Magic Tutorial #W-1: Design-Rule Extensions

Don Stark

Western Research Laboratory
Digital Equipment Corporation
Palo Alto, CA 94301

This tutorial corresponds to Magic version 7.

Tutorials to read first:

Magic Tutorial #6: Design-Rule Checking
Magic Tutorial #9: Format Conversion for CIF and Calma
Magic Maintainer's Manual #2: The Technology File

Commands introduced in this tutorial:

(None)

Macros introduced in this tutorial:

(None)

1 Introduction

Magic's original design rule checker has proved inadequate to implement all the rules found in advanced technologies. The rules described in this section allow more complicated configurations to be analyzed. Two new rules check a region's area and its maximum width. In addition, width, spacing, area, and maxwidth checks may now be performed on cif layers.

2 Area Rules

The **area** rule is used to check the minimum area of a region. Its syntax is:

area *types minarea minedge why*

Types is a list of types that compose the region, all of which must be on the same plane. *Minarea* is the minimum area that a region must have, while *minedge* is the minimum length of an edge for the region. This second dimension is basically an optimization to make the design rule checker run faster; without it, the checker has to assume that a region 1 lambda wide and *minarea* long is legal, and it must examine a much larger area when checking the interaction between cells. Specifying *minedge* reduces this interaction distance. An example rule is:

area (emitter,em1c)/npoly 6 2 "emitter must be at least 2x3"

Figure 1: Example of the area rule.

3 Maxwidth Rules

Sometimes a technology requires that a region not be wider than a certain value. The **maxwidth** rule can be used to check this.

maxwidth *layers mwidth bends why*

Layers, the types that compose the region, must all be in the same plane. The region must be less than *mwidth* wide in either the horizontal or vertical dimension. *Bends* takes one of two values, **bend_illegal** and **bend_ok**. For **bend_illegal** rules, the checker forms a bounding box around all contiguous tiles of the correct type, then checks this box's width. For example:

**maxwidth (emitter,em1c)/npoly 2 bend_illegal **
"emitter width cannot be over 2"

bend_ok rules are used to check structures where the region must be locally less than maxwidth, but may contain bends, T's, and X's.

maxwidth trench 2 bend_ok "trench must be exactly 2 wide"

Warning: the bend_ok rule is basically a kludge, and may fail for regions composed of more than one type, or for intersections more complicated than T's or X's. Figure 3 shows some examples of both types of rules.

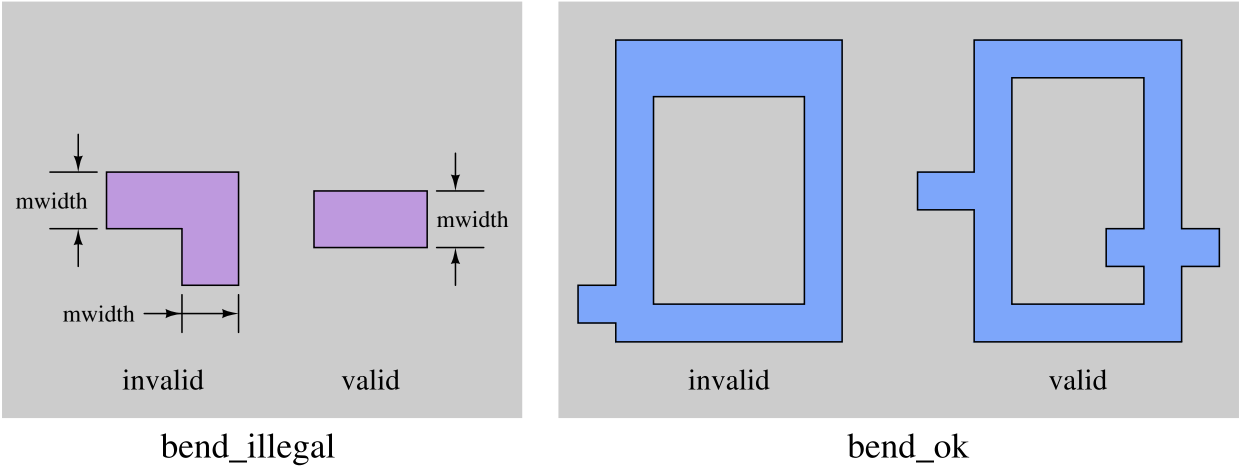

Figure 2: Examples of the maxwidth rule. The dogleg at the left would be ok in a **bend_ok** rule, but fails in a **bend_illegal** one, where the region's bounding box is checked. For **bend_ok** rules, each tile in the region is checked. The left shape fails in two places: the top horizontal part is too thick and the stub at the bottom intersects the region in a shape other than a T or X.

4 Rules on CIF layers

For technologies with complicated generated layers, it is often difficult to check design rules on the abstract types that are drawn in Magic. To ameliorate this problem, the extended checker allows simple checks to be performed on cif layers. The rules that can be checked are width, spacing, area, and maxarea. Since checking rules on the cif layers requires that these layers be generated, these checks are considerably slower than the normal ones, and should only be used when absolutely necessary.

4.1 Setting the CIF style

The **cifstyle** rule is used to select which **cifoutput** style is used.

cifstyle *cif_style*

Cif_style must be one of the cif styles included in the cifoutput section. In the current implementation, the cif checker generates all the layers in the style regardless of whether they are actually used in design-rule checks; for speed, defining a separate cif style for design rule checking it may be worthwhile when only a few layers are checked. Any layer in the cif style, defined by either a *layer* or a *templayer* rule, may be checked.

4.2 Width Checks

The syntax for **cifwidth** is analogous to that of the regular width rule:

cifwidth *layer width why*

Layer is a single cif layer. (To do width checks with more than one cif layer, **or** all the layers into a new *templayer*). *Width* is the minimum width of the region in centimicrons.

4.3 Spacing Checks

The **cifspacing** rule is also very similar to the regular rule:

cifspacing *layer1 layer2 separation adjacency why*

Layer1 and *layer2* are both cif layers. If *adjacency* is **touching_ok**, then layer1 must equal layer2. For **touching_illegal** rules, *layer1* and *layer2* may be any two cif layers. *Separation* is given in centimicrons.

4.4 Area Checks

The area rule is:

cifarea *layer minarea minedge why*

Layer is again a single cif layer. *minedge* is expressed in centimicrons, and *minarea* is given in square centimicrons.

4.5 Maxwidth Checks

The maxwidth rule is:

cifmaxwidth *layer mwidth bends why*

Again, *layer* is a single cif layer, and *mwidth* is given in centimicrons.