

1 icc2_shell> man set_floorplan_spacing_rules

2 2. Synopsys Commands

Command Reference

3 set_floorplan_spacing_rules

4
5 NAME

6 set_floorplan_spacing_rules

7 Defines a spacing floorplan rule in the **design**.

8
9 SYNTAX

10 set_floorplan_spacing_rules

11 -from_object_types from_type_list

12 -to_object_types to_type_list

13 -from_lib_cells lib_cells

14 -to_lib_cells lib_cells

15 -directions direction_list

16 [-orientation_types orientation_list]

17 -name rule_name

18 [-shielding_object_types type_list]

19 [-shielding_lib_cells lib_cells]

20 [-between_lib_cells lib_cells]

21 [-min_parallel_run_length distance]

22 [-max_parallel_run_length distance]

23 [-follow_rotations]

24 [-ignore_rotate90]

25 [-no_overlap]

26 [-no_overlap_policy no_overlap_policy_type]

27 [-identical]

28 [-mirror]

29 [-from_layers from_layer_list]

30 [-to_layers to_layer_list]

31 [-check_same_object]

32 [-check_same_object_policy check_same_object_policy_type]

33 [-forbidden_list distance_list]

34 [-forbidden_ranges {{low high} {low1 high1} ... }]

35 [-max distance]

36 [-min distance]

37 [-offset distance]

38 [-step distance]

39 [-valid_list distance_list]

40 [-valid_ranges {{low high} {low1 high1} ... }]

41 [-offset_ranges {{low high} {low1 high1} ... }]

42
43 Data Types

44 from_type_list list

45 to_type_list list

46 lib_cells collection

47 orientation_list list

48 type_list list

49 direction_list list

50 no_overlap_policy_type **string**

51 check_same_object_policy_type **string**

52 rule_name **string**

53 distance float

54 from_layer_list list

55 to_layer_list list

56 distance_list list

57 low float

58 high float

59 low1 float

60 high1 float

61
62 ARGUMENTS

63 -from_object_types from_type_list

64 Specifies the list of "from" object types **for** the spacing floor-
65 plan rule. Spacing between these **type** of objects **and** other
66 objects specified **with** -to_object_types **or library** cells speci-
67 fied **with** -to_lib_cells is checked. Valid values **for this** option
68 are block_boundary, hard_macro, placement_blockage, rout-
69 ing_blockage, shape, soft_macro, unplaceable_area, bound-

```

70         ary_cell_region, std_cell_area, io_pad and cover_bump. This
71         option is mutually exclusive with -from_lib_cells and you must
72         specify one or the other.
73
74     -to_object_types to_type_list
75         Specifies the list of "to" object types for the spacing floor-
76         plan rule. Spacing between these type of objects and other
77         objects specified with -from_object_types or library cells spec-
78         ified with -from_lib_cells is checked. Valid values for this
79         option are block_boundary, hard_macro, placement_blockage, rout-
80         ing_blockage, shape, soft_macro, unplaceable_area, bound-
81         ary_cell_region, std_cell_area, io_pad and cover_bump. This
82         option is mutually exclusive with -to_lib_cells and you must
83         specify one or the other.
84
85     -from_lib_cells lib_cells
86         Specifies the collection of library cells for the spacing floor-
87         plan rule. Spacing between these library cells and other objects
88         specified with -to_object_types or library cells specified with
89         -to_lib_cells is checked. This option is mutually exclusive with
90         -from_object_types and you must specify one or the other.
91
92     -to_lib_cells lib_cells
93         Specifies the collection of library cells for the spacing floor-
94         plan rule. Spacing between these library cells and other objects
95         specified with -from_object_types or library cells specified
96         with -from_lib_cells is checked. This option is mutually exclu-
97         sive with -to_object_types and you must specify one or the
98         other.
99
100     -shielding_object_types type_list
101         Specifies the collection of object types for the halo floorplan
102         rule as shielding objects so that the rule is not applied when
103         these object types form a shield between "from" and "to"
104         objects. Valid values for this option are hard_macro and
105         std_cell_area. This is an optional option.
106
107     -shielding_lib_cells lib_cells
108         Specifies the collection of library cells for the spacing floor-
109         plan rule as shielding objects so that the rule is not applied
110         when these lib cells form a shield between "from" and "to"
111         objects.
112
113     -between_lib_cells lib_cells
114         Specifies the collection of library cells for the spacing floor-
115         plan rule to check when this collection of cells is between the
116         specified "from" and "to" objects.
117
118     -directions direction_list
119         Specifies the sides or directions in which spacing between
120         "from" object or from library cells and "to" object or to
121         library cells needs to be checked. Valid values are any, hori-
122         zontal, vertical, left, right, bottom, top and nearest_corners.
123         The horizontal argument includes both left and right. Similarly,
124         the vertical argument includes both bottom and top. This is a
125         mandatory option.
126
127     -orientation_types orientation_list
128         Specifies the orientation of the two objects for the check to be
129         enabled. Valid values are align, mirror and partial. align means
130         both the objects should be of same orientation like R0, MX, MY
131         or R180. partial means the orientation pair should be R0-R180 or
132         MX-MY. mirror means the objects are mirrored in checked direc-
133         tion.
134
135     -name rule_name
136         Specifies the name of the spacing floorplan rule. This is a
137         mandatory option.
138

```

```

139 -min_parallel_run_length distance
140     Specifies the minimum overlap length of two "to" objects or to
141     library cells kept side-by-side. This is an optional option.
142
143 -max_parallel_run_length distance
144     Specifies the maximum overlap length of two "to" objects or to
145     library cells kept side-by-side. This is an optional option.
146
147 -follow_rotations
148     Specifies whether mentioned sides should follow the rotations of
149     library cells, that is, if meaning of horizontal or vertical
150     should change when library cell has a 90-degree rotations. This
151     option must be used together with -to_lib_cells or
152     -from_lib_cells or -to_object_types hard_macro or
153     -to_object_types soft_macro or -from_object_types hard_macro or
154     -from_object_types soft_macro. This is an optional option.
155
156 -ignore_rotate90
157     Specifies whether this rule can be ignored for library cells
158     with a 90-degree rotation. This option must be used together
159     with -to_lib_cells or -from_lib_cells or -to_object_types
160     hard_macro or -to_object_types soft_macro or -from_object_types
161     hard_macro or -from_object_types soft_macro. This is an optional
162     option.
163
164 -no_overlap
165     Specifies whether the shapes can overlap. By default the shapes
166     can overlap. This is an optional option and mutually exclusive
167     with -no_overlap_policy.
168
169 -no_overlap_policy no_overlap_policy_type
170     Specifies whether the shapes can overlap or not or internal
171     shapes need to be excluded. By default, the shapes can overlap.
172     This is an optional option and mutually exclusive with -no_over-
173     lap.
174
175 -identical
176     Specifies whether this rule applies to hard macros of same ref-
177     erence. This is an optional option.
178
179 -mirror
180     Specifies whether this rule applies when hard macros face each
181     other mirrored. This is an optional option.
182
183 -from_layers from_layer_list
184     Specifies the routing layers to be considered for
185     -from_object_types routing_blockage or -from_object_types shape.
186     This option must be used along with -from_object_types rout-
187     ing_blockage or -from_object_types shape. This is an optional
188     option.
189
190 -to_layers to_layer_list
191     Specifies the routing layers to be considered for
192     -to_object_types routing_blockage or -to_object_types shape
193     object types. This option must be used along with
194     -to_object_types routing_blockage or -to_object_types shape.
195     This is an optional option.
196
197 -check_same_object
198     Specifies whether this rule checks the spacing between edges of
199     same objects. This option must be used along with
200     -from_object_types std_cell_area or boundary_cell_region and
201     -to_object_types std_cell_area or boundary_cell_region. This is
202     an optional option and mutually exclusive with
203     -check_same_object_policy.
204
205 -check_same_object_policy check_same_object_policy_type
206     Specifies whether this rule will check the inside or outside
207     spacing between edges of same objects. This option must be used

```

208 along **with** **-from_object_types** **std_cell_area** **or** bound-
 209 ary_cell_region **and** **-to_object_types** **std_cell_area** **or** bound-
 210 ary_cell_region. This is an optional option **and** mutually exclu-
 211 sive **with** **-check_same_object**.
 212

213 **-forbidden_list distance_list**
 214 Specifies a list of distances that are **not** allowed between
 215 "from" objects **or** "from" **library cell** **and** "to" object **or** "to"
 216 **library cell**. This option is mutually exclusive **with**
 217 **-valid_list**. Values specified cannot be negative. This is an
 218 optional option.
 219

220 **-forbidden_ranges {{low high} {low1 high1} ... }**
 221 Specifies a list of distance ranges that are **not** allowed between
 222 "from" objects **or** "from" **library cell** **and** "to" object **or** to
 223 **library cell**. The distance must **not** lie **within** any of low **and**
 224 high in the specified list of ranges. This option is mutually
 225 exclusive **with** **-valid_ranges**. Values specified cannot be nega-
 226 tive. This is an optional option.
 227

228 **-max distance**
 229 Specifies the maximum distance between "from" object **or** "from"
 230 **library cell** **and** "to" object **or** "to" **library cell**. The distance
 231 cannot be greater than **this** value. The specified value cannot be
 232 negative. If **-min** is also specified then **this** value must be
 233 greater than the min value. This is an optional option.
 234

235 **-min distance**
 236 Specifies the minimum distance between "from" object **or** "from"
 237 **library cell** **and** "to" object **or** "to" **library cell**. The distance
 238 cannot be less than **this** value. The specified value cannot be
 239 negative. If **-max** is also specified then **this** value must be
 240 lesser than the max value. This is an optional option.
 241

242 **-offset distance**
 243 Specifies a **parameter** in distance calculation between "from" **and**
 244 "to" objects. This option must be used together **with** **-step**. This
 245 **implies** that the distance has to be an **integer** multiple of step
 246 value plus offset value. This option is mutually exclusive **with**
 247 **-offset_ranges**. Value specified cannot be negative.
 248

249 **-step distance**
 250 Specifies a **parameter** in distance calculation between "from" **and**
 251 "to" objects. This option must be used together **with** **-offset**.
 252 This **implies** that the distance has to be an integral multiple of
 253 step value plus offset value **or** distance has to be in range of
 254 an integral multiple of step value plus **offset_ranges** value.
 255 Value specified must be greater than zero. This is an optional
 256 option.
 257

258 **-valid_list distance_list**
 259 Specifies a list of legal separation distances between the
 260 "from" object **or** "from" **library cell** **and** "to" object **or** "to"
 261 **library cell**. This option is mutually exclusive **with** **-forbid-**
 262 **den_list**. Values specified cannot be negative. This is an
 263 optional option.
 264

265 **-valid_ranges {{low high} {low1 high1} ... }**
 266 Specifies a list of distance ranges between **with** the "from"
 267 object **or** "from" **library cell** **and** "to" object **or** "to" **library**
 268 **cell** must be separated. The distance must lie **within** any of low
 269 **and** high in the specified list of ranges. This option is mutu-
 270 ally exclusive **with** **-forbidden_ranges**. Values specified cannot
 271 be negative. This is an optional option.
 272

273 **-offset_ranges {{low high} {low1 high1} ... }**
 274 Specifies a list of distance ranges. This **implies** that the dis-
 275 tance has to be in range of an integral multiple of step value
 276 plus **offset_ranges** value. Values specified can't be negative.

This option must be used along **with** **-step**. This is an optional option.

DESCRIPTION

The `set_floorplan_spacing_rules` command defines a named spacing floorplan rule in the current **design**. The defined rule is persistent. If another floorplan rule by the same name exists then the command errors out.

There is a difference between the object **type** `core_area` **and** `std_cell_area`. The `core_area` object **type** means core boundary region without cutting out any blockages **and** is typically applicable **for** top level whereas `std_cell_area` object **type** means core boundary region after cutting out all blockages **and** is typically applicable **for** block level.

If a spacing rule is defined **for** a **library cell** **and** another spacing rule is defined **for** a hard macro then the spacing rule defined **for** the **library cell** takes precedence over the spacing rule defined **for** hard macro when checks are done **for** that **library cell**.

If the measured value falls **inside** valid range **or** is a member of the valid list then there is no violation given by `check_floorplan_rules` regardless of other constraints like min, max, **and** so on. If **this** measured value is outside valid range **or** list then a violation is reported **if** other constraints are specified **and** they are **not** met **or if** no other constraints are specified.

EXAMPLES

The following example creates a spacing rule named `s1` to check spacing between the standard **cell** area **and** the block boundary in the vertical direction, both top **and** bottom. The spacing must be at least **5**.

```
prompt> set_floorplan_spacing_rules -name s1 \  
-from_object_types std_cell_area -to_object_types block_boundary \  
-directions vertical -min 5
```

SEE ALSO

- `set_floorplan_area_rules` (2)
- `set_floorplan_enclosure_rules` (2)
- `set_floorplan_extension_rules` (2)
- `set_floorplan_exception_rules` (2)
- `set_floorplan_forbidden_rules` (2)
- `set_floorplan_halo_rules` (2)
- `set_floorplan_length_rules` (2)
- `set_floorplan_width_rules` (2)
- `remove_floorplan_rules` (2)
- `report_floorplan_rules` (2)

Version S-2021.06-SP5

Copyright (c) 2022 Synopsys, Inc. All rights reserved.

icc2_shell>