Checking Design Rules

Introduction

Modern fabrication processes are *pattern independent*: there is a clean separation between the design effort that creates the patterns to be fabricated and the actual process of fabrication.

This independence requires the designer to have a precise knowledge of the capabilities of the processing line, usually in the form of geometrical constraints that satisfy both the resolution of the fabrication process and the device physics governing the proper operation of transistors and interconnections. These constraints are expressed as sets of *design rules*.

Design rules, in their simplest form, are usually minimum allowable values for certain *widths*, *separations*, *extensions*, and *overlaps* of and between geometrical objects. The exact nature of design rules is dependent on specifications supplied by the foundry to which the design will be submitted for fabrication.

Tools > DRC (for whole cells) and **Tools > DRC Box** (for limited regions) run a *design rule checker* which determines whether a design obeys a specific set of rules. Design rule violations can be reported directly on the layout, in a text file, or both.

Design Rule Sets

Sets of design rules do not typically have to be created from scratch. If the rules to be used are sufficiently similar to those for a previous design, then the previous rule set can be modified as required.

In general, creating or editing a design rule set involves three steps.

- Determine the rules which have to be specified. Fabrication services are typically able to provide design rule sets, perhaps even with illustrative examples.
- Determine which generated layers, if any, will be needed to implement each rule in the set. Define these layers with **Setup > Layers** , and generate them with **Tools > Generate Layers**.
- Enter the rules with **Setup > DRC**.

Setups

The design rule set is part of the *setup* specification that characterizes every L-Edit design. The setup should be established before the new design is begun. There are several ways of doing this, described below.

Copying Setup Information to a New File

A *copy* of an existing file automatically contains the same setup information as the original.

Another method is to use **File > New** to create a new file while a file with the desired setup information is active. The new file automatically contains the same setup information.

The design rules contained in the new or copied file's setup information can be modified with **Setup > DRC**.

Combining Rules from Different Files

Design rules from the setups of different TDB-format or TTX-format files can be combined as follows.

- Delete existing design rules with **Setup > DRC**: click **Delete Rule** for each rule to be deleted.
- Create any additional needed layers, including generated layers, and remove any unused layers. Care should be taken during this step to create *all* and *only* the required layers. Design rules associated with missing layers will not work properly. No design rules will be specified for extra layers, so layout errors on those layers will not be detected.

- Specify the TDB or TTX file from which existing rules are to be taken with **File > Replace Setup**. Enter the name of the file, uncheck all the options except **DRC rules**, and click **OK**. The specified setup, including design rules, is read into the active file.

Generated Layers

Design rules may be specified for generated layers just as for other layers. When a generated layer is used in the specification of a design rule, the rule checker generates it before proceeding, and then automatically deletes it when the check is complete.

Rule Lists

The **Write to file** button in the **Setup > DRC** dialog outputs the entire design rule set in list form to a text file. The name of the output file is ***design*.rul**, where ***design*** is the base name of the active design file.

Each design rule has an entry in the rule file, in the following form:

```
name
Type: type, Distance: distance unit
          Layer: layer1
          Layer: layer2
```

where the first two text lines specify the ***name***, ***type*** (including any exceptions), and ***distance*** (with the associated ***unit***) of the design rule, and the next lines specify the involved layers, one per line. Definitions for generated layers are also shown.

For example:

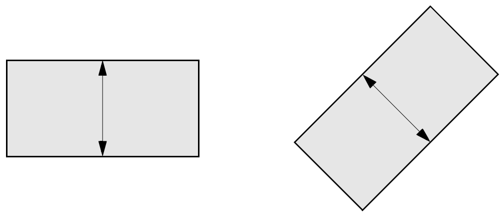
```
8.4a Via to Poly Spacing
Type: Spacing:E, Distance: 2 Lambda
      Layer: ViaNotOnPad
            Layer: Via
            AND
            NOT Layer: Pad Comment
      Layer: Poly
```

Design Rule Types

Seven types of design rules are supported. Each of these rule types is described below; the figures include specific examples.

Minimum Width

Minimum width rules specify the minimum width of all objects, in any direction, on the named layer.

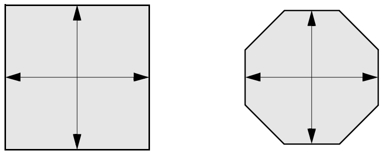


Poly Minimum Width = 2

Certain exceptions can be specified.

Exact Width

Exact width rules specify the exact width of all objects on the named layer. The width of octagons is measured between parallel sides.



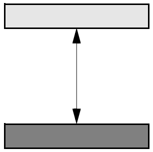
Poly Contact Exact Width = 2

Not Exist

Not exist rules specify that no objects should exist on the named layer. Not exist rules are unique in having no associated distance.

Spacing

Spacing rules specify the minimum distance that should separate all pairs of objects either on the same layer or two different layers.

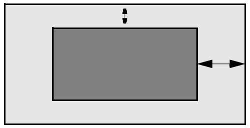


Via to Poly Contact Spacing = 2

Certain exceptions can be specified.

Surround

Surround rules specify that objects on one layer must be completely surrounded by objects on another layer.

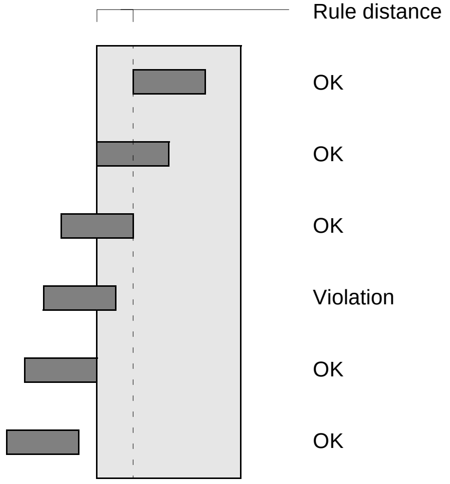


Metal2 Surround Via = 1

Certain exceptions can be specified.

Overlap

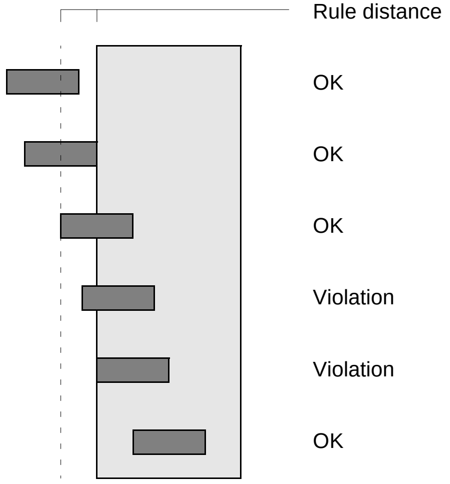
Overlap rules specify the minimum amount that an object on one layer must overlap an object on another layer (when there is an overlap). Objects which overlap more than the distance, or whose edges coincide, are not considered in violation of overlap rules.



Poly Contact Overlap Poly = 2

Extension

Extension rules specify the minimum amount that an object on one layer must extend beyond the edge of an object on another layer. Objects which extend more than the distance, or which have a coincident edge but are otherwise *outside*, or which are entirely surrounded, are not considered in violation of extension rules.



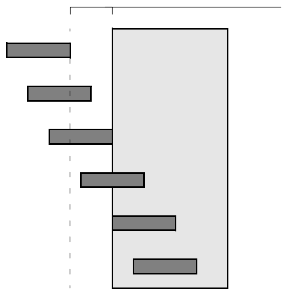
Poly Contact Extend Poly = 2

Rule Exceptions

Some rules can be fine-tuned by specifying certain layout conditions that are *not* to be reported as violations. These conditions are represented by the **Ignore** options in the **Setup > DRC** dialog:

Condition	*Description*	*Applicable rules*
Coincidences	Coincident edges between objects are ignored.	Surround
Intersections	Intersections between objects are ignored.	Surround
If layer 2 completely encloses layer 1	Objects on one layer *entirely surrounded* by objects on another layer are ignored.	Spacing
If layer 1 completely outside layer 2	Objects on one layer *entirely outside* objects on another layer are ignored.	Surround
45 degree acute angles	Objects with angles of 45° (or less) are ignored.	Minimum width Spacing Surround

Some illustrations of these exception conditions for spacing and surround rules.



Spacing rule distance

Always OK

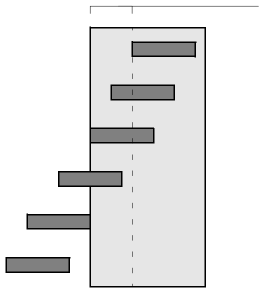
Always a violation

OK if **Coincidences** checked — else a violation

OK if **Intersections** checked — else a violation

OK if **If layer 2 … encloses layer 1** checked — else a violation

OK if **If layer 2 … encloses layer 1** checked — else a violation



Surround rule distance

Always OK

Always a violation

OK if **Coincidences** checked — else a violation

OK if **Intersections** checked — else a violation

OK if **If layer 1 … outside layer 2** checked — else a violation

OK if **If layer 1 … outside layer 2** checked — else a violation

Checking the Layout

After a layout is complete, a check for design rule violations should be made before the layout is sent to the chip foundry for fabrication. If a chip is fabricated with design rule violations, it may fail to function as designed.

L-Edit can check only boxes and 45°/90° polygons and wires for design rule violations; it cannot check circles or all-angle polygons and wires.

Full-Cell Check

Tools > DRC performs a *full-cell* check on the active cell's layout.

The layout is divided into a grid of square *bins*. The **Bin size** (the length of a bin side) is measured in Locator Units. The full-cell check is performed bin by bin, beginning in the lower left corner of the layout and moving to the right. When the right edge is reached, the check moves up to the next row, beginning again on the left.

Thus two objects placed relatively far apart will never be checked for design rule correctness. This poses no danger, since objects far apart always satisfy any applicable spacing rules.

Errors can be reported in three ways: marked by (1) ports or (2) objects placed at the sites of violations, or (3) written to a text file.

When the check is completed, L-Edit displays the total number of errors detected. Each bin containing a rule violation is shown as a box on the Error layer, marked with an × (two wires along the diagonals).

Region-Only Check

Tools > DRC Box performs *region-only* checking within the active cell's layout.

A region-only check is used when a restricted area or group of objects in the layout needs to be checked for design rule errors. This is useful for checking a region during the creation of some layout or after correcting a design rule violation in the region.

Use the DRAW mouse button to drag a rectangular region surrounding the layout to be checked. When the DRAW button is released, the same dialog as for the full-chip check appears, and errors are reported identically.

When the check is complete, the region checked is represented by a box on the Error layer.

Sporadic region-only checking is not a substitute for a full-cell check. A full-cell check should always be performed after the last layout alteration and before submitting the design for fabrication.

Working with 45° Layout

Generation of layers containing 45° objects can produce off-grid vertices due to off-grid intersections of 45° polygons or conversion of 45° wires to polygons.

The coordinates of off-grid vertices are rounded to the nearest Internal Unit while still preserving angles of edges. If the dimensions of the source objects (measured in Internal Units) are small, then the resulting polygons may be distorted.

To prevent distortion of generated objects due to rounding, a *subgrid* must be maintained. The subgrid is the smallest possible non-zero value, in Internal Units, of any edge length, wire width, distance between any two objects, or Grow distance.

The best way to maintain a subgrid is to set the mouse snap grid to prevent objects or spacings smaller than a specified value from being created. A snap grid equivalent to at least 100 Internal Units is recommended for designs containing 45° layout.

Correcting Errors

Error Markers

When specified in the **Tools > DRC** dialog, *error ports* and *error objects* are placed on the Error layer at violation locations (in addition to the diagonally marked boxes that always appear, representing bins containing violations).

An error port's name consists of the *name* of the violated design rule accompanied by a bracketed *expression* indicating the nature of the error.

For example, an error port named **8.4c Via to Active Spacing [1 < 2]** shows that the associated violation involves a spacing of 1, when the minimum Via-Active spacing should be 2.

An error object is a wire indicating any distance that violated a design rule.

Error ports and objects can be moved, deleted, and hidden and shown.

Error Files

Errors can also be written into a specified result file. This is a plain text file (with default name ***cell*.drc**, where ***cell*** is the name of the checked cell) in the following format:

```
DRC Errors in cell cell of file file.
rule = value unit; (x1,y1)->(x2,y2)
...

number errors.
...
```

The first line names the checked ***cell*** and ***file***. The second line is repeated once for every error: the name of the ***rule*** violated, the required ***value*** for the specified rule, the ***unit*** of the specified value, and coordinates (***x1,y1***) and (***x2,y2***) indicating the location of the error on the layout. The last line shows the total ***number*** of errors reported. There is some further information about timing.

Finding Error Markers

Edit > Find, along with **Edit > Find Next** and **Edit > Find Previous**, can locate reported rule violations.

Specify a **Port** search (for error ports) or a **Wire** search (for error objects), with the **Pan** or **Pan & Zoom** options, on the Error layer.

Because of the default rendering setup of the Error layer, a port's text is only visible when the port is selected. Thus, as the search commands go through ports on the Error layer, the names of the error ports become visible one at a time.

Clearing Error Markers

Tools > Clear Error Layer removes error markers (ports and objects).

It is not necessary to clear the Error layer before running a design rule check. **Tools > DRC** and **Tools > DRC Box** automatically clear the Error layer.

Optimizing Performance

Checking for design rule violations is a complicated, computation-intensive process, involving large numbers of comparisons and measurements. This can result in very long execution times. This section outlines several ways to improve the time spent on the DRC process.

Reducing Distances

The time required to perform a design rule check is related to the greatest distance specified in the rule set.

When a region or bin is checked, its boundaries are *bloated*, or increased, by a distance $B = \max(R,G)$, where R is the maximum distance specified in the design rule set and G is the sum of the absolute values of all Grow amounts specified in generated-layer definitions. This prevents errors that might be recorded because objects extend past the boundaries of a bin.

As the distances specified in design rules or Grow amounts increase, so does the time required for rule checking. To minimize this effect, keep the distances specified in the design rules and the number of Grow operations as small as possible.

Checking Incrementally

The time required for checking also varies with the square of the number of objects to be checked in each bin or region. Thus, as the design size increases, the time needed for a complete check increases dramatically.

Use region-only checks at convenient stopping points in the design process. This will also help to prevent compounding of errors which might require extensive layout modification to correct. Use the full-cell check for completed cells and the final design.

Hiding Layers

Rules which involve layers hidden at execution time are not checked. After a full-cell check has been done and the violations have been examined and repaired, the layers not involved in the violations can be hidden. Then the checker can be used to verify repairs on the affected layers only, speeding up the process.

The design rule checker automatically disables rules containing references to hidden layers.

Disabling Rules

The **Enable** option in the **Setup > DRC** dialog can be turned off for rules which do not need to be checked.

For example, pad spacing rules typically involve distances far greater than other rules. A rule like *Pad Comment Width* might be set for 100 Internal Units; if this rule were left active, *each bin* to be checked would be bloated by 100 Internal Units *on each side*, to nine times the original area. Effectively, nine bins would need to be checked for each bin in the original grid. Since the time required to perform a design rule check varies directly with the square of the number of objects to be checked, the performance time would increase (assuming constant object density) by approximately a factor of eighty-one.