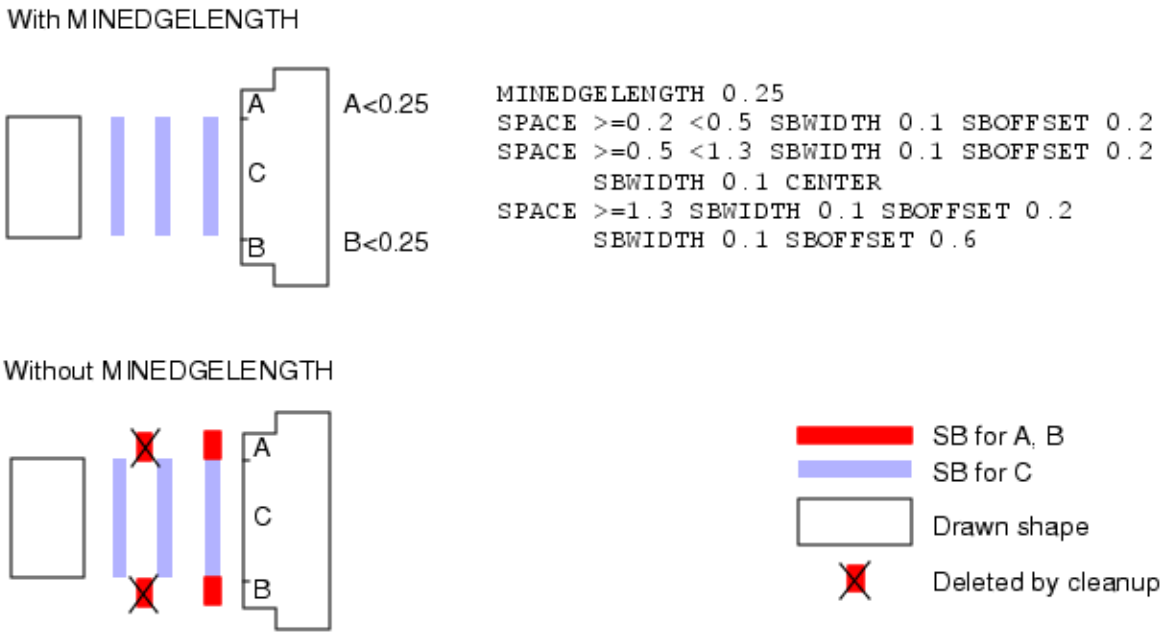


Figure 3-23 demonstrates the impact of MINEDGELENGTH where scattering bars are generated for an original edge that is classified into three edges based on spacing conditions. In the figure, A and B are shorter than MINEDGELENGTH.

- With MINEDGELENGTH, no scattering bars will be generated for A or B.
- Without MINEDGELENGTH, two scattering bars are generated for each of the edge segments A and B, although only the first bars remain after cleanup.

Figure 3-23. Impact of MINEDGELENGTH



MINSBLENGTH

OPCSBAR keyword: Manufacturing Constraint Arguments

Defines minimum length of scattering bars.

Usage

MINSBLENGTH *value*

Arguments

- *value*

  A floating point number with a default value of MAXSBWIDTH plus two database units.

Description

An optional minimum length requirement for all scattering bars created by this operation. The value applies to the total length of a scattering bar, not necessarily to each segment. Calibre OPCsbar requires MINSBLENGTH to be greater than SBWIDTH, and MINSBLENGTH has to be greater than MAXSBWIDTH otherwise Calibre OPCsbar confuses line-ends during cleanup. This operation affects all scattering bars created by Calibre OPCsbar.

When ANGLED is specified, the portion of the angled line that meets a specific SPACE condition can be quite small, even smaller than MINSBLENGTH. Because of this, Calibre OPCsbar interprets the minimum length requirement differently for angled lines. When ANGLED is specified, boundary edges less than MINSBLENGTH are treated as belonging to the adjacent space. Calibre appends multiple small boundary edges until the resulting classified edge is equal to or greater than MINSBLENGTH, at which time it generates the scattering bar for the compound edge.

MINSBOFFSET

OPCSBAR keyword: Manufacturing Constraint Arguments

Defines a minimum offset.

Usage

MINSBOFFSET [*condition*] *value* [OPPOSITE]

Arguments

- *condition*

  A range of floating point numbers that specifies the widths of the scattering bars for which the rule will be applied. This has no default value.

- *value*

  A floating point number that specifies the minimum distance allowed between any scattering bar and non-perpendicular original polygon edges. There is no default for value but if MINSBOFFSET is not explicitly set, value is set to the smallest SBOFFSET in any rule.

- OPPOSITE

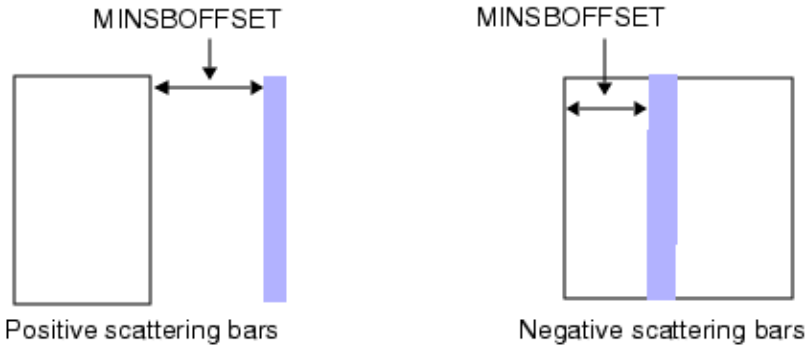
  The optional OPPOSITE keyword causes the OPPOSITE measurement metric to be used when enforcing offset spacing. For a discussion of this metric, see the metric parameters of the External operation in the *Standard Verification Rule Format (SVRF) Manual*. The default is to use the EUCLIDEAN metric.

Description

An option providing minimum offset requirements for all scattering bars generated by Calibre OPCsbar.

MINSBOFFSET is enforced identically for positive and negative scattering bars as shown in Figure 3-24.

Figure 3-24. Where MINSBOFFSET is Measured

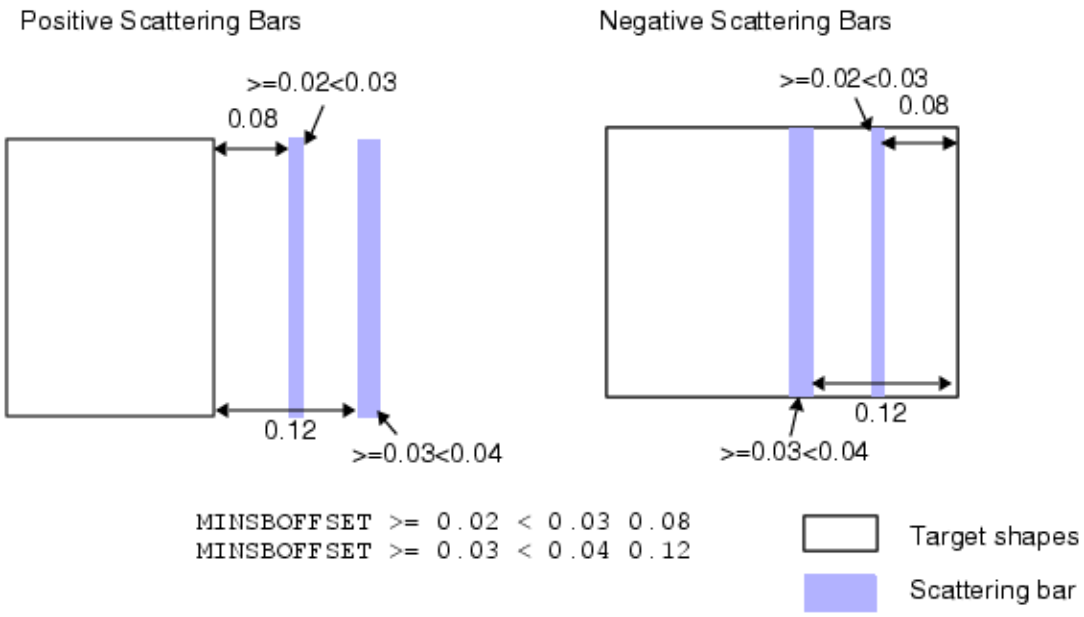


When multiple conditions are specified, different minimum offset values can be applied to scattering bars of different widths.

Examples

Figure 3-25 demonstrates scattering bars of differing widths, where widths greater than or equal to 0.02 but less than 0.03 must be at least 0.08 from the edge of a feature. Scattering bars of widths greater than or equal to 0.03 but less than 0.04 must be at least 0.12 from the edge of a feature.

Figure 3-25. MINSBOFFSET with Multiple Conditions



MINSBSPACE

OPCSBAR keyword: Manufacturing Constraint Arguments

Defines minimum scattering bar spacing.

Usage

MINSBSPACE *value*

Arguments

- *value*

  A floating point number with a default value of MINSBOFFSET.

Description

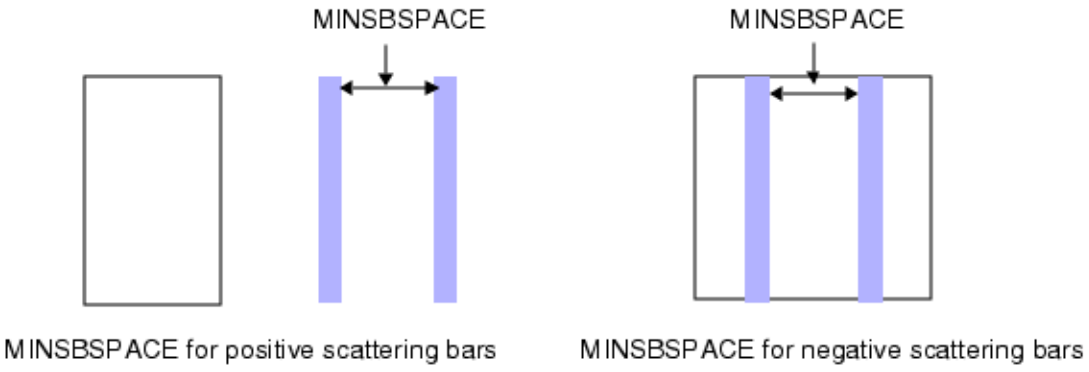
An optional minimum spacing requirement between adjacent scattering bars created by Calibre OPCsbar. This operation affects all scattering bars created by Calibre OPCsbar.

MINSBSPACE is enforced identically for positive and negative scattering bars.

In specific cases where cells are rotated, the scattering bars generated in the rotated cell may result in minor inconsistencies due to MINSBSPACE enforcement during cleanup control. More consistent results are achieved by specifying OPTION MINSBSPACE_CLEANUP_TIEBREAKER.

Figure 3-26 demonstrates MINSBSPACE (minimum scattering bar space).

Figure 3-26. Where MINSBSPACE is Measured



MINSBWIDTH

OPCSBAR keyword: Manufacturing Constraint Arguments

Defines minimum scattering bar width.

Usage

MINSBWIDTH *value*

Arguments

- *value*

  A floating point number with a default value of the smallest SBWIDTH in any SPACE or SPACELAYER keyword minus one database unit.

Description

An optional minimum width requirement that affects all scattering bars created by Calibre OPCsbar.

Set MINSBWIDTH explicitly to prevent mismatching widths between your scattering bars and the SBWIDTH value. If MINSBWIDTH is not set, it defaults to the smallest value of SBWIDTH minus one database unit. The one database unit is used to account for angled scattering bars and also results in a width differential between your final scattering bars and SBWIDTH.

Figure 3-27 demonstrates MINSBWIDTH measurement.

Figure 3-27. Where MINSBWIDTH is Measured

Style Control Arguments

Calibre OPCsbar provides detailed control to augment scattering bar placement.

For organizational keyword information, refer to “OPCSBAR” on page 35.

Table 3-6. Style Control Arguments

Argument	Description
ANGLED	Generates 45-degree scattering bars.
ANGLEEDGE	Generates scattering bars from angled target edges.
HORIZONTAL	Generates scattering bars from horizontal target only.
INTERSECTION	Define intersection types between scattering bars.
NEGATIVE	Generates negative scattering bars.
OPPOSITEEXTENDED	Generates scattering bars based on space.
STRICTCENTER	Generates scattering bars between two edges.
VERTICAL	Generates scattering bars from vertical target only.
WITHWIDTH	Defines the SVRF style of polygonal width classification.

ANGLED

OPCSBAR keyword: Style Control Arguments

Generates 45-degree scattering bars.

Usage

ANGLED

Arguments

None.

Description

An optional flag that directs Calibre OPCsbar to generate scattering bars adjacent to 45-degree boundary edges (angle measured with respect to the database axes) as well as the standard orthogonal boundary edges. Calibre OPCsbar generates scattering bars parallel to skewed target edges while not exceeding one DBU from a multiple of 45 degrees with respect to the database axes. This operation affects all scattering bars created by Calibre OPCsbar.

By default, Calibre OPCsbar only creates scattering bars for orthogonal edges, even if you supply a SPACELAYER containing angled edges.

Because Calibre builds scattering bars for angled edges out of small abutting or overlapping edges, these scattering bars are removed during cleanup unless JOG FILL is also specified. Even with JOG FILL specified, only angles that are multiples of 45 degrees are considered valid.

When ANGLED is specified, the portion of the angled line that meets a specific SPACE condition can be quite small, even smaller than MINSBLENGTH. Because of this, Calibre OPCsbar interprets the minimum length requirement differently for angled lines. When ANGLED is specified, boundary edges less than MINSBLENGTH are treated as belonging to the adjacent space. Calibre appends multiple small boundary edges until the resulting classified edge is equal to or greater than MINSBLENGTH, at which time it generates the scattering bar for the compound edge.

ANGLEEDGE

OPCSBAR keyword: Style Control Arguments

Generates scattering bars from angled target edges.

Usage

ANGLEEDGE

Arguments

None.

Description

An optional keyword that applies conditions of the SPACE keyword to those edges derived from angled features. Vertical or horizontal edges that are classified by this rule are ignored unless a similar condition exists specifying the VERTICAL or HORIZONTAL keywords. This option is only available when used with SPACE.

HORIZONTAL

OPCSBAR keyword: Style Control Arguments

Generates scattering bars from horizontal target only.

Usage

HORIZONTAL

Arguments

None.

Description

An optional flag that enables the application of the present rule only to those edges derived from horizontal features. Vertical or angled edges that are classified by this rule are ignored unless a similar rule exists specifying the VERTICAL or ANGLEEDGE keywords.

INTERSECTION

OPCSBAR keyword: Style Control Arguments

Define intersection types between scattering bars.

Usage

INTERSECTION {L [*u_bottom_length*] | N | P | T}

Arguments

There are four valid intersection types:

- L [*u_bottom_length*]

  L-shaped intersections are allowed. This is the default.

  The optional *u_bottom_length* parameter is a positive, floating-point number, interpreted as a minimum length in user units of the bottom of a U-shaped polygon. If two L-shaped polygons abut to form a U-shaped polygon, the distance between the parallel bars of the polygon is measured. If that distance is less than *u_bottom_length*, the bottom of the U is removed.

- N

  No intersections are allowed.

- P

  Plus-shaped, T-shaped, and L-shaped intersections are allowed.

- T

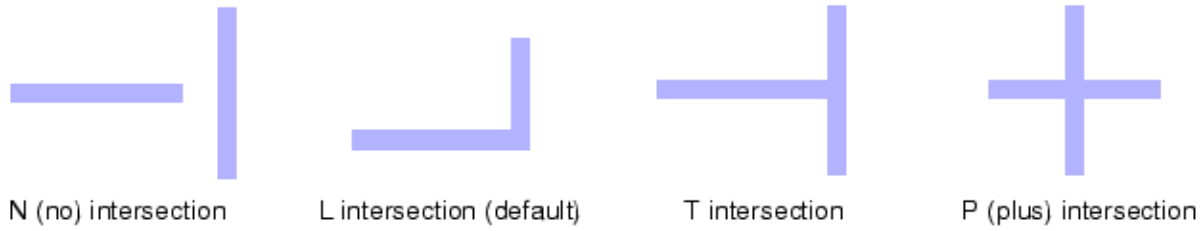
  T-shaped and L-shaped intersections are allowed.

Description

An optional keyword that defines the types of intersections allowed between scattering bars. This operation affects all scattering bars created by Calibre OPCsbar.

Figure 3-28 depicts the four intersection options.

Figure 3-28. INTERSECTION Options



When Calibre encounters two scattering bars with violating intersections, it trims the overlaps to less than or equal to MINSBLENGTH, and cuts areas of violation to generate an allowed

intersection. For more detail, refer to “Cleanup of Intersection Violations” in the section entitled “Scattering Bar Cleanup Order” on page 129.

NEGATIVE

OPCSBAR keyword: Style Control Arguments

Generates negative scattering bars.

Usage

NEGATIVE

Arguments

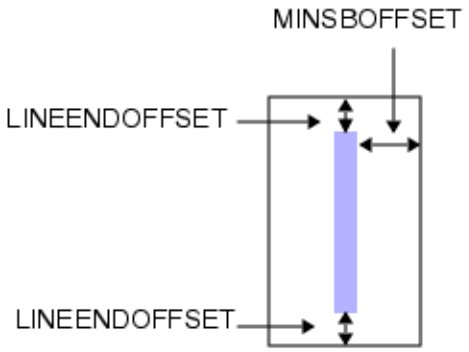
None.

Description

An optional keyword that directs Calibre OPCsbar to generate negative scattering bars. Negative scattering bars are non-printing assist features inside the original feature. The default is to generate positive scattering bars. This keyword controls all scattering bars created by the operation; you cannot generate both negative and positive scattering bars in a single operation. This operation affects all scattering bars created by Calibre OPCsbar.

To generate partial-cut negative scattering bars, use MINSBOFFSET and LINEENDOFFSET as shown in Figure 3-29.

Figure 3-29. Partial-Cut Negative Scattering Bars



OPPOSITEEXTENDED

OPCSBAR keyword: Style Control Arguments

Generates scattering bars based on space.

Usage

OPPOSITEEXTENDED *value*

Arguments

- *value*

  A non-zero positive floating point number.

Description

An optional flag that defines the method used to classify edges based on space. This operation affects all scattering bars created by Calibre OPCsbar. When OPPOSITEEXTENDED is specified, edges participating in space searches are extended by the specified *value*.

You must specify at least one SPACE keyword when using OPPOSITEEXTENDED.

Note

OPPOSITEEXTENDED and STRICTCENTER cannot be used together.

STRICTCENTER

OPCSBAR keyword: Style Control Arguments

Generates scattering bars between two edges.

Usage

STRICTCENTER

Arguments

None.

Description

An optional argument that determines whether a scattering bar is generated between two edges, if the SPACE and one WIDTH condition are satisfied.

When only one WIDTH condition is satisfied and if STRICTCENTER is not specified, a scattering bar is placed in the center between both edges. When STRICTCENTER is specified, a scattering bar is not generated.

Note

STRICTCENTER and OPPOSITEEXTENDED cannot be used together.

VERTICAL

OPCSBAR keyword: Style Control Arguments

Generates scattering bars from vertical target only.

Usage

VERTICAL

Arguments

None.

Description

An optional flag that enables the application of the present rule only to those edges derived from vertical features. Horizontal or angled edges that are classified by this rule are ignored unless a similar rule exists specifying the HORIZONTAL or ANGLEEDGE style controls.

WITHWIDTH

OPCSBAR keyword: Style Control Arguments

Defines the SVRF style of polygonal width classification.

Usage

WITHWIDTH

Arguments

None.

Description

An optional keyword that defines the method used to classify edges based on width. When WITHWIDTH is not specified, INTERNAL OPPOSITE is used to classify edges. When WITHWIDTH is specified, classification of the edges behaves the same as the WITH WIDTH operation in SVRF.

For a detailed discussion of the differences between INTERNAL OPPOSITE and WITHWIDTH, refer to “How WIDTH is Measured” on page 123. This operation affects all scattering bars generated by Calibre OPCsbar.

There must be at least one SPACE and WIDTH classification when using WITHWIDTH.

Cleanup Control Arguments

Calibre OPCsbar provides post-placement cleanup to ensure MRC-clean scattering bar output.

For organizational keyword information, refer to “OPCSBAR” on page 35.

Table 3-7. Cleanup Control Arguments

Argument	Description
CENTERMERGE	Merges adjacent scattering bars by center.
COLINEARIZE	Merges adjacent scattering bars.
EXTENSION	Adjusts ends of scattering bars.
JOG FILL	Cleans overlapping scattering bars.
LINEENDMERGE	Merges adjacent scattering bars by end.
MINJOGWIDTH	Defines the minimum jog width.
NOCLEANUP	Inhibits cleanup of scattering bars.
OPENT	Cleans up T-shaped scattering bars.
OPTION CAREFUL_CLEANUP	Cleans scattering bars with increased stringency.
OPTION CLEAN_RESEAT	Enables generation of a fuller set of scattering bars.
OPTION FAST_MRC	Reduces cleaning stringency of scattering bars.
OPTION INCLUSIVE_JOG_LENGTH	Dispositions cleanup for scattering bars in jog conditions.
OPTION MINSBSPACE_CLEANUP_TIEBREAKER	Cleans scattering bars with increased consistency between orthogonal data orientation.
OPTION PRIORITIZE_SPACE	Prioritizes negative scattering bars.
PRIORITIZE BY LAYER	Defines priorities for generating scattering bars.
PRIORITYCENTER	Prioritizes the generation of scattering bars.
SMOOTH	Smooths edges of input target.

CENTERMERGE

OPCSBAR keyword: Cleanup Control Arguments

Merges adjacent scattering bars by center.

Usage

CENTERMERGE *dist* [*width*]

Arguments

- *dist*

  A required floating point number specifying the maximum merge distance to be considered.

- *width*

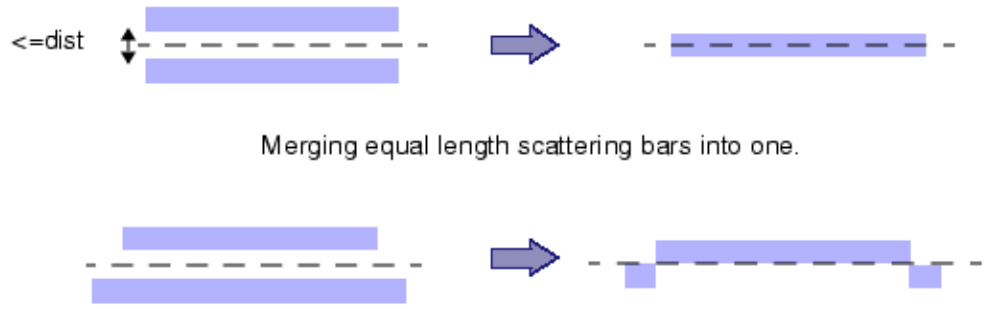
  An optional floating point number that controls the width of scattering bars created when performing center merge. This value is MAXSBWIDTH by default.

Description

An optional keyword that directs Calibre OPCsbar to merge adjacent scattering bars that are separated from one another by less than dist. Note that *dist* has no relation to any of the spacing or offset parameters. The default is no center merging.

The two scattering bars to be merged must have the same priority, but need not have the same length.

Figure 3-30. CENTERMERGE



COLINEARIZE

OPCSBAR keyword: Cleanup Control Arguments

Merges adjacent scattering bars.

Usage

COLINEARIZE *length* [*width*]

Arguments

- *length*

  A required positive floating point number specifying the maximum individual length of two overlapping scattering bars to be merged. When both overlapping scattering bars individually exceed the length value, no alignment or merging is performed.

- *width*

  An optional positive floating point number specifying the final width of merged scattering bars. The default for width is MINSBWIDTH.

Description

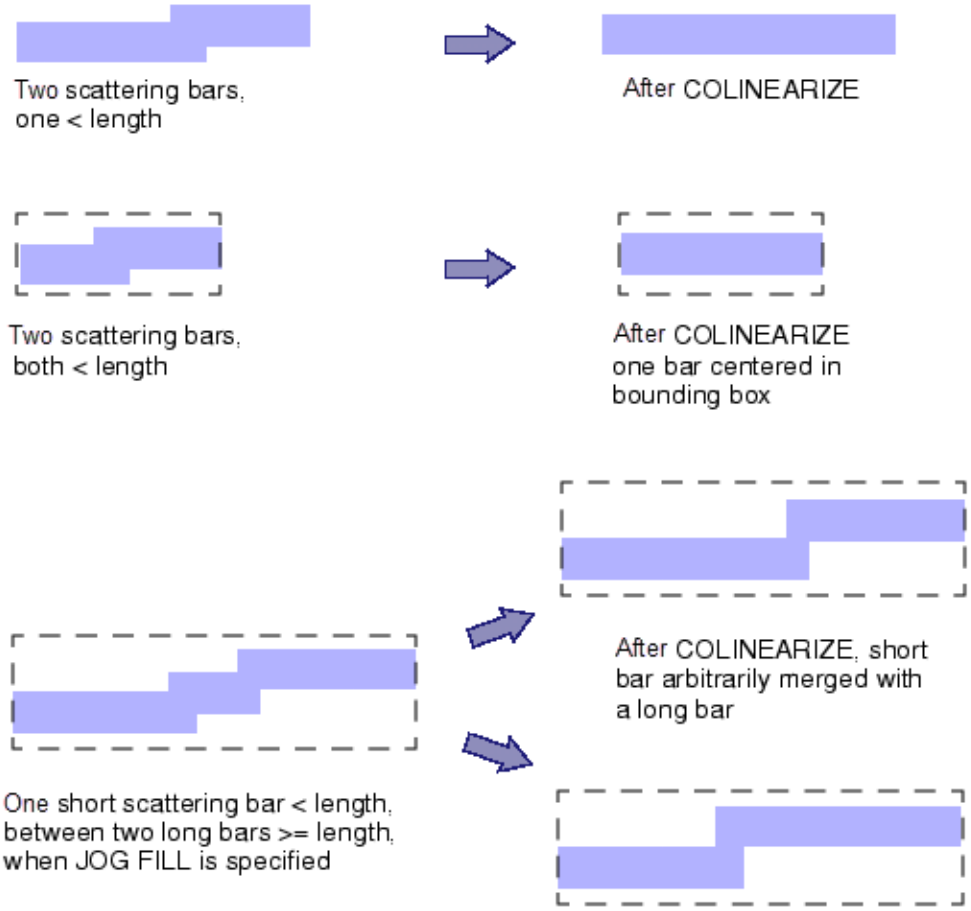
An optional keyword that directs Calibre OPCsbar to align and merge overlapping scattering bars that are misaligned with each other if at least one of the bars is less than or equal to the specified length value. To align and merge scattering bars, these conditions must be satisfied:

- At least one scattering bar is shorter than the specified length.
- They must have the same priority.
- They must overlap by at least two points. Singularity (kissing corners) is not considered overlapping.

Merging with the COLINEARIZE option occurs before JOG FILL. Jogs are only created when JOG FILL is enabled. COLINEARIZE is not enabled by default.

Figure 3-31 demonstrates the length argument control of merging scattering bars when applying COLINEARIZE.

Figure 3-31. COLINEARIZE Example



EXTENSION

OPCSBAR keyword: Cleanup Control Arguments

Adjusts ends of scattering bars.

Usage

EXTENSION *value* [BEFORECLEANUP]

Arguments

- *value*

  A floating point number with a default of 0. A positive value results in OPCSBAR extending valid bar ends. A negative value results in OPCSBAR shortening valid bar ends.

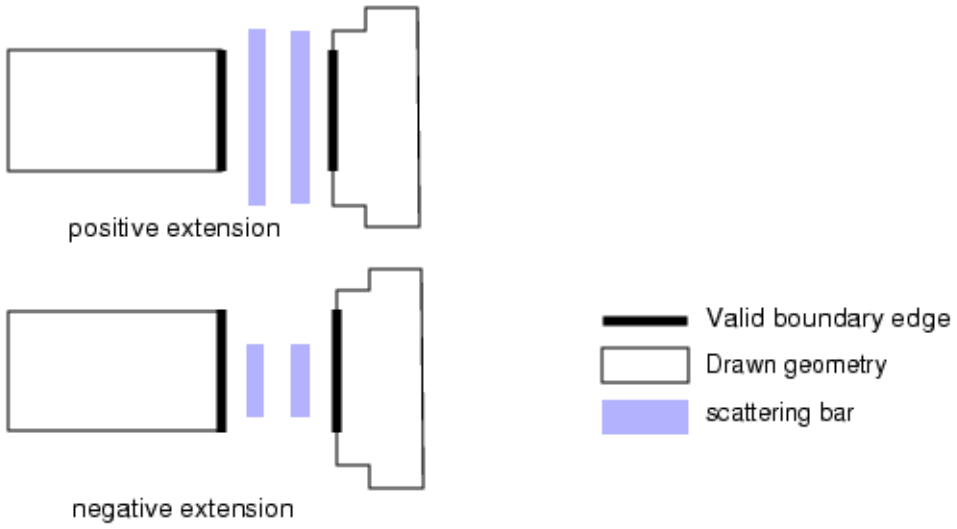
- BEFORECLEANUP

  The optional BEFORECLEANUP keyword causes extensions to be adjusted when scattering bars are generated. This means scattering bars and their adjusted extensions have the same priority during the cleanup phase. By default, original scattering bars have a higher priority than extensions during the cleanup phase.

Description

An optional keyword that extends or shortens scattering bar ends by the specified amount relative to their respective target edges. The EXTENSION keyword only applies to scattering bars generated by the SBOFFSET and SBWIDTH SPACE keywords.

Figure 3-32. Extensions on Positive Scattering Bars



JOG FILL

OPCSBAR keyword: Cleanup Control Arguments

Cleans overlapping scattering bars.

Usage

JOG FILL *dist* [*width*]

Arguments

- *dist*

  Required floating point value that directs JOG FILL to correct spacing violations between adjacent scattering bars that are within the value of dist to each other. This value must be greater than or equal to MINSBLENGTH.

- *width*

  The optional width argument controls the width of scattering bars created when resolving violations. The default is MINSBWIDTH.

Description

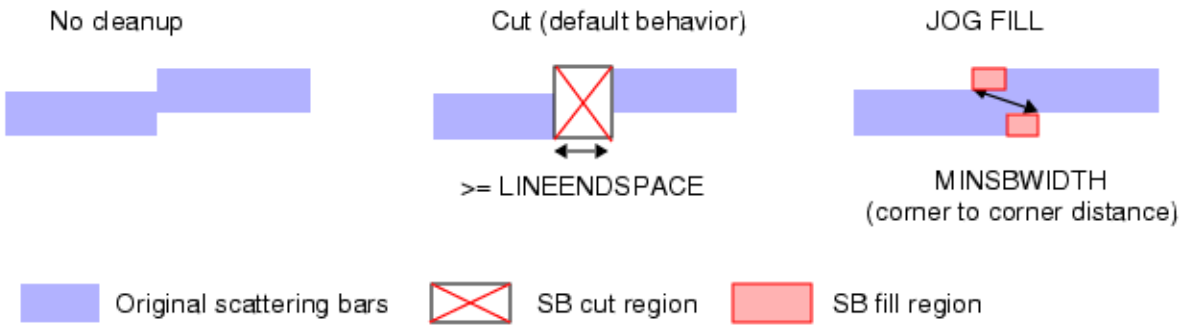
An optional control that cleans up scattering bars that are overlapping, touching, have singularities (kissing corners), or those within the value of dist to another. Jog filling only occurs between scattering bars of equal priority.

Note

During nmOPC and SRAF Print Avoidance (SPA) after SRAF generation, fragments near concave corners cannot be aligned and eliminated. SPA cannot resolve printing at or near jogs due to the inability to align fragments near concave corners.

By default, Calibre OPCsbar trims scattering bars that violate spacing rules to form an MRC-free separation. JOG FILL handles scattering bars equally to ensure that the resulting overlap is equal to MINSBWIDTH.

Figure 3-33. Styles of Jog Interaction



JOG FILL cleans up overlapping or touching scattering bars when the overlap is less than MINJOGWIDTH. The value of dist must be greater than or equal to MINSBLENGTH. Refer to Figure 4-27 for more information.

Given multiple ways to fill jogs, Calibre OPCsbar arbitrarily chooses the method. For more detail regarding how JOG FILL resolves space violations, refer to “Cleanup of Illegal Jogs” in the section entitled “Scattering Bar Cleanup Order” on page 129.

LINEENDMERGE

OPCSBAR keyword: Cleanup Control Arguments

Merges adjacent scattering bars by end.

Usage

LINEENDMERGE *value*

Arguments

- *value*

  Floating point number. The default is 0 (no merging).

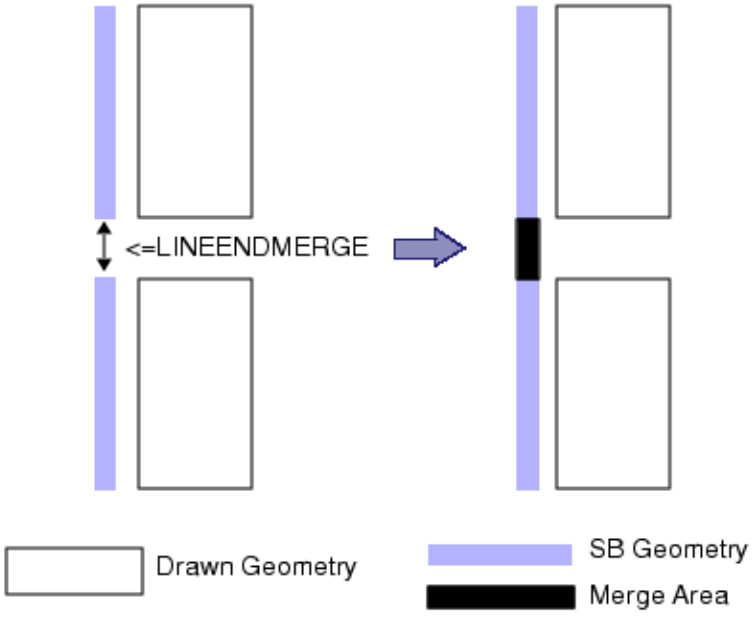
Description

An optional keyword that directs Calibre OPCsbar to merge scattering bars when they meet the following conditions:

- The line-ends (a scattering bar line-end is defined as greater than or equal to MINSBWIDTH and less than or equal to MAXSBWIDTH) of the two scattering bars are lined up perfectly, so that no jogs are created by merging.
- The distance between the two scattering bars is less than or equal to value.

Line-end merging is demonstrated in Figure 3-34.

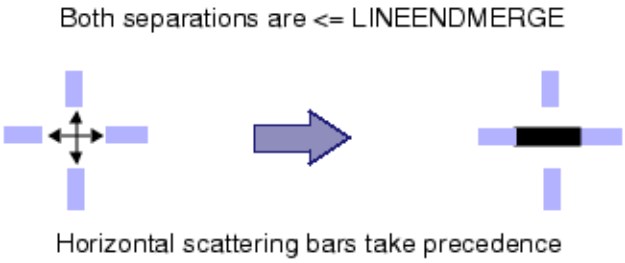
Figure 3-34. Merging at Line-ends



As with other cleanup operations, the application assigns the highest priority to horizontal scattering bars, and the lowest priority to angled scattering bars. If two pair of scattering bars

meet the requirements for merging, but merging would cause the resulting scattering bars to conflict, only the horizontal scattering bars will be merged as depicted in Figure 3-35.

Figure 3-35. Resolving Merge Conflicts



MINJOGWIDTH

OPCSBAR keyword: Cleanup Control Arguments

Defines the minimum jog width.

Usage

MINJOGWIDTH *value*

Arguments

- *value*

  A floating point number that must be less than or equal to MAXSBWIDTH.

Description

An optional keyword that defines the minimum overlap required for a jog to be legal. Calibre OPCsbar removes any scattering bars with SBWIDTH less than MINJOGWIDTH. If the width of the scattering bar jog is smaller than MINJOGWIDTH, the scattering bars associated with the jog are removed when JOG FILL is specified. If the jog is larger, it is considered to be legal and the scattering bars are retained.

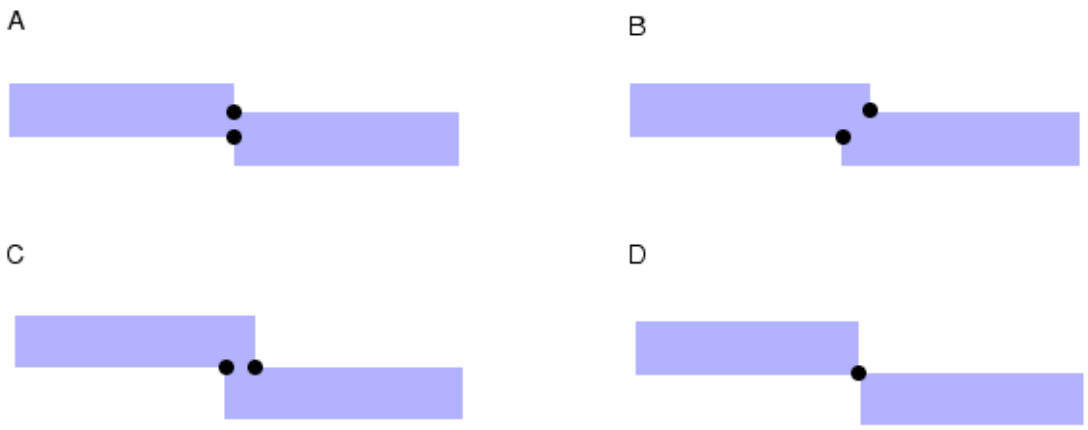
Note

During nmOPC and SRAF Print Avoidance (SPA) after SRAF generation, fragments near concave corners cannot be aligned and eliminated. SPA cannot resolve printing at or near jogs due to the inability to align fragments near concave corners.

This operation affects all scattering bars created by Calibre OPCsbar. The default MINJOGWIDTH without JOG FILL is 0. The default with JOG FILL is MAXSBWIDTH.

Figure 3-36 shows four jog configurations. The jog width is measured from point to point. In the case of the singularity (configuration D, kissing corners), the width is always 0, and JOG FILL removes these associated scattering bars.

Figure 3-36. Measuring Jog Widths



If the width of the jog is greater than or equal to MINJOGWIDTH, the jog is legal and is retained. If the width is less than MINJOGWIDTH, the jog is illegal and the associated scattering bars are trimmed to a legal spacing.

NOCLEANUP

OPCSBAR keyword: Cleanup Control Arguments

Inhibits cleanup of scattering bars.

Usage

NOCLEANUP [*priority*]

Arguments

- *priority*

  An optional integer referring to the priority of the scattering bars to act upon. When used without priority, Calibre OPCsbar returns all raw scattering bars.

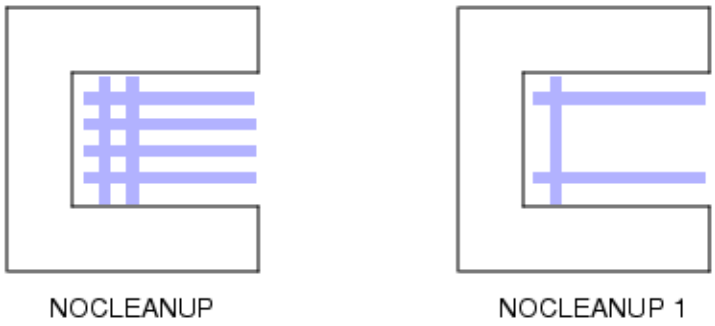
  Calibre OPCsbar assigns priorities at run time to the scattering bars it creates. The highest priority scattering bars have the priority 1, while the lowest priority scattering bars have the largest assigned priority number. When NOCLEANUP is used with priority, Calibre returns only the raw scattering bars with this priority number. If priority is greater than the highest priority number assigned at run time, Calibre returns the raw scattering bars with the largest assigned priority number.

Description

An optional keyword that directs Calibre to stop processing after generating the raw scattering bars and performing extensions and line-end merging. When specified, Calibre does not enforce spacing or width rules and does not enforce allowed intersection rules. The default is to perform scattering bar cleanup. This operation affects all scattering bars generated by Calibre OPCsbar.

This keyword is used only for debugging purposes and is demonstrated in Figure 3-37.

Figure 3-37. Using NOCLEANUP



OPENT

OPCSBAR keyword: Cleanup Control Arguments

Cleans up T-shaped scattering bars.

Usage

OPENT

Arguments

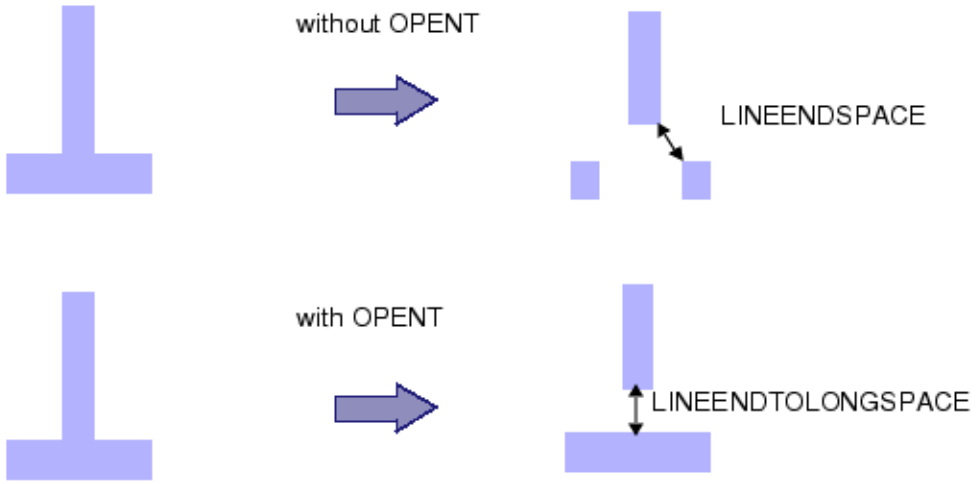
None.

Description

An optional keyword controlling how Calibre OPCsbar cleans up T-shaped scattering bars when no intersections are allowed. This operation affects all scattering bars generated by Calibre OPCsbar.

By default, the Calibre OPCsbar cleanup operation trims the center of the intersection away equally. If you specify OPENT, the Calibre OPCsbar cleanup operation trims away stems of T-shaped intersections.

Figure 3-38. Cleanup with and without OPENT using INTERSECTION N



When cleaning up L intersections, Calibre OPCsbar always trims the stems of T intersections. In this situation, the OPENT keyword does not affect the results of cleanup.

OPTION CAREFUL_CLEANUP

OPCSBAR keyword: Cleanup Control Arguments

Cleans scattering bars with increased stringency.

Usage

OPTION CAREFUL_CLEANUP {0 | 1}

Arguments

- 0 | 1

  Specifies the level of scattering bars that are to be recaptured. Setting this value to 0 disables it and does not recapture scattering bars. The default cleanup value of 1 recaptures some of the lost scattering bars and provides better coverage than the value of 0.

Description

An optional keyword specifying the level that scattering bars will be recaptured. Scattering bars are recaptured by comparing the original raw scattering bars to the cleaned up scattering bars.

This option can also be set using the OPCSBAR_CAREFUL_CLEANUP environment variable although the option takes precedence over the environment variable setting. To set this option using an environment variable, set OPCSBAR_CAREFUL_CLEANUP to 0 in a shell:

```
setenv OPCSBAR_CAREFUL_CLEANUP 0
```

Note

Siemens EDA recommends setting OPTION CAREFUL_CLEANUP to to 0. This forces OPTION FAST_MRC to be set to off by default. This setting provides the optimal scattering bar cleanup setting.

OPTION CLEAN_RESEAT

OPCSBAR keyword: Processing Mode Arguments

Enables generation of a fuller set of scattering bars.

Usage

OPTION CLEAN_RESEAT {NO | YES}

Arguments

NO | YES

Description

An optional keyword that enables generation of a fuller set of scattering bars.

Siemens EDA recommends setting OPTION CLEAN_RESEAT to YES. OPTION CLEAN_RESEAT is set to NO by default.

Examples

Use OPTION CLEAN_RESEAT for better scattering bar output:

```
richer_sbars = OPCSBAR input
   <MRC RULES ...>
   <SBAR recipe ...>
   OPTION CLEAN_RESEAT YES
```

OPTION FAST_MRC

OPCSBAR keyword: Cleanup Control Arguments

Reduces cleaning stringency of scattering bars.

Usage

OPTION FAST_MRC {NO | YES}

Arguments

- NO | YES

  Specifies whether the last pass of MRC cleanup is skipped.

Description

An optional keyword that skips the last pass of MRC cleanup in the design. To use OPTION FAST_MRC, OPTION CAREFUL_CLEANUP must be set to 1. If OPTION CAREFUL_CLEANUP is set to 0, OPTION FAST_MRC is disabled by default and a warning message is issued to the transcript.

This option can also be set using the OPCSBAR_FAST_MRC environment variable although the option takes precedence over the environment variable setting. To disable this option using an environment variable, set OPCSBAR_FAST_MRC to NO in a shell:

```
setenv OPCSBAR_FAST_MRC NO
```

OPTION INCLUSIVE_JOG_LENGTH

OPCSBAR keyword: Cleanup Control Arguments

Dispositions cleanup for scattering bars in jog conditions.

Usage

OPTION INCLUSIVE_JOG_LENGTH {NO | YES}

Arguments

NO | YES

Description

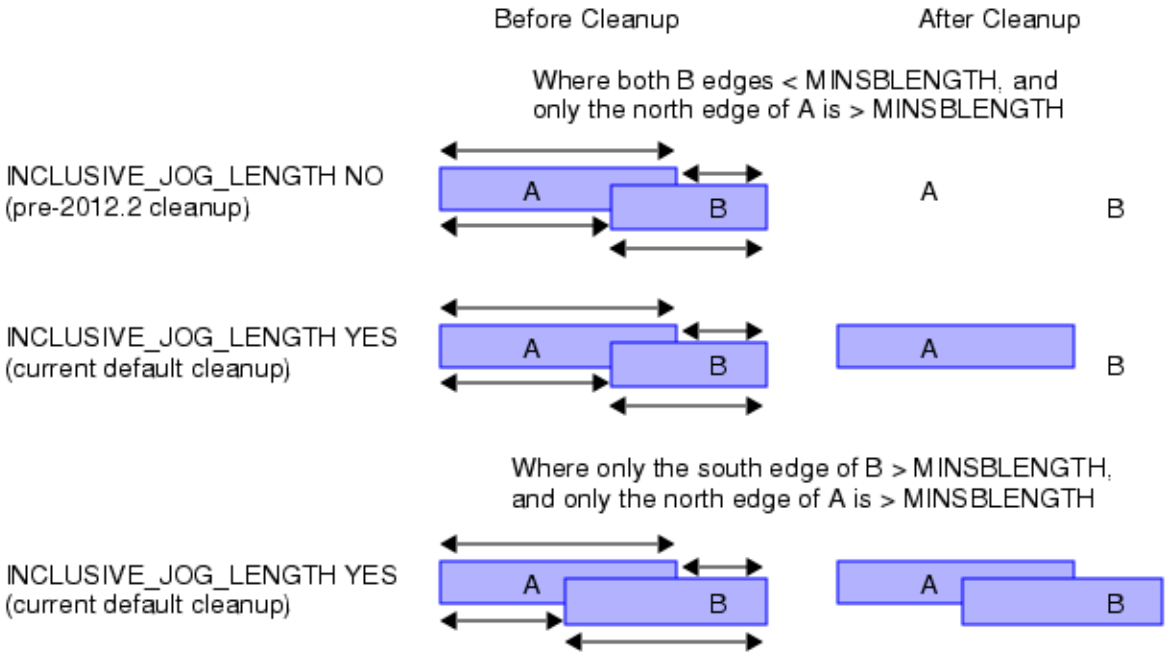
An optional keyword that deletes scattering bars in jog conditions if both the length edges fall below the minimum scattering bar length established with MINSBLENGTH. Setting this option to NO reverts jog cleaning to behavior prior to 2012.2. Previous behavior deletes the scattering bar if either edge is less than the minimum scattering bar length. You may see minor differences in the scattering bar output in jog conditions from previous scattering bar generation using this option.

This option can also be set using an environment variable. The option takes precedence over the environment variable setting. To set this option using an environment variable, set OPCSBAR_INCLUSIVE_JOG_LENGTH to NO in the program shell:

```
setenv OPCSBAR_INCLUSIVE_JOG_LENGTH NO
```

Calibre OPCsbar trims SRAF depending on this keyword as illustrated in Figure 3-39.

Figure 3-39. INCLUSIVE_JOG_LENGTH



OPTION MINSBSPACE_CLEANUP_TIEBREAKER

OPCSBAR keyword: Cleanup Control Arguments

Cleans scattering bars with increased consistency between orthogonal data orientation.

Usage

OPTION MINSBSPACE_CLEANUP_TIEBREAKER {NO | YES}

Arguments

- NO | YES

  Specifies an improved consistency algorithm for SRAF undergoing cleanup. Setting this value to NO disables it and does not enable the cleanup algorithm. Setting this value to YES enables it and performs the cleanup algorithm.

Description

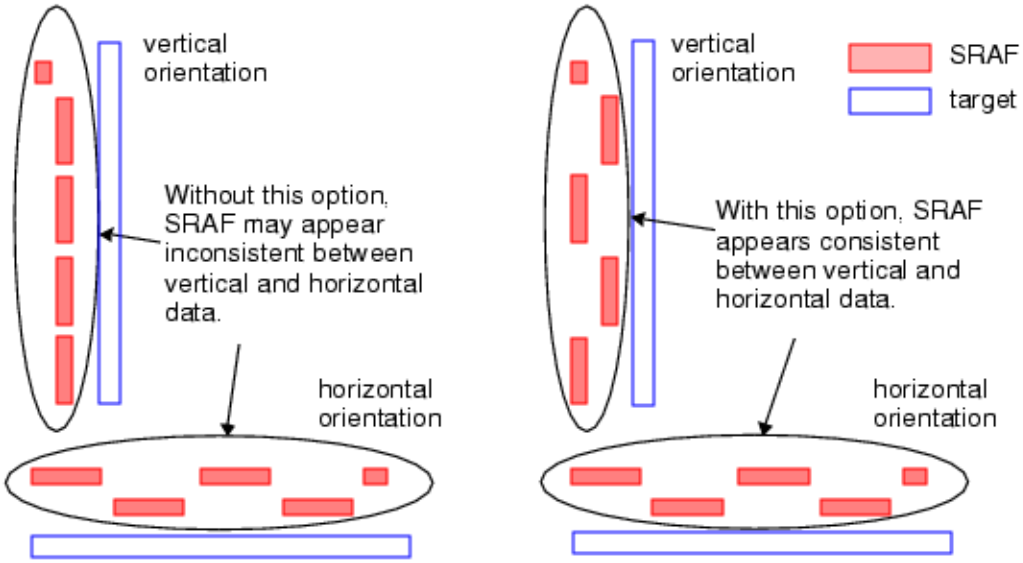
An optional keyword enabling more consistent SRAF cleanup control. In many cases where data runs vertically, generated scattering bars may appear inconsistent to identical horizontally oriented data due to MINSBSPACE enforcement during cleanup control. More consistent results may be achieved by setting OPTION MINSBSPACE_CLEANUP_TIEBREAKER to YES. This option has no effect with non-orthogonal data.

You can set this option using the OPCSBAR_MINSBSPACE_CLEANUP_TIEBREAKER environment variable. The option takes precedence over the environment variable. You enable the environment variable in a shell. For example:

```
setenv OPCSBAR_MINSBSPACE_CLEANUP_TIEBREAKER 1
```

Behavior of OPTION MINSBSPACE_CLEANUP_TIEBREAKER is depicted in Figure 3-40.

Figure 3-40. OPTION MINSBSPACE_CLEANUP_TIEBREAKER



OPTION PRIORITIZE_SPACE

OPCSBAR keyword: Cleanup Control Arguments

Prioritizes negative scattering bars.

Usage

OPTION PRIORITIZE_SPACE {NO | YES}

Arguments

NO | YES

Description

An optional keyword that prioritizes negative scattering bars based on the SPACE condition in a rule. Negative scattering bars generated from smaller SPACE conditions have a higher priority than negative scattering bars generated from larger SPACE conditions. Negative scattering bars that are generated with the same SPACE condition have the same priority regardless of any WIDTH conditions.

This option can also be enabled using the OPCSBAR_PRIORITIZE_SPACE environment variable although the option takes precedence over the environment variable setting. To enable this option using an environment variable, set OPCSBAR_PRIORITIZE_SPACE to 1 in a shell:

```
setenv OPCSBAR_PRIORITIZE_SPACE 1
```

PRIORITIZE BY LAYER

OPCSBAR keyword: Cleanup Control Arguments

Defines priorities for generating scattering bars.

Usage

PRIORITIZE BY LAYER *layer1* [*layerN* …]

Arguments

- *layer1*

  Specifies the input polygon layers to prioritize by. The default is to prioritize by position.

Description

An optional control that defines a prioritization scheme to use when resolving intersections or spacing violations. When PRIORITIZE BY LAYER is specified, Calibre prioritizes the edges based on interaction with additional input layers, called island layers. This operation affects all scattering bars created by Calibre OPCsbar.

By default, the system assigns the highest priority to the scattering bars that are closest to the input geometry edges, second highest priority to the next closest scattering bars, and so on.

When you use the PRIORITIZE BY LAYER keyword, you must supply at least one island layer.

PRIORITYCENTER

OPCSBAR keyword: Cleanup Control Arguments

Prioritizes the generation of scattering bars.

Usage

PRIORITYCENTER

Arguments

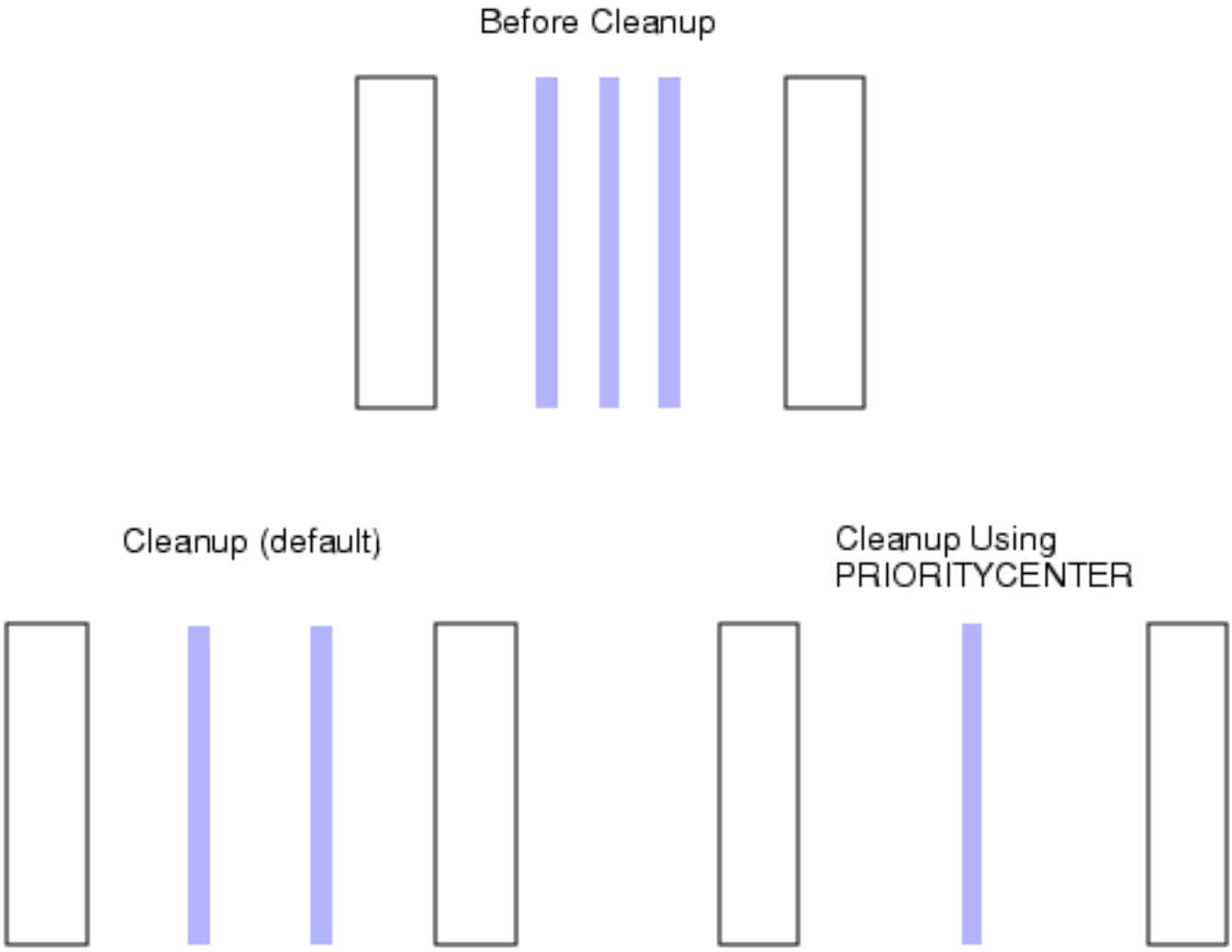
None.

Description

An optional keyword that gives centered scattering bars the highest priority during the cleanup process instead of scattering bars closest to any valid edge.

By default, Calibre performs an ordered prioritization where the highest priority scattering bars are those closest to the valid edge (see “Scattering Bar Cleanup Order” on page 129 for more detail). The next highest priority scattering bars are those that are second closest to the valid edge. When using PRIORITYCENTER, Calibre OPCsbar maintains center scattering bars at the expense of non-centered scattering bars as illustrated in Figure 3-41.

Figure 3-41. PRIORITYCENTER



SMOOTH

OPCSBAR keyword: Cleanup Control Arguments

Smooths edges of input target.

Usage

SMOOTH *value*

Arguments

- *value*

  A positive floating point number with no default value.

Description

An optional keyword used to smooth out small target edges based on constraints you specify with SPACE. If the length of a target edge is less than the *value* specified, the edge will be merged with its neighbor due to it being closer. For layers such as contact, it is recommended that *value* is set to the original layer edge length.

Processing Mode Arguments

Calibre OPCsbar uses various run modes for optimal processing time.

For organizational keyword information, refer to “OPCSBAR” on page 35.

Table 3-8. Processing Mode Arguments

Argument	Description
OPTION NO_PUSH	Enables data pushing, which may improve run time when invoking litho flat processing mode.
OPTION PROCESSING_MODE	Enables litho flat processing mode.
OPTION TILEMICRONS	Specifies a tile size.
OPTION VERIFICATION_MODE	Finds deleted scattering bars.

OPTION NO_PUSH

OPCSBAR keyword: Processing Mode Arguments

Enables data pushing, which may improve run time when invoking litho flat processing mode.

Usage

OPTION NO_PUSH {NO | YES}

Arguments

- NO | YES

  NO enables data pushing. YES disables data pushing. The default is NO.

Description

An optional keyword that disables pushing data to various hierarchical levels that may reduce run time for litho flat processing. This option is only meaningful when running OPTION PROCESSING_MODE FLAT (litho flat processing mode).

OPTION PROCESSING_MODE

OPCSBAR keyword: Processing Mode Arguments

Enables litho flat processing mode.

Usage

OPTION PROCESSING_MODE {FLAT | HIERARCHICAL | HYBRID}

Arguments

- FLAT

  Enables litho flat processing mode.

- HIERARCHICAL

  Specifies hierarchical processing mode. This is the default mode.

- HYBRID

  Specifies hybrid processing mode.

Description

An optional command that enables litho flat processing mode.

HIERARCHICAL mode splits the layout into artificial cells. Each artificial cell is split into tiles that are processed separately on remote machines.

FLAT mode enables litho flat processing, and splits the design into tiles which are processed separately on remote machines.

HYBRID mode enables litho hybrid processing, and examines the layout hierarchy to determine on a per-cell basis whether standard hierarchical or litho flat processing provides the best runtime performance.

To enable litho flat processing you must also specify LAYOUT ULTRA FLEX YES. Refer to the *Calibre Post-Tapeout Flow User's Manual* for more information. Additionally, specifying OPTION NO_PUSH 1 may reduce run time when running litho flat processing mode. See OPTION NO_PUSH for more information.

Examples

```
LAYOUT ULTRA FLEX YES
sbar_layer = OPCSBAR ...
   OPTION PROCESSING_MODE FLAT ...
```

OPTION TILEMICRONS

OPCSBAR keyword: Processing Mode Arguments

Specifies a tile size.

Usage

OPTION TILEMICRONS *value*

Arguments

- *value*

  Specifies a value for tiles in units of microns.

Description

Specifies the size of an OPCSBAR tile. Large tiles are broken into tiles of the value specified. The default is 100 microns. Use this keyword to improve processing scalability on designs with a very large amount of flat input data.

Examples

```
sbar_layer = OPCSBAR ...
   OPTION TILEMICRONS 35
   ...
```

OPTION VERIFICATION_MODE

OPCSBAR keyword: Processing Mode Arguments

Finds deleted scattering bars.

Usage

OPTION VERIFICATION_MODE {NO | YES}

Arguments

NO | YES

Description

An optional keyword that finds deleted scattering bars on the SBLAYER and copies them to the output layer. OPTION VERIFICATION_MODE requires an SBLAYER layer name to be declared within the OPCSBAR command. For example:

```
my_deleted_sbars = OPCSBAR input
   <MRC RULES ...>
   <SBAR recipe ...>
   OPTION VERIFICATION_MODE YES
   SBLAYER my_sbars
```

Scattering bars that satisfy MRC rules, copied to the SBLAYER my_sbars, are also copied to the output layer my_deleted_sbars.

OPTION VERIFICATION_MODE cannot be used with the SBLAYER optional argument CLEANSBLAYER, or with keyword NEGATIVE.

OPTION VERIFICATION_MODE is NO by default.

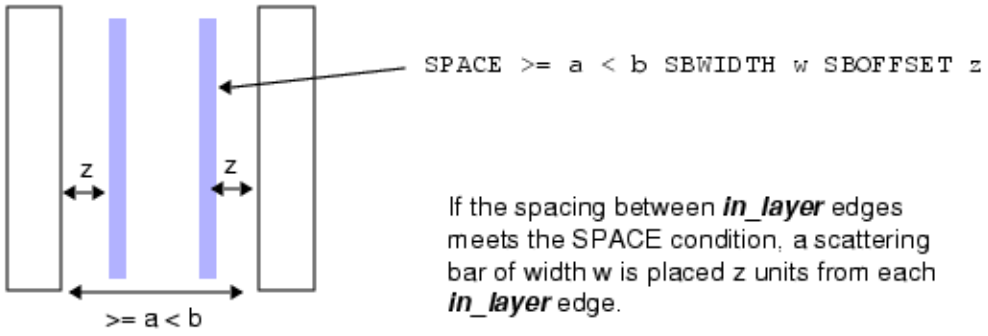
SPACE and SPACELAYER Overview

The SPACE and SPACELAYER keywords control the placement of scattering bars.

These keywords are mutually exclusive within a single OPCSBAR command. With SPACE you specify a dimensional condition and the scattering bar width with either an offset, pitch or center positioning for the generated scattering bars. The width of the target shapes can be optionally considered. The syntax is shown here with a typical SPACE rule demonstrated in Figure 3-42.

```
SPACE condition [WIDTH condition] {{SBWIDTH value {SBOFFSET value |
   SBPITCH value | CENTER [center_offset] | CENTERPITCH
[center_offset]}}…}
```

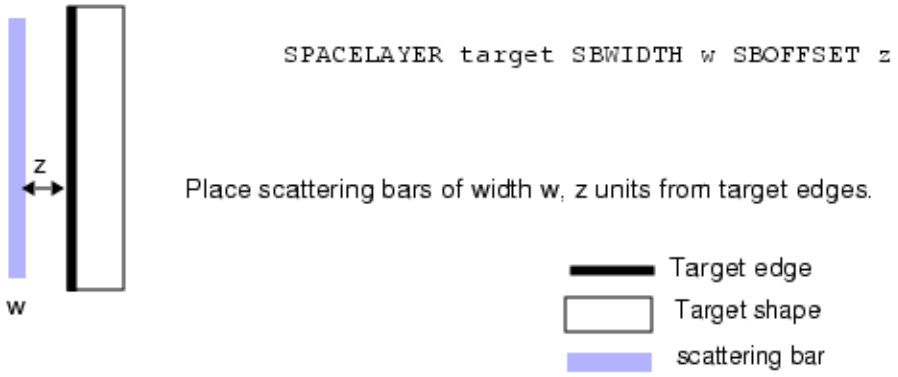
Figure 3-42. Typical SPACE Rule



With SPACELAYER you specify a layer composed of edges and the width and offset of the generated scattering bar. The syntax is shown here with a typical SPACELAYER rule demonstrated in Figure 3-43.

```
SPACELAYER layer {{SBWIDTH value SBOFFSET value}…}
```

Figure 3-43. Typical SPACELAYER Rule



Refer to Table 4-2 for a comparison of these keywords.

Data Returned by Calibre OPCsbar

Calibre OPCsbar generates layers that contain new scattering bars.

You must merge the layer containing the new scattering bars with the original mask data. To save the output, you must store it as a derived layer or with a DRC CHECK MAP command using SVRF. For more information, refer to “The Results Database” on page 15.

- To input results from an initial OPCSBAR command to a second one, assign the data to a new derived layer:

```
assist1 = OPCSBAR gate_poly poly <initial recipe>  //output as
                                                   //derived layer
assist {OPCSBAR poly
   SBLAYER assist1                                 //use as input
      PRIORITIZE BY LAYER active
      <second recipe> }
```

- To save the results of an OPCSBAR command, assign the data using a rulecheck statement, then specify a layer number for that rulecheck using the DRC CHECK MAP statement (detailed in the *Standard Verification Rule Format (SVRF) Manual*). The DRC CHECK MAP statement writes the results of the rule to the specified layer in the results database:

```
assist {OPCSBAR poly gate <sbar recipe>}  // assign to rulecheck

DRC CHECK MAP assist 10                   // specify layer name
```

- To input results to another command and save it, store OPCSBAR results as a derived layer so you can input them to another operation, then use the COPY command (detailed in the *Standard Verification Rule Format (SVRF) Manual*) to copy the derived layer to a rulecheck and use DRC CHECK MAP to assign it to a layer in the results database:

```
assist1 = OPCSBAR gate_poly poly <first sbar recipe>
                                     // output as derived layer
assist {OPCSBAR poly
   SBLAYER assist1                   // use as input
   PRIORITIZE BY LAYER active
   <second sbar recipe> }

sb_layer {COPY assist1}              // rulecheck generated
                                     // from derived layer

DRC CHECK MAP assist 10
DRC CHECK MAP sb_layer 30            // rulecheck written to
                                     // layout database
```

Chapter 4
Calibre OPCsbar Concepts

Understanding Calibre OPCsbar concepts speed development of OPCSBAR recipes.

Measurement Concepts 111
Types of Scattering Bars 112
Scattering Bar Type and Arguments 113
Definition of Scattering Bar Treatment 115
SPACE Keyword 115
SPACELAYER Keyword 125
Scattering Bar Cleanup 128
How Priority Affects Cleanup 140
Scattering Bar Cleanup Priority 141

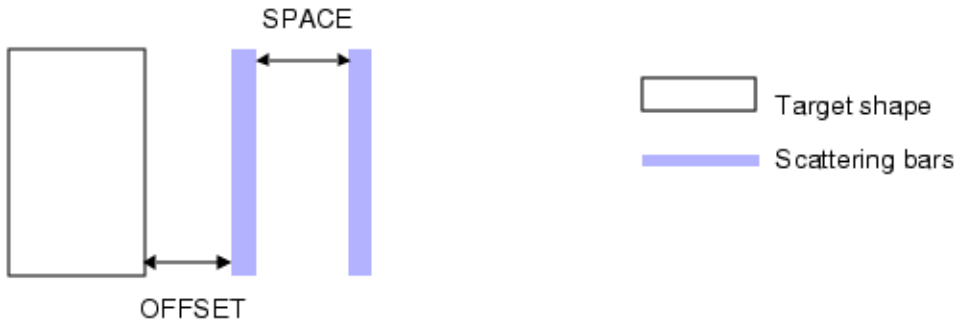
Measurement Concepts

OFFSET and SPACE are universally used by Calibre OPCsbar.

See Figure 4-1 for an example use.

- **OFFSET —** The distance between a scattering bar and the edge of a target shape.
- **SPACE —** The distance between two objects of the same type. For example, two scattering bars or two target shapes.

Figure 4-1. Where OFFSET and SPACE are Measured



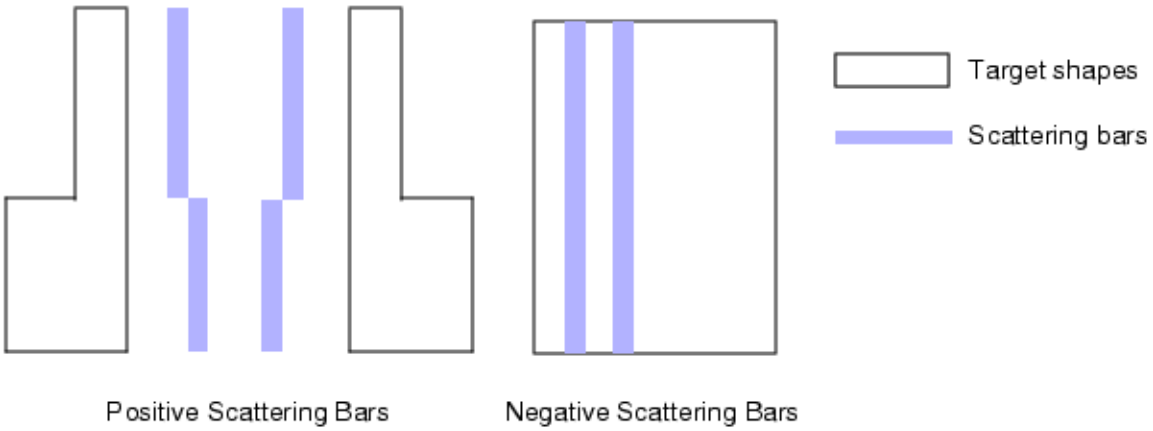
Types of Scattering Bars

Using Calibre OPCsbar, you can generate two types of scattering bars.

- Positive scattering bars, SRAFs added to the mask design.
- Negative scattering bars, representing areas removed from shapes.

These scattering bar types are depicted in Figure 4-2:

Figure 4-2. Two Types of Scattering Bars



By default, positive scattering bars are generated. To generate negative scattering bars, use the NEGATIVE keyword, a style control of the OPCSBAR command.

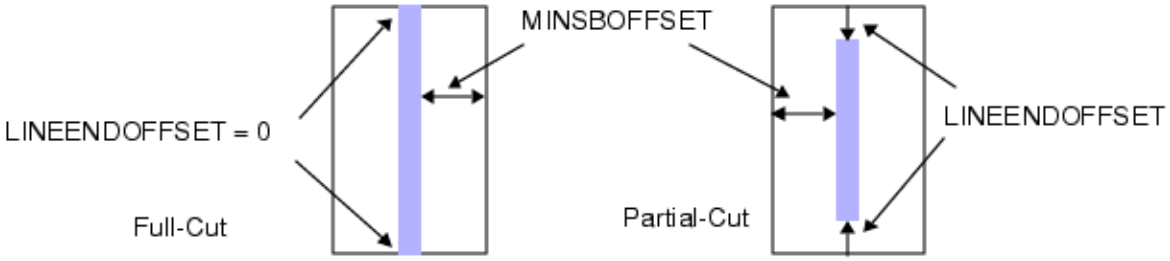
You can generate either type of scattering bar using either the SPACE or SPACELAYER keywords. SPACELAYER provides a greater level of customization. See “SPACELAYER Keyword” on page 125 for more information.

Using Calibre OPCsbar, you can generate either full-cut or partial-cut negative scattering bars.

- **Full-cut —** Negative scattering bars that extend the length of the feature. To generate full-cut negative scattering bar, set LINEENDOFFSET to 0, as shown in the following figure.
- **Partial-cut —** Negative scattering bars that are shorter than the original feature. When merged with the original data using the Boolean NOT operation, they result in holes in the feature.

To generate full-cut or partial-cut negative scattering bars, use MINSBOFFSET and LINEENDOFFSET as shown in Figure 4-3.

Figure 4-3. Full- and Partial-Cut Negative Scattering Bars



Scattering Bar Type and Arguments

OPCSBAR arguments interact with positive and negative scattering bars in various ways.

Table 4-1 lists the interactions of four arguments with positive and negative scattering bars.

Table 4-1. How Arguments Impact Scattering Bar Type

Argument	Positive Scattering Bars	Negative Scattering Bars
SBOFFSET	Measured from an edge outward	Measured from an edge inward
CENTER (SPACE rules only)	Meaningful for all bounded spaces	Meaningful for all bounded spaces
MINSBOFFSET	Measured from an edge outward	Measured from an edge inward
LINEENDOFFSET	Measured from all edges	Measured from all edges

Figures Figure 4-4 and Figure 4-5 illustrate how MINSBOFFSET and SBOFFSET are treated differently for positive and negative scattering bars.

Figure 4-4. Comparing MINSBOFFSET Keepout Areas

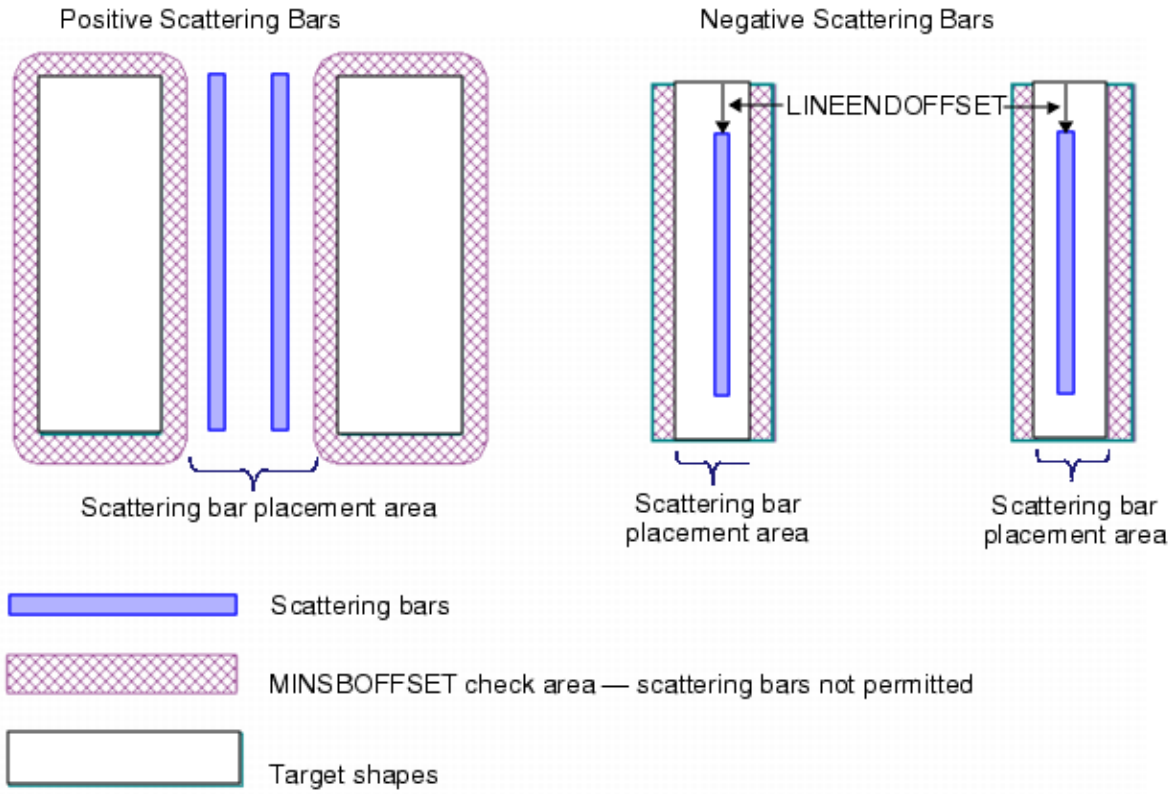
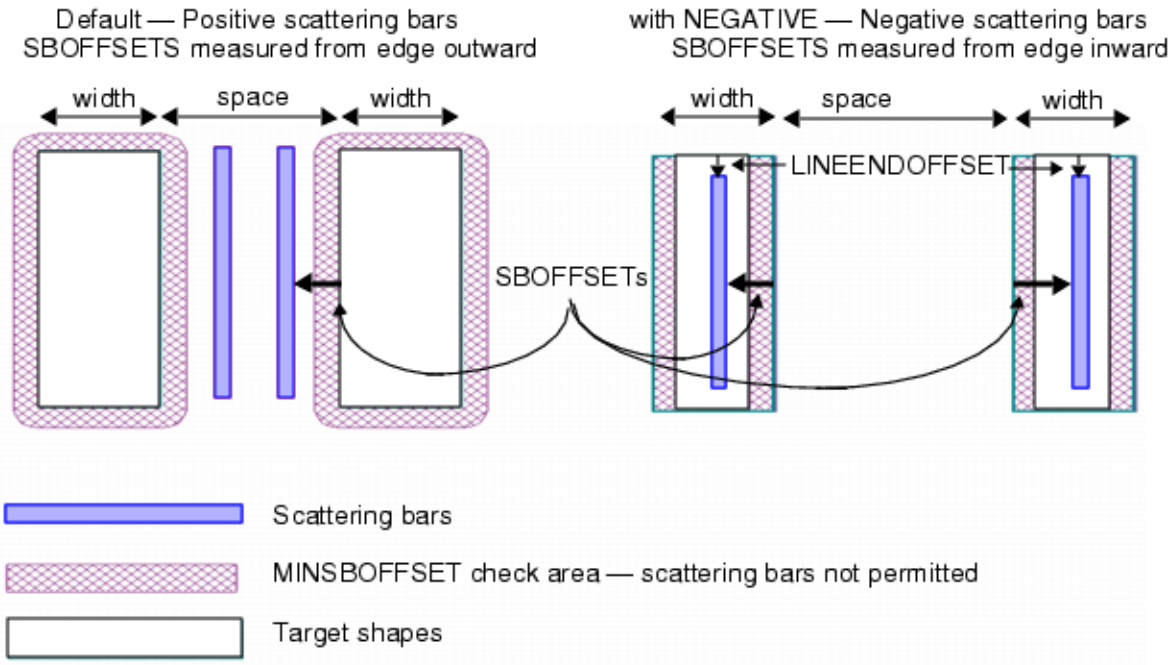


Figure 4-5. Measuring Positive and Negative SBOFFSET Values



Definition of Scattering Bar Treatment

Calibre OPCsbar provides two mutually exclusive methods for specifying which edges require scattering bar enhancement.

- SPACE — You translate tables into rules using the SPACE keyword. See "SPACE Keyword" on page 115 for more information. As the Calibre OPCsbar operation processes the data, it evaluates all edges and classifies them according to the rules you have written to calculate the exact scattering bar treatment to apply, if any.
- SPACELAYER — To write more involved rules with a greater number of processing steps or measurement ranges use SPACELAYER. See "SPACELAYER Keyword" on page 125 for more information. You isolate those edges requiring scattering bar treatment, and write rules defining that treatment using the SPACELAYER keyword. SPACELAYER does not evaluate the edges or classify them; it only applies the scattering bar treatment you specified for that input layer.

Table 4-2 gives a comparison between the SPACE and SPACELAYER keywords.

Table 4-2. Comparing SPACE and SPACELAYER Keywords

	SPACE	SPACELAYER
Categorize edges by space	Yes	Yes, using SVRF outside the Calibre OPCsbar operation
Categorize edges by width	Yes, if using WIDTH	Yes, using SVRF outside the Calibre OPCsbar operation
Categorize edges by other characteristics	Yes, using SVRF outside the Calibre OPCsbar operation	Yes, using SVRF outside the Calibre OPCsbar operation
Treat edges as pairs, with both edges receiving the same treatment	Yes	No
Treat edges individually	If ref_layer is supplied	Yes
Pass in edge layer for treatment	Yes	Yes
Center scattering bars between edges	Yes	No
Difficulty	Basic	Advanced knowledge of SVRF required

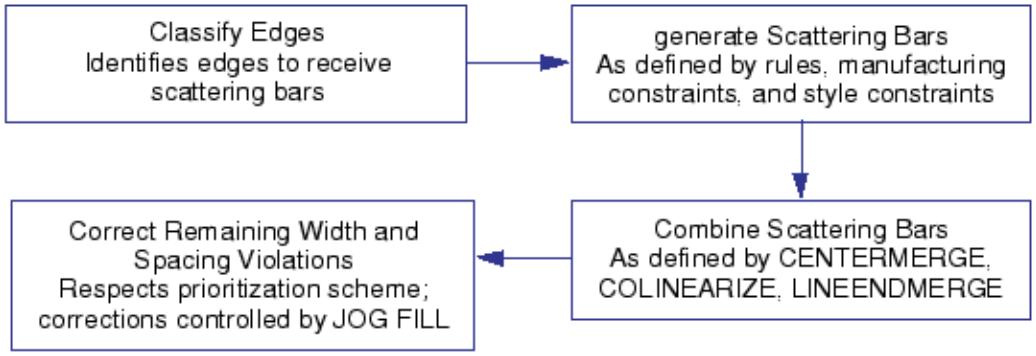
SPACE Keyword

Defines a constraining spatial range between edges to qualify and generate scattering bars.

When you use the SPACE keyword, OPCSBAR classifies edges based on the distance to the nearest edge.

Figure 4-6 shows the flow of scattering bar generation from the SPACE keyword.

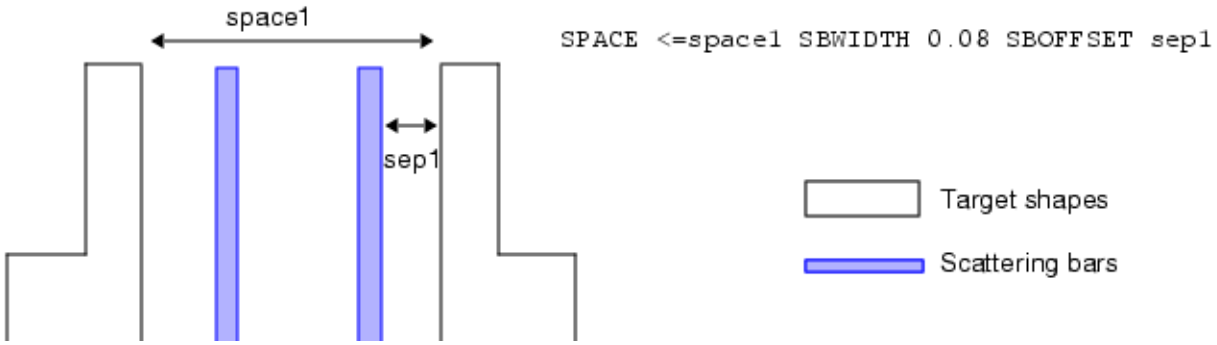
Figure 4-6. How OPCSBAR Generates Scattering Bars using SPACE



You define the classes of edges by writing rules, with each rule defining one class in terms of a range of distances (spaces). Each rule also defines the treatment that class of edges is to receive. To generate scattering bars for a shape, you specify the width of the scattering bar, using SBWIDTH, and the placement of the scattering bar with respect to the shape edges, using either SBOFFSET or CENTER.

Figure 4-7 shows an example of specifying edges using SPACE.

Figure 4-7. Specifying Edges With SPACE



Understanding Spaces

When classifying edges, Calibre OPCsbar looks at edges and spaces.

- An edge is any polygon edge on either the ***in_layer*** or the optional *ref_layer*.
- A space is an area between two adjacent edges.
- The boundary edges for a space are the pair of adjacent facing edges on either side of the space.

Calibre OPCsbar finds spaces as follows: For every edge on the ***in_layer***, Calibre extends measurement regions from the exterior sides of edges outward for both positive and negative scattering bars. It then searches the measurement region for the nearest edge on the required layer, which varies according to the layers you input to OPCSBAR.

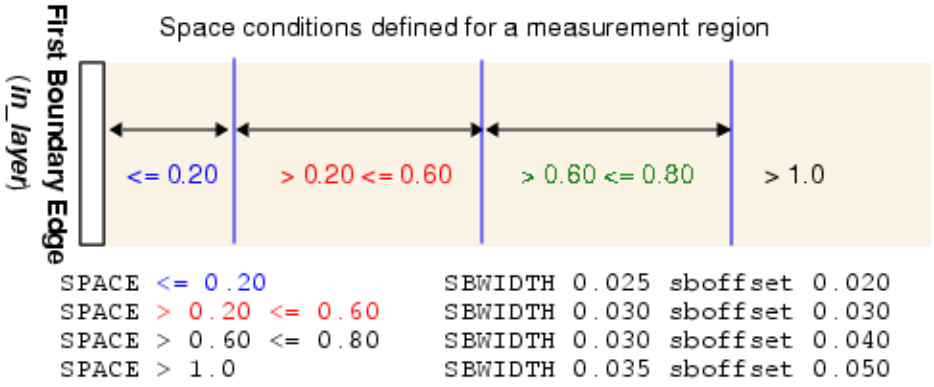
Table 4-3 shows space measurements for edge classification.

Table 4-3. Measuring Spaces for Edge Classification

Layers input to OPCSBAR	Space is measured…
polygon ***in_layer*** alone	Between an edge on the ***in_layer*** and the nearest edge on the ***in_layer***
polygon ***in_layer*** with *ref_layer*	Between an edge of the ***in_layer*** shape and the nearest edge on the *ref_layer*
edge ***in_layer*** with *ref_layer*	Between an edge of the ***in_layer*** and the nearest edge on the *ref_layer*

Calibre OPCsbar classifies edges according to size of the space, that is, the distance between boundary edges. Each rule defines a class of edges based on a range of space sizes. Defining multiple rules with mutually exclusive, contiguous SPACE ranges ensures that you address the requirements for all spaces requiring scattering bars. Calibre OPCsbar ignores any spaces that do not fit within any of the SPACE ranges in the rules. Figure 4-8 demonstrates a graphical example of contiguous SPACE conditions.

Figure 4-8. Defining Multiple Contiguous SPACE conditions



Types of Spaces

Calibre OPCsbar recognizes two types of spaces:

- **Bounded Spaces —** Spaces that extend from one edge to an adjacent edge. For bounded spaces, Calibre OPCsbar creates scattering bars parallel to each valid boundary edge (see "Valid Boundary Edges" for more information), with the placements as specified by SBOFFSET or CENTER.

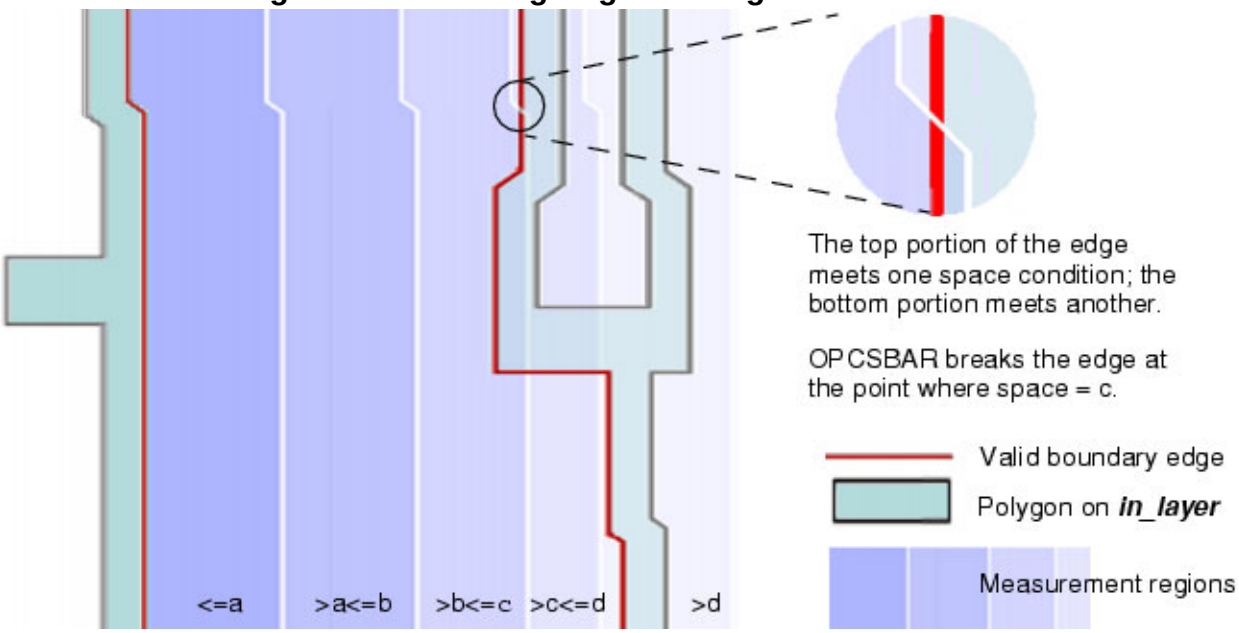
- **Unbounded Spaces —** Spaces that extend from an edge to the outer boundary of the mask. For unbounded spaces, Calibre OPCsbar creates scattering bars parallel to the single valid boundary edge, with the placements as specified by SBOFFSET. Placements specified by CENTER have no meaning for unbounded spaces.

Because a space fits within an unbounded range does not make it an unbounded space. Any space having two boundary edges is bounded. For example, assume you have a rule with SPACE greater than 17 user units. If two edges are 20 user units away from each other, the rule applies, and the space is considered bounded. If a boundary edge faces the outer edge of the mask, the rule also applies, but the space is considered unbounded.

Understanding Edges

When classifying edges based on SPACE rules, Calibre OPCsbar does not necessarily apply the same rule to an entire edge. Instead, it breaks[1]edges into edge fragments so that each resulting edge fragment meets only one space condition. Figure 4-9 shows facing edges on adjacent polygons (shown in red) in which some edges must be broken to respond to spacing variations. The resulting edge fragments on either side of the break are treated differently.

Figure 4-9. Breaking Edges During Classification.



Figures 4-10 and 4-11 show how the edges on the right would be classified. If both polygons are on the ***in_layer***, both sets of edges are classified based on the spaces between.

1. Edge breaking is for internal processing only and does not affect your data.

Figure 4-10. Scattering Bar Treatment by Classification

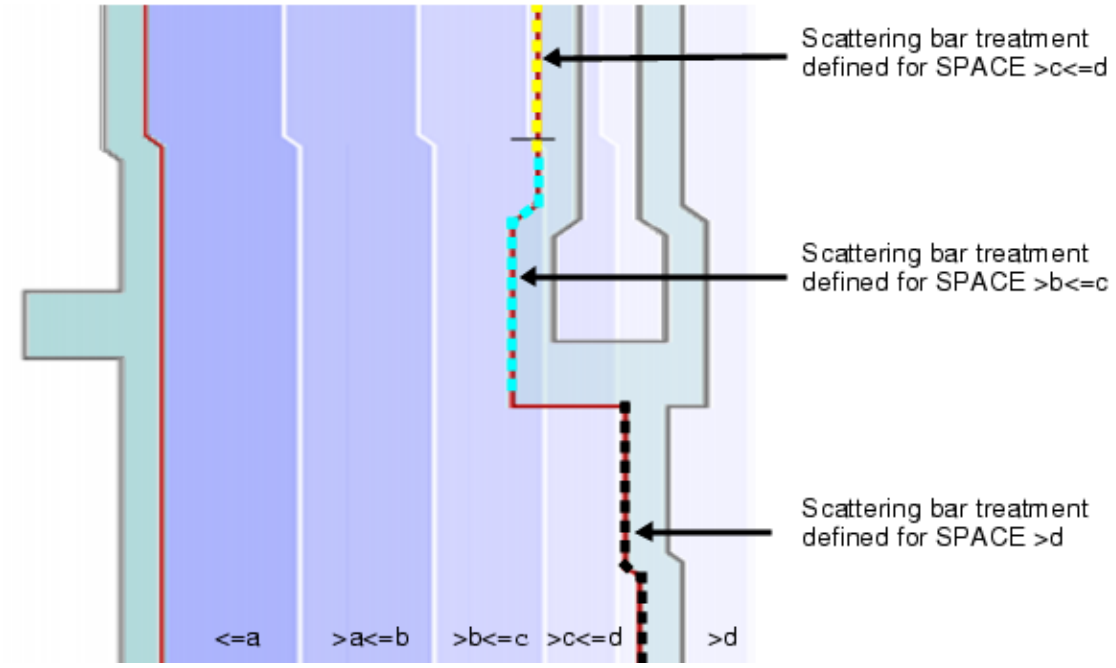


Figure 4-11. Facing Edges Classified by Space

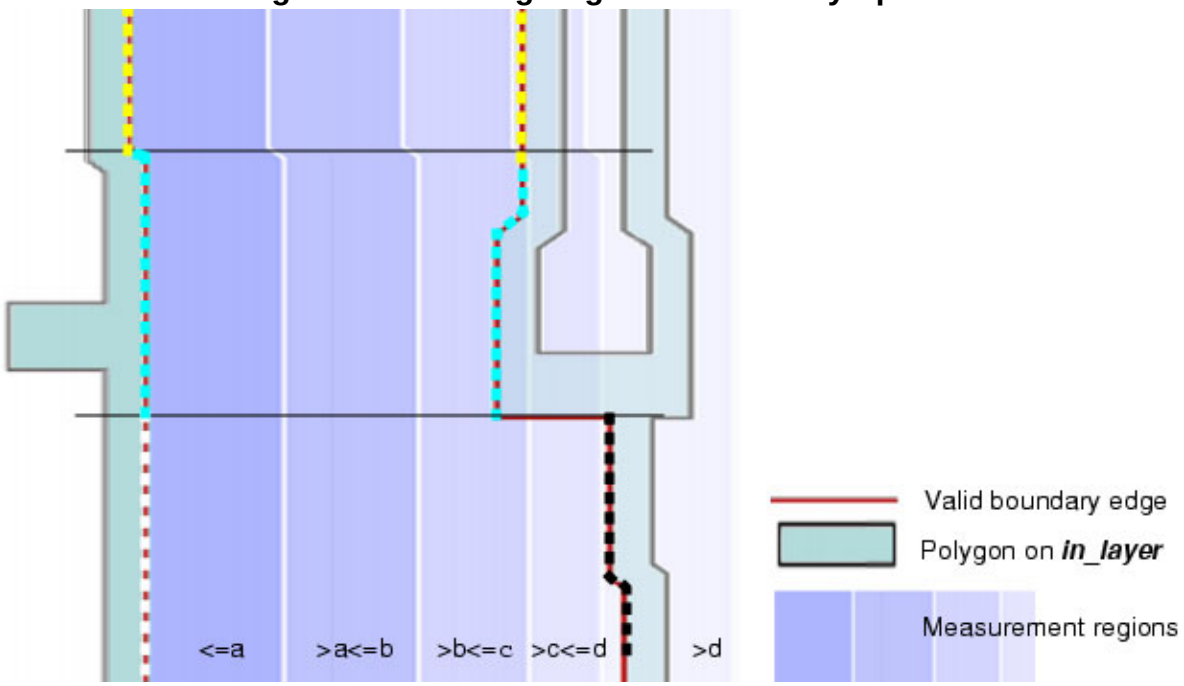
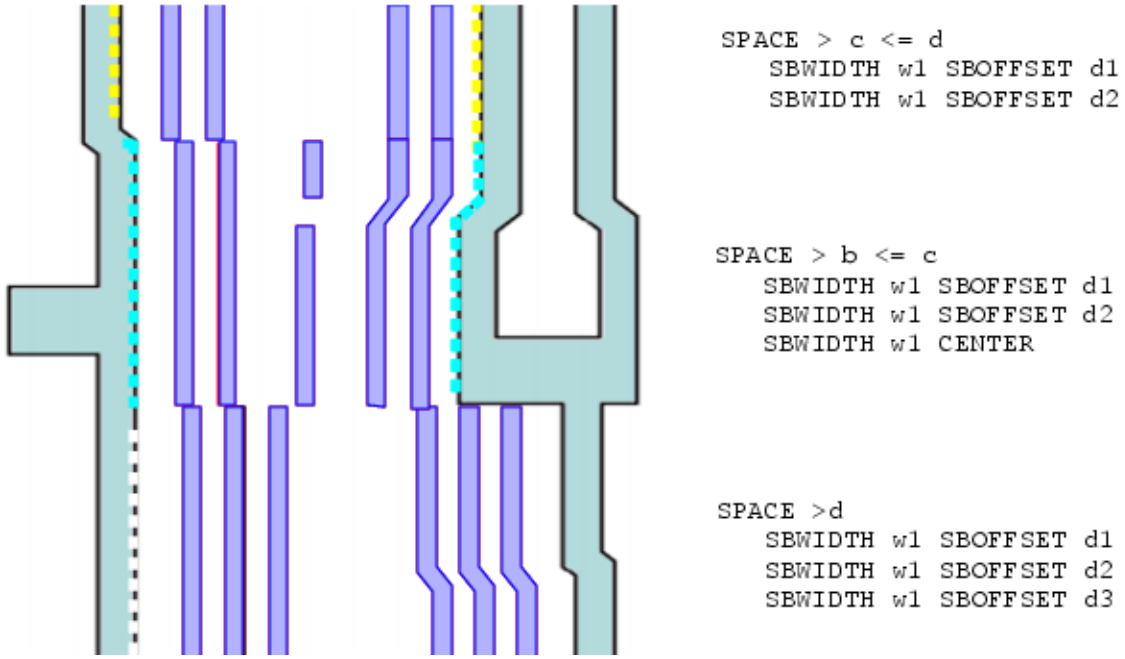


Figure 4-12 shows how the scattering bars generated based on this type of classification can contain jogs, even when the edge for which the bars are generated is straight.

Figure 4-12. Scattering Bars Generated by Classification



Valid Boundary Edges

You define the placement of scattering bars with respect to valid boundary edges. Not all boundary edges are valid. In order for a boundary edge to be valid, it must meet the following conditions:

- It must be on the in_layer.
- It must be drawn at an allowed angle relative to the X- or Y-axis.
- It must meet the MINEDGELENGTH requirement, if one is defined.

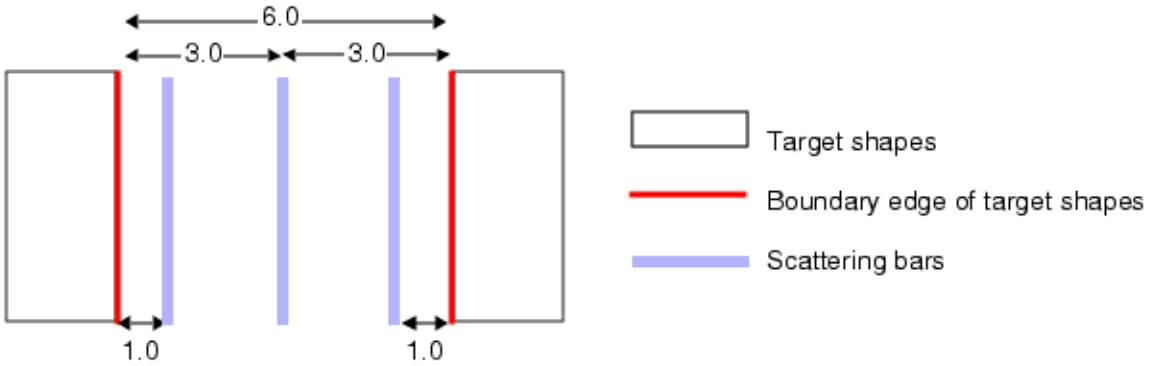
Boundary Edges and Layers

Unless you specify a reference layer, Calibre OPCsbar identifies spaces based on pairs of opposite-facing adjacent edges on the ***in_layer***. When this is the case, both boundary edges automatically meet the ***in_layer*** requirement.

Since SBOFFSET defines scattering bar placement with respect to any valid boundary edges, this placement parameter causes two scattering bars to be created within the space, one for each of the valid boundary edges.

Figure 4-13 shows how a rule defining one CENTER scattering bar and one SBOFFSET scattering bar results in the creation of three scattering bars when both boundary edges are on the ***in_layer***.

Figure 4-13. Scattering Bars Created for Two Boundary Edges on the in_layer

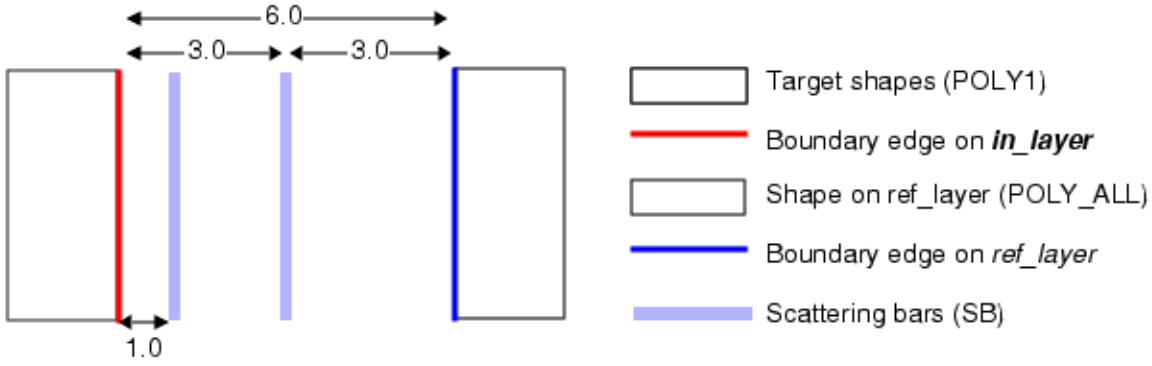



```
SB=OPCSBAR POLY1
    SPACE>5<=6 SBWIDTH 0.1 CENTER         //Creates 1 SB between 2 edges
    SPACE>5<=6 SBWIDTH 0.1 SBOFFSET 1.0   //Creates 2 SB between 2 edges
```

When you specify a reference layer, Calibre OPCsbar identifies spaces having one boundary edge on the ***in_layer*** and the opposite boundary edge on the *ref_layer*. When this is the case, only one of the boundary edges meets the ***in_layer*** requirement.

Again, since SBOFFSET defines scattering bar placement only with respect to valid boundary edges, this placement parameter causes a single scattering bar to be created within the space, with its placement relative to the one valid boundary edge. Figure 4-14 shows how a rule defining one CENTER scattering bar and one SBOFFSET scattering bar results in the creation of two scattering bars when only one boundary edge is on the ***in_layer***.

Figure 4-14. Scattering Bars Generated for a Single Edge




```
SB=OPCSBAR POLY1 POLY_ALL
  SPACE>5.0<=6.0 SBWIDTH 0.1 CENTER         //Creates 1 SB between 2 edges
  SPACE>5.0<=6.0 SBWIDTH 0.1 SBOFFSET 1.0   //Creates 1 SB between 2 edges
```

Boundary Edges and Angles

The Calibre OPCsbar operation recognizes three types of edges:

- **Orthogonal —** Edges that are orthogonal to either the X or the Y database axis.
- **45 —** Edges that are at a 45-degree angle to the X- and Y-axis.
- **Other —** Edges that are at a non-45-degree angle relative to the X- or Y-axis.

By default, the Calibre OPCsbar operation only creates scattering bars parallel to orthogonal edges. Specifying ANGLED overrides this default behavior and forces the Calibre OPCsbar operation to generate scattering bars parallel to 45-degree edges. The Calibre OPCsbar operation never creates scattering bars for edges drawn at other angles.

How to use WIDTH with SPACE Together

When you use the optional WIDTH condition with the SPACE condition, an edge must satisfy both conditions to receive scattering bar treatment.

It can be helpful to think of SPACE rules as tables:

- Every rule must differ from all other rules in at least one condition.
- The rows partition external SPACE.
- The columns partition internal space (WIDTH).
- Each cell in the table represents MRC-compliant scattering bar treatment for each edge that meets the constraints for that row and column.
- For every WIDTH there must be an associated SPACE, but it is possible to have a SPACE that does not have an associated WIDTH.

Figure 4-15 provides an illustration of WIDTH and SPACE usage for positive scattering bars.

Figure 4-15. WIDTH Example for Positive Scattering Bars

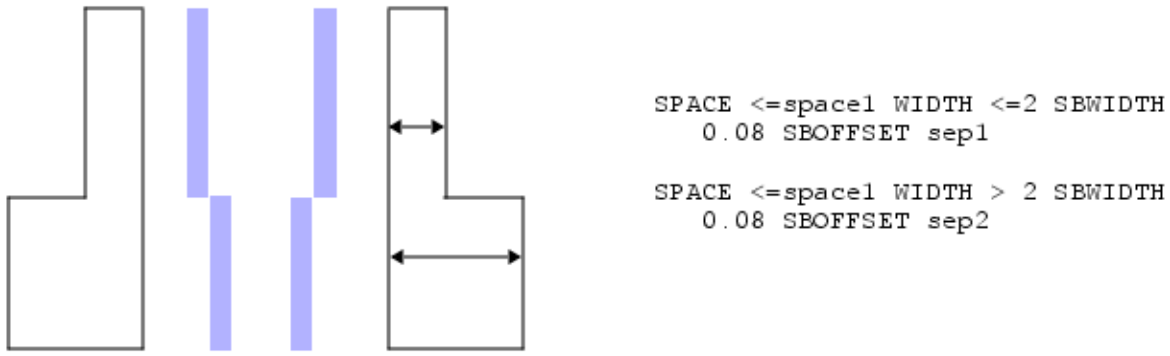
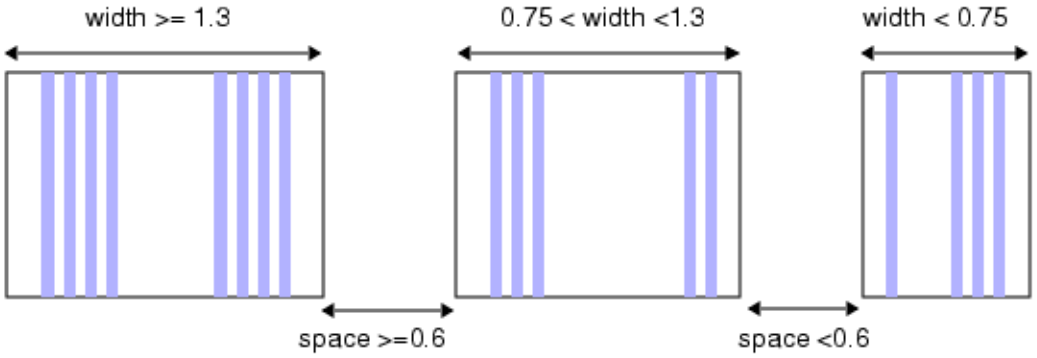


Figure 4-16 provides an illustration of WIDTH and SPACE usage for negative scattering bars.

Figure 4-16. WIDTH Example for Negative Scattering Bars

```
SB = OPCSBAR sample1
     negative
     space < 0.6 width < 0.75    SBOFFSET 0.1 SBWIDTH 0.05
     space < 0.6 width >= 0.75   SBOFFSET 0.1 SBWIDTH 0.05
                                 SBOFFSET 0.2 SBWIDTH 0.05
     space >= 0.6 width < 1.3    SBOFFSET 0.1 SBWIDTH 0.05
                                 SBOFFSET 0.2 SBWIDTH 0.05
                                 SBOFFSET 0.3 SBWIDTH 0.05
     space >= 0.6 width >= 1.3   SBOFFSET 0.1 SBWIDTH 0.05
                                 SBOFFSET 0.2 SBWIDTH 0.05
                                 SBOFFSET 0.3 SBWIDTH 0.05
                                 SBOFFSET 0.4 SBWIDTH 0.05
```



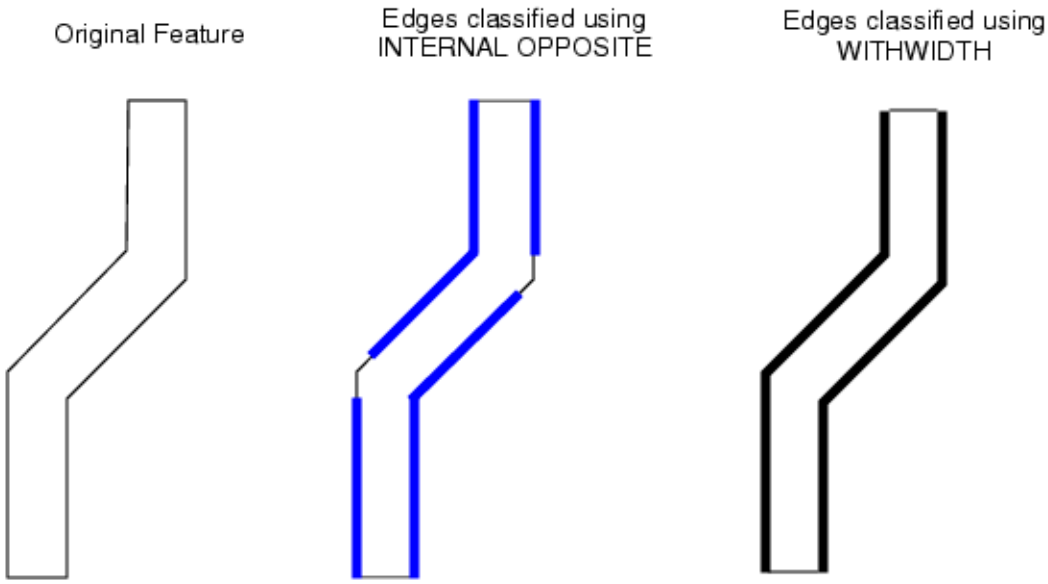
How WIDTH is Measured

You can choose how the Calibre OPCsbar operation measures width when classifying edges for SPACE rules.

- **Default —** Equivalent to the INTERNAL OPPOSITE operation in SVRF.
- **When WITHWIDTH is specified —** Equivalent to the WITH WIDTH operation in SVRF.

The primary difference between the two methods is that INTERNAL OPPOSITE uses a strict definition of width, based on measuring the shortest internal distance. WITHWIDTH uses a more intuitive definition of width that encompasses situations in which fixed width shapes bend and turn corners, as illustrated in Figure 4-17.

Figure 4-17. Comparing WIDTH Classification Options



WITHWIDTH is most helpful when creating scattering bars for bent gates that contain 45-degree edges or edges orthogonal to the database axis.

Refer to With Width in the *Standard Verification Rule Format (SVRF) Manual* for a detailed discussion of the differences between it and INTERNAL projection.

Impact of WIDTH on Scattering Bar Placement

Using the WIDTH condition impacts the CENTER keyword as well.

- For positive, or outside, scattering bars, when you use the WIDTH condition with the CENTER keyword, the application only creates a centered scattering bar if both edges meet the criteria for the same rule, that is, the same WIDTH category and the same SPACE category.

SPACE Keyword Methodology for Negative Scattering Bars

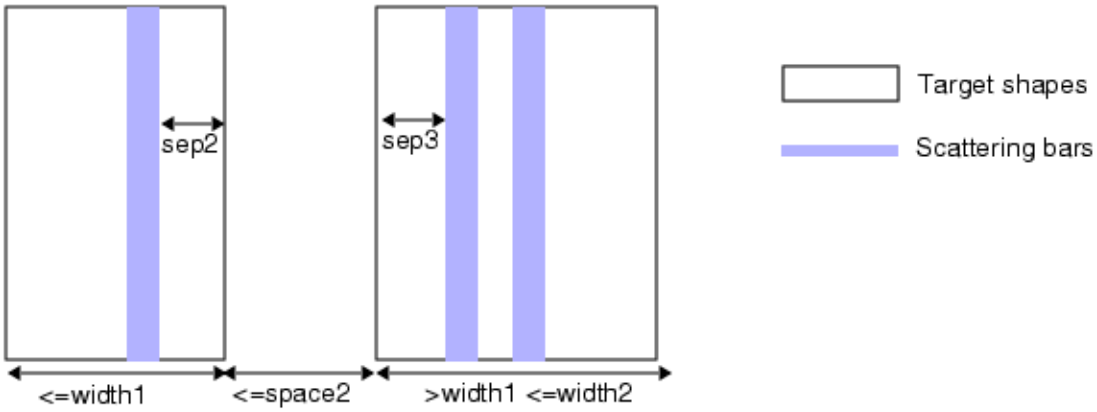
1. Use SVRF to derive an ***in_layer*** containing those shapes that are candidates for negative scattering bars, and input them to the OPCSBAR command. You then input both the in_layer containing the features requiring negative scattering bars, and the LINEENDOFFSET containing all features on the mask layer.
2. Add the keyword NEGATIVE to the OPCSBAR command. All scattering bars generated are negative scattering bars.
3. Write one rule for every cell in the rules table. The rules must define the SPACE condition to be met and the corresponding correction.

When creating negative scattering bars using SPACE rules:

- SBOFFSET edge measurements are performed differently for negative and positive scattering bars:
  - For positive scattering bars, the offset is measured from an edge outward.
  - For negative scattering bars, the offset is measured from an edge inward.
- MINSBOFFSET edges are also measured differently for different scattering bars:
  - For positive scattering bars, the offset is measured from an edge outward.
  - For negative scattering bars, the offset is measured from an edge inward.

Figure 4-18 shows the relationship between the SPACE keyword and negative scattering bars.

Figure 4-18. SPACE Keyword and Negative Scattering Bars




```
SPACE <=space2 WIDTH <=width1 SBWIDTH 0.08 SBOFFSET sep2
SPACE <=space2 WIDTH >width1 <= width2 SBWIDTH 0.08
   SBOFFSET sep3 SBWIDTH 0.08
```

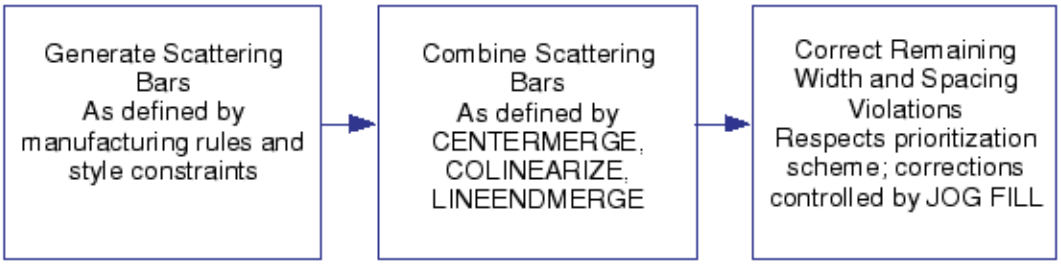
SPACELAYER Keyword

The SPACELAYER keyword specifies layers requiring scattering bars.

The SPACELAYER keyword allows greater flexibility for customization of conditions, and permits input of pre-classified edges to OPCSBAR.

Figure 4-19 depicts this flow.

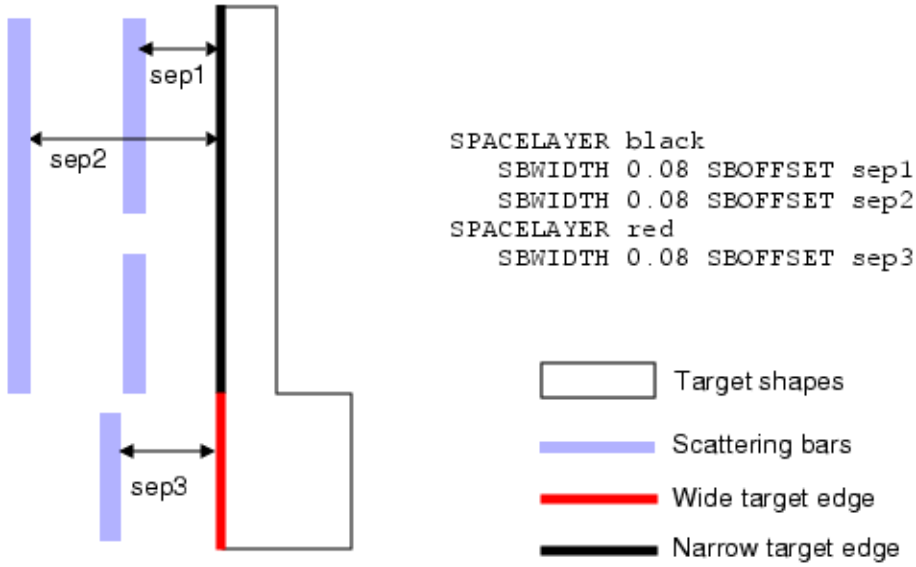
Figure 4-19. Generation of Scattering Bars from SPACELAYER



Isolate the edges to receive scattering bars by using SVRF operations, then pass them to Calibre OPCsbar in the form of edge layers. Since different edges require different scattering bar treatment, you typically generate several different edge layers, each containing a set of related edges.

All edges on a single SPACELAYER receive the same treatment. You define this treatment in terms of how wide the scattering bar(s) should be and how far to offset them from the edge as shown in Figure 4-20.

Figure 4-20. Scattering Bars Generated using SPACELAYER



Note that the presence of an edge in the SPACELAYER does not guarantee that Calibre OPCsbar generates a scattering bar for it. For an edge to receive scattering bar treatment, it must meet the following conditions:

- The edge cannot be skew[2].
- The edge must meet the MINEDGELENGTH requirement, if one is defined.

2. At an angle other than a multiple of 45 degrees, relative to the X and Y axis.

- If its angle relative to the design axes is 45 degrees (or 135, 225, or 315 degrees) ANGLED must be specified.

When you use SPACELAYER rules, you must still supply an ***in_layer***. This layer contains the design data that you are enhancing with scattering bars and is a superset of all the SPACELAYERs. The Calibre OPCsbar operation ignores the geometry on the ***in_layer*** until it is time to check for spacing and offset violations. At that time, it checks MINSBOFFSET and LINEENDOFFSET rules by measuring the distances between each scattering bar and geometry on the ***in_layer***.

SPACELAYER Keyword Methodology for Negative Scattering Bars

1. Use SVRF operations outside the call to OPCSBAR to isolate those edges requiring scattering bar treatment. Generate one edge layer for every set of edges to receive the same treatment.
2. Add the keyword NEGATIVE to the OPCSBAR command. Note that all scattering bars generated are negative scattering bars.
3. Write one rule for every edge layer. Each rule defines the scattering bars to generate for the edges on the specified layer.

When generating negative scattering bars with the SPACELAYER keyword, consider that:

- You cannot generate scattering bars centered within a feature using SPACELAYER rules.
- You generate partial-cut negative scattering bars using MINSBOFFSET and LINEENDOFFSET as shown in Figure 3-29.
- SBOFFSET for edges are measured differently for different scattering bars:
  - For positive scattering bars, the offset is measured from an edge outward.
  - For negative scattering bars, the offset is measured from an edge inward.
- MINSBOFFSET edges are also measured differently for different scattering bars:
  - For positive scattering bars, the offset is measured from an edge outward.
  - For negative scattering bars, the offset is measured from an edge inward.

For illustrations of these differences, refer to “Scattering Bar Type and Arguments”.

More About SPACELAYER

Other considerations:

- The layer you pass to the OPCSBAR command with the SPACELAYER method must be an SVRF edge layer; it cannot be a polygon layer.

- When using SPACELAYER, you cannot define the scattering bar position to be CENTER.
- SPACELAYER and SPACE are mutually exclusive; you cannot use both within the same OPCSBAR run.
- When you pass in an edge layer using the SPACELAYER method, you must specify only one polygon input layer. The application uses the polygon input layer during the cleanup process to identify spacing violations.

Scattering Bar Cleanup

After Calibre OPCsbar generates scattering bars, it cleans up the design based on instructions provided with various keywords.

Cleanup is an iterative process. Cleaning up a group of scattering bars often results in another group requiring cleanup. Calibre OPCsbar performs several cleanup iterations, where scattering bars are modified to meet all constraints, ensuring that the final output is DRC and MRC compliant.

For example, Figure 4-21 demonstrates a cleanup cycle to resolve the spacing violation by shortening the lower scattering bar. If the remainder is less than MINSBLENGTH, Calibre OPCsbar removes the scattering bar completely.

Figure 4-21. Cleaning up DRC and MRC Violations

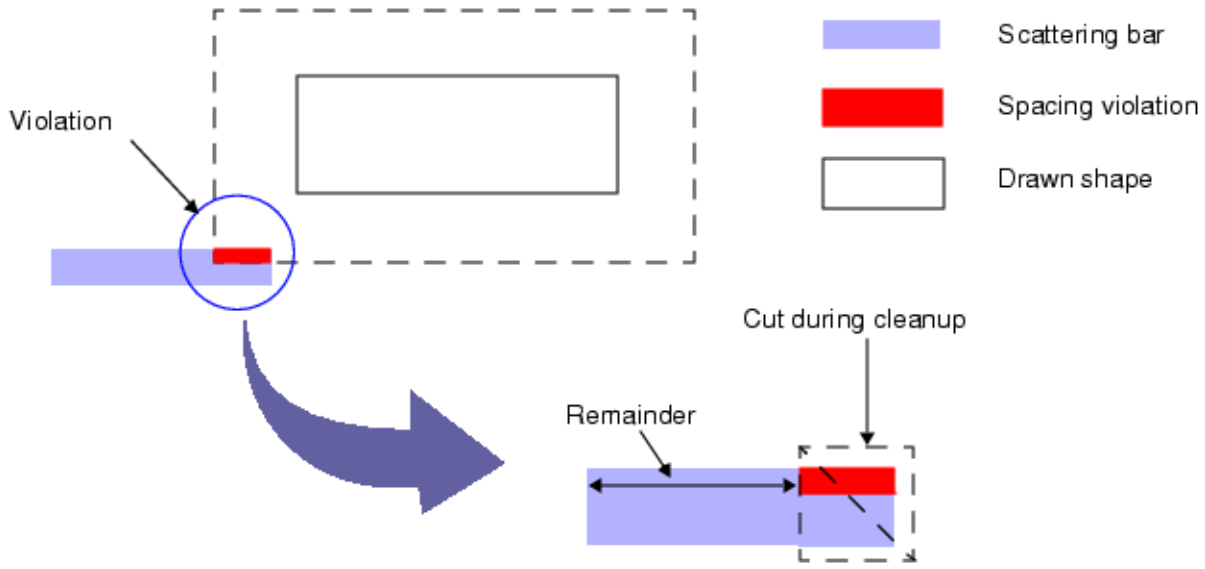


Table 4-4 lists all keywords you use for scattering bar cleanup work.

Table 4-4. Cleanup Keywords

Cleanup Task	Keyword	Applies to:

Table 4-4. Cleanup Keywords (cont.)

Treatment of nearby scattering bars	LINEENDMERGE	adjacent, parallel, and offset in the line-end[1] direction
	CENTERMERGE	adjacent, parallel, and offset in the space direction
Treatment of intersecting scattering bars	COLINEARIZE	parallel overlapping sbars
	INTERSECTION	orthogonal and angled sbars that intersect
	OPENT	T intersections when no intersections are allowed
	JOG FILL	parallel overlapping sbars
	MINJOGWIDTH	parallel overlapping sbars
Extension	EXTENSION	all scattering bars generated
Width, Length, Spacing, and Offset constraints	MINSBOFFSET	all scattering bars; checked at every step of the cleanup process
	MINSBSPACE	
	LINEENDSPACE	
	LINEENDTOLONGSPACE	
	LINEENDOFFSET	
	MINSBWIDTH	
	MAXSBWIDTH	
	MINSBLENGTH	

1. A scattering bar line-end is defined as >= MINSBWIDTH and <=MAXSBWIDTH.

Scattering Bar Cleanup Order

Scattering bar cleanup requires several steps. Calibre OPCsbar performs these cleanup steps in this order:

1. CENTERMERGE
2. COLINEARIZE
3. LINEENDMERGE
4. Cleanup of Illegal Jogs (for scattering bars that do not touch)
5. Cleanup of Illegal Jogs (for scattering bars that touch)
6. EXTENSION
7. INTERSECTION

Cleanup of Width, Offset, and Spacing Violations

At each of the cleanup phases, Calibre OPCsbar checks to see whether or not resulting scattering bars meet the space, width, and offset constraints. When it encounters violations, the operation resolves the problems by modifying or removing the offending scattering bars.

Calibre OPCsbar checks for violations of the space and offset constraints by calculating measurement regions and checking for edges within these regions. To check the space and offset constraints for standard scattering bars, it creates measurement regions extending from the outside edges of each figure. To check the space and offset constraints for negative scattering bars, it creates measurement regions extending from the inside edges of each figure.

Calibre OPCsbar creates the measurement region for line-ends using the Euclidean metric, and for edges using the Opposite metric. For a detailed discussion of spacing checks and measurement regions, refer to the EXTERNAL operation in the *Standard Verification Rule Format (SVRF) Manual*.

Figure 4-22 shows Euclidean measurements for OFFSET and SPACE keywords.

Figure 4-22. Measurement Regions for OFFSET and SPACE Rules

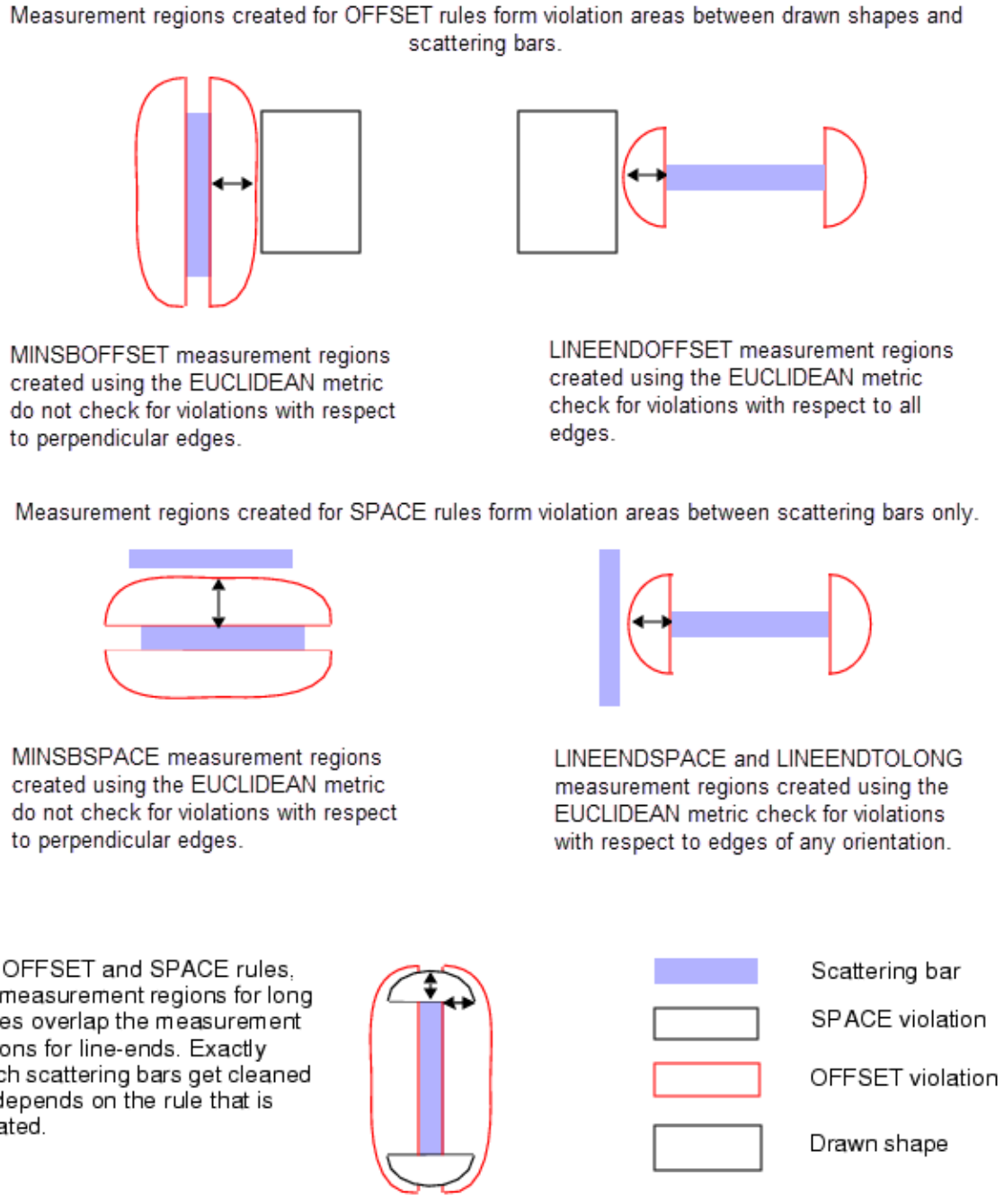


Figure 4-23 depicts spacings and lengths common to scattering bars and target shapes.

Figure 4-23. Parameters Used to Define Edge Spacing and Length

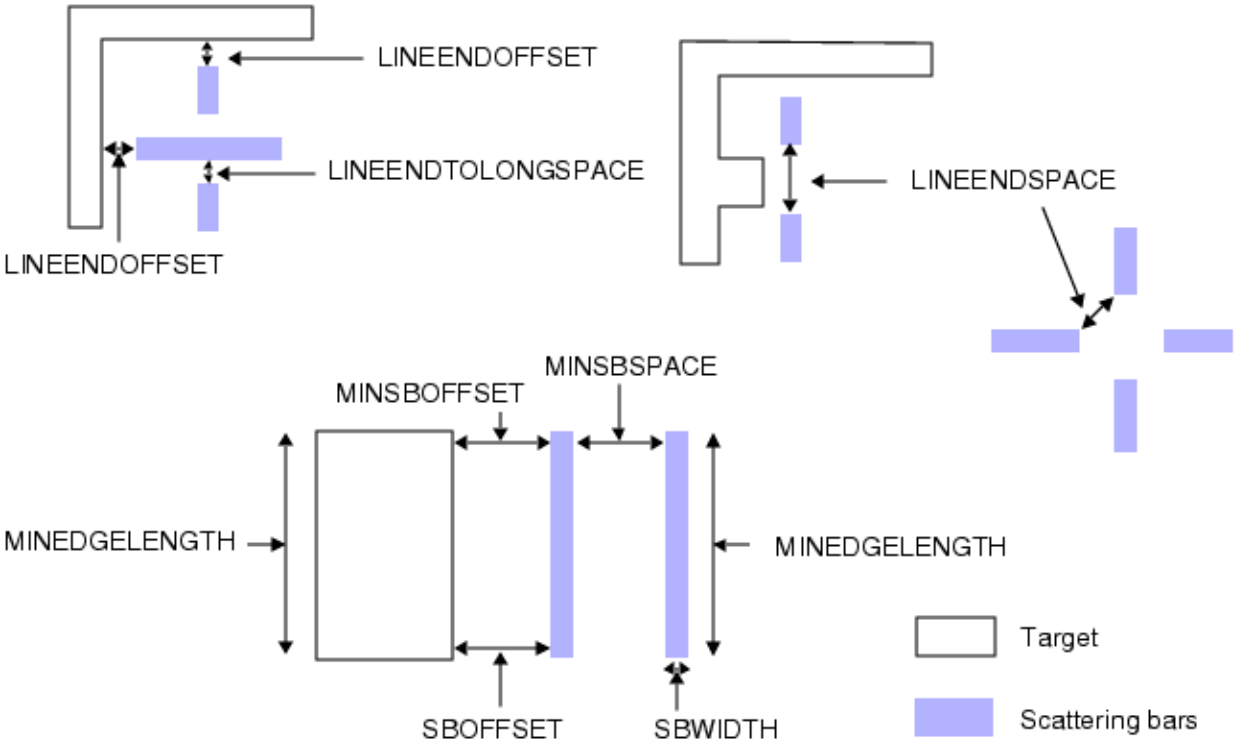


Table 4-5 provides a matrix of constraints between various keywords and scattering bar edge types.

Table 4-5. Defining Spacing Constraints Between Different Types of Edges

SB Type	Any input layer edge	SB edge	SB line-end
Edge	MINSBOFFSET	MINSBSPACE	LINEENDTOLONGSPACE
Line-end	LINEENDOFFSET	LINEENDTOLONGSPACE	LINEENDSPACE
Corner	LINEENDOFFSET	LINEENDSPACE	LINEENDSPACE

A scattering bar line-end is defined as >= MINSBWIDTH and <=MAXSBWIDTH.

Cleanup Techniques to Keep Scattering Bars Rectangular

The scattering bars resulting from the generation step or from resolving spacing or offset violations are often characterized by interesting non-rectangular shapes. Rectangularization is the process of trimming raw or intermediate scattering bars to end up with the rectangular shapes typical of scattering bars.

Whenever the Calibre OPCsbar operation creates a non-rectangular scattering bar, it immediately modifies it to be rectangular.

- Non-rectangular raw scattering bars are rectangularized as part of the creation step.

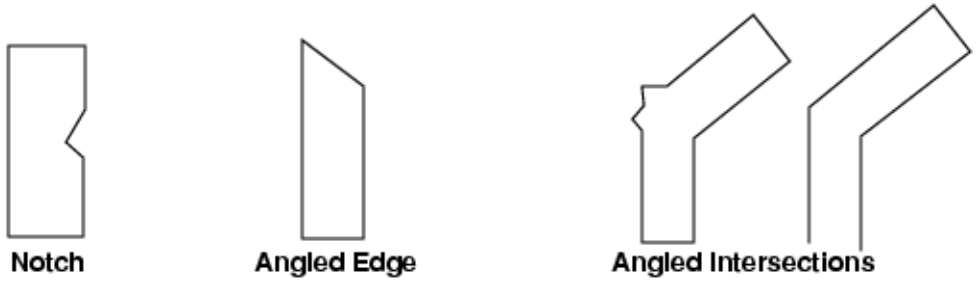
- Non-rectangular scattering bars created as a result of correcting spacing or offset violations are rectangularized before moving on to the next cleanup step.

Uncorrected scattering bars can contain non-rectilinear angles in three configurations:

- **Notches —** A rectangular shape with a small notched-shaped section missing.
- **Angled Edges —** A parallelogram or trapezoid having one or more sides at other than 90- or 270-degrees to the remaining sides.
- **Angled Intersections —** Two rectangular scattering bars intersecting at a non-90- or 270-degree angle.

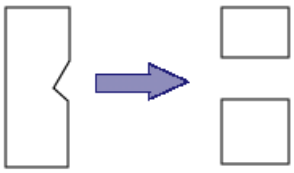
The following figure provides an illustration of these three configurations.

Figure 4-24. Raw Scattering bars with Non-Orthogonal Angles

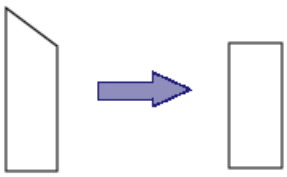


Raw scattering bar errors are corrected as follows:

- Calibre OPCsbar cleans up notches by trimming away the notched portion of the raw scattering bar, leaving two smaller scattering bars.

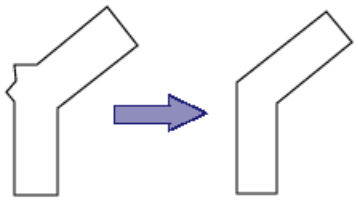


- Calibre OPCsbar cleans up angled ends by trimming the angled end, leaving a rectangular end.

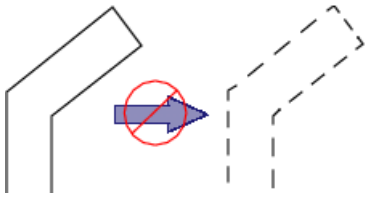


- Calibre OPCsbar cleans up angled intersections only when portions of the intersecting scattering bars extend beyond the intersection, forming small wing-shaped extensions. Cleanup involves trimming away those ends. The remaining scattering bar still contains

non-90 and non-270 degree angles at the intersection point but all corners angles are 90 degrees.



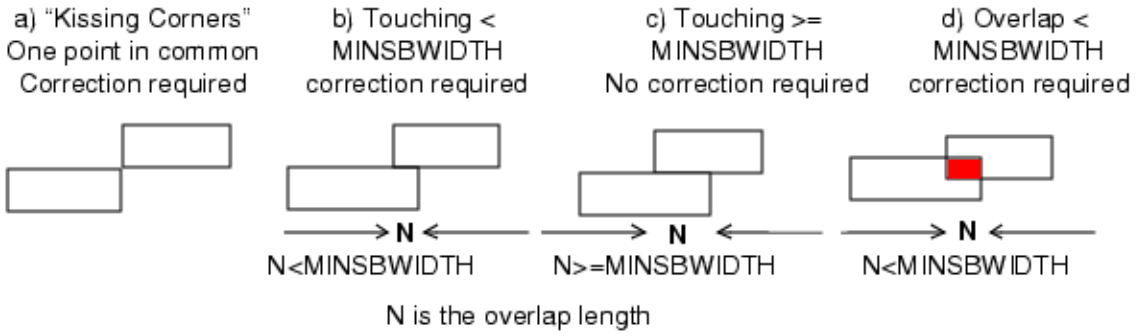
- The second angled intersection in Figure 4-24 does not require correction as it does not include any wing-shaped extensions.



Cleanup of Illegal Jogs

When two scattering bars touch or overlap, they may form illegal jogs that are too small for mask fabrication or inspection. To avoid this, you define the minimum length allowed for the shared region with MINSBWIDTH. Figure 4-25 shows how this parameter relates to jogs.

Figure 4-25. MINSBWIDTH and Jogs



Calibre resolution methodology for touching or overlapping problems depends on whether JOG FILL is specified, and the priorities of the scattering bars. This is based on shape position relative to original edges, since JOG FILL does not consider PRIORITIZE BY LAYER. See "Prioritization by Order" on page 141 for more information.

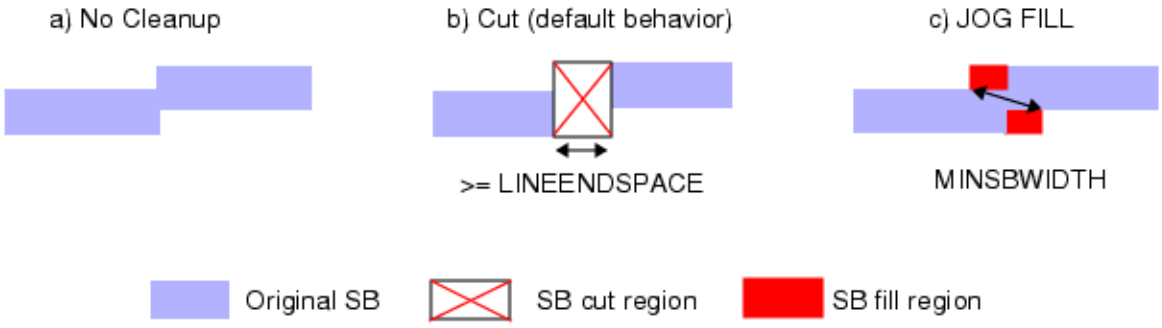
If the scattering bars are the same priority, Calibre OPCsbar checks JOG FILL.

- If JOG FILL is specified, Calibre OPCsbar extends both scattering bars equally to generate a jog equal to the value you specify for the fill width. (If the fill width is not specified, it defaults to the value specified for MINSBWIDTH.)

- If JOG FILL is not specified, or if the scattering bars are different priorities, Calibre OPCsbar trims away the scattering bars to leave an MRC-compliant separation.

Figure 4-26 demonstrates jog interaction styles.

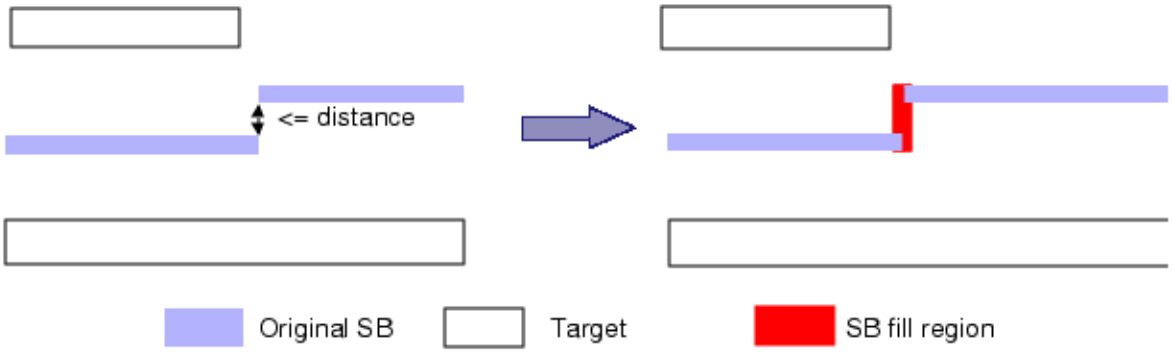
Figure 4-26. Jog Interaction Styles



Used alone, JOG FILL can generate scattering bars with jogs that are legal by filling jog areas formed by touching or overlapping scattering bars. If you specify a value for line-end distance, JOG FILL fills the gap between adjacent scattering bars that are not touching, if they are within the specified distance from each other. This distance must be less than or equal to MINSBSPACE.

Figure 4-27 demonstrates this behavior.

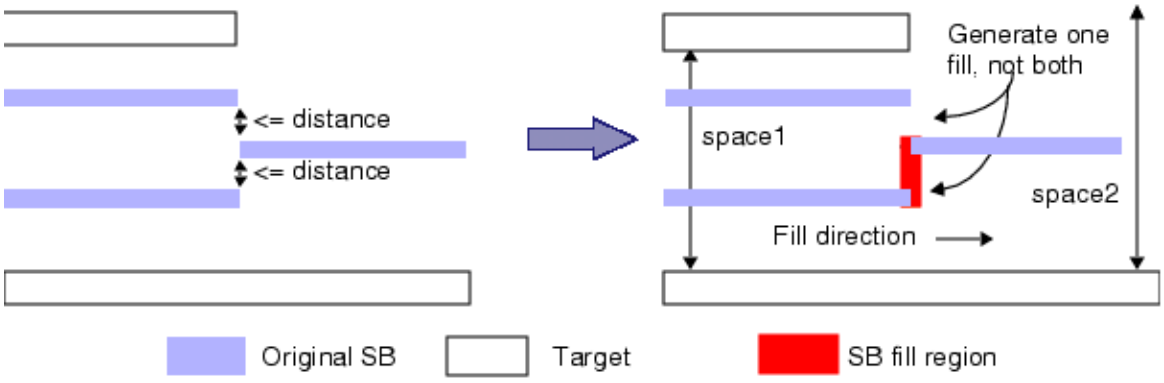
Figure 4-27. Jog Fill Between Non-Touching Scattering Bars



The application fills in both directions. When scattering bars are configured such that there are multiple ways of adding jog fill, the application chooses the configuration arbitrarily.

In Figure 4-28, OPCSBAR can generate either the upper jog fill or the lower jog fill, but not both.

Figure 4-28. Jog Fill Between Non-Touching Scattering Bars (cont.)



When you specify ANGLED, you must also specify JOG FILL, or cleanup removes angled scattering bars. Even with JOG FILL specified, only angles that are multiples of 45 degrees are considered valid. Table 4-6 lists the angles that remain after cleanup for the various combinations of ANGLED, JOG FILL, and INTERSECTION.

Table 4-6 lists how angles are handled by the cleanup algorithm.

Table 4-6. Internal Angles Remaining After Cleanup

JOG FILL			
ANGLED		YES	NO
	YES	90, 135, 225, 270	90, 270
	NO	L,P or T Intersections: 90, 270 N Intersections: 90	L, P, or T Intersections: 90, 270 N Intersections: 90

Cleanup of Intersection Violations

The INTERSECTION keyword allows you to control the types of intersections allowed after cleanup. Allowed styles are

- **N —** No intersections are allowed.
- **L —** Only L-shaped intersections are allowed.
- **T —** Both L- and T-shaped intersections are allowed.
- **P —** L-, T- and plus-shaped intersections are allowed.

Note that you can only specify a single intersection style. If scattering bars intersect in a style that is not allowed, the application cuts away portions of the intersection to generate an intersection that is allowed. It makes the cuts such that the spacing between any remaining shapes is equal to MINSBSPACE[3]or LINEENDSPACE, whichever applies.

3. When not specified, MINSBSPACE defaults to MINSBOFFSET which defaults to the smallest SBOFFSET defined in any of the rules for the Calibre OPCsbar operation.

Basic Intersection Cleanup

Intersection resolution occurs after trimming away any portions of the scattering bars that violate the MINSBOFFSET rule. The final shape is dependent on the relative priority, original length, and orientation of the scattering bars.

- When higher and lower scattering bars interact, if it is not possible to preserve both, Calibre OPCsbar preserves higher scattering bars at the expense of lower scattering bars.
- When the priorities are the same, Calibre OPCsbar cuts away both scattering bars such that the separation between the bars is equal to MINSBSPACE/LINEENDSPACE.
- If, after cutting away both scattering bars, the length of one scattering bar(s) is less than MINSBLENGTH, Calibre OPCsbar removes the short bar(s), then returns the other scattering bar to its original length.
- If, after cutting away the scattering bars, the lengths of all remaining scattering bars are less than MINSBLENGTH, Calibre OPCsbar resolves the conflict automatically by preserving either the vertical or horizontal scattering bar and removing the other scattering bar. To avoid this automatic resolution, use SBLAYER.

Figure 4-29 shows how Calibre OPCsbar resolves various intersection problems to achieve an “N” intersection. Note that in two situations, the bars are not cut away equally. The first situation is demonstrated when resolving a “T” intersection. Calibre OPCsbar trims the stem of the “T” to preserve the crossbar. The second situation is demonstrated when the length of one portion of a “+” intersection is less than MINSBLENGTH.

Figure 4-29. Achieving an “N” Intersection

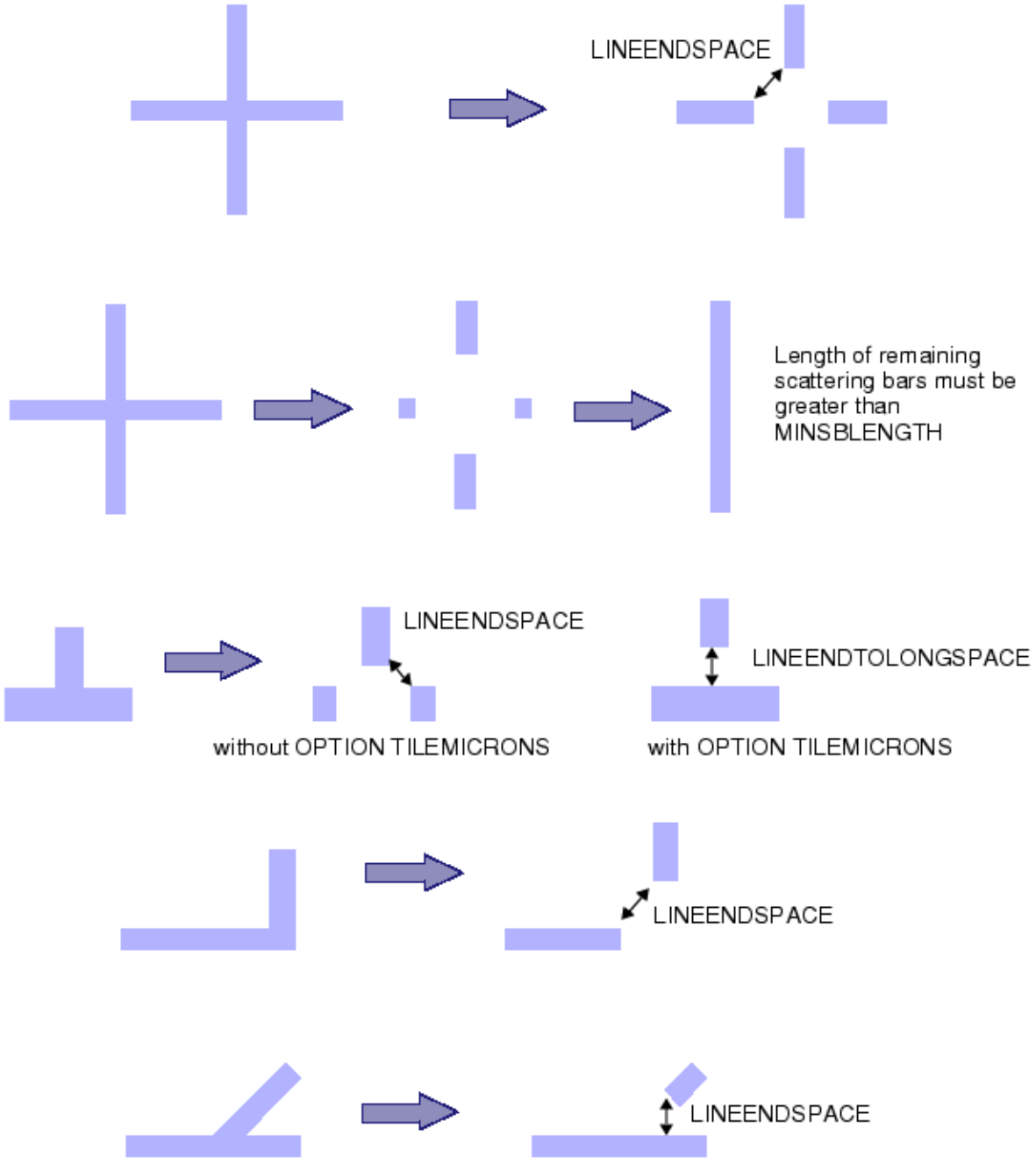
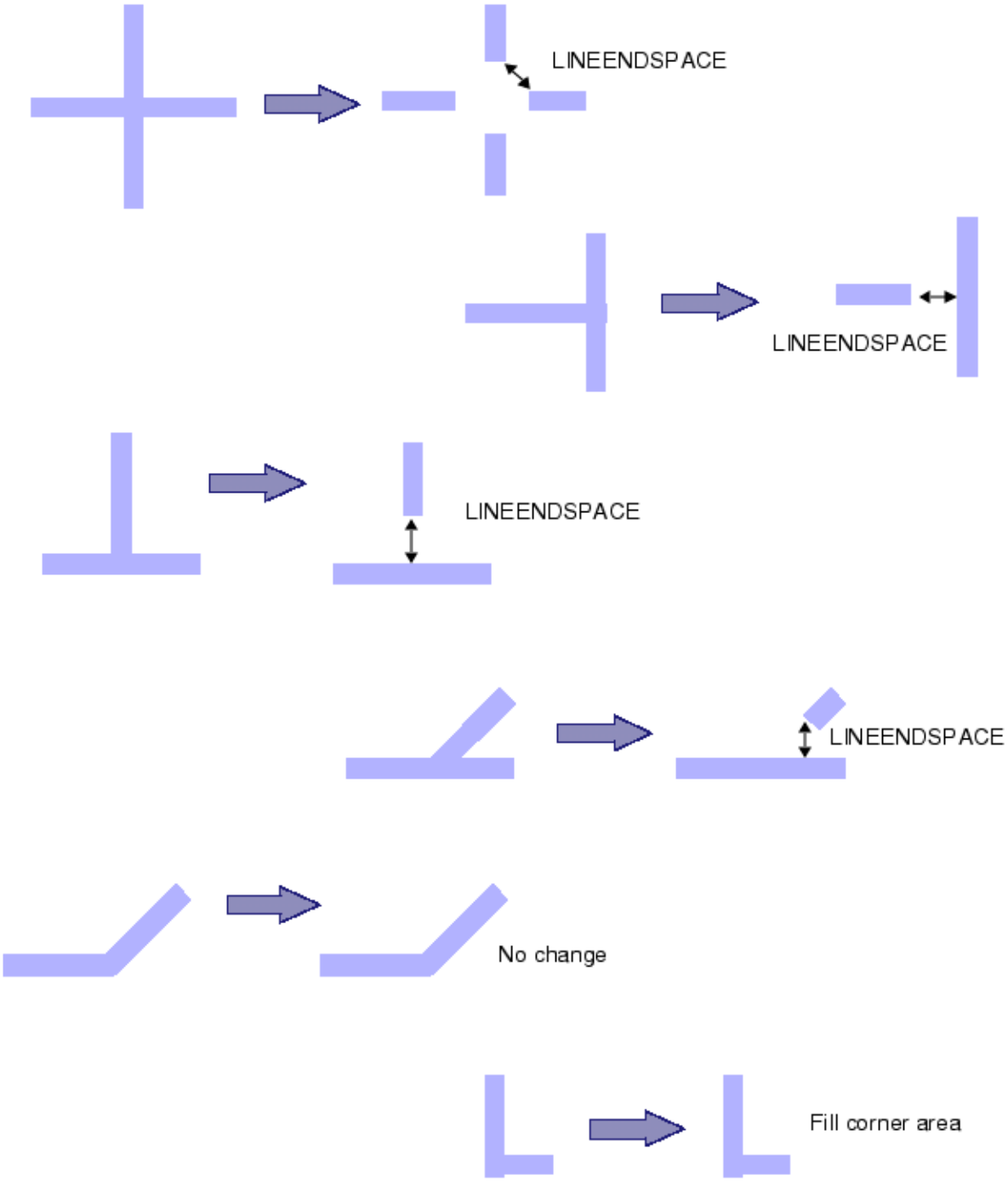


Figure 4-30 shows how Calibre OPCsbar resolves various intersection problems to achieve an “L” intersection. Note that an “L” formed by an orthogonal scattering bar with an angled scattering bar is considered a valid “L.”

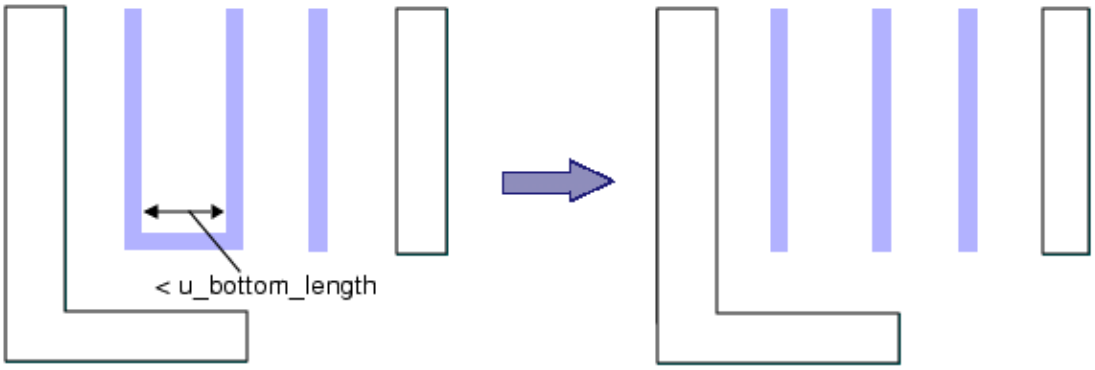
Figure 4-30. Achieving “L” Intersections



Cleanup of U Configurations

When two “L” intersections face each other, they create a “U” shape. When specifying an intersection of type “L”, you can supply an optional value, *u_bottom_length*, which defines a minimum distance between the middle scattering bars in a U configuration. If the minimum distance is not met, the bottom of the U is removed as shown in Figure 4-31.

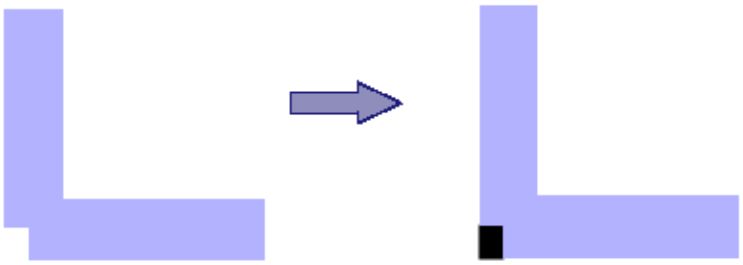
Figure 4-31. Resolving “L” Intersections that Result in “U”



Cleanup of Overlapping Intersections

When L intersections are allowed and two perpendicular scattering bars overlap but do not intersect completely, Calibre OPCsbar fills as needed to generate an L shape as depicted in Figure 4-32.

Figure 4-32. Cleaning of Overlapping Scattering Bars



How Priority Affects Cleanup

The cleanup phase of raw scattering bars involves their prioritization, finding their illegal interactions, and finally modifying or removing them to produce MRC-correct results.

Calibre OPCsbar modifies or removes scattering bars based on their established priority (also referred to as relative ranking). Prioritization can be established by the scattering bar proximity to target edge (order), a layer name or the SBLAYER name.

Conflict Resolution With Same Priority

When two scattering bars with the same priority ranking interact, Calibre OPCsbar resolves conflicts as follows:

- Calibre OPCsbar examines the angles of the edges. It assigns scattering bars generated for edges that are orthogonal to the X or the Y axis a higher priority than those generated for angled lines.
- If both edges are orthogonal or both are angled, Calibre OPCsbar trims both away equally, then examines the lengths:
  - If both are greater than MINSBLENGTH, both remain intact and shortened symmetrically.
  - If one scattering bar is now less than MINSBLENGTH, the application removes the entire scattering bar and returns the other to its full (original) length.
  - If both scattering bars are now less than MINSBLENGTH, the application removes the entire vertical scattering bar and returns the horizontal one to its full (original) length.

Conflict Resolution with Differing Priority

By default, whenever there is a conflict between scattering bars with different priorities, the cleanup operation trims or removes the lower priority scattering bars to preserve the higher priority scattering bars. The higher priority scattering bars are never modified to correct interactions with lower priority scattering bars.

Scattering Bar Cleanup Priority

Calibre OPCsbar always prioritizes scattering bars as a first step in the cleanup process.

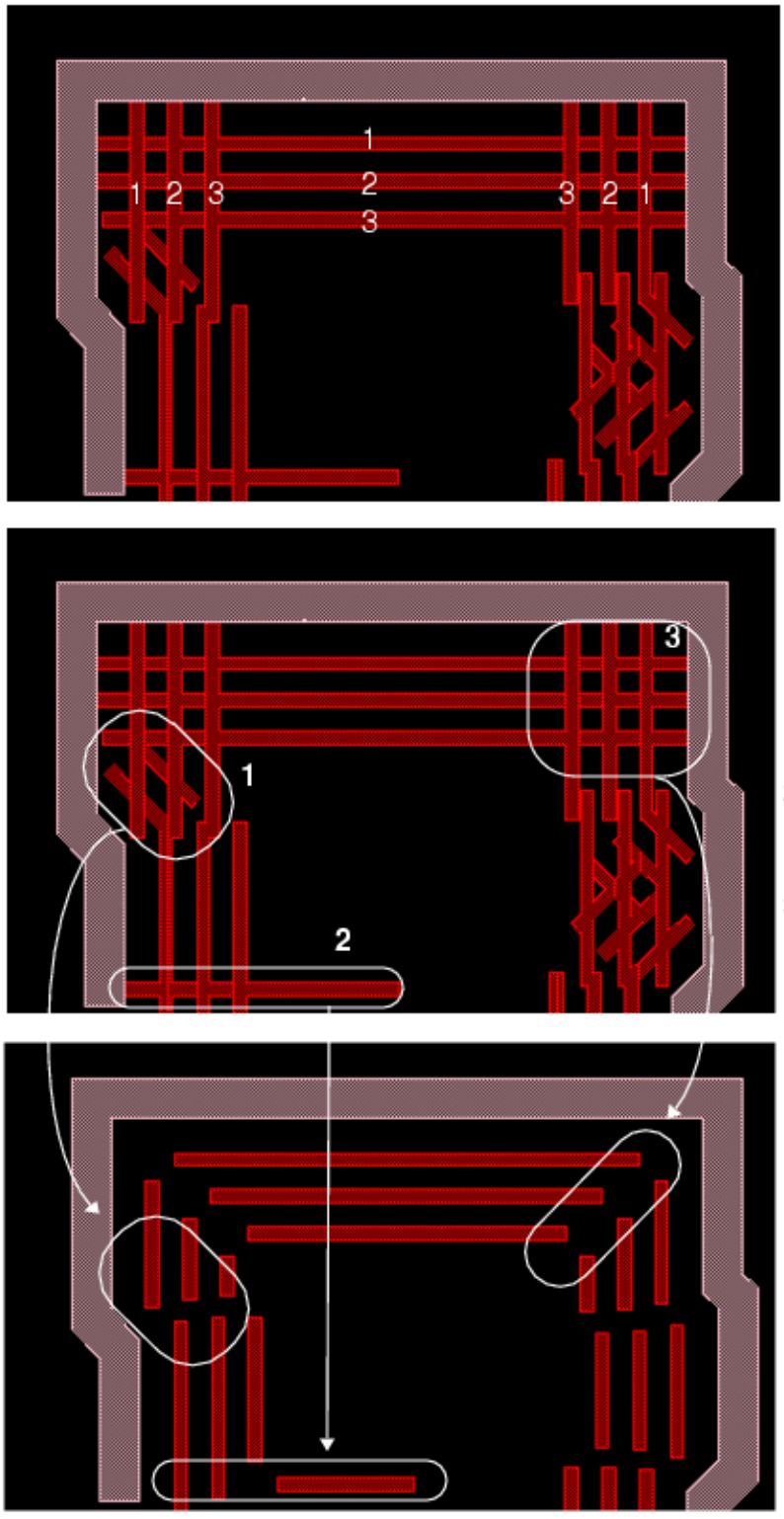
You can influence this prioritization using the SBLAYER and PRIORITIZE BY LAYER keywords. There are three prioritization options available.

Prioritization by Order

Calibre OPCsbar performs prioritization based on scattering bar proximity to target edge. Scattering bars closest to the target edge take the highest priority.

Figure 4-33 demonstrates prioritization and prioritization-based cleanup.

Figure 4-33. Cleanup Based on Prioritization



Scattering bars are classified in order of impact to the target, 1 being highest.

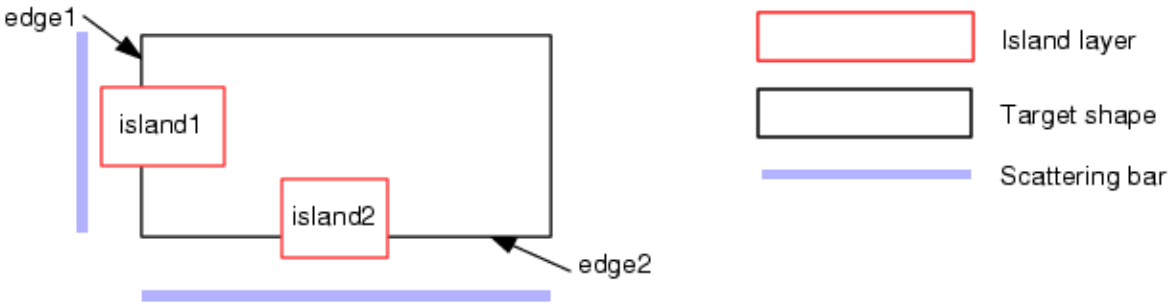
1. Angled scattering bars are trimmed to preserve orthogonal scattering bars of like priority.
2. Lower priority scattering bars are trimmed to preserve higher order scattering bars.
3. Orthogonal scattering bars of the same order are trimmed equally.

Prioritization by Layer

The PRIORITIZE BY LAYER keyword lets you define a finer granularity of prioritization based on additional input layers, called island layers. An island layer is a layer used to mark certain portions of the design as requiring special treatment. The order in which you supply the island layers defines the priority ranking. That is, the first island layer defines the highest island-layer priority, the second defines the next highest island-layer priority, and so on.

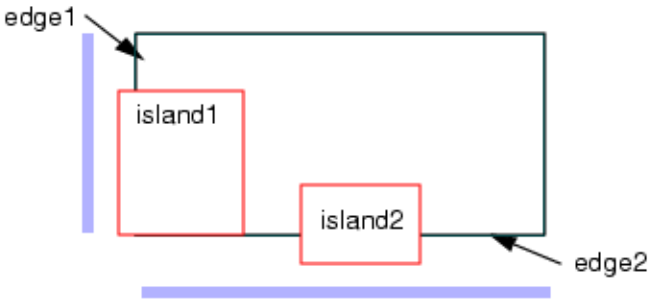
Scattering bars inherit their island-layer priority based on interactions between boundary edges and polygons on the island layers. If any portion of an edge lies inside a polygon on an island layer, the entire edge, plus any scattering bars generated for that edge inherit the priority for that island layer. If an edge intersects polygons on two or more island layers, the scattering bars inherit the higher priority. In Figure 4-34, edge1 (and its associated scattering bars) inherits its priority from island1, which intersects it.

Figure 4-34. Island Layer Prioritization



Note that coincidence or sharing a single point with an island layer does not count as being inside that island layer; as shown in Figure 4-35 edge2 still inherits its priority only from island2.

Figure 4-35. Island Layer Prioritization (cont.)

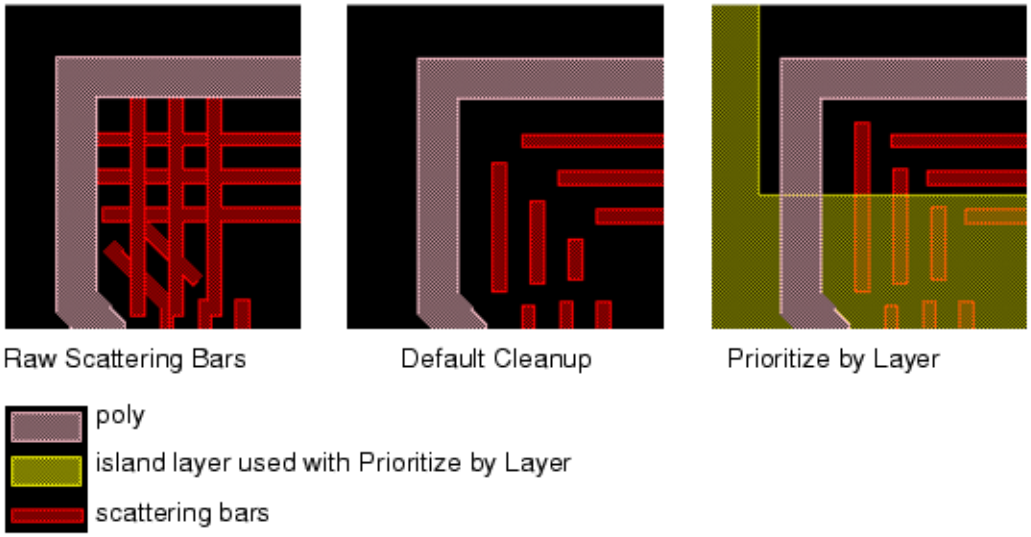


When using island layer prioritization, Calibre OPCsbar calculates the priority rankings based on a combination of order plus island-layer priority. Within the first order scattering bars, which are closest to the edge, Calibre OPCsbar assigns a relative priority based on the island layer rankings. Then Calibre OPCsbar assigns a relative priority within the second order scattering

bars based on the island layer ranking, and so on. For example, if the rules define a maximum of three scattering bars per edge, and three island layers (island1, island2, and island3) for prioritization, the number of different priority levels equals:

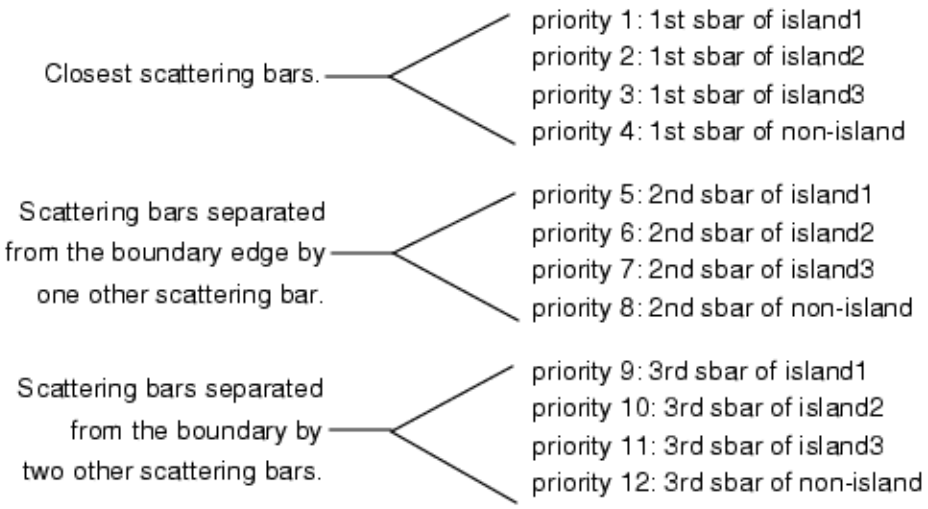
Figure 4-36 demonstrates prioritization using a layer.

Figure 4-36. Cleanup Based on Prioritization by Layer



The highest priority scattering bars have the priority number 1, while the lowest priority scattering bars have the highest priority number as shown in Figure 4-37.

Figure 4-37. Scattering Bar Prioritization Graph

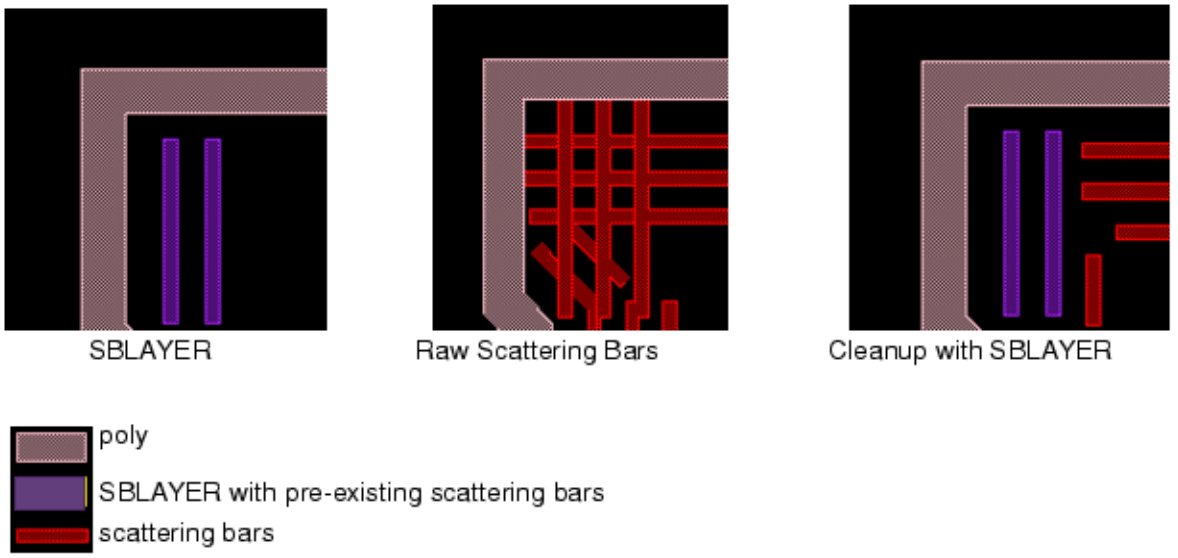


Prioritization Using SBLAYER

Cleanup that is either algorithmically-based, based on use of SBLAYER or based on PRIORITIZE BY LAYER use, calculates priority of scattering bars based on their proximity to the edge for which they were created. You can assign a priority that is not based on proximity by inputting existing scattering bars to Calibre OPCsbar using the SBLAYER keyword. See "Cleanup Based on Prioritization" and Prioritization by Layer for more information.

Calibre OPCsbar recognizes the shapes on SBLAYER as previously generated scattering bars and assigns them the highest level of priority as shown in Figure 4-38.

Figure 4-38. Cleanup Based on SBLAYER Prioritization



Chapter 5
Calibre OPCsbar Best Practices and Problem Resolution

There are several best practices and troubleshooting tips you can follow to create the most effective Calibre OPCsbar rule file.

Best Practices **147**
Recommendations for Scattering Bar Generation **148**
Problem Resolution **149**

Best Practices

These general recommendations facilitate the development of effective Calibre OPCsbar rule files.

- Calibre OPCsbar returns layer data containing the scattering bars it creates. These scattering bars are on a separate layer. To combine the scattering bars with the design data, you must merge the scattering bar layer with the original layer using Boolean operations. For positive scattering bars use the OR operation. For negative scattering bars use the NOT operation.
- The ref_layer must be a superset of the in_layer; that is, it must contain all shapes on the ***in_layer***. You cannot specify *ref_layer* when using SPACELAYER.
- Use OPTION VERIFICATION_MODE to find deleted scattering bars from the SBLAYER of an executed Calibre OPCsbar operation.
- Use OPTION TILEMICRONS to improve processing scalability on designs with very large amounts of flat input data.
- STRICTCENTER and OPPOSITEEXTENDED are mutually exclusive. It is possible OPPOSITEEXTENDED may change the width of target shapes, disabling STRICTCENTER from generating scattering bars.
- Specify MAXSBWIDTH and MINSBWIDTH values. This is important in cases where a single SBWIDTH is available in the OPCSBAR command. If not specified, MAXSBWIDTH defaults to the largest SBWIDTH plus two database units and MINSBWIDTH defaults to the smallest SBWIDTH minus one database unit in any SPACE keyword.
- Always enable LAYOUT CLONE TRANSFORMED PLACEMENTS (detailed in the *Standard Verification Rule Format (SVRF) Manual*) when running OPCSBAR. This

permits the Calibre database constructor to clone cells that have rotated or mirrored placements, so that subsequent hierarchical simulation of rotated structures is correct.

- Siemens EDA recommends setting OPTION CAREFUL_CLEANUP to to 0. This forces OPTION FAST_MRC to be set to off by default. This setting provides for optimal scattering bar cleanup.

Recommendations for Scattering Bar Generation

Following recommended practices optimizes Calibre OPCsbar scattering bar generation.

Table 5-1. Recommendations for Scattering Bars

When creating...	Recommendation
SRAFs for 45 degree or skew edges	Use the ANGLED option to generate scattering bars for 45 degree or skew edges. By default, Calibre OPCsbar only creates scattering bars for orthogonal edges, even if you supply a SPACELAYER containing angled edges.
SRAFs for bent gates	Use the measurement metric WITHWIDTH when creating scattering bars for bent gates that contain 45 degree edges or edges orthogonal to the database axis. The default measurement metric is INTERNAL OPPOSITE.
SRAFs for gates only	Isolate the gate poly, then pass it to OPCSBAR as the in_layer and pass OPCSBAR the poly layer as the ref_layer.
SRAFs for gates and critically dimensioned features	Add scattering bars to the entire poly layer, using SPACE and WIDTH to ensure that the operation only creates scattering bars for critically dimensioned poly. Since scattering bars for gates are more important than for other narrow features, you must give gate scattering bars a higher priority. This can be also done by using PRIORITIZE BY LAYER.
SRAFs for iso features	Start by creating simple rules classifying edges based on distances to nearby features.
Partial-cut negative SRAFs	Starting from the 2009.2 release, to generate partial-cut negative scattering bars, use MINSBOFFSET and LINEENDOFFSET. Prior to 2009.2, negative scattering bar generation with EXTENSION may be used.
SRAFs incrementally	Pass in previously created scattering bars using SBLAYER, and generate the new scattering bars around them.
Cleanup settings	Siemens EDA recommends setting OPTION CAREFUL_CLEANUP to to 0. This forces OPTION FAST_MRC to be set to off by default. This setting provides the optimal scattering bar cleanup setting.

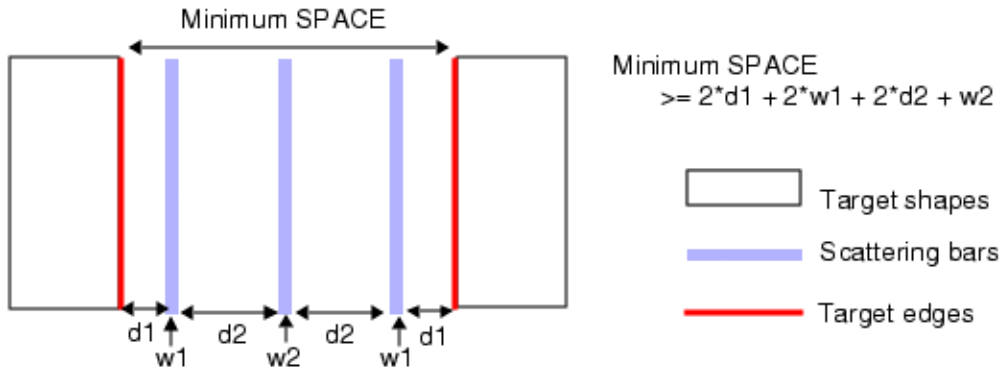
Problem Resolution

Systematic approaches are provided to resolve problems you encounter with your rules.

Quick Checks for Rules

Minimum SPACE values should be greater than or equal to the sum of the scattering bar widths and spaces between them and the target shapes. Figure 5-1 shows a calculation of minimum space.

Figure 5-1. Calculating Minimum SPACE



For REGULAR scattering bars, (defined with the same width, spaced a single regular distance apart) you can define variables and calculate values in your rule deck as shown in Table 5-2 and Table 5-3:

Table 5-2. Variables to Define

Variable	Definition	Comment
sbWidth	Scattering Bar width	use this for all SBWIDTHs
sbDist	Distance from feature to sb	typically larger than MINSBOFFSET
minGap	Minimum space between scattering bars	typically = MINSBSPACE

Table 5-3. Values for Low End of SPACE Constraint

Configuration	Minimum Space Values	Maximum Space Values
CENTER	2*sbDist+sbWidth	2*sbDist+2*sbWidth+minGap
OFFSET	2*sbDist+2*sbWidth+minGap	2*sbDist+3*sbWidth+2*minGap
OFFSET + CENTER	2*sbDist+3*sbWidth+2*minGap	2*sbDist+4*sbWidth+3*minGap
2 OFFSETS	2*sbDist+4*sbWidth+3*minGap	2*sbDist+5*sbWidth+4*minGap
2 OFFSETS + CENTER	2*sbDist+5*sbWidth+4*minGap	2*sbDist+6*sbWidth+5*minGap
3 OFFSETS	2*sbDist+6*sbWidth+5*minGap	2*sbDist+7*sbWidth+6*minGap

Unexpected Placement of Scattering Bars

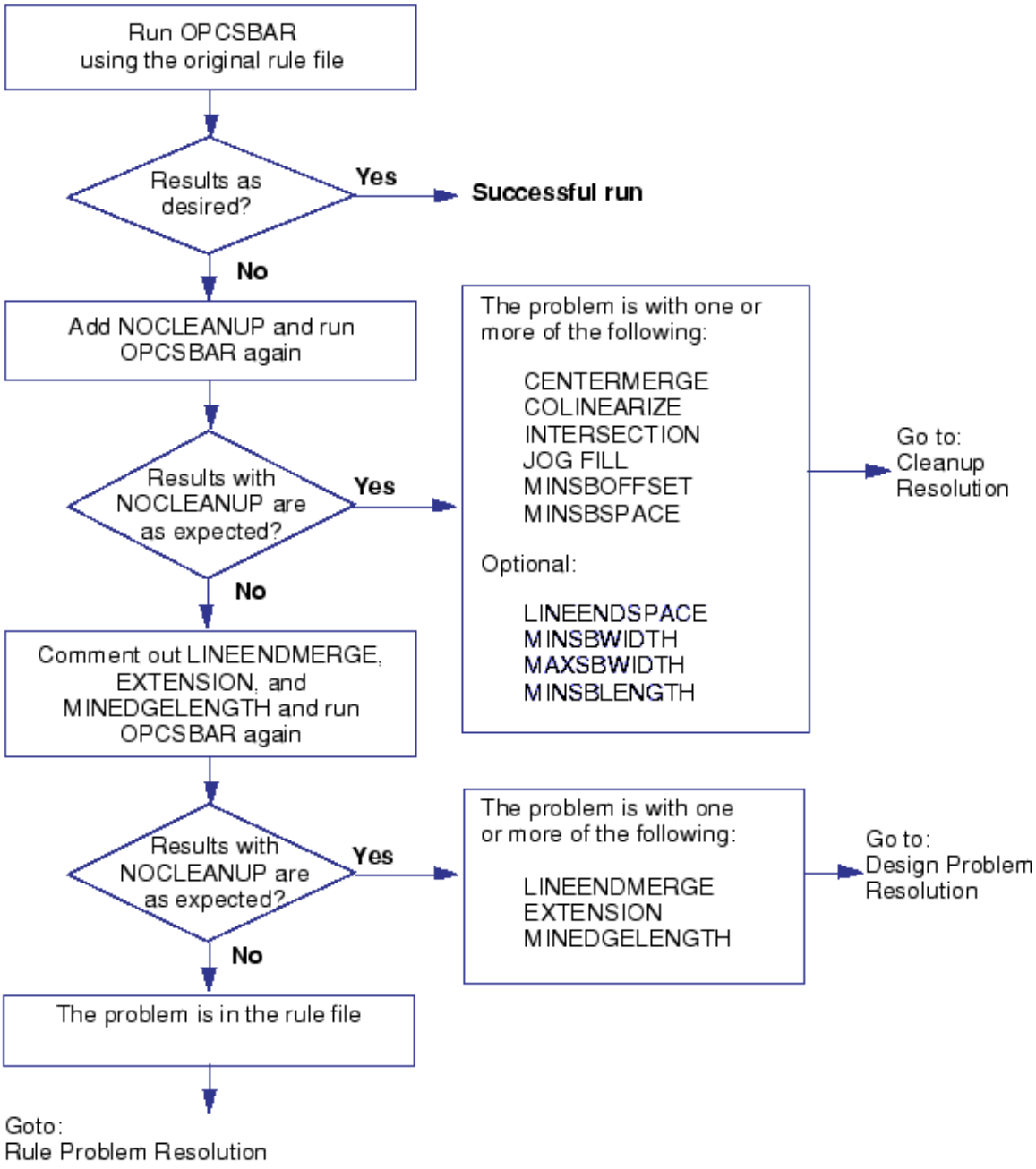
When scattering bars are not where you expected, there are two possible causes:

1. The rules are wrong, leading to unexpected raw scattering bars.
2. The cleanup algorithm modified the raw scattering bars and they no longer match your expectations.

Determine which of these is the cause of your unexpected scattering bar placement.

The flow in Figure 5-2 shows how to determine unexpected scattering bar isolation.

Figure 5-2. Isolation of Incorrectly Placed Scattering Bars



Cleanup Resolution

Once you have identified a problem due to cleanup options, turn on NOCLEANUP, re-run, and compare results to those of the first run. Follow these general steps as an aid:

1. Find one area where the original results were incorrect.
2. Isolate the failure during the cleanup option that caused the problem.

3. Find the problem in Table 5-4 and consider the possible solutions.
4. Test the recommended solution:
    a. Comment out NOCLEANUP in your rule file.
    b. Modify the keyword(s) as suggested.
    c. Re-run OPCSBAR.
5. Check the new results.
6. Continue modifying keyword values until the problem is resolved.

After you have resolved one problem, if the output is still not what you want, repeat the process for another problem, this time comparing the most recent results to the results with NOCLEANUP.

Table 5-4. Resolving Cleanup Problems

Problem	Try this…
Two parallel bars replaced by one	Comment out CENTERMERGE.
Overlapping bars trimmed away	Add JOG FILL or increase MINJOGWIDTH.
Overlapping bars replaced by one	Comment out COLINEARIZE.
Small scattering bars missing	Reduce MINSBLENGTH.
Missing first priority scattering bars	Reduce MINSBOFFSET or LINEENDOFFSET.
Missing second or third priority scattering bars	Reduce MINSBSPACE or LINEENDSPACE.
Scattering bars shortened oddly	Turn off SBLAYER or PRIORITIZE BY LAYER.

Table 5-4 provides a list of cleanup problems.

Design Problem Resolution

Once you have identified the problem as stemming from MINEDGELENGTH, EXTENSION, or LINEENDMERGE, you should experiment with these keywords to see what you must do to get the desired results.

1. Find one area where the original results were not what you wanted.
2. Compare the results from a run using NOCLEANUP with these options (layoutA) to the results from a run using NOCLEANUP without these options (layoutB).
3. Once you identify a problem, look it up in Table 5-5. This table suggests possible solutions.

4. Test the recommended solution:
    a. Beginning with the script used to generate layoutA, modify the keyword(s) as suggested.
    b. Re-run OPCSBAR.
5. Compare this new layout to layoutA and layoutB to see how much this helped, if at all.
6. Continue tweaking the keyword value until the problem is resolved.

After you have resolved one problem, if the output is still not what you want, repeat the process for another problem, this time comparing the most recent results with the layoutB results.

Table 5-5. Resolving Design Problems

Problem in LayoutA	Try this…
Not all scattering bars created	Run with the original values for EXTENSION and LINEENDMERGE, but without MINEDGELENGTH. If this helps, you can still use MINEDGELENGTH, but it will need to be smaller than the value you used originally. Experiment to find the best value.
Scattering bars created the wrong length	Adjust the value for EXTENSION.
One long scattering bar where there should be two	Run with the original values for EXTENSION and MINEDGELENGTH, but without LINEENDMERGE. If this helps, adjust EXTENSION and LINEENDMERGE values so scattering bars are only merged where you want them to be.

Rule Problem Resolution

When the problem is with the rules, you must look for the solution in one or more of the following places:

- Input layers: in_layer and ref_layer
- Placement arguments: SBOFFSET and CENTER
- SPACE and WIDTH conditions (refer to Figure 4-23)

Start by looking at one set of scattering bars at a time. You do this by setting the priority level for NOCLEANUP as shown in Table 5-6.

Table 5-6. Scattering Bars Returned Based on Priority

priority	Rules alone	with SBLAYER	with one PRIORITIZE BY LAYER	with SBLAYER and one PRIORITIZE BY LAYER
1	closest to edge	on SBLAYER	closest to edges touching island layer	on SBLAYER

Table 5-6. Scattering Bars Returned Based on Priority (cont.)

priority	Rules alone	with SBLAYER	with one PRIORITIZE BY LAYER	with SBLAYER and one PRIORITIZE BY LAYER
2	second from edge	closest to edge	closest to other edges	closest to edges touching island layer
3	third from edge	second from edge	second from edges touching island layer	closest to other edges
4		third from edge	second from other edges	second from edges touching island layer
5			third from edges touching island layer	second from other edges
6			third from other edges	third from edges touching island layer
7				third from other edges

1. Find one area where the original results were not what you wanted.
2. Edit the rule file used to generate a layout using NOCLEANUP without EXTENSION, LINEENDMERGE, or MINEDGELENGTH (layoutB), setting NOCLEANUP priority to 1.

   ```
   NOCLEANUP 1
   ```

3. Run OPCSBAR with this rule file to generate layoutC.
4. Compare layoutC to your expectations. Check Table 5-6 to make sure you know exactly what scattering bars you should be seeing.
5. Once you identify a problem, look it up in Table 5-7. This table suggests probable solutions.
6. Test the recommended solution:
   a. Beginning with the script used to generate layout C, modifying the input layers or rules as suggested.
   b. Re-run OPCSBAR.
7. Compare this new layout to layoutA and layoutB to see how much this helped, if at all.
8. Continue adjusting the rules until this set of scattering bars appear the way you want them. Repeat for every other set of scattering bars until the final results are the RAW scattering bars you expected.

Table 5-7. Resolving Rules Problems

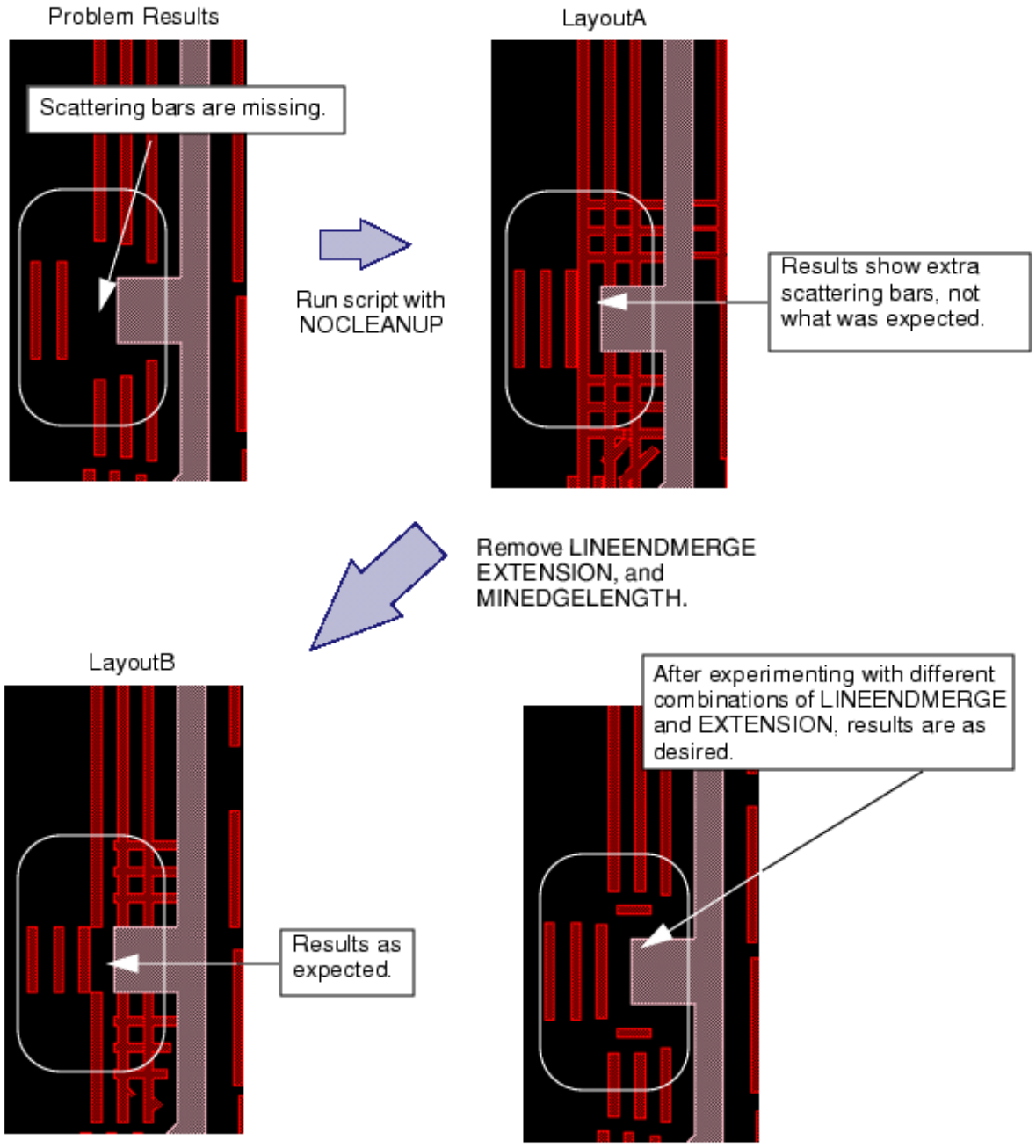
Problem	Try this…
Wrong number of scattering bars	If using *ref_layer*, run OPCSBAR without it. If not using *ref_layer*, look for unbounded spaces.
Scattering bars in wrong place	Check the SBOFFSET in SPACE or SPACELAYER that was used to generate these scattering bars.
Missing scattering bars here and there	If using SPACE rules, check the WIDTH conditions used to generate these scattering bars.
Scattering bars generated for edges that should not be treated.	Create a DRC CHECK MAP statement (detailed in the *Standard Verification Rule Format (SVRF) Manual*) to write the ***in_layer*** as a separate layer in your output database. Perform a visual inspection to make sure it contains expected data. If ***in_layer*** is a polygon layer, use SVRF to derive an edge layer to pass only the edges requiring scattering bar treatment.

Using NOCLEANUP to Resolve Scattering Bar Generation Failure

When expected scattering bars fail to be generated, the problem is typically caused by the cleanup phase of raw scattering bars.

Figure 5-3 shows how you can find the cause of the problem with NOCLEANUP. This sample data was designed to illustrate how to debug missing scattering bars and should not be interpreted as a production scattering bar placement recipe.

Figure 5-3. NOCLEANUP Usage



Performance Issues

The leading cause of performance problems is passing flattened (or partially flattened) data to the Calibre OPCsbar command. Some flattening is always likely to occur as you prepare the original layers using SVRF commands. When performance is an issue, flatten the data as little as possible.

Tracking down flattening problems involves studying the Calibre transcript. The Calibre nmDRC hierarchical engine automatically generates this text transcript at run-time and sends it

to stdout, which is typically the shell window from which you invoked the application. It is good practice to save the transcript by redirecting the output to a file, which you can do when you invoke the Calibre nmDRC hierarchical engine, by using the pipe/tee syntax, as follows:

```
calibre -drc -hier -turbo sbar.drc | tee sbar.log
```

In the transcript, you see a separate set of statistics for every layer operation:

- The first line shows the layer operation as it appears in the rule file.
- This line is followed by a dashed line.
- The second line shows the name of the generated layer followed by several statistics.

Statistics of interest are HGC and FGC:

- HGC: The number of hierarchical objects on the layer.
- FGC: The estimated number of flat objects on the layer.

For every layer operation, you can calculate the degree of flattening as the ratio of HGC to FGC:

$$\frac{4578}{316325} = 0.014$$

For a more complete description of the Calibre transcript, refer to the "nmDRC Results" chapter of the *Calibre Verification User's Manual.*

```
nplus = sdm NOT ppm
------------------------
nplus (HIER TYP=1 CFG=1 HGC=4578 FGC=316325 VHC=F VPC=F)CPU TIME = 4 REAL
TIME = 4 LVHEAP = 27/31/31
```

Common Causes of Flattening

Flattening is contextual. Some operations applied to certain designs causes significant flattening. With other designs, the same operation may cause minimal flattening. The following sections describe operations that may cause flattening.

Isolation of Edges using EXTERNAL

Flattening is caused by creating a very large measurement region for the External dimension check detailed in the *Standard Verification Rule Format (SVRF) Manual*. When the measurement region extends beyond a cell boundary, flattening is required to check edges not in the cell:

```
temp_layer = EXTERNAL orig_layer > 2 <=4
```

By using an EXTERNAL dimension check, followed by a logical NOT, you are only requiring Calibre to search in small measurement regions. The chances that the region extends beyond a cell boundary are greatly reduced, so flattening is also greatly reduced:

```
temp_layer1 = EXTERNAL orig_layer <= 2
temp_layer = orig_layer NOT temp_layer1
```

Isolation of Critically Dimensioned Features using SIZE

SIZE with UNDEROVER finds critically dimensioned features by reducing features to the point where all features smaller than 2*size_by_value disappear. When the second half of the operation enlarges the features back to their original sizes, the critically dimensioned features no longer exist. Using the Boolean NOT, you can then remove all features that are not critically dimensioned from the original layer. This process can cause a significant amount of flattening wherever polygons cross cell boundaries:

```
temp_layer1 = SIZE orig_layer BY 0.07 UNDEROVER
temp_layer = orig_layer NOT temp_layer1
```

INTERNAL finds critically dimensioned features more directly by finding internally facing edges nearer than the specified amount. Calibre does not usually need to flatten data to perform this operation because the dimension check is within a single feature, and hence a single cell. However, this is not as accurate because it measures distances perpendicular to an original edge, which will not always be the shortest distance between the two edges:

```
temp_layer = INTERNAL orig_layer <= 0.14
temp_layer = orig_layer WITH WIDTH <= 0.14
```

A third approach, WITH WIDTH finds critically dimensioned features more directly by isolating those features that are narrower than the specified amount.

This approach is most accurate because it uses the shortest distance between two edges as the width, regardless of the orientation. However, it also causes the most flattening because performing a shortest distance calculation requires that all portions of a polygon must be placed on the same hierarchical level.

Table 5-8. Comparing Methods for Finding CD Features

Operation	Performance	Accuracy
INTERNAL	Best	Worst
SIZE UNDEROVER	Better	Better
WITH WIDTH	Worst	Best

Table 5-8 shows this trade-off.

Extensive Edge Classification

For some designs, Calibre OPCsbar causes large amounts of flattening. You can minimize this in two ways:

- Supply an ***in_layer*** containing only those portions of the design requiring scattering bar treatment. This can cause less flattening if it cuts down on the number of edges to classify and on the scope of the measurement regions created in the process:

```
OPCSBAR orig_layer
   <sbar recipe>
```

- Use SPACELAYER rules to avoid performing edge classification altogether. Note, however, that using this method, the pre-processing you do may end up flattening the data just as much or more than Calibre OPCsbar:

```
OPCSBAR temp_poly_layer orig_layer
   <sbar recipe using SPACE rules>

OPCSBAR temp_edge_layer orig_layer
   <sbar recipe using SPACE rules>

OPCSBAR orig_layer
   <sbar recipe using SPACELAYER rules>
```

Appendix A
OPCSBAR Command Keywords

OPCSBAR keywords support various methods of generating scattering bars.

OPCSBAR Summary Table **161**

OPCSBAR Summary Table

Use this table as a handy quick-reference for all Calibre OPCsbar commands.

Table A-1. OPCSBAR Keywords

Keywords	Type	Default	Description
ANGLED	flag	Only orthogonal scattering bars allowed	45 degree edges allowed or not.
ANGLEEDGE	keyword	No default	Applies the present rule only to edges derived from angled features.
CENTERMERGE	positive floating pt number	Do not merge	Merges two scattering bars into one if they are closer than value.
COLINEARIZE	positive floating pt number	Do not colinearize	Indicates that overlapping scattering bars that are offset from one another are merged if at least one is shorter than value.
EXTENSION	non-zero floating pt number	Not extended	Edge extended or shortened prior to creating scattering bars.
HORIZONTAL	keyword	No default	Applies the present rule only to edges derived from horizontal features.
INTERSECTION	key letter	L	Types of intersections allowed between scattering bars.

Table A-1. OPCSBAR Keywords (cont.)

Keywords	Type	Default	Description
JOG FILL	flag and positive floating pt number(s)	Jog is cut, not filled	Indicates jog is filled or cut. When value is supplied, fills jogs between nearby (not touching) scattering bars when closer than value.
LINEENDMERGE	positive floating pt number	Do not merge	Merge adjacent scattering bars if they are lined up perfectly and if the distance between them is less than value.
LINEENDOFFSET	positive floating pt number	MINSBOFFSET	Minimum separation between the line-end of one scattering bar and the edge of an original feature.
LINEENDTOLONGSPACE	positive floating pt number	LINEENDSPACE	Minimum spacing requirement between the line-end edge of one scattering bar created by this operation and the long edge of another scattering bar.
LINEENDSPACE	positive floating pt number	MINSBSPACE	Minimum separation between the line-ends of two scattering bars.
MAXSBLENGTH	positive floating pt number	No default	Largest allowed size for final scattering bar.
MAXSBWIDTH	positive floating pt number	Largest SBWIDTH in any rule plus two database units.	Largest allowed size for final scattering bar.
MINEDGELENGTH	positive floating pt number	0	Minimum length of classified edges for which scattering bars will be generated.
MINJOGWIDTH	positive floating pt number	MINSBWIDTH	Defines the minimum overlap required for a jog to be legal.
MINSBLENGTH	positive floating pt number	Smallest SBWIDTH	Minimum length of scattering bar outside jogs.
MINSBOFFSET	positive floating pt number	Smallest SBOFFSET	Minimum spacing from original polygon to scattering bar.

Table A-1. OPCSBAR Keywords (cont.)

Keywords	Type	Default	Description
MINSBSPACE	positive floating pt number	MINSBOFFSET	Smallest allowed distance between scattering bars.
MINSBWIDTH	positive floating pt number	Smallest SBWIDTH in any rule minus one database unit	Smallest allowed size for final scattering bar.
NEGATIVE	flag	Not negative — Generate scattering bars relative to outside edge	Indicates that scattering bars should be created relative to the inside edge of the polygon.
NOCLEANUP	flag optional positive integer	Clean and output all scattering bars	Specifies whether to output raw or cleaned up edges. If cleaned up edges are output, specifies which ones (by priority).
OFFSETLAYER	layer	No default	Defines areas where scattering bars cannot be created. Scattering bars cannot be created inside or within MINSBOFFSET or LINEENDOFFSET of these areas.
OPENT	literal	Trims away the intersection	Instructs the operation to cleanup T intersections by shortening the stem of the T.
OPPOSITEEXTENDED	non-zero floating pt number	No default	Defines the method used to classify edges based on space.
OPTION CAREFUL_CLEANUP	keyword, optional	Use the default scattering bar cleanup.	Compares the original scattering bars to the cleaned up scattering bars to ensure that none were lost during cleanup.
OPTION CAREFUL_CLEANUP	optional keyword	1	Specifies the level that scattering bars will be recaptured.
OPTION INCLUSIVE_JOG_LENGTH	optional keyword	YES	Dispositions cleanup for scattering bars in jog conditions.
OPTION PROCESSING_MODE	optional keyword	HIERARCHICAL	Enables litho flat processing mode.

Table A-1. OPCSBAR Keywords (cont.)

Keywords	Type	Default	Description
OPTION PRIORITIZE_SPACE	optional keyword	NO (off)	Prioritizes negative scattering bars based on the SPACE condition in a rule.
OPTION VERIFICATION_MODE	optional keyword	NO (off)	Finds the deleted scattering bars of an executed Calibre OPCsbar operation.
PRIORITIZE BY LAYER	layer	No layers: Priority by order	Prioritization scheme for resolving conflicts based on island layers.
PRIORITYCENTER	keywords	No default	Gives the center scattering bar the highest priority during the cleanup process.
SBLAYER	layer	No default	Adds a polygon layer containing pre-existing scattering bars to be factored into calculations when generating additional scattering bars.
SMOOTH	positive floating pt number	Do not smooth	Smooths target edges formed by edge classification.
SPACE	constraint: range of floating pt numbers	No default	Identifies layers by spacing and other parameters for classification and subsequent scattering bar treatment. Mutually exclusive with SPACELAYER.
SPACELAYER	layer	No default	Identifies edges to be treated. Mutually exclusive with SPACE.
OPTION TILEMICRONS	keyword	100 microns	Specifies a tile size.
VERTICAL	keyword	No default	Applies the present rule only to edges derived from vertical features.
WITHWIDTH	literal	INTERNAL OPPOSITE	Classifies edges using WITH WIDTH.

Appendix B
Calibre OPCsbar Rule File Template

Templates can be used as a starting point for OPCSBAR file development.

A Simple OPCSBAR Implementation 165

A Simple OPCSBAR Implementation

Use this example to begin development of your own OPCSBAR implementation.

```
//*********************************************************************
//*************************** Input Section ***************************
//*********************************************************************

LAYOUT SYSTEM GDSII
LAYOUT PATH "myGds.gds"
//          ^-----------------------Write a GDS file name here
LAYOUT PRIMARY "top"
//              ^--------------------Write its Top Cell name
FLAG SKEW YES
LAYOUT ERROR ON INPUT YES
PRECISION 1000
//        ^--------- Write the database precision (1/grid)
RESOLUTION 1
LAYER feature  1
//             ^------Select a layer (i.e. 1, 2, etc)
//                     to receive scattering bars

//*********************************************************************
//*************************** Output Section **************************
//*********************************************************************

DRC MAXIMUM RESULTS ALL
DRC RESULTS DATABASE "myGds_withSB.gds" GDSII
//                   ^---------------Write an Output GDS
DRC SUMMARY REPORT "/dev/null"
DRC MAXIMUM VERTEX 199
DRC KEEP EMPTY YES
DRC CHECK MAP feature 1
//                   ^-------Layer number in the output GDS for the
//                            original design layer
DRC CHECK MAP assist  2
//                   ^-------Layer number in the output GDS for the
//                            assist features
```

```
//*********************************************************************
//************************** Variables ********************************
//*********************************************************************

//*********************************************************************
// THESE PARAMETERS DEFINE THE ASPECT OF THE SCATTERING BARS
//*********************************************************************

// Used to avoid rounding errors.
VARIABLE eps 0.001

// Scattering Bar width
VARIABLE sbWidthV 0.06

// Distance from feature to sb.
VARIABLE sbDist 0.16

// Minimum space between sb for each existing one
VARIABLE minSbGap 0.096

// Minimum Distance between Critical Main Feature and Scattering Bar
VARIABLE minOff 0.11

// Minimum Length for the Scattering Bars
VARIABLE minLength 0.130

// Minimum Jog Leg Size to be "pre-cleaned" using COLINEARIZE
// It cannot be larger than sbWidthV (SBwidth)
VARIABLE minJogLeg 0.015

//*********************************************************************
// THESE PARAMETERS DEFINE THE PLACEMENT OF THE SCATTERING BARS
// Note: They can be defined explicitly or generally using the
//       previously defined parameters
//*********************************************************************

// Min Value is 2*sbDist + sbWidth
// If space < space1, no sb.
VARIABLE space1 2*sbDist + sbWidthV

// Space between space1 and space2, 1 centered sb.
// Min Value is 2* sbDist + 2*sbWidthV + minSbGap
// > space2, each edge gets 1 sb.
VARIABLE space2 2*sbDist+2*sbWidthV+minSbGap

// > spacer3 and < space4 leaves room for a single
// bar centered between 2 bars 2*sbDist+3*sbWidthV+2*minSbGap
VARIABLE space3 2*sbDist+3*sbWidthV+2*minSbGap

// > space 4 has room for at least an additional bar
VARIABLE space4 2*sbDist+4*sbWidthV+3*minSbGap

// SPACE5 has room for a single bar centered between two second bars
VARIABLE space5 2*sbDist+5*sbWidthV+4*minSbGap

// SPACE6 has room for a feature to have 3 bars
VARIABLE space6 2*sbDist+6*sbWidthV+5*minSbGap
```

```
//*********************************************************************
// THESE PARAMETERS DEFINE THE CLEANING AND MANUFACTURABILITY OF
//                      THE SCATTERING BARS
//*********************************************************************

// Join SBs if end to end space <= to this.
VARIABLE sbJoinDist 1.3

// Extend SBs by this amount
VARIABLE sbExtend 0.10

// Variable to round to user units
VARIABLE gdsPrecision 1000

//*********************************************************************
//********************** The DRC RECIPE STARTS HERE ******************
//*********************************************************************
// Add any layer preprocessing here.  If you generate new layers, be
// sure to modify the call to OPCSBAR to reflect these layers

//*********************************************************************
//*************************** OUTPUT LAYERS ***************************
//*********************************************************************

assist {OPCSBAR feature
   MINSBOFFSET  minOff
   MINSBSPACE   minSbGap
   MINSBWIDTH   sbWidthV-eps
   MAXSBWIDTH   sbWIdthV+eps
   MINSBLENGTH  minLength
   EXTENSION    sbExtend
   INTERSECTION N
   ANGLED

   SPACE >=space1 <space2 SBWIDTH sbWidthV  CENTER

   SPACE >=space2 <space3 SBWIDTH sbWidthV  SBOFFSET sbDist

   SPACE >=space3 <space4 SBWIDTH sbWidthV  SBOFFSET sbDist
      SBWIDTH sbWidthV  CENTER

   SPACE >=space4 <space5 SBWIDTH sbWidthV  SBOFFSET sbDist
      SBWIDTH sbWidthV  SBOFFSET sbDist+sbWidthV+minSbGap

   SPACE >=space5 <space6 SBWIDTH sbWidthV  SBOFFSET sbDist
      SBWIDTH sbWidthV  SBOFFSET sbDist+sbWidthV+minSBGap
      SBWIDTH sbWidthV  CENTER

   SPACE >=space6 SBWIDTH sbWidthV  SBOFFSET sbDist
      SBWIDTH sbWidthV  SBOFFSET sbDist+sbWidthV+minSBGap
      SBWIDTH sbWidthV  SBOFFSET sbDist+2*sbWidthV+2*minSBGap

   LINEENDMERGE sbJoinDist
}
```

Index

— Symbols —

[], 18
{}, 18
|, 18

— A —

ANGLED, 69
  affect on JOG FILL, 136
  affect on MINSBLENGTH, 70
ANGLEEDGE, 70
Angles
  and Calibre OPCsbar, 122

— B —

Bold words, 18
Boundary edges, 117

— C —

Calibre command syntax, 14
Calibre nmDRC hierarchical engine
  input, 14
  rule files, 14
Calibre OPCsbar
  operation, 33
  parameters illustrated, 132
  tasks performed by, 13
Calibre transcript, 15
CENTER
  with WIDTH, 124
CENTERMERGE, 80
CLEANSBLAYER, 42
Cleanup
  intersection, 136
  jog fill, 134
  of non-rectangular shapes, 132
  order, 129
  parameters used for, 128
COLINEARIZE, 81
Command reference, 18
Courier font, 18

— D —

Databases
  Calibre nmDRC hierarchy, 14
  results, 14, 15
Double pipes, 18
DRC CHECK MAP, 110, 155
DRC results summary, 15

— E —

Edges
  boundary, 120
  related to OPCsbar, 116
Euclidean metric, 130
Event log, 16
Existing scattering bars, 42
EXTENSION, 83
Extensions, negative, 83

— G —

GDS Files
  as input to Calibre, 14
  for results database, 15

— H —

Heavy font, 18
HORIZONTAL, 71

— I —

INTER SECTION
  keyword for OPCSBAR, 136
INTERSECTION, 72
Intersection cleanup
  definition, 137
  illustration, 138
Intersections
  LINEENDSPACE and correcting, 56
Island layers
  OPCSBAR cleanup, 143
Italic font, 18

— J —

JOG FILL, 84
  and ANGLED, 136
  illustration, 135
  of non-touching scattering bars, 135
Jogs
  corrections require d wi th OPCSBAR, 134
  generated for straight edges, 120

— K —

Keepout areas, 40

— L —

L intersections
  illustrations of cleanup, 140
  when two meet, 129
Layer prioritization, 141
Layer statistics, 157
Layers
  creating scattering bars, 115
LINEENDMERGE, 86
LINEENDOFFSET, 55
  and keepout areas, 40
LINEENDSPACE, 56
LINEENDTOLONGSPACE, 57
Long edge, 57

— M —

MAXSBLENGTH, 58
MAXSBWIDTH, 60
Measurement regions
  with Calibre OPCsbar, 130
Merging
  line-ends, 86
  overlapping scattering bars, 81
  parallel scattering bars, 81
Merging scattering bars
  illustra tions, 86
MI NEDGELENGTH, 61
MINEDGELENGTH, 62
  illustration, 61
Minimum keyword, 18
MINJOGWIDTH, 88
MINSBLENGTH, 63
MINSBOFFSET, 64
  and keepout areas, 40
MINSBSPACE, 66
Multi-threaded processing, 15

— N —

NEGATIVE, 74
Negative extensions, 83
Negative scattering bars
  checking space and offset constraints, 130
  creating with SPACE rules, 125
  creating with SPACELAYER rules, 127
  enforcing rules for, 130
  full cut, 113
  illustration showing offsets, 114
  illustration showing WIDTH, 125
  keywords impacted by, 125
  merging with original layer data, 17
  NEGATIVE parameter, 163
  partial cut, 113
NOCLEANUP, 90
  flow using, 151
  troubleshooting, 156

— O —

Offset
  constraints, 129
  definition, 111
OFFSETLAYER, 40
OPC operations
  Calibre OPCsbar, 33
OPCSBAR reference layer, 41
OPENT, 91
OPPOSITEEXTEND, 75
OPTION CAREFUL_CLEANUP, 92
OPTION FAST_MRC, 94
OPTION I NCLUSIVE_JOG_LENGTH, 95
OPTION
    MINSBSPACE_CLEANUP_TIEBRE
    AKER, 97
OPTION PRIORITIZE_SPACE, 99
OPTION TILEMICRONS, 106, 107
OPTION VERIFICATION_MODE, 93, 108
Order for cleanup, 129

— P —

Parentheses, 18
Partial negative-cut scattering bars, 113

Pipes, 18
Prioritization
  by layer, 141
PRIORITIZE BY LAYER, 100, 143
Prioritizing scattering bars
  impact on intersection cleanup, 137
Priority
  based on order, 141
  related to NOCLEANUP, 153
  when using PRIORITIZE BY LAYER, 144
PRIORITYCENTER, 101

— Q —

Quotation marks, 18

— R —

Raw scattering bars
  returning all, 90
Rectangularity, 132
Reference layer
  illustration, 122
  spaces with Calibre OPCsbar, 120
Results database, 15
Rule files
  functions of, 14
Rule-based scattering bars, 115
Rulecheck
  statement, 110
Rules
  for Calibre OPCsbar, 117
Runtime summary, 15
Runtime warnings, 15

— S —

SBLAYER, 42
SBOFFSET, 116
  negative scattering bars, 47, 52
  positive scattering bars, 47, 52
SBWIDTH, 116
Scattering bars
  creating, 128
  existing used as input, 42
  not where expected, 151
Slanted words, 18
SMOOTH, 103
SPACE, 45
  CENTER, 48
  CENTERPITCH, 48
  definition, 111
  explanation, 116
  illustration, 116
  SBOFFSET, 47
  SBPITCH, 47
  SBWIDTH, 46
  WIDTH, 45
Space constraints, 129
SPACELAYER, 52
  rules for, 115
  SBOFFSET, 52
  SBWIDTH, 52
  tips for, 127
Spaces
  unbounded, 118
Square parentheses, 18
Statistics
  layer, 157
STRICTCENTER, 76
SVRF Statements
  DRC CHECK MAP, 110
SVRF statements
  DRC CHECK MAP, 110, 155
  rulecheck statement, 110

— T —

Table-based rules, 115
Transcript, Calibre, 15
-turbo, 15
-turbo_all, 15
-turbo_litho, 15

— U —

u_bottom_length, 140
Unbounded spaces, 118
Underlined words, 18

— V —

VERTICAL, 77

— W —

Warning messages, 16
WIDTH
  impact on CENTER, 124
Width constraints, 129

WITHWID TH, 78

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.