```
icc2_shell> man write_floorplan
2.  Synopsys Commands                                   Command Reference
                                write_floorplan

NAME
       write_floorplan
              Writes floorplan information to files.

SYNTAX
       status write_floorplan
              [-include include_list]
              [-exclude exclude_list]
              [-output dir_name]
              [-force]
              [-nosplit]
              [-format icc | icc2]
              [-objects objects_list]
              [-blocks blocks_list]
              [-def_units 100 | 200 | 1000 | 2000 | 10000 | 20000]
              [-compress gzip | none]
              [-add_def_dependencies true | false]
              [-via_as_fixed]
              [-routed_nets]
              [-include_physical_status include_physical_status_list]
              [-exclude_physical_status exclude_physical_status_list]
              [-cell_types cell_types_list]
              [-net_types net_types_list]
              [-read_def_options option_string]
              [-def_version version]

   Data Types
       include_list                 list
       exclude_list                 list
       dir_name                     string
       objects_list                 collection
       blocks_list                  collection
       include_physical_status_list list
       exclude_physical_status_list list
       cell_types_list              list
       net_types_list               list
       option_string                string
       version                      string

ARGUMENTS
       -include include_list
              Specifies the types of data to include in the output. This
              option cannot be specified together with the -exclude or
              -objects options. If this option is not specified, all types are
              included by default.

              The supported types are:

              o blockages

              o bounds

              o cells

              o corridors

              o die_area

              o edit_groups

              o io_guides

              o macros

              o module_boundaries
```

- nets
- pin_guides
- pins
- route_guides
- rows
- tracks
- vias
- scan_chains
- routing_directions
- voltage_areas
- fills
- pg_metal_fills
- routing_rules
- pg_regions
- placement_attractions
- bump_regions
- anchors

Specify **-include** macros to **output** only macro cells **and** skip standard cells. Specify **-include** cells to **output** all cells. If you **specify -include** {macros cells}, the command issues a warning message that macros will be ignored.

-exclude exclude_list
Specifies the types of data to exclude from the **output**. This option cannot be specified **with** the **-include** option **or** -objects options. If **this** option is specified, the command writes all types, except the types specified in exclude_list. The allowed data types are the same as **for** the **-include** option.

-output dir_name
Specifies the name of the directory in which to write the floorplan files. The **default** is **./**floorplan. If the directory already exists, the command issues an error message, unless **-force** is also specified.

-force Overwrites floorplan files in the **./**floorplan directory, **or** in the directory specified by the **-output** dir_name option. By **default**, the command issues an error message **and** exits **if** the directory exists.

-nosplit
Writes out long lines, even **if** the line length is greater than 80 columns. Use **this** option to simplify post-processing of the **output constraint** files **or** when comparing the **constraint** files by using the UNIX diff command. If **this** option is **not** specified, the command breaks long lines near column 80 **and** inserts a line continuation character **(**\).

-format icc **|** icc2
Specifies the target tool **for** which to write the floorplan. Valid values are icc **and** icc2. By **default**, the value is icc2.

To take floorplan information to the Design Compiler tool (in topographical mode) for RTL resynthesis with floorplan information such as macro placement), use the -format icc option. The EXAMPLES section contains an example of this flow.

-objects objects_list
Specifies the objects to include in the output. Supported object types are cells, ports, blockages, voltage areas, bounds, edit groups, routing corridors, module boundaries, PG regions, site rows, bump_regions, anchors and site arrays. Objects of other types are ignored.

This option cannot be specified with the -include, -exclude, or -blocks options.

-blocks blocks_list
Specifies the blocks to output hierarchically. If this option is not specified, only the top-level block is written. This option cannot be combined with the -objects option.

-def_units 100 | 200 | 1000 | 2000 | 10000 | 20000
Specifies the number of units per micron in the DEF file. All database length units are scaled accordingly in the output DEF. If not specified, the units per micron is obtained from the technology length precision. Valid values are 100, 200, 1000, 2000, 10000, and 20000. This option is passed to the DEF writer and the behavior is identical to the -units option of the write_def command.

-compress gzip | none
Specifies the compression type to use when writing the .DEF file. By default, no compression is performed.

-add_def_dependencies true | false
Specifies whether to add dependencies (sections not requested) to the .DEF file to make it self-sufficient. For example, if this option is set to true, if nets are in the -include list but vias are not, the VIAS section will be added anyway; if the option is set to false, it will not be added. The default is true.

-via_as_fixed
Indicates that all the via definitions are written out as fixed vias in the output file.

-routed_nets
Indicates that all the nets written out in the output are routed nets, that is, each net has at least one shape associated it.

-include_physical_status include_physical_status_list
Specifies which cells to include in the output based on their physical status. When you use this option, all of the specified cells are included in the generated DEF file. The physical status can be one or more of the following:

all
application_fixed
fixed
legalize_only
locked
placed
unplaced

-exclude_physical_status exclude_physical_status_list
Specifies which cells to exclude in the output file based on their physical status. When you use this option, all of the specified cell are included in the generated DEF file except for those whose physical status is specified in the exclude_physi-

cal_status_list.

-include_physical_status and -exclude_physical_status are mutually exclusive.

-cell_types cell_types_list
Specifies which cells to include in the output DEF file based on their types. When you use this option, all of the specified cells are excluded in the generated file except for those whose cell types is specified in the cell_types_list. These are the specifiable cell types:

corner
end_cap
pad
flip_chip_pad
flip_chip_driver
pad_spacer
macro
lib_cell

-net_types net_types_list
Specifies which nets to include in the output DEF file based on their types. When you use this option, all of the specified nets are excluded in the generated file except for those whose net types is specified in the net_types_list. The net types can be one or more of the following:

clock
ground
power
signal

-read_def_options option_string
Specifies options to include with the read_def command that is written to the floorplan file.

-def_version version
Specifies the DEF syntax version used to create floorplan.def file. Valid values are 5.7 and 5.8. The default is DEF version 5.8.

DESCRIPTION

The write_floorplan command writes the floorplan.tcl Tcl script and floorplan.def DEF file. The floorplan.def file generally includes the objects that are numerous, as sourcing the corresponding Tcl would be time-consuming. The floorplan.tcl contains the remaining floorplan constructs and a read_def command to load the floorplan.def file. To restore the floorplan, source the floorplan.tcl Tcl script.

By default, the floorplan files are written to a directory named ./floorplan. The destination can be changed by using the -output option. The command also writes a third file, floorplan_compare_data.txt, that contains information to use when comparing the output with a new design by using the compare_floorplans command. If the -blocks option is used, the command creates subdirectories with names corresponding to the list of blocks.

The write_floorplan command writes out user attributes associated with objects to the Tcl script. In the IC Compiler tool, there is no shape or via object associated with a terminal. In this case, the script sets user attributes from the terminal and shape or via object on the terminal for the IC Compiler tool. If there is a conflict between values of the same attribute, a higher precedence is given to the value set on the terminal.

This command returns 1 on success, 0 otherwise.

EXAMPLES

```
277         The following example writes floorplan files to the script directory.
278
279           prompt> write_floorplan -output script
280
281         The following example writes only cells and blockages.
282
283           prompt> write_floorplan -include {cells blockages}
284
285         The following example writes macros hierarchically for all blocks.
286
287           prompt> write_floorplan -include {macros} -blocks [get_blocks *:*]
288
289         The  following  example writes the selected objects and ignores objects
290         that belong to unsupported types.
291
292           prompt> write_floorplan -objects [get_selection]
293
294         The following example writes floorplan information in icc format.  This
295         could  be  used  to export the floorplan information to the Design Com-
296         piler tool (in topographical mode) where the RTL is resynthesized  with
297         this  floorplan  information.  Non-PG  nets and non-fixed cells are not
298         needed for re-synthesis and so they can be filtered out of the DEF file
299         by using post-processing.
300
301           prompt> write_floorplan -format icc
302
303         The  following example writes out the floorplan file for all blocks. To
304         load back floorplan file for each  block,  one  method  is  source  the
305         floorplan.tcl  Tcl  script, the other method is use set_constraint_map-
306         ping_file floorplan/mapfile and load_block_constraints -type  FLOORPLAN
307         -all_blocks.
308
309           prompt> write_floorplan -output floorplan -blocks [lsort -unique \
310              [get_attribute [get_cells -hierarchical -filter is_soft_macro] ref_name]
311
312         The  following  example  writes out the floorplan file and includes the
313         -add_def_only_objects  {cells}  and  -no_incremental  options  to  the
314         read_def command written to the floorplan file.
315
316           prompt> write_floorplan -output "dirName" -nosplit -verbosity low \
317              -def_version 5.8 -read_def_options {-add_def_only_objects {cells} -
                no_incremental}
318
319   SEE ALSO
320         compare_floorplans(2)
321         read_def(2)
322         write_def(2)
323
324                              Version S-2021.06-SP5
325              Copyright (c) 2022 Synopsys, Inc. All rights reserved.
326   icc2_shell>
327
```