LVR Command Reference

Layout Verification Reporter
Complete reference for all 441 commands

LVR 2.0 · Documentation Rev A
August 2026

Ramtera Design Automation, Inc.
Bangalore, India | Lewes, Delaware, USA
www.ramtera.com | support@ramtera.com

Legal Notice

> **Preliminary.** Argument default values and behavioural notes in this revision are being verified against the shipping build. Where a default stated here differs from observed behaviour, the observed behaviour is correct; please report the discrepancy to support@ramtera.com so it can be corrected in the next revision.

Ramtera Design Automation, Inc.
16192 Coastal Highway, Lewes, Delaware 19958, County of Sussex, USA
www.ramtera.com | support@ramtera.com

Revision History

Revision	Date	Product	Summary
Rev A	August 2026	2.0	First release of the consolidated command reference. All 441 commands documented with argument tables, defaults, mode applicability and worked examples. Adds the LRF rule language reference.

This document describes LVR 2.0 (build 2.0.0). Verify against the release notes for the build you are running.

Contents

1 About This Manual

This manual documents every command accepted by the LVR command interpreter. LVR (Layout Verification Reporter) is Ramtera's physical verification platform, covering DRC, LVS, MLVS, SVS, ERC, XOR, density, DFM, antenna and parasitic extraction from a single SVRF-native engine.

Commands are grouped into 27 functional categories. Within each category, entries appear in the order a script would normally use them rather than alphabetically; the alphabetical index in Chapter 3 gives the fastest route to a single command.

1.1 Notation

Convention	Meaning
`monospace`	A command name, argument value, file name or literal script text.
`<angle brackets>`	A placeholder to be replaced with a real value.
`required`	The command fails if the argument is omitted.
`optional`	The argument may be omitted; the stated default applies.
`Default`	The value in effect when the command is not issued at all.
`Applies to`	The <code>init_tool_db</code> modes in which the command has effect. A command issued in a mode it does not apply to is accepted and ignored.

1.2 Argument types

Type	Accepted form	Notes
`string`	Bare word or path	Quoting is not required and not interpreted. Paths may be absolute or relative to the invocation directory.
`integer`	Signed decimal	Boolean options take 1 for enabled and 0 for disabled.
`double`	Decimal, optionally in exponential form	Units are stated per command; they are never inferred.

1.3 Defaults

Where a command is not issued, the value shown in the Default column of its argument table is in effect. Integer switches default to 0 (disabled), doubles to 0.0, and string arguments are unset. A default of none on a required argument means there is no fallback: the command must supply a value.

A default of 0 on a limit or resource command generally means *unbounded* rather than *zero* — an uncapped marker count, or a partition size derived by LVR rather than fixed by the script. Each entry states which reading applies.

1.4 Comments

A line whose first non-blank character is # is a comment. There is no block comment form and no line-continuation character; each command occupies exactly one line.

2 Structure of a Command Script

An LVR script is an ordered sequence of commands. The order is not arbitrary: the interpreter builds state incrementally, and several commands are only meaningful once earlier state exists. A script that reads a layout before initializing the tool database, or runs a check before committing the database, will fail or silently produce nothing.

2.1 The seven phases

Every LVR script follows the same seven phases. Not all phases are present in every script, but those that are present must appear in this order.

#	Phase	Commands	Constraint
1	Initialize	`init_tool_db`	Must be the first command. Selects the mode.
2	Configure	`set_technode, set_top, set_num_cores, set_die_area`	Before any read, so parsing has the layer map and hierarchy root.
3	Read rules	`read_svrf, read_lrf`	Layer declarations before the rules that use them.
4	Read design	`read_gds, read_spice, read_verilog, read_lef, read_def`	After rules, so the layer map is known during parsing.
5	Commit	`load_db, load_def_db`	Boundary between read and run. Nothing may be read after it.
6	Run	`drc, lvs, erc, xor_check, run_density, pex`	Requires a committed database.
7	Report and review	`set_report_file_*, report_*, load_gui`	After the run. load_gui is normally last.

2.2 Worked example: GF12LP+ DRC

The following is a production DRC script for a GlobalFoundries 12LP+ certification layout, abridged in the rule-file block. It illustrates all seven phases.

```
# ---- phase 1: initialize ------------------------------------------
init_tool_db DRC

# ---- phase 2: configure -------------------------------------------
set_technode sg13g2
set_top chip_top
set_num_cores 1
set_die_area -2000 0 2000 2000

# ---- phase 3: read rules ------------------------------------------
# layer declarations first, then the rule files that use them
read_svrf ../rule/foundry_deck/Include/tech.layers.svrf
read_svrf ../rule/foundry_deck/Include/tech_boolean.drc.svrf
read_svrf ../rule/foundry_deck/Include/tech_derived_layers.drc.svrf
read_svrf ../rule/foundry_deck/Include/tech_beol_m1.drc.svrf
read_svrf ../rule/foundry_deck/Include/tech_crackstop.drc.svrf
```

```
# ... one read_svrf per include file of the foundry deck ...
read_svrf ../rule/foundry_deck/Include/tech_pad.drc.svrf

# ---- phase 4: read design ----------------------------------------
read_gds ../gds/chip_top.gds

# ---- phase 5: commit ---------------------------------------------
load_db DRC

# ---- phase 6: run ------------------------------------------------
no_compare 1
drc_max_error 100
drc chip_top

# ---- phase 7: report and review ----------------------------------
set_report_file_drc ./out/drc.rpt
report_drc
load_gui
```

2.3 Reading a foundry deck

A production foundry deck is delivered as a top-level file that includes several dozen others. LVR does not follow those includes itself; the script issues one read_svrf per include file. Order matters, because a rule file may reference a layer that an earlier file derived. Read them in the same order the deck's own top-level file lists them.

Where a deck contains files not required for the current run — a density deck during a geometry-only run, for example — comment out the corresponding read_svrf line rather than deleting it, so the script remains a faithful record of the deck it was built from.

2.4 Running headless

For regression and CI use, replace load_gui with cmd_line_only and enable the two determinism settings so that successive runs produce byte-comparable output:

```
cmd_line_only
set_testmode 1
disable_logger_timestamp
```

Invoke the script from the command line:

```
% LVR my_script.cmd
```

3 Alphabetical Command Index

All 441 commands, with the argument signature and the chapter in which the full entry appears.

Command	Arguments	Ch.
antenna	<top_cell:str>	8
cell_orient_scheme	<scheme:int>	7
check_allpins	<flag:int>	12
check_allrect	<flag:int>	12
check_allsegs	<flag:int>	12
check_allvias	<flag:int>	12
check_antenna	<flag:int>	12
check_antenna_cut	<flag:int>	12
check_area	<flag:int>	12
check_cut_enclosure	<flag:int>	12
check_cut_enclosureedge	<flag:int>	12
check_cut_enclosureedge_opposite	<flag:int>	12
check_cut_forbidden_spacing	<flag:int>	12
check_cut_on_centerline	<flag:int>	12
check_cut_short	<flag:int>	12
check_cut_spacing	<flag:int>	12
check_dangling_wire	<flag:int>	12
check_enclosure_to_joint	<flag:int>	12
check_eol_extn_spacing	<flag:int>	12
check_eol_keepout	<flag:int>	12
check_floating_patch	<flag:int>	12
check_invalid_cut	<flag:int>	12
check_loop	<flag:int>	12
check_metal_area_spacing	<flag:int>	12
check_metal_eol_spacing	<flag:int>	12
check_metal_forbidden_spacing	<flag:int>	12
check_metal_joint_corner_spacing	<flag:int>	12
check_metal_spacing	<flag:int>	12
check_min_cut	<flag:int>	12
check_min_step	<flag:int>	12
check_minimal_width	<flag:int>	12
check_nsmetal	<flag:int>	12

Command	Arguments	Ch.
check_off_grid	<flag:int>	12
check_open_check	<flag:int>	12
check_out_of_die	<flag:int>	12
check_powersegs	<flag:int>	12
check_rect_only	<flag:int>	12
check_shorts	<flag:int>	12
check_span_length_table	<flag:int>	12
check_table_cut_spacing	<flag:int>	12
check_table_prl_spacing	<flag:int>	12
check_unconnpin	<flag:int>	12
check_via_in_pin	<flag:int>	12
check_widthtable	<flag:int>	12
check_wrongway	<flag:int>	12
cmd_line_only	(no arguments)	4

Command	Arguments	Ch.
compare_spefs	<spef1:str> <spef2:str>	6
density	<top_cell:str>	8
density_layer	<layer:str>	13
density_pixel	<size:int>	13
density_window	<width:int> <height:int> <step_x:int> <step_y:int> <mode:int>	13
dfm	<top_cell:str>	8
disable_logger_timestamp	(no arguments)	14
drc	<top_cell:str>	8
drc_max_error	<count:int>	10
dump_nets	<path:str>	5
enable_logger_timestamp	(no arguments)	14
erc	<top_cell:str>	8
expected_file	<path:str>	5
gui_display_all_layers	(no arguments)	14
gui_display_single_layer	<layer:str>	14
help	(no arguments)	4
init_tool_db	<mode:str>	4
load_db	<mode:str>	4
load_def_db	(no arguments)	4
load_gui	(no arguments)	14
lvr_ground	<name:str>	15
lvr_kind	<name:str> <kind:str>	15
lvr_layer	<name:str> <number:int> <datatype:int>	15
lvr_node	<name:str>	15
lvr_pglayer	<name:str> <number:int>	15
lvr_pinlayer	<name:str> <number:int> <datatype:int>	15
lvr_portlayer	<name:str> <number:int> <datatype:int>	15
lvr_power	<name:str>	15
lvr_text	<name:str> <number:int> <texttype:int>	15
lvr_version	<version:int>	15
lvs	<top_cell:str>	8
lvs_check_macro	<flag:int>	11
lvs_check_matching	<flag:int>	11
lvs_check_open	<flag:int>	11
lvs_check_shorts	<flag:int>	11

Command	Arguments	Ch.
lvs_check_unconnected_pins	<flag:int>	11
lvs_consider_macro_pins	<flag:int>	11
lvs_detect_open	<flag:int>	11
lvs_detect_short	<flag:int>	11
lvs_enable_ports	<flag:int>	11
lvs_ground_pin	<name:str>	11
lvs_mapping_pin_to_terminal	<pin:str> <terminal:str>	11
lvs_max_error	<count:int>	10
lvs_option_max_error	<count:int>	10
lvs_power_pin	<name:str>	11
lvs_set_datc_flow	<flag:int>	11

Command	Arguments	Ch.
lvs_set_macros_cdl_file_name	<path:str>	11
lvs_set_schematic_file_name	<path:str>	11
lvs_set_spice_file_name	<path:str>	11
lvs_set_verilog_netlist_file_name	<path:str>	11
lvs_timeout	<seconds:int>	10
lvs_use_datc_flow	<flag:int>	11
lvs_waiver_macro_net	<macro:str> <net:str>	11
lvs_waiver_netlist_net	<net:str>	11
lvs_write_layout_from_db	<path:str>	6
lvs_write_schematic_from_spice	<in_spice:str> <out_path:str>	6
mlvs	<top_cell:str>	8
no_cells	<flag:int>	7
no_compare	<flag:int>	7
no_obs	<flag:int>	7
no_pwr	<flag:int>	7
no_rect	<flag:int>	7
no_vias	<flag:int>	7
pex	<top_cell:str>	8
query_rect	<x1:int> <y1:int> <x2:int> <y2:int>	7
read_def	<path:str>	5
read_gds	<path:str>	5
read_gds2	<path:str>	5
read_lef	<path:str>	5
read_lrf	<path:str>	5
read_spice	<path:str>	5
read_svrf	<path:str>	5
read_verilog	<path:str>	5
rep_antenna	(no arguments)	9
rep_density	(no arguments)	9
report_antenna	(no arguments)	9
report_density	(no arguments)	9
report_dfm	(no arguments)	9
report_drc	(no arguments)	9
report_erc	(no arguments)	9
report_fastxor	(no arguments)	9

Command	Arguments	Ch.
report_lefdef_drc	(no arguments)	9
report_lvs	(no arguments)	9
report_mlvs	(no arguments)	9
report_svs	(no arguments)	9
report_xor	(no arguments)	9
run_antenna	(no arguments)	8
run_density	(no arguments)	8
run_dfm	(no arguments)	8
run_drc	(no arguments)	8
run_erc	(no arguments)	8
run_fastxor	(no arguments)	8

Command	Arguments	Ch.
run_lefdef_drc	(no arguments)	8
run_lvs	(no arguments)	8
run_qdrc	(no arguments)	8
run_svs	(no arguments)	8
run_xor	(no arguments)	8
set_debug_mode_3049	<level:int>	4
set_die_area	<x1:int> <y1:int> <x2:int> <y2:int>	7
set_erc_check_dangling_output	<flag:int>	27
set_erc_check_floating_bulk	<flag:int>	27
set_erc_check_floating_gate	<flag:int>	27
set_erc_check_multi_driver	<flag:int>	27
set_erc_check_power_short	<flag:int>	27
set_erc_check_short_to_ground	<flag:int>	27
set_erc_check_short_to_power	<flag:int>	27
set_erc_check_unconnected_input	<flag:int>	27
set_erc_check_undriven_power	<flag:int>	27
set_erc_max_error	<count:int>	10
set_erc_report_info	<flag:int>	27
set_gds_flow	<flag:int>	7
set_gui_xor_result	<flag:int>	14
set_halo_width	<width:int>	7
set_layer_map_file	<path:str>	5
set_lefdef_flow	<flag:int>	7
set_log_flush_time	<seconds:int>	4
set_lrf_flow	<flag:int>	30
set_lvs_abort_on_first_error	<flag:int>	10
set_lvs_alias_vgnd	<name:str>	18
set_lvs_alias_vnb	<name:str>	18
set_lvs_alias_vpb	<name:str>	18
set_lvs_alias_vpwr	<name:str>	18
set_lvs_assign_internal_nets	<flag:int>	22
set_lvs_cache_size_mb	<value:int>	25
set_lvs_cdl_export_before_run	<flag:int>	28
set_lvs_cell_depth	<value:int>	19
set_lvs_cell_filter	<pattern:str>	19

Command	Arguments	Ch.
set_lvs_check_all_cells	<flag:int>	19
set_lvs_check_top_only	<flag:int>	19
set_lvs_chunk_size	<value:int>	25
set_lvs_connect_all_metals	<flag:int>	22
set_lvs_connect_layer	<layer:str>	18
set_lvs_connect_power_global	<flag:int>	22
set_lvs_csv_file	<path:str>	17
set_lvs_db_file	<path:str>	17
set_lvs_dbu	<value:dou>	7
set_lvs_def_file	<path:str>	16
set_lvs_detailed_log_file	<path:str>	17

Command	Arguments	Ch.
set_lvs_device_report_file	<path:str>	17
set_lvs_enable_csv_export	<flag:int>	23
set_lvs_enable_db_export	<flag:int>	23
set_lvs_enable_marker_file	<flag:int>	23
set_lvs_enable_waivers	<flag:int>	24
set_lvs_erc_report_file	<path:str>	17
set_lvs_error_on_mismatch	<flag:int>	10
set_lvs_error_on_open	<flag:int>	10
set_lvs_error_on_short	<flag:int>	10
set_lvs_error_on_unmatched_inst	<flag:int>	10
set_lvs_extract_bjt	<flag:int>	20
set_lvs_extract_bulk_terminal	<flag:int>	20
set_lvs_extract_capacitor	<flag:int>	20
set_lvs_extract_diode	<flag:int>	20
set_lvs_extract_ldd	<flag:int>	20
set_lvs_extract_nmos	<flag:int>	20
set_lvs_extract_nwell_terminal	<flag:int>	20
set_lvs_extract_pmos	<flag:int>	20
set_lvs_extract_resistor	<flag:int>	20
set_lvs_extract_sub_terminal	<flag:int>	20
set_lvs_extracted_netlist	<path:str>	17
set_lvs_fin_pitch	<value:dou>	20
set_lvs_flatten_cells	<flag:int>	19
set_lvs_flush_time	<value:int>	25
set_lvs_gds_export_before_run	<flag:int>	28
set_lvs_gds_file	<path:str>	16
set_lvs_global_net	<name:str>	18
set_lvs_ground_pin	<name:str>	18
set_lvs_ground_pin1	<name:str>	18
set_lvs_ground_pin2	<name:str>	18
set_lvs_ground_pin3	<name:str>	18
set_lvs_ground_pin4	<name:str>	18
set_lvs_hierarchical	<flag:int>	19
set_lvs_inst_match_timeout	<seconds:int>	10
set_lvs_internal_net_prefix	<prefix:str>	18

Command	Arguments	Ch.
set_lvs_l_tolerance	<value:dou>	20
set_lvs_layer_map_file	<path:str>	16
set_lvs_lef_file	<path:str>	16
set_lvs_log_file	<path:str>	17
set_lvs_lrf_file	<path:str>	16
set_lvs_macro_cell	<cell:str>	19
set_lvs_macro_timeout	<seconds:int>	10
set_lvs_macros_cdl_file	<path:str>	16
set_lvs_marker_file	<path:str>	17
set_lvs_match_allow_prefix	<flag:int>	21
set_lvs_match_bipartite	<flag:int>	21

Command	Arguments	Ch.
set_lvs_match_by_name	<flag:int>	21
set_lvs_match_by_topology	<flag:int>	21
set_lvs_match_case_sensitive	<flag:int>	21
set_lvs_match_ignore_bulk	<flag:int>	21
set_lvs_match_min_percentage	<value:int>	21
set_lvs_match_prefix	<prefix:str>	21
set_lvs_match_report_file	<path:str>	17
set_lvs_match_report_unmatched	<flag:int>	21
set_lvs_match_sort_based	<flag:int>	21
set_lvs_match_symmetric_sd	<flag:int>	21
set_lvs_max_error	<count:int>	10
set_lvs_max_error_per_macro	<count:int>	10
set_lvs_max_error_per_net	<count:int>	10
set_lvs_memory_limit_mb	<value:int>	25
set_lvs_net_report_file	<path:str>	17
set_lvs_num_threads	<value:int>	25
set_lvs_only_cell	<cell:str>	19
set_lvs_open_check_all_nets	<flag:int>	22
set_lvs_open_check_schematic	<flag:int>	22
set_lvs_open_max_per_net	<count:int>	10
set_lvs_open_net	<net:str>	22
set_lvs_open_report_file	<path:str>	17
set_lvs_open_report_path	<flag:int>	22
set_lvs_open_use_via	<flag:int>	22
set_lvs_power_pin	<name:str>	18
set_lvs_power_pin1	<name:str>	18
set_lvs_power_pin2	<name:str>	18
set_lvs_power_pin3	<name:str>	18
set_lvs_power_pin4	<name:str>	18
set_lvs_power_pin5	<name:str>	18
set_lvs_power_pin6	<name:str>	18
set_lvs_report_all_errors	<flag:int>	23
set_lvs_report_coordinates	<flag:int>	23
set_lvs_report_device_counts	<flag:int>	23
set_lvs_report_erc	<flag:int>	23

Command	Arguments	Ch.
set_lvs_report_fail_cells	<flag:int>	23
set_lvs_report_file	<path:str>	17
set_lvs_report_match_percent	<flag:int>	23
set_lvs_report_net_names	<flag:int>	23
set_lvs_report_opens	<flag:int>	23
set_lvs_report_pass_cells	<flag:int>	23
set_lvs_report_shorts	<flag:int>	23
set_lvs_report_timing	<flag:int>	23
set_lvs_report_unconnected	<flag:int>	23
set_lvs_rtree_node_size	<value:int>	25
set_lvs_short_check_all_layers	<flag:int>	22

Command	Arguments	Ch.
set_lvs_short_layer	<layer:str>	22
set_lvs_short_max_per_layer	<count:int>	10
set_lvs_short_report_file	<path:str>	17
set_lvs_short_report_nets	<flag:int>	22
set_lvs_short_use_rtree	<flag:int>	22
set_lvs_signature_file	<path:str>	16
set_lvs_skill_callback	<procedure:str>	28
set_lvs_skill_integration	<flag:int>	28
set_lvs_skip_cell	<cell:str>	19
set_lvs_skip_empty_cells	<flag:int>	19
set_lvs_skip_fill_cells	<flag:int>	19
set_lvs_spice_file	<path:str>	16
set_lvs_spice_file1	<path:str>	16
set_lvs_spice_file2	<path:str>	16
set_lvs_summary_file	<path:str>	17
set_lvs_summary_only	<flag:int>	23
set_lvs_svrf_file	<path:str>	16
set_lvs_tech_file	<path:str>	16
set_lvs_tech_node	<node:str>	7
set_lvs_top_cdl_file	<path:str>	16
set_lvs_top_cell	<cell:str>	19
set_lvs_total_timeout	<seconds:int>	10
set_lvs_unit	<unit:str>	7
set_lvs_use_cache	<flag:int>	25
set_lvs_use_dbu	<flag:int>	7
set_lvs_use_legacy_extraction	<flag:int>	20
set_lvs_use_pins_for_nets	<flag:int>	22
set_lvs_use_rtree	<flag:int>	25
set_lvs_use_svrf_connect	<flag:int>	22
set_lvs_use_svrf_device_rules	<flag:int>	20
set_lvs_verbose	<flag:int>	23
set_lvs_verilog_file	<path:str>	16
set_lvs_verilog_file1	<path:str>	16
set_lvs_verilog_file2	<path:str>	16
set_lvs_virtuoso_cell	<cell:str>	28

Command	Arguments	Ch.
set_lvs_virtuoso_lib	<library:str>	28
set_lvs_virtuoso_markers	<flag:int>	28
set_lvs_virtuoso_view	<view:str>	28
set_lvs_w_tolerance	<value:dou>	20
set_lvs_waiver_audit_trail	<flag:int>	24
set_lvs_waiver_expiry_check	<flag:int>	24
set_lvs_waiver_file	<path:str>	16
set_lvs_waiver_owner	<name:str>	24
set_lvs_waiver_spatial_filter	<flag:int>	24
set_lvs_waiver_wildcard	<flag:int>	24
set_lvs_warning_as_error	<flag:int>	10

Command	Arguments	Ch.
set_mlvs_report_file	<path:str>	17
set_num_bins	<count:int>	7
set_num_cores	<count:int>	7
set_num_cpu	<count:int>	7
set_num_layers	<count:int>	7
set_num_regions	<count:int>	7
set_perform_lrf_check	<flag:int>	4
set_perform_lvs	<flag:int>	26
set_perform_lvs_antenna	<flag:int>	26
set_perform_lvs_datc_flow	<flag:int>	26
set_perform_lvs_device_extract	<flag:int>	26
set_perform_lvs_erc	<flag:int>	26
set_perform_lvs_ground_check	<flag:int>	26
set_perform_lvs_macro	<flag:int>	26
set_perform_lvs_matching	<flag:int>	26
set_perform_lvs_opens	<flag:int>	26
set_perform_lvs_parallel_reduce	<flag:int>	26
set_perform_lvs_port_check	<flag:int>	26
set_perform_lvs_power_check	<flag:int>	26
set_perform_lvs_property_check	<flag:int>	26
set_perform_lvs_series_reduce	<flag:int>	26
set_perform_lvs_shorts	<flag:int>	26
set_perform_lvs_soft_connect	<flag:int>	26
set_perform_lvs_svs	<flag:int>	26
set_perform_lvs_unconnected_pin	<flag:int>	26
set_perform_lvs_well_check	<flag:int>	26
set_perform_svrf_device_parse	<flag:int>	26
set_pex_cap_unit	<unit:str>	29
set_pex_cg_max_iter	<value:int>	29
set_pex_cg_tolerance	<value:dou>	29
set_pex_dbu_to_um	<value:dou>	29
set_pex_design_name	<name:str>	29
set_pex_dump_rc	<flag:int>	29
set_pex_em_verbose_segments	<value:int>	29
set_pex_em_violation_ratio	<value:dou>	29

Command	Arguments	Ch.
set_pex_exclude_layers	<layers:str>	29
set_pex_exclude_nets	<pattern:str>	29
set_pex_extract_bump	<flag:int>	29
set_pex_extract_capacitance	<flag:int>	29
set_pex_extract_coupling_cap	<flag:int>	29
set_pex_extract_fringe_cap	<flag:int>	29
set_pex_extract_ground_cap	<flag:int>	29
set_pex_extract_resistance	<flag:int>	29
set_pex_extract_tap	<flag:int>	29
set_pex_extract_via_resistance	<flag:int>	29
set_pex_flush_time	<value:int>	29

Command	Arguments	Ch.
set_pex_include_layers	<layers:str>	29
set_pex_inter_net_coupling_only	<flag:int>	29
set_pex_ir_drop_threshold_mv	<value:dou>	29
set_pex_ir_verbose_nodes	<value:int>	29
set_pex_ladder_seg_len_um	<value:dou>	29
set_pex_log_file	<path:str>	29
set_pex_max_coupling_dist_um	<value:dou>	29
set_pex_max_coupling_errors	<count:int>	10
set_pex_max_em_errors	<count:int>	10
set_pex_max_errors	<count:int>	10
set_pex_max_ir_errors	<count:int>	10
set_pex_max_ladder_segs	<value:int>	29
set_pex_min_cap_ff	<value:dou>	29
set_pex_min_coupling_cap_ff	<value:dou>	29
set_pex_min_resistance_ohm	<value:dou>	29
set_pex_net_filter	<pattern:str>	29
set_pex_num_threads	<value:int>	29
set_pex_rc_reduction	<flag:int>	29
set_pex_report_dspf	<path:str>	29
set_pex_report_em	<path:str>	29
set_pex_report_file	<path:str>	29
set_pex_report_ir	<path:str>	29
set_pex_report_markers	<path:str>	29
set_pex_report_sdf	<path:str>	29
set_pex_report_spef	<path:str>	29
set_pex_res_unit	<unit:str>	29
set_pex_run_em	<flag:int>	29
set_pex_run_extraction	<flag:int>	29
set_pex_run_ir_drop	<flag:int>	29
set_pex_skip_power_nets	<flag:int>	29
set_pex_spef_vendor	<vendor:str>	29
set_pex_spef_version	<version:str>	29
set_pex_supply_voltage	<value:dou>	29
set_pex_tech_file	<path:str>	29
set_pex_time_unit	<unit:str>	29

Command	Arguments	Ch.
set_pex_verbosity	<value:int>	29
set_report_file_antenna	<path:str>	9
set_report_file_density	<path:str>	9
set_report_file_dfm	<path:str>	9
set_report_file_drc	<path:str>	9
set_report_file_erc	<path:str>	9
set_report_file_fastxor	<path:str>	9
set_report_file_lefdef_drc	<path:str>	9
set_report_file_lvs	<path:str>	9
set_report_file_mlvs	<path:str>	9
set_report_file_svs	<path:str>	9

Command	Arguments	Ch.
set_report_file_xor	<path:str>	9
set_sch_file	<path:str>	5
set_svs1_spice	<path:str>	5
set_svs1_verilog	<path:str>	5
set_svs2_spice	<path:str>	5
set_svs2_verilog	<path:str>	5
set_svs_report_file	<path:str>	17
set_technode	<node:str>	4
set_testmode	<flag:int>	4
set_tool_db	<mode:str>	4
set_top	<cell:str>	4
set_xor1_gds	<path:str>	5
set_xor2_gds	<path:str>	5
sky130_to_spice	<in_file:str> <out_file:str>	6
split_cdl_asap7	<in_cdl:str> <out_top:str> <out_macros:str> <top:str>	6
split_cdl_gpdk	<in_cdl:str> <out_top:str> <out_macros:str> <top:str>	6
split_cdl_ng15	<in_cdl:str> <out_top:str> <out_macros:str> <top:str>	6
split_cdl_ng45	<in_cdl:str> <out_top:str> <out_macros:str> <top:str>	6
split_cdl_saed32	<in_cdl:str> <out_top:str> <out_macros:str> <top:str>	6
split_cdl_sky130	<in_cdl:str> <out_top:str> <out_macros:str> <top:str>	6
split_cdl_xh018	<in_cdl:str> <out_top:str> <out_macros:str> <top:str>	6
svs	<top_cell:str>	8
test_lvs	<layout:str> <schematic:str>	6
use_max	<flag:int>	7
verilog2sch	<in_verilog:str> <out_sch:str>	6
write_gds	<path:str>	5
xor_check	<top_cell:str>	8

4 Initialization and Database Management

Every LVR command script begins here. **init_tool_db** creates the in-memory tool database and selects the operating mode; that mode determines which rule constructs are honoured, which engines are armed, and which report writers are available. No read_* command may be issued before init_tool_db, and no run_* or check command may be issued before load_db.

help

Prints the built-in command list to the console. Issued with no arguments. Useful for confirming that a build supports a given command before committing a script to a regression queue.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: all

```
help
```

See also: —

init_tool_db

Creates the tool database and sets the operating mode. This must be the first command in every script. The mode argument arms the corresponding engine and selects the rule-file dialect that subsequent read_svrf and read_lrf calls will be interpreted under. Re-issuing init_tool_db discards the existing database.

Argument	Type	Required	Default	Notes
mode	string	required	none	Operating mode: DRC, LVS, MLVS, SVS, ERC, XOR, FASTXOR, DFM, ANTENNA, DENSITY, PEX

Applies to: all

```
init_tool_db DRC
```

See also: load_db, set_tool_db, set_top

set_tool_db

Switches the active database context to a database that has already been initialized. Used in multi-engine scripts that run, for example, DRC and then antenna checks against the same loaded geometry without re-reading the GDS.

Argument	Type	Required	Default	Notes
mode	string	required	none	Mode of an existing database

Applies to: all

```
set_tool_db ANTENNA
```

See also: init_tool_db, load_db

load_db

Commits all data read so far into the internal database and builds the geometry, layer and hierarchy structures the engines operate on. This is the boundary between the read phase and the run phase: all read_* and layer-mapping commands must precede it, and all run_*, check and report commands must follow it.

Argument	Type	Required	Default	Notes
mode	string	required	none	Must match the init_tool_db mode

Applies to: all

```
load_db DRC
```

See also: init_tool_db, read_gds, read_svrf

load_def_db

Commits a LEF/DEF design into the internal database. This is the LEF/DEF equivalent of load_db and is used after read_lef and read_def in the LEF/DEF DRC flow.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: LEF/DEF flows

```
load_def_db
```

See also: read_lef, read_def, set_lefdef_flow, run_lefdef_drc

set_top

Names the top cell of the design. The value must match a structure name present in the GDS or OASIS file exactly, including case. Setting it early lets LVR pre-size its hierarchy tables and gives every later command a consistent hierarchy root. A mismatch here is the most common cause of an empty run.

Argument	Type	Required	Default	Notes
cell	string	required	none	Top cell name as it appears in the layout

Applies to: all

```
set_top chip_top
```

See also: read_gds, set_lvs_top_cell

set_technode

Selects the technology node. This drives the default layer map, database unit, and device-recognition defaults. Set it immediately after init_tool_db and before any rule or layout read, since the layer map is consulted while parsing.

Argument	Type	Required	Default	Notes
node	string	required	none	Technology identifier, e.g. sky130, sg13g2, generic22, generic12, gf180mcu, asap7, ng45

Applies to: all

```
set_technode sg13g2
```

See also: set_layer_map_file, lvr_node

set_debug_mode_3049

Enables internal diagnostic tracing. Intended for use when working with Ramtera support on a reproducer; the output format is not stable between releases and should not be parsed by scripts.

Argument	Type	Required	Default	Notes
level	integer	optional	0	0 disables, higher values increase internal tracing

Applies to: all

```
set_debug_mode_3049 1
```

See also: set_lvs_verbose, set_pex_verbosity

set_testmode

Enables regression test mode, which suppresses timestamps and other run-varying fields from reports so that outputs can be diffed byte-for-byte between runs. Recommended for every automated regression job.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables regression test mode

Applies to: all

```
set_testmode 1
```

See also: disable_logger_timestamp, expected_file

set_perform_lrf_check

Validates the rule file for syntactic and semantic correctness before the run begins, rather than failing partway through. Costs a little startup time and is worth enabling whenever a rule deck has just been edited.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables rule-file validation

Applies to: all

```
set_perform_lrf_check 1
```

See also: read_lrf, read_svrf

cmd_line_only

Forces headless operation. The GUI is not initialized even if load_gui is later encountered, and no X11 or Wayland display is required. Use this in CI pipelines and on compute farm nodes.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: all

```
cmd_line_only
```

See also: load_gui

set_log_flush_time

Sets how often the log buffer is flushed to disk. A small value gives near real-time progress in a tailed log at some I/O cost; the default lets the logger flush when its buffer fills, which is faster for long batch runs.

Argument	Type	Required	Default	Notes
seconds	integer	optional	0	Flush interval; 0 means flush on buffer fill

Applies to: all

```
set_log_flush_time 5
```

See also: set_lvs_flush_time, set_pex_flush_time

5 Input File Reading

All reads occur between init_tool_db and load_db. Rule files are read before layout so that layer definitions and derivations are known when geometry is ingested. read_svrf is repeatable and is normally issued once per include file of a foundry deck, in the order the deck's own top-level file includes them.

read_svrf

Reads an SVRF rule file. The command is repeatable and is the normal way to load a production foundry deck: issue one call per include file, in the same order the deck's top-level file includes them, since layer derivations depend on declaration order. Layer declaration files must be read before the rule files that reference those layers.

Argument	Type	Required	Default	Notes
path	string	required	none	Path to an SVRF rule file

Applies to: all

```
read_svrf ../rule/foundry_deck/Include/tech.layers.svrf
read_svrf ../rule/foundry_deck/Include/tech_boolean.drc.svrf
read_svrf ../rule/foundry_deck/Include/tech_beol_m1.drc.svrf
```

See also: read_lrf, set_perform_lrf_check

read_lrf

Reads a rule file in LVR's native LRF format. LRF and SVRF decks may not be mixed within a single run. See the LRF Reference for the operator set.

Argument	Type	Required	Default	Notes
path	string	required	none	Path to an LRF rule file

Applies to: all

```
read_lrf ../rule/sky130_DRC001.lrf
```

See also: read_svrf, set_perform_lrf_check

read_gds

Reads a GDSII layout file hierarchically. Cell structures, instance placements, arrays, transforms, text and property records are preserved. Issue after the rule files so that the layer map is known during parsing, and before load_db.

Argument	Type	Required	Default	Notes
path	string	required	none	Path to a GDSII file

Applies to: all

```
read_gds ../gds/chip_top.gds
```

See also: read_gds2, write_gds, set_top, set_layer_map_file

read_gds2

Reads a second GDSII file as the comparison side of a two-layout flow. Used by XOR and FastXOR to establish the reference against which the primary layout is differenced.

Argument	Type	Required	Default	Notes
path	string	required	none	Path to a second GDSII file

Applies to: XOR, FASTXOR

```
read_gds2 ../gds/revB.gds
```

See also: read_gds, set_xor1_gds, set_xor2_gds, run_xor

read_lef

Reads a LEF technology and cell library file. Repeatable, so a technology LEF and one or more macro LEFs can be loaded in sequence. Required before read_def.

Argument	Type	Required	Default	Notes
path	string	required	none	Path to a LEF file

Applies to: LEF/DEF flows

```
read_lef ../lef/tech.lef
read_lef ../lef/stdcells.lef
```

See also: read_def, set_lefdef_flow, load_def_db

read_def

Reads a DEF placement and routing file. All applicable LEF files must be read first, since DEF references macros and layers defined in LEF.

Argument	Type	Required	Default	Notes
path	string	required	none	Path to a DEF file

Applies to: LEF/DEF flows

```
read_def ../def/design.def
```

See also: read_lef, load_def_db, run_lefdef_drc

read_spice

Reads a SPICE or CDL source netlist as the schematic side of a comparison. Subcircuit definitions, device statements, parameter assignments and m= multipliers are parsed; multiplier expansion is applied during load_db.

Argument	Type	Required	Default	Notes
path	string	required	none	Path to a SPICE or CDL netlist

Applies to: LVS, MLVS, SVS, ERC

```
read_spice ../netlist/DiffAmp.cdl
```

See also: set_lvs_spice_file, lvs_set_spice_file_name, read_verilog

`read_verilog`

Reads a structural Verilog netlist as the schematic side of a comparison. The module hierarchy must correspond to the layout hierarchy for a hierarchical comparison; use set_lvs_flatten_cells where it does not.

Argument	Type	Required	Default	Notes
`path`	string	required	`none`	Path to a Verilog netlist

Applies to: LVS, MLVS, SVS

```
read_verilog ../netlist/design.v
```

See also: read_spice, verilog2sch, set_lvs_verilog_file

`write_gds`

Writes the current database contents to a GDSII file, including any geometry created by rule-file derivations or by edits made in the AMS Layout Editor. Hierarchy is preserved.

Argument	Type	Required	Default	Notes
`path`	string	required	`none`	Output GDSII path

Applies to: all

```
write_gds ./out/derived.gds
```

See also: read_gds, lvs_write_layout_from_db

`dump_nets`

Writes the extracted connectivity to a text file: one record per net with its member shapes, layers and connecting devices. Primarily a debug aid for diagnosing shorts and opens, and for confirming that a connect statement behaved as intended.

Argument	Type	Required	Default	Notes
`path`	string	required	`none`	Output path for the net dump

Applies to: LVS, MLVS, SVS, ERC

```
dump_nets ./out/nets.txt
```

See also: set_lvs_net_report_file, set_lvs_extracted_netlist

`expected_file`

Names a golden results file for the current run. LVR compares its own results against this file and reports the delta, which is the mechanism used by the regression suite to detect correlation drift.

Argument	Type	Required	Default	Notes
`path`	string	required	`none`	Path to a golden results file

Applies to: all

```
expected_file ../golden/gcd1_DRC001.exp
```

See also: set_testmode, no_compare

set_layer_map_file

Supplies an explicit layer map, overriding the default map implied by set_technode. Required when a design uses a non-standard layer/datatype assignment, and when merging GDS from multiple sources.

Argument	Type	Required	Default	Notes
path	string	required	none	Path to a layer map file

Applies to: all

```
set_layer_map_file ../maps/generic22.map
```

See also: set_technode, lvr_layer, lvr_kind

set_sch_file

Names the schematic file for the comparison. A legacy alias retained for compatibility with older scripts; new scripts should prefer set_lvs_spice_file or set_lvs_verilog_file.

Argument	Type	Required	Default	Notes
path	string	required	none	Path to the schematic netlist

Applies to: LVS, MLVS, SVS

```
set_sch_file ../netlist/top.cdl
```

See also: set_lvs_spice_file, set_lvs_verilog_file

set_xor1_gds

Names the first of the two layouts to be differenced. Together with set_xor2_gds this is the preferred way to set up an XOR run, since it makes the pairing explicit in the script.

Argument	Type	Required	Default	Notes
path	string	required	none	First layout of the XOR pair

Applies to: XOR, FASTXOR

```
set_xor1_gds ../gds/revA.gds
```

See also: set_xor2_gds, run_xor, run_fastxor

set_xor2_gds

Names the second of the two layouts to be differenced. Geometry present in one layout but not the other is reported as an XOR marker.

Argument	Type	Required	Default	Notes
path	string	required	none	Second layout of the XOR pair

Applies to: XOR, FASTXOR

```
set_xor2_gds ../gds/revB.gds
```

See also: set_xor1_gds, run_xor, set_gui_xor_result

set_svs1_spice

Names the first SPICE netlist of a schematic-versus-schematic comparison. SVS compares two netlists directly with no layout involved, which is the fastest way to validate a netlist transformation such as a synthesis or ECO step.

Argument	Type	Required	Default	Notes
path	string	required	none	First SPICE netlist

Applies to: SVS

```
set_svs1_spice ../netlist/golden.cdl
```

See also: set_svs2_spice, run_svs

set_svs2_spice

Names the second SPICE netlist of a schematic-versus-schematic comparison.

Argument	Type	Required	Default	Notes
path	string	required	none	Second SPICE netlist

Applies to: SVS

```
set_svs2_spice ../netlist/revised.cdl
```

See also: set_svs1_spice, run_svs

set_svs1_verilog

Names the first Verilog netlist of a schematic-versus-schematic comparison. Verilog and SPICE sides may be mixed, which allows a gate-level Verilog netlist to be compared against a transistor-level CDL.

Argument	Type	Required	Default	Notes
path	string	required	none	First Verilog netlist

Applies to: SVS

```
set_svs1_verilog ../netlist/golden.v
```

See also: set_svs2_verilog, run_svs

set_svs2_verilog

Names the second Verilog netlist of a schematic-versus-schematic comparison.

Argument	Type	Required	Default	Notes
path	string	required	none	Second Verilog netlist

Applies to: SVS

```
set_svs2_verilog ../netlist/revised.v
```

See also: set_svs1_verilog, run_svs

6 Netlist Preparation Utilities

These commands transform third-party netlist formats into the flat, single-hierarchy CDL that the LVS comparison engine consumes. They run standalone: they do not require a loaded database and produce files on disk that a later script reads back with read_spice. The split_cdl_* family shares one signature and differs only in the device-model conventions of the target PDK.

split_cdl_asap7

Splits an ASAP7 CDL netlist into a top-level netlist and a macro library, applying ASAP7 device-model naming conventions. The two outputs are read back with set_lvs_top_cdl_file and set_lvs_macros_cdl_file.

Argument	Type	Required	Default	Notes
in_cdl	string	required	none	Source CDL
out_top	string	required	none	Output top-level CDL
out_macros	string	required	none	Output macro CDL
top	string	required	none	Top cell name

Applies to: LVS, MLVS

```
split_cdl_asap7 in.cdl top.cdl macros.cdl my_top
```

See also: set_lvs_top_cdl_file, set_lvs_macros_cdl_file

split_cdl_gpdk

Splits a Cadence GPDK CDL netlist into top-level and macro netlists using GPDK device-model conventions.

Argument	Type	Required	Default	Notes
in_cdl	string	required	none	Source CDL
out_top	string	required	none	Output top-level CDL
out_macros	string	required	none	Output macro CDL
top	string	required	none	Top cell name

Applies to: LVS, MLVS

```
split_cdl_gpdk in.cdl top.cdl macros.cdl my_top
```

See also: split_cdl_asap7

split_cdl_ng15

Splits a NanGate 15 nm CDL netlist into top-level and macro netlists.

Argument	Type	Required	Default	Notes
in_cdl	string	required	none	Source CDL
out_top	string	required	none	Output top-level CDL
out_macros	string	required	none	Output macro CDL
top	string	required	none	Top cell name

Applies to: LVS, MLVS

```
split_cdl_ng15 in.cdl top.cdl macros.cdl my_top
```

See also: split_cdl_asap7

split_cdl_ng45

Splits a NanGate 45 nm CDL netlist into top-level and macro netlists.

Argument	Type	Required	Default	Notes
in_cdl	string	required	none	Source CDL
out_top	string	required	none	Output top-level CDL
out_macros	string	required	none	Output macro CDL
top	string	required	none	Top cell name

Applies to: LVS, MLVS

```
split_cdl_ng45 in.cdl top.cdl macros.cdl my_top
```

See also: split_cdl_asap7

split_cdl_saed32

Splits a Synopsys SAED32 CDL netlist into top-level and macro netlists.

Argument	Type	Required	Default	Notes
in_cdl	string	required	none	Source CDL
out_top	string	required	none	Output top-level CDL
out_macros	string	required	none	Output macro CDL
top	string	required	none	Top cell name

Applies to: LVS, MLVS

```
split_cdl_saed32 in.cdl top.cdl macros.cdl my_top
```

See also: split_cdl_asap7

split_cdl_sky130

Splits a SKY130 CDL netlist into top-level and macro netlists, handling the SKY130 four-terminal device conventions and the VPWR/VGND/VPB/VNB supply pin naming.

Argument	Type	Required	Default	Notes
in_cdl	string	required	none	Source CDL
out_top	string	required	none	Output top-level CDL
out_macros	string	required	none	Output macro CDL
top	string	required	none	Top cell name

Applies to: LVS, MLVS

```
split_cdl_sky130 in.cdl top.cdl macros.cdl gcd1_sky130hd
```

See also: sky130_to_spice, set_lvs_alias_vpwr

split_cdl_xh018

Splits an X-FAB XH018 CDL netlist into top-level and macro netlists.

Argument	Type	Required	Default	Notes
in_cdl	string	required	none	Source CDL
out_top	string	required	none	Output top-level CDL
out_macros	string	required	none	Output macro CDL
top	string	required	none	Top cell name

Applies to: LVS, MLVS

```
split_cdl_xh018 in.cdl top.cdl macros.cdl my_top
```

See also: split_cdl_asap7

sky130_to_spice

Converts a SKY130 netlist into the canonical SPICE form the comparison engine expects, normalizing device model names and terminal ordering.

Argument	Type	Required	Default	Notes
in_file	string	required	none	SKY130 source netlist
out_file	string	required	none	Output SPICE netlist

Applies to: LVS, MLVS, SVS

```
sky130_to_spice in.spice out.spice
```

See also: split_cdl_sky130, read_spice

verilog2sch

Converts a structural Verilog netlist into a schematic netlist suitable for comparison. Use where the reference is gate-level Verilog but the flow expects a CDL-style schematic side.

Argument	Type	Required	Default	Notes
in_verilog	string	required	none	Source Verilog netlist
out_sch	string	required	none	Output schematic netlist

Applies to: LVS, MLVS, SVS

```
verilog2sch design.v design.sch
```

See also: read_verilog, lvs_write_schematic_from_spice

test_lvs

Runs a self-contained LVS comparison of the two named inputs without requiring the full script setup. Intended for quick sanity checks and for the built-in regression suite, not for sign-off.

Argument	Type	Required	Default	Notes
layout	string	required	none	Layout-side input
schematic	string	required	none	Schematic-side input

Applies to: LVS

```
test_lvs layout.gds schematic.cdl
```

See also: run_lvs, lvs

lvs_write_layout_from_db

Writes the layout-extracted netlist from the database to a file. This is the netlist the comparison engine actually sees from the layout side, so it is the first thing to inspect when device counts do not match.

Argument	Type	Required	Default	Notes
path	string	required	none	Output netlist path

Applies to: LVS, MLVS

```
lvs_write_layout_from_db ./out/layout_extracted.cdl
```

See also: set_lvs_extracted_netlist, lvs_write_schematic_from_spice

`lvs_write_schematic_from_spice`

Writes the normalized schematic-side netlist derived from a SPICE input. Comparing this against the output of lvs_write_layout_from_db is the standard way to isolate whether a mismatch originates in extraction or in the source netlist.

Argument	Type	Required	Default	Notes
`in_spice`	string	required	`none`	Source SPICE netlist
`out_path`	string	required	`none`	Output netlist path

Applies to: LVS, MLVS

```
lvs_write_schematic_from_spice in.cdl ./out/schematic_norm.cdl
```

See also: lvs_write_layout_from_db

`compare_spefs`

Compares two SPEF parasitic files and reports per-net differences in resistance and capacitance. Used to correlate LVR PEX output against a reference extractor and to quantify the effect of an extraction setting change.

Argument	Type	Required	Default	Notes
`spef1`	string	required	`none`	First SPEF file
`spef2`	string	required	`none`	Second SPEF file

Applies to: PEX

```
compare_spefs golden.spef lvr.spef
```

See also: set_pex_report_spef, run_density

7 Flow Selection and Resources

These commands select the input flow and size the run. The flow switches are mutually exclusive: set exactly one of set_gds_flow or set_lefdef_flow. Resource commands should be set before load_db so that the partitioner can size its structures correctly.

set_gds_flow

Selects the GDS/OASIS input flow. Mutually exclusive with set_lefdef_flow. This is the flow used for all foundry sign-off decks.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 selects the GDS flow

Applies to: all

```
set_gds_flow 1
```

See also: set_lefdef_flow, read_gds

set_lefdef_flow

Selects the LEF/DEF input flow, which enables the check_* command family and the run_lefdef_drc engine. Mutually exclusive with set_gds_flow.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 selects the LEF/DEF flow

Applies to: LEF/DEF flows

```
set_lefdef_flow 1
```

See also: set_gds_flow, read_lef, read_def, run_lefdef_drc

set_num_cpu

Sets the number of CPUs made available to the run. On most systems this is a synonym for set_num_cores; where the two differ, set_num_cpu bounds the process and set_num_cores bounds the worker pool.

Argument	Type	Required	Default	Notes
count	integer	optional	1	Number of CPUs

Applies to: all

```
set_num_cpu 16
```

See also: set_num_cores, set_lvs_num_threads, set_pex_num_threads

set_num_cores

Sets the size of the worker pool used for tiled, multithreaded checking. Each worker owns a region of the layout and executes the rule set within it independently; boundary interactions are resolved by the halo mechanism. Scaling is close to linear up to the point where the design no longer partitions evenly.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`1`	Worker thread count

Applies to: all

```
set_num_cores 16
```

See also: set_num_regions, set_halo_width, set_num_cpu

set_num_regions

Sets the number of regions the layout is partitioned into. More regions than cores gives finer-grained load balancing on designs with uneven density, at the cost of more boundary work. Leaving this at the default lets LVR derive a partition from the die area and core count.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`0`	Region count; 0 lets LVR choose

Applies to: all

```
set_num_regions 64
```

See also: set_num_cores, set_die_area, set_halo_width

set_num_layers

Pre-sizes the internal layer tables. Rarely needed, since the count is derived from the layer declarations in the rule file; supply it only when working with a deck that generates a very large number of derived layers.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`0`	Layer count hint; 0 derives from the rule file

Applies to: all

```
set_num_layers 512
```

See also: lvr_layer, set_layer_map_file

set_num_bins

Sets the number of bins used for the marker distribution histograms shown in the GUI Histograms tab. Higher counts give finer spatial resolution of where violations cluster.

Argument	Type	Required	Default	Notes
count	integer	optional	0	Histogram bin count

Applies to: all

```
set_num_bins 100
```

See also: load_gui, set_num_regions

set_die_area

Declares the die bounding box in database units. This bounds the partitioner and defines the extent used by boundary-derived layers. Coordinates may be negative. If the box is smaller than the actual design, geometry outside it is not checked, so this is worth verifying whenever a run reports fewer markers than expected.

Argument	Type	Required	Default	Notes
x1	integer	required	none	Lower-left X in database units
y1	integer	required	none	Lower-left Y in database units
x2	integer	required	none	Upper-right X in database units
y2	integer	required	none	Upper-right Y in database units

Applies to: all

```
set_die_area -2000 0 2000 2000
```

See also: set_num_regions, query_rect

set_halo_width

Sets the overlap band each region reads beyond its own boundary so that rules spanning a region edge are evaluated correctly. It must be at least as large as the largest interaction distance in the rule deck; too small a halo produces missed violations at region boundaries, while too large a halo increases redundant work.

Argument	Type	Required	Default	Notes
width	integer	optional	0	Halo width in database units

Applies to: all

```
set_halo_width 5000
```

See also: set_num_regions, set_num_cores

`cell_orient_scheme`

Selects how instance orientations are composed through the hierarchy. The default follows the GDSII convention in which reflection is applied before rotation. Change this only when reading a database written by a tool with a different convention, since an incorrect setting mirrors or rotates whole sub-blocks.

Argument	Type	Required	Default	Notes
`scheme`	integer	optional	`0`	Orientation composition scheme

Applies to: all

```
cell_orient_scheme 1
```

See also: read_gds, set_top

`use_max`

Allows the run to use the maximum available resources rather than the conservative defaults. Appropriate on a dedicated machine, less so on a shared compute farm node.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 enables maximum-resource mode

Applies to: all

```
use_max 1
```

See also: set_num_cores, set_lvs_memory_limit_mb

`no_compare`

Suppresses comparison against a golden results file and reports LVR's own results only. Set this for any production run that is not part of the correlation regression.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 suppresses golden comparison

Applies to: all

```
no_compare 1
```

See also: expected_file, set_testmode

`no_cells`

Skips standard-cell instances during LEF/DEF processing. Used to isolate routing-layer violations from cell-internal ones when triaging a large DEF.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 skips cell processing

Applies to: LEF/DEF flows

```
no_cells 1
```

See also: no_obs, no_pwr, no_vias, no_rect

no_obs

Skips LEF OBS obstruction geometry. Useful when a library's obstruction modelling is known to be pessimistic and is masking real routing violations.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 skips obstructions

Applies to: LEF/DEF flows

```
no_obs 1
```

See also: no_cells, no_pwr

no_pwr

Excludes power and ground nets from processing. Because supply nets are large and highly connected, excluding them substantially shortens a signal-routing debug loop.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 skips power nets

Applies to: LEF/DEF flows

```
no_pwr 1
```

See also: set_lvs_skip_power_nets, check_powersegs

no_vias

Skips via geometry. Isolates metal-layer violations from cut-layer violations during triage.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 skips via processing

Applies to: LEF/DEF flows

```
no_vias 1
```

See also: check_allvias, check_invalid_cut

no_rect

Skips rectangle-only geometry during LEF/DEF processing.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 skips rectangle-only geometry

Applies to: LEF/DEF flows

```
no_rect 1
```

See also: check_allrect, check_rect_only

set_lvs_dbu

Sets the database unit used by the LVS engine, in database units per micron. Leave at the default to inherit the value from the layout file; override only when comparing databases written at different precisions.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Database units per micron

Applies to: LVS, MLVS, SVS, ERC

```
set_lvs_dbu 1000.0
```

See also: set_lvs_use_dbu, set_lvs_unit, set_pex_dbu_to_um

set_lvs_use_dbu

Forces use of the explicit value set by set_lvs_dbu instead of the value carried in the layout file.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 forces the explicit DBU value

Applies to: LVS, MLVS, SVS, ERC

```
set_lvs_use_dbu 1
```

See also: set_lvs_dbu

set_lvs_unit

Sets the length unit used in LVS report output. Affects presentation only, not the internal integer geometry.

Argument	Type	Required	Default	Notes
unit	string	optional	um	Length unit for reports

Applies to: LVS, MLVS, SVS, ERC

```
set_lvs_unit um
```

See also: set_lvs_dbu, set_pex_cap_unit

set_lvs_tech_node

Overrides the technology node for the LVS engine only, leaving the global set_technode value in place for the geometry engines.

Argument	Type	Required	Default	Notes
node	string	optional	none	Technology node for the LVS engine

Applies to: LVS, MLVS, SVS, ERC

```
set_lvs_tech_node sg13g2
```

See also: set_technode, set_lvs_tech_file

`query_rect`

Queries the database for all geometry intersecting the given rectangle, in database units, and reports what is found by layer. A debug aid for confirming that geometry was read at the coordinates expected, and for investigating a marker at a known location.

Argument	Type	Required	Default	Notes
`x1`	integer	required	`none`	Lower-left X
`y1`	integer	required	`none`	Lower-left Y
`x2`	integer	required	`none`	Upper-right X
`y2`	integer	required	`none`	Upper-right Y

Applies to: all

```
query_rect 0 0 10000 10000
```

See also: set_die_area, load_gui

8 Run Commands

Each engine is invoked either by its run_* form, which takes no argument and operates on the top cell set by set_top, or by its short form, which takes the top cell name directly. The two forms are equivalent; the short form is the one used throughout the shipped example scripts. All run commands require a prior load_db.

run_drc

Runs design rule checking on the loaded layout using the loaded rule deck. Work is distributed across the worker pool by region. Results are held in the results database until a report_* command writes them out or load_gui displays them.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: DRC

```
run_drc
```

See also: drc, report_drc, drc_max_error

drc

Runs design rule checking on the named top cell. Equivalent to run_drc with the top cell given inline.

Argument	Type	Required	Default	Notes
top_cell	string	required	none	Top cell name

Applies to: DRC

```
drc chip_top
```

See also: run_drc, set_top, drc_max_error

run_qdrc

Runs a reduced quick-DRC pass covering the geometry checks that can be evaluated without full connectivity extraction. Intended for fast iteration during layout editing, not for sign-off.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: DRC

```
run_qdrc
```

See also: run_drc

run_lvs

Runs layout versus schematic: extracts devices and connectivity from the layout, normalizes the schematic netlist, and compares the two graphs. Requires both a layout and a netlist to have been read.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: LVS

```
run_lvs
```

See also: lvs, report_lvs, set_perform_lvs

lvs

Runs layout versus schematic on the named top cell. Equivalent to run_lvs with the top cell given inline.

Argument	Type	Required	Default	Notes
top_cell	string	required	none	Top cell name

Applies to: LVS

```
lvs DiffAmp_NOFILL
```

See also: run_lvs, set_lvs_top_cell

mlvs

Runs macro LVS on the named top cell. Macro LVS treats hard macros as black boxes matched by pin interface rather than by internal topology, which is the correct mode for a top level that instantiates IP delivered without transistor-level netlists.

Argument	Type	Required	Default	Notes
top_cell	string	required	none	Top cell name

Applies to: MLVS

```
mlvs top_soc
```

See also: report_mlvs, lvs_check_macro, lvs_consider_macro_pins

run_svs

Runs schematic versus schematic, comparing the two netlists named by the set_svs* commands. No layout is involved.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: SVS

```
run_svs
```

See also: svs, set_svs1_spice, set_svs2_spice, report_svs

svs

Runs schematic versus schematic on the named top cell.

Argument	Type	Required	Default	Notes
top_cell	string	required	none	Top cell name

Applies to: SVS

```
svs gcd1
```

See also: run_svs, set_svs1_spice

run_erc

Runs electrical rule checking on the extracted connectivity: floating gates, shorts to supply, undriven nets, multiple drivers and the other conditions armed by the set_erc_check_* family.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: ERC

```
run_erc
```

See also: erc, report_erc, set_erc_check_floating_gate

erc

Runs electrical rule checking on the named top cell.

Argument	Type	Required	Default	Notes
top_cell	string	required	none	Top cell name

Applies to: ERC

```
erc gcd1_sky130hd
```

See also: run_erc, set_erc_max_error

run_xor

Runs a full-precision geometric XOR between the two layouts, reporting every region present in one and absent in the other. This is the exact comparison used for ECO verification and revision sign-off.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: XOR

```
run_xor
```

See also: xor_check, run_fastxor, set_xor1_gds, report_xor

xor_check

Runs a full-precision geometric XOR on the named top cell.

Argument	Type	Required	Default	Notes
top_cell	string	required	none	Top cell name

Applies to: XOR

```
xor_check gcd1_sky130hd
```

See also: run_xor, set_gui_xor_result

run_fastxor

Runs an accelerated XOR that reports whether and approximately where two layouts differ, trading exact marker geometry for speed. Appropriate as a screening pass across many cells; follow up with run_xor on any cell that reports a difference.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: FASTXOR

```
run_fastxor
```

See also: run_xor, report_fastxor

run_dfm

Runs design for manufacturability checks: recommended-rule and yield-detractor patterns that are advisory rather than blocking.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: DFM

```
run_dfm
```

See also: dfm, report_dfm, drc_max_error

dfm

Runs design for manufacturability checks on the named top cell.

Argument	Type	Required	Default	Notes
top_cell	string	required	none	Top cell name

Applies to: DFM

```
dfm top_soc
```

See also: run_dfm, report_dfm

run_antenna

Runs antenna ratio checking: accumulates conductor area connected to each gate at each metallization stage and flags nets whose antenna ratio exceeds the process limit.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: ANTENNA

```
run_antenna
```

See also: antenna, report_antenna, check_antenna

antenna

Runs antenna ratio checking on the named top cell.

Argument	Type	Required	Default	Notes
top_cell	string	required	none	Top cell name

Applies to: ANTENNA

```
antenna top_soc
```

See also: run_antenna, set_report_file_antenna

run_density

Runs metal and diffusion density checking. The die is divided into windows by density_window; the fill ratio of each window is measured against the rule minimum and maximum and out-of-range windows are reported as markers.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: DENSITY

```
run_density
```

See also: density, density_window, density_layer, report_density

density

Runs density checking on the named top cell.

Argument	Type	Required	Default	Notes
top_cell	string	required	none	Top cell name

Applies to: DENSITY

```
density analog_core
```

See also: run_density, density_window

pex

Runs parasitic extraction on the named top cell, producing resistance and capacitance for each net according to the set_pex_* settings, and writing SPEF, DSPF or SDF as configured.

Argument	Type	Required	Default	Notes
top_cell	string	required	none	Top cell name

Applies to: PEX

```
pex top_soc
```

See also: set_pex_run_extraction, set_pex_report_spef

run_lefdef_drc

Runs the LEF/DEF design rule checks armed by the check_* family against a design loaded with read_lef, read_def and load_def_db.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: LEF/DEF flows

```
run_lefdef_drc
```

See also: load_def_db, report_lefdef_drc, check_metal_spacing

9 Reporting

Each engine has a matched pair of commands: a set_report_file_* that names the output path and a report_* that writes it. Both follow the run. Reports are written in the Calibre-compatible ASCII RDB format where the engine has a Calibre analogue, which allows LVR output to be loaded directly into existing marker viewers and correlation scripts.

report_drc

Writes the DRC results held in the results database to the file named by set_report_file_drc. If no report file has been named, output goes to the console. Issue after the corresponding run command. DRC results are written in Calibre-compatible ASCII RDB format.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: DRC

```
report_drc
```

See also: set_report_file_drc, run_drc

report_lvs

Writes the LVS results held in the results database to the file named by set_report_file_lvs. If no report file has been named, output goes to the console. Issue after the corresponding run command.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: LVS

```
report_lvs
```

See also: set_report_file_lvs, run_lvs

report_mlvs

Writes the MLVS results held in the results database to the file named by set_report_file_mlvs. If no report file has been named, output goes to the console. Issue after the corresponding run command.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: MLVS

```
report_mlvs
```

See also: set_report_file_mlvs, run_mlvs

report_dfm

Writes the DFM results held in the results database to the file named by set_report_file_dfm. If no report file has been named, output goes to the console. Issue after the corresponding run command.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: DFM

```
report_dfm
```

See also: set_report_file_dfm, run_dfm

report_erc

Writes the ERC results held in the results database to the file named by set_report_file_erc. If no report file has been named, output goes to the console. Issue after the corresponding run command.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: ERC

```
report_erc
```

See also: set_report_file_erc, run_erc

report_xor

Writes the XOR results held in the results database to the file named by set_report_file_xor. If no report file has been named, output goes to the console. Issue after the corresponding run command.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: XOR

```
report_xor
```

See also: set_report_file_xor, run_xor

report_fastxor

Writes the FastXOR results held in the results database to the file named by set_report_file_fastxor. If no report file has been named, output goes to the console. Issue after the corresponding run command.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: FASTXOR

```
report_fastxor
```

See also: set_report_file_fastxor, run_fastxor

report_svs

Writes the SVS results held in the results database to the file named by set_report_file_svs. If no report file has been named, output goes to the console. Issue after the corresponding run command.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: SVS

```
report_svs
```

See also: set_report_file_svs, run_svs

report_antenna

Writes the antenna results held in the results database to the file named by set_report_file_antenna. If no report file has been named, output goes to the console. Issue after the corresponding run command.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: ANTENNA

```
report_antenna
```

See also: set_report_file_antenna, run_antenna

report_density

Writes the density results held in the results database to the file named by set_report_file_density. If no report file has been named, output goes to the console. Issue after the corresponding run command.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: DENSITY

```
report_density
```

See also: set_report_file_density, run_density

report_lefdef_drc

Writes the LEF/DEF DRC results held in the results database to the file named by set_report_file_lefdef_drc. If no report file has been named, output goes to the console. Issue after the corresponding run command.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: LEF/DEF DRC

```
report_lefdef_drc
```

See also: set_report_file_lefdef_drc, run_lefdef_drc

rep_antenna

Short alias for report_antenna, retained for compatibility with older scripts.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: ANTENNA

```
rep_antenna
```

See also: report_antenna

rep_density

Short alias for report_density, retained for compatibility with older scripts.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: DENSITY

```
rep_density
```

See also: report_density

set_report_file_drc

Names the output file for the DRC report. Set before report_drc. Existing files are overwritten.

Argument	Type	Required	Default	Notes
path	string	required	none	Output report path

Applies to: DRC

```
set_report_file_drc ./out/drc_summary.rpt
```

See also: report_drc

set_report_file_lvs

Names the output file for the LVS report. Set before report_lvs. Existing files are overwritten.

Argument	Type	Required	Default	Notes
path	string	required	none	Output report path

Applies to: LVS

```
set_report_file_lvs ./out/lvs_summary.rpt
```

See also: report_lvs

set_report_file_mlvs

Names the output file for the MLVS report. Set before report_mlvs. Existing files are overwritten.

Argument	Type	Required	Default	Notes
path	string	required	none	Output report path

Applies to: MLVS

```
set_report_file_mlvs ./out/mlvs_summary.rpt
```

See also: report_mlvs

set_report_file_dfm

Names the output file for the DFM report. Set before report_dfm. Existing files are overwritten.

Argument	Type	Required	Default	Notes
path	string	required	none	Output report path

Applies to: DFM

```
set_report_file_dfm ./out/dfm_summary.rpt
```

See also: report_dfm

set_report_file_erc

Names the output file for the ERC report. Set before report_erc. Existing files are overwritten.

Argument	Type	Required	Default	Notes
path	string	required	none	Output report path

Applies to: ERC

```
set_report_file_erc ./out/erc_summary.rpt
```

See also: report_erc

set_report_file_xor

Names the output file for the XOR report. Set before report_xor. Existing files are overwritten.

Argument	Type	Required	Default	Notes
path	string	required	none	Output report path

Applies to: XOR

```
set_report_file_xor ./out/xor_summary.rpt
```

See also: report_xor

set_report_file_fastxor

Names the output file for the FastXOR report. Set before report_fastxor. Existing files are overwritten.

Argument	Type	Required	Default	Notes
path	string	required	none	Output report path

Applies to: FASTXOR

```
set_report_file_fastxor ./out/fastxor_summary.rpt
```

See also: report_fastxor

set_report_file_svs

Names the output file for the SVS report. Set before report_svs. Existing files are overwritten.

Argument	Type	Required	Default	Notes
path	string	required	none	Output report path

Applies to: SVS

```
set_report_file_svs ./out/svs_summary.rpt
```

See also: report_svs

set_report_file_antenna

Names the output file for the antenna report. Set before report_antenna. Existing files are overwritten.

Argument	Type	Required	Default	Notes
path	string	required	none	Output report path

Applies to: ANTENNA

```
set_report_file_antenna ./out/antenna_summary.rpt
```

See also: report_antenna

set_report_file_density

Names the output file for the density report. Set before report_density. Existing files are overwritten.

Argument	Type	Required	Default	Notes
path	string	required	none	Output report path

Applies to: DENSITY

```
set_report_file_density ./out/density_summary.rpt
```

See also: report_density

set_report_file_lefdef_drc

Names the output file for the LEF/DEF DRC report. Set before report_lefdef_drc. Existing files are overwritten.

Argument	Type	Required	Default	Notes
path	string	required	none	Output report path

Applies to: LEF/DEF DRC

```
set_report_file_lefdef_drc ./out/lefdef_drc_summary.rpt
```

See also: report_lefdef_drc

10 Violation Limits

Limits bound how many markers an engine records. They exist to keep a run against a broken deck from filling a disk, not to shape sign-off results. A capped run produces a subset of the true marker set, so when correlating against a golden database the cap must either be lifted on both sides or accounted for explicitly.

`drc_max_error`

Caps the number of DRC violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`0`	Maximum markers; 0 means unlimited

Applies to: DRC

```
drc_max_error 1000
```

See also: drc_max_error, lvs_max_error

`lvs_max_error`

Caps the number of LVS violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`0`	Maximum markers; 0 means unlimited

Applies to: LVS

```
lvs_max_error 1000
```

See also: drc_max_error, lvs_max_error

`lvs_option_max_error`

Caps the number of LVS option-level violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`0`	Maximum markers; 0 means unlimited

Applies to: LVS

```
lvs_option_max_error 1000
```

See also: drc_max_error, lvs_max_error

set_erc_max_error

Caps the number of ERC violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`0`	Maximum markers; 0 means unlimited

Applies to: ERC

```
set_erc_max_error 1000
```

See also: drc_max_error, lvs_max_error

set_lvs_max_error

Caps the number of LVS violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`0`	Maximum markers; 0 means unlimited

Applies to: LVS, MLVS

```
set_lvs_max_error 1000
```

See also: drc_max_error, lvs_max_error

set_lvs_max_error_per_macro

Caps the number of per-macro LVS violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`0`	Maximum markers; 0 means unlimited

Applies to: LVS, MLVS

```
set_lvs_max_error_per_macro 1000
```

See also: drc_max_error, lvs_max_error

set_lvs_max_error_per_net

Caps the number of per-net LVS violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
count	integer	optional	0	Maximum markers; 0 means unlimited

Applies to: LVS, MLVS

```
set_lvs_max_error_per_net 1000
```

See also: drc_max_error, lvs_max_error

set_pex_max_errors

Caps the number of PEX violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
count	integer	optional	0	Maximum markers; 0 means unlimited

Applies to: PEX

```
set_pex_max_errors 1000
```

See also: drc_max_error, lvs_max_error

set_pex_max_coupling_errors

Caps the number of PEX coupling violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
count	integer	optional	0	Maximum markers; 0 means unlimited

Applies to: PEX

```
set_pex_max_coupling_errors 1000
```

See also: drc_max_error, lvs_max_error

set_pex_max_ir_errors

Caps the number of PEX IR-drop violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`0`	Maximum markers; 0 means unlimited

Applies to: PEX

```
set_pex_max_ir_errors 1000
```

See also: drc_max_error, lvs_max_error

set_pex_max_em_errors

Caps the number of PEX electromigration violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`0`	Maximum markers; 0 means unlimited

Applies to: PEX

```
set_pex_max_em_errors 1000
```

See also: drc_max_error, lvs_max_error

set_lvs_short_max_per_layer

Caps the number of per-layer short violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
`count`	integer	optional	`0`	Maximum markers; 0 means unlimited

Applies to: LVS, MLVS

```
set_lvs_short_max_per_layer 1000
```

See also: drc_max_error, lvs_max_error

set_lvs_open_max_per_net

Caps the number of per-net open violations recorded. Once the cap is reached the run stops recording further markers of that class, so a capped run reports a subset and its counts must not be compared directly against an uncapped golden result. Leave at the default for correlation work; set a cap only to bound a debug run against a badly broken deck.

Argument	Type	Required	Default	Notes
count	integer	optional	0	Maximum markers; 0 means unlimited

Applies to: LVS, MLVS

```
set_lvs_open_max_per_net 1000
```

See also: drc_max_error, lvs_max_error

lvs_timeout

Bounds the total wall-clock time of the LVS comparison. On expiry the run is abandoned and partial results are reported.

Argument	Type	Required	Default	Notes
seconds	integer	optional	0	Timeout; 0 means no limit

Applies to: LVS, MLVS

```
lvs_timeout 3600
```

See also: set_lvs_total_timeout, set_lvs_macro_timeout

set_lvs_total_timeout

Bounds the total wall-clock time of the entire LVS run including extraction, not just the comparison stage.

Argument	Type	Required	Default	Notes
seconds	integer	optional	0	Total timeout; 0 means no limit

Applies to: LVS, MLVS

```
set_lvs_total_timeout 7200
```

See also: lvs_timeout, set_lvs_inst_match_timeout

set_lvs_inst_match_timeout

Bounds the instance-to-instance graph matching stage. One hour is normally sufficient for a large hierarchical design; a timeout here usually indicates a genuine topology mismatch that has made the search space explode rather than a resource shortage.

Argument	Type	Required	Default	Notes
seconds	integer	optional	3600	Instance matching timeout

Applies to: LVS, MLVS

```
set_lvs_inst_match_timeout 3600
```

See also: set_lvs_macro_timeout, set_lvs_break_ambig_max

set_lvs_macro_timeout

Bounds the time spent resolving any single macro block. Increase for designs containing large or deeply nested hard macros.

Argument	Type	Required	Default	Notes
seconds	integer	optional	1	Per-macro resolution timeout

Applies to: LVS, MLVS

```
set_lvs_macro_timeout 60
```

See also: set_lvs_inst_match_timeout, lvs_check_macro

set_lvs_abort_on_first_error

Stops the run at the first error rather than continuing to collect the full error set. Useful for a fast fail in a gating CI job.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 aborts on the first error

Applies to: LVS, MLVS

```
set_lvs_abort_on_first_error 1
```

See also: set_lvs_warning_as_error, set_lvs_error_on_mismatch

set_lvs_warning_as_error

Promotes every LVS warning to error severity, which causes the run to report failure on conditions that would otherwise be advisory. Appropriate for a sign-off gate, not for early debug.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 promotes warnings to errors

Applies to: LVS, MLVS

```
set_lvs_warning_as_error 1
```

See also: set_lvs_error_on_mismatch

set_lvs_error_on_mismatch

Reports a netlist mismatch as an error rather than a warning.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 makes a mismatch an error

Applies to: LVS, MLVS

```
set_lvs_error_on_mismatch 1
```

See also: set_lvs_warning_as_error

set_lvs_error_on_unmatched_inst

Reports an instance present on one side but not the other as an error.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 makes an unmatched instance an error

Applies to: LVS, MLVS

```
set_lvs_error_on_unmatched_inst 1
```

See also: set_lvs_match_report_unmatched

set_lvs_error_on_open

Reports a detected open as an error rather than a warning.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 makes an open an error

Applies to: LVS, MLVS

```
set_lvs_error_on_open 1
```

See also: lvs_check_open, lvs_detect_open

set_lvs_error_on_short

Reports a detected short as an error rather than a warning.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 makes a short an error

Applies to: LVS, MLVS

```
set_lvs_error_on_short 1
```

See also: lvs_check_shorts, lvs_detect_short

11 LVS Options: Legacy Command Set

This family predates the set_lvs_* configuration commands and is retained so that existing scripts continue to run unchanged. Where a legacy command and a set_lvs_* command control the same behaviour, the later of the two in the script wins. New scripts should use the set_lvs_* forms.

`lvs_enable_ports`

Includes top-level ports in the comparison. Required for any block that will be integrated at a higher level, since without it a port naming or ordering error will not be caught.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 includes ports in the comparison

Applies to: LVS, MLVS, SVS

```
lvs_enable_ports 1
```

See also: set_perform_lvs_port_check, lvs_ignore_ports

`lvs_detect_short`

Enables detection of unintended connections between nets. Legacy form of lvs_check_shorts.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 enables short detection

Applies to: LVS, MLVS

```
lvs_detect_short 1
```

See also: lvs_check_shorts, set_lvs_short_report_nets

`lvs_detect_open`

Enables detection of nets that are electrically discontinuous. Legacy form of lvs_check_open.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 enables open detection

Applies to: LVS, MLVS

```
lvs_detect_open 1
```

See also: set_perform_lvs_opens, lvs_detect_open

lvs_power_pin

Declares a net as a power supply. Legacy form of set_lvs_power_pin.

Argument	Type	Required	Default	Notes
name	string	required	none	Power pin name

Applies to: LVS, MLVS, ERC

```
lvs_power_pin VPWR
```

See also: set_lvs_power_pin, lvs_ground_pin

lvs_ground_pin

Declares a net as a ground supply. Legacy form of set_lvs_ground_pin.

Argument	Type	Required	Default	Notes
name	string	required	none	Ground pin name

Applies to: LVS, MLVS, ERC

```
lvs_ground_pin VGND
```

See also: set_lvs_ground_pin, lvs_power_pin

lvs_check_shorts

Enables detection of unintended electrical connections between nets that the schematic keeps distinct. Arm the stage with set_perform_lvs_shorts.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LVS, MLVS

```
lvs_check_shorts 1
```

See also: set_perform_lvs_shorts, lvs_detect_short

lvs_check_open

Enables detection of nets that are electrically discontinuous in the layout. Arm the stage with set_perform_lvs_opens.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LVS, MLVS

```
lvs_check_open 1
```

See also: set_perform_lvs_opens, lvs_detect_open

lvs_check_macro

Checks macro-level connectivity, matching hard macros on their pin interface rather than their internal topology.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LVS, MLVS

```
lvs_check_macro 1
```

See also: mlvs, set_lvs_macro_cell

lvs_check_matching

Performs instance matching between the layout and schematic netlists.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LVS, MLVS

```
lvs_check_matching 1
```

See also: set_perform_lvs_matching, set_lvs_match_by_topology

lvs_check_unconnected_pins

Detects pins with no connection, which are often the visible symptom of a missing route or a mislabelled port.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LVS, MLVS

```
lvs_check_unconnected_pins 1
```

See also: set_perform_lvs_unconnected_pin, set_lvs_report_unconnected

lvs_consider_macro_pins

Includes macro pin connectivity in the comparison, so that hard macros are matched on their pin interface. Central to the MLVS flow.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 considers macro pins

Applies to: LVS, MLVS

```
lvs_consider_macro_pins 1
```

See also: mlvs, set_lvs_macro_cell

lvs_set_datc_flow

Enables DATC reference-benchmark flow compliance, aligning output format and check selection with the public DATC LVS benchmark definitions.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables DATC flow

Applies to: LVS, MLVS

```
lvs_set_datc_flow 1
```

See also: lvs_use_datc_flow, lvs_set_datc_flow

lvs_use_datc_flow

Activates the DATC flow for the current run.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 uses the DATC flow

Applies to: LVS, MLVS

```
lvs_use_datc_flow 1
```

See also: lvs_set_datc_flow

lvs_set_schematic_file_name

Names the schematic file. Legacy form; new scripts should use set_lvs_spice_file or set_lvs_verilog_file.

Argument	Type	Required	Default	Notes
path	string	required	none	Schematic file path

Applies to: LVS, MLVS

```
lvs_set_schematic_file_name top.cdl
```

See also: set_lvs_spice_file

lvs_set_spice_file_name

Names the SPICE netlist. Legacy form of set_lvs_spice_file.

Argument	Type	Required	Default	Notes
path	string	required	none	SPICE file path

Applies to: LVS, MLVS

```
lvs_set_spice_file_name DiffAmp.cdl
```

See also: set_lvs_spice_file

lvs_set_macros_cdl_file_name

Names the macro CDL library. Legacy form of set_lvs_macros_cdl_file.

Argument	Type	Required	Default	Notes
path	string	required	none	Macro CDL path

Applies to: LVS, MLVS

```
lvs_set_macros_cdl_file_name macros.cdl
```

See also: set_lvs_macros_cdl_file

lvs_set_verilog_netlist_file_name

Names the Verilog netlist. Legacy form of set_lvs_verilog_file.

Argument	Type	Required	Default	Notes
path	string	required	none	Verilog netlist path

Applies to: LVS, MLVS

```
lvs_set_verilog_netlist_file_name design.v
```

See also: set_lvs_verilog_file

lvs_mapping_pin_to_terminal

Maps a layout pin name onto a schematic terminal name where the two differ. Repeatable, one call per mapping. The usual reason to need this is a library whose layout pin labels were never aligned with its symbol terminal names.

Argument	Type	Required	Default	Notes
pin	string	required	none	Layout pin name
terminal	string	required	none	Schematic terminal name

Applies to: LVS, MLVS

```
lvs_mapping_pin_to_terminal VDD VPWR
```

See also: set_lvs_match_prefix

lvs_waiver_macro_net

Waives a specific net within a specific macro from LVS checking. Waivers are recorded with a SHA-256 content hash so that a waiver silently stops applying if the underlying geometry changes.

Argument	Type	Required	Default	Notes
macro	string	required	none	Macro cell name
net	string	required	none	Net name to waive

Applies to: LVS, MLVS

```
lvs_waiver_macro_net sram_1024x32 VDDA
```

See also: lvs_waiver_netlist_net, set_lvs_waiver_file, set_lvs_enable_waivers

`lvs_waiver_netlist_net`

Waives a named netlist net from LVS checking.

Argument	Type	Required	Default	Notes
`net`	string	required	`none`	Net name to waive

Applies to: LVS, MLVS

```
lvs_waiver_netlist_net VDDA_NOCONN
```

See also: lvs_waiver_macro_net, set_lvs_waiver_file

12 LEF/DEF Design Rule Checks

This family arms the individual checks of the LEF/DEF DRC engine. Each is a boolean switch that defaults to disabled, so a LEF/DEF run checks only what the script explicitly enables. All require set_lefdef_flow 1, a design loaded with read_lef and read_def, and load_def_db, and are evaluated by run_lefdef_drc.

check_open_check

Enables open-net checking. Reports nets whose routing is discontinuous.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_open_check 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_loop

Enables loop checking. Reports routing loops, which waste area and can create unintended antenna paths.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_loop 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_area

Enables minimum-area checking. Reports metal islands smaller than the layer minimum area.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_area 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_allvias

Enables via dumping. Dumps every via in the design for inspection rather than reporting violations.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_allvias 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_allrect

Enables rectangle dumping. Dumps every rectangle in the design.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_allrect 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_allsegs

Enables segment dumping. Dumps every routing segment in the design.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_allsegs 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_powersegs

Enables power segment dumping. Dumps every power and ground routing segment.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_powersegs 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_allpins

Enables pin dumping. Dumps every pin in the design.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_allpins 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_wrongway

Enables wrong-way routing checking. Reports routing against the preferred direction of a layer. Distinct from off-grid checking, which concerns track alignment rather than direction.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_wrongway 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_shorts

Enables short checking. Reports unintended electrical connections between distinct nets.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_shorts 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_off_grid

Enables off-grid checking. Reports geometry not aligned to the manufacturing grid.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_off_grid 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_out_of_die

Enables out-of-die checking. Reports geometry falling outside the declared die area.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_out_of_die 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_via_in_pin

Enables via-in-pin checking. Reports vias placed within pin shapes where the library does not permit it.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_via_in_pin 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_unconnpin

Enables unconnected pin checking. Reports pins with no routing attached.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_unconnpin 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_dangling_wire

Enables dangling wire checking. Reports routing segments connected at only one end.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_dangling_wire 1
```

See also: run_lefdef_drc, set_lefdef_flow

`check_nsmetal`

Enables non-sufficient metal checking. Reports metal coverage below what the layer requires for reliable connection.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 enables the check

Applies to: LEF/DEF flows

```
check_nsmetal 1
```

See also: run_lefdef_drc, set_lefdef_flow

`check_cut_short`

Enables cut short checking. Reports cut-layer geometry creating an unintended connection.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 enables the check

Applies to: LEF/DEF flows

```
check_cut_short 1
```

See also: run_lefdef_drc, set_lefdef_flow

`check_antenna`

Enables antenna checking. Reports nets whose accumulated conductor area exceeds the process antenna ratio.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 enables the check

Applies to: LEF/DEF flows

```
check_antenna 1
```

See also: run_lefdef_drc, set_lefdef_flow

`check_antenna_cut`

Enables cut-layer antenna checking. Reports antenna violations accumulated through cut layers.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 enables the check

Applies to: LEF/DEF flows

```
check_antenna_cut 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_rect_only

Enables rectangle-only checking. Reports geometry on layers restricted to rectangular shapes.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_rect_only 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_minimal_width

Enables minimum-width checking. Reports geometry narrower than the layer minimum width.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_minimal_width 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_widthtable

Enables width-table checking. Reports widths not present in the layer's permitted width table.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_widthtable 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_min_step

Enables minimum-step checking. Reports edge steps shorter than the layer minimum, which are difficult to resolve lithographically.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_min_step 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_floating_patch

Enables floating patch checking. Reports metal patches with no electrical connection.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_floating_patch 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_metal_spacing

Enables metal spacing checking. Reports metal-to-metal spacing below the layer minimum.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_metal_spacing 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_cut_spacing

Enables cut spacing checking. Reports cut-to-cut spacing below the layer minimum.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_cut_spacing 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_eol_keepout

Enables end-of-line keepout checking. Reports geometry intruding into the keepout region at a line end.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_eol_keepout 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_table_prl_spacing

Enables parallel run length spacing checking. Reports spacing violations from the parallel-run-length table, in which the required spacing increases with the length over which two edges run parallel.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_table_prl_spacing 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_table_cut_spacing

Enables cut spacing table checking. Reports cut spacing violations from the layer's spacing table.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_table_cut_spacing 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_min_cut

Enables minimum cut count checking. Reports connections made with fewer cuts than the layer requires for the conducting width involved.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_min_cut 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_cut_enclosure

Enables cut enclosure checking. Reports insufficient metal enclosure of a cut.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_cut_enclosure 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_metal_eol_spacing

Enables metal end-of-line spacing checking. Reports spacing violations at line ends, where the requirement is normally larger than the general spacing rule.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_metal_eol_spacing 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_metal_joint_corner_spacing

Enables joint corner spacing checking. Reports spacing violations at corner joints.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_metal_joint_corner_spacing 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_metal_forbidden_spacing

Enables forbidden metal spacing checking. Reports spacings falling inside a forbidden range, which some processes define between the minimum and a larger permitted value.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_metal_forbidden_spacing 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_eol_extn_spacing

Enables end-of-line extension spacing checking. Reports insufficient extension beyond a line end.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_eol_extn_spacing 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_cut_forbidden_spacing

Enables forbidden cut spacing checking. Reports cut spacings falling inside a forbidden range.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_cut_forbidden_spacing 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_metal_area_spacing

Enables metal area spacing checking. Reports spacing violations conditioned on the area of the interacting shapes.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_metal_area_spacing 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_cut_on_centerline

Enables cut centerline checking. Reports cuts not aligned to the required centerline.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_cut_on_centerline 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_cut_enclosureedge

Enables cut enclosure edge checking. Reports insufficient enclosure measured at a specific cut edge.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_cut_enclosureedge 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_cut_enclosureedge_opposite

Enables opposite-edge cut enclosure checking. Reports insufficient enclosure measured on opposite cut edges, which is the form most processes use for via landing rules.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_cut_enclosureedge_opposite 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_span_length_table

Enables span length table checking. Reports violations of the layer's span-length table.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_span_length_table 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_enclosure_to_joint

Enables enclosure-to-joint checking. Reports insufficient enclosure measured to a routing joint.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_enclosure_to_joint 1
```

See also: run_lefdef_drc, set_lefdef_flow

check_invalid_cut

Enables invalid cut checking. Reports cuts placed where the layer stack does not permit a connection.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables the check

Applies to: LEF/DEF flows

```
check_invalid_cut 1
```

See also: run_lefdef_drc, set_lefdef_flow

13 Density and DFM

Density checking measures the fill ratio of each layer within a sliding window and reports windows outside the permitted range. The window geometry is set once with density_window; layers are selected with repeated density_layer calls.

density_layer

Adds a layer to the density check set. Repeatable, one call per layer. Only layers named here are measured.

Argument	Type	Required	Default	Notes
layer	string	required	none	Layer name

Applies to: DENSITY, DFM

```
density_layer Metal1
density_layer Metal2
density_layer Metal3
```

See also: density_window, density_pixel, run_density

density_window

Defines the measurement window and how it is stepped across the die. A step smaller than the window gives overlapping windows, which is what most foundry decks require and which detects local density excursions that a tiled non-overlapping scan would miss. The window and step must match the foundry rule exactly or the resulting density values will not correlate.

Argument	Type	Required	Default	Notes
width	integer	required	none	Window width in database units
height	integer	required	none	Window height in database units
step_x	integer	required	none	X step between windows
step_y	integer	required	none	Y step between windows
mode	integer	required	none	Window evaluation mode

Applies to: DENSITY, DFM

```
density_window 100000 100000 50000 50000 1
```

See also: density_layer, density_pixel, run_density

`density_pixel`

Sets the sampling pixel size used when computing window fill ratios. A smaller pixel is more accurate and slower; the value must be small relative to the smallest feature contributing to density or the ratio will be quantized.

Argument	Type	Required	Default	Notes
`size`	integer	optional	`0`	Pixel size in database units

Applies to: DENSITY, DFM

```
density_pixel 1000
```

See also: density_window, density_layer

14 GUI, Logging and Display

These commands control the interactive session and the log stream. load_gui is normally the last command in a script: it opens the graphics view and error browser on the results just computed. In a headless environment, cmd_line_only suppresses it.

load_gui

Opens the LVR graphical interface on the current database and results. Placed last in a script, this gives an interactive review session immediately after the run; omitted or overridden by cmd_line_only, the run is fully headless. See the GUI Reference for the interface itself.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: all

```
load_gui
```

See also: cmd_line_only, gui_display_all_layers, set_gui_xor_result

gui_display_single_layer

Makes only the named layer visible when the graphics view opens. Useful for scripting a review session that lands directly on the layer of interest.

Argument	Type	Required	Default	Notes
layer	string	required	none	Layer name

Applies to: all

```
gui_display_single_layer Metal1
```

See also: gui_display_all_layers, load_gui

gui_display_all_layers

Makes all layers visible when the graphics view opens.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: all

```
gui_display_all_layers
```

See also: gui_display_single_layer, load_gui

set_gui_xor_result

Loads XOR difference regions into the graphics view as markers, so differences can be navigated the same way DRC violations are.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 displays XOR results in the GUI

Applies to: XOR, FASTXOR

```
set_gui_xor_result 1
```

See also: run_xor, load_gui

enable_logger_timestamp

Prefixes every log line with a timestamp. Useful for profiling where wall-clock time is being spent within a run.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: all

```
enable_logger_timestamp
```

See also: disable_logger_timestamp, set_log_flush_time

disable_logger_timestamp

Suppresses timestamps from log output, which makes logs from successive runs directly diffable. Enable this in regression jobs.

Argument	Type	Required	Default	Notes
(none)	—	—	–	This command takes no arguments.

Applies to: all

```
disable_logger_timestamp
```

See also: enable_logger_timestamp, set_testmode

15 Layer and Technology Declaration

The lvr_* commands declare the technology's layer set to LVR independently of a rule file. They are most often placed in a technology setup file that is sourced by every script for a given node. All must precede load_db, since the layer table is consulted while geometry is read.

`lvr_version`

Declares the version of the technology declaration file, allowing LVR to apply the correct interpretation as the format evolves.

Argument	Type	Required	Default	Notes
`version`	integer	optional	`0`	Technology file version

Applies to: all

```
lvr_version 2
```

See also: lvr_node

`lvr_node`

Declares the technology node name within a technology file. Equivalent in effect to set_technode but expressed as part of the technology declaration rather than the run script.

Argument	Type	Required	Default	Notes
`name`	string	required	`none`	Technology node name

Applies to: all

```
lvr_node sg13g2
```

See also: set_technode, lvr_version

`lvr_layer`

Declares a drawing layer and binds it to a GDS layer number and datatype. The name declared here is the name rule files refer to.

Argument	Type	Required	Default	Notes
`name`	string	required	`none`	Layer name
`number`	integer	required	`none`	GDS layer number
`datatype`	integer	required	`none`	GDS datatype

Applies to: all

```
lvr_layer Metal1 8 0
```

See also: lvr_kind, lvr_pinlayer, lvr_text, set_layer_map_file

lvr_kind

Classifies a declared layer. The class drives default behaviour in connectivity extraction, antenna accumulation and density measurement, so misclassifying a layer produces plausible but wrong results rather than an error.

Argument	Type	Required	Default	Notes
name	string	required	none	Layer name
kind	string	required	none	Layer class, e.g. metal, cut, diffusion, poly, well, implant, text

Applies to: all

```
lvr_kind Metal1 metal
```

See also: lvr_layer, lvr_pglayer

lvr_text

Declares a text layer used for net labelling. Net names picked up from this layer seed the connectivity extraction, so an incorrect texttype produces unnamed nets and a large crop of spurious LVS mismatches.

Argument	Type	Required	Default	Notes
name	string	required	none	Text layer name
number	integer	required	none	GDS layer number
texttype	integer	required	none	GDS texttype

Applies to: all

```
lvr_text Metal1_txt 8 0
```

See also: lvr_layer, lvr_portlayer

lvr_portlayer

Declares a port layer, marking geometry that constitutes a top-level interface port.

Argument	Type	Required	Default	Notes
name	string	required	none	Port layer name
number	integer	required	none	GDS layer number
datatype	integer	required	none	GDS datatype

Applies to: all

```
lvr_portlayer Metal1_port 8 2
```

See also: lvr_pinlayer, lvs_enable_ports

lvr_pinlayer

Declares a pin layer. Pin geometry is used to associate a net name with the shapes forming a cell terminal, and is the basis of the geometric pin capture used in hierarchical LVS.

Argument	Type	Required	Default	Notes
name	string	required	none	Pin layer name
number	integer	required	none	GDS layer number
datatype	integer	required	none	GDS datatype

Applies to: all

```
lvr_pinlayer Metal1_pin 8 1
```

See also: lvr_portlayer, lvr_text

lvr_pglayer

Declares a power or ground layer, so that supply geometry is recognized as such during extraction and ERC.

Argument	Type	Required	Default	Notes
name	string	required	none	Power/ground layer name
number	integer	required	none	GDS layer number

Applies to: all

```
lvr_pglayer Metal1_pg 8
```

See also: lvr_power, lvr_ground, lvr_kind

lvr_power

Declares a global power net name. Repeatable. Nets named here are treated as supplies throughout extraction, ERC and antenna accumulation.

Argument	Type	Required	Default	Notes
name	string	required	none	Power net name

Applies to: all

```
lvr_power VDD
```

See also: lvr_ground, set_lvs_power_pin, set_lvs_global_net

`lvr_ground`

Declares a global ground net name. Repeatable.

Argument	Type	Required	Default	Notes
`name`	string	required	`none`	Ground net name

Applies to: all

```
lvr_ground VSS
```

See also: lvr_power, set_lvs_ground_pin

16 LVS Input Files

These commands name every input the LVS and MLVS engines consume. Each is a simple path assignment with no side effects; the files are opened during load_db. Paths may be absolute or relative to the directory the script is run from.

set_lvs_gds_file

Names the layout GDSII file for the comparison.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_gds_file ../gds/DiffAmp.gds
```

See also: read_gds, set_lvs_top_cell

set_lvs_spice_file

Names the SPICE or CDL schematic netlist for the comparison. This is the preferred modern form; lvs_set_spice_file_name is the legacy equivalent.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_spice_file UNKNOWN_CDL
```

See also: read_spice, set_lvs_top_cdl_file

set_lvs_spice_file1

Names the first SPICE netlist in a two-netlist comparison.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_spice_file1 golden.cdl
```

See also: set_lvs_spice_file2, set_svs1_spice

set_lvs_spice_file2

Names the second SPICE netlist in a two-netlist comparison.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_spice_file2 revised.cdl
```

See also: set_lvs_spice_file1

set_lvs_verilog_file

Names the structural Verilog netlist for the comparison.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_verilog_file design.v
```

See also: read_verilog, verilog2sch

set_lvs_verilog_file1

Names the first Verilog netlist in a two-netlist comparison.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_verilog_file1 golden.v
```

See also: set_lvs_verilog_file2, set_svs1_verilog

set_lvs_verilog_file2

Names the second Verilog netlist in a two-netlist comparison.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_verilog_file2 revised.v
```

See also: set_lvs_verilog_file1

set_lvs_lef_file

Names a LEF file for the LVS engine, used where cell abstracts supply pin locations that the GDS does not label.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_lef_file ../lef/stdcells.lef
```

See also: read_lef, set_lvs_def_file

set_lvs_def_file

Names a DEF file for the LVS engine.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_def_file ../def/design.def
```

See also: read_def, set_lvs_lef_file

set_lvs_svrf_file

Names the SVRF rule file used for device recognition and connectivity by the LVS engine.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_svrf_file ../rule/extract.cal
```

See also: read_svrf, set_lvs_use_svrf_device_rules

set_lvs_lrf_file

Names the LRF rule file used by the LVS engine.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_lrf_file ../rule/sky130.lrf
```

See also: read_lrf

set_lvs_layer_map_file

Names a layer map file for the LVS engine, overriding the global map for extraction purposes only.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_layer_map_file ../maps/lvs.map
```

See also: set_layer_map_file

set_lvs_macros_cdl_file

Names the macro CDL library used to resolve hard-macro references. Without it, macro instances cannot be matched and every macro pin appears as an unresolved reference.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_macros_cdl_file macros.cdl
```

See also: set_lvs_top_cdl_file, split_cdl_sky130, lvs_check_macro

set_lvs_top_cdl_file

Names the top-level CDL netlist, the counterpart to the macro library produced by the split_cdl_* utilities.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_top_cdl_file top.cdl
```

See also: set_lvs_macros_cdl_file, split_cdl_sky130

set_lvs_tech_file

Names a technology file for the LVS engine, supplying device model definitions and terminal orders.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_tech_file ../tech/sg13g2.tech
```

See also: set_lvs_tech_node, set_technode

set_lvs_waiver_file

Names the waiver database file. Waivers are matched by SHA-256 content hash, so a waiver stops applying automatically once the geometry it was written against changes.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_waiver_file ./waivers/lvs_waivers.db
```

See also: set_lvs_enable_waivers, set_lvs_waiver_expiry_check, lvs_waiver_macro_net

set_lvs_signature_file

Names the run signature file, which records a hash of every input so that a later run can prove it operated on identical data. Required for a reproducible sign-off record.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_signature_file ./out/run.sig
```

See also: set_lvs_waiver_audit_trail

17 LVS Output Files

Each of these names one output artifact. Nothing is written until the run completes, and empty or missing files after a run indicate the run did not reach the stage that produces them. Set these before the run command.

set_lvs_report_file

Names the primary LVS report: pass or fail status, device and net counts, and a summary of every mismatch class.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_report_file ./out/lvs.rpt
```

See also: report_lvs, set_lvs_summary_file

set_lvs_summary_file

Names the condensed summary report, suitable for a regression dashboard.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_summary_file ./out/lvs_summary.rpt
```

See also: set_lvs_summary_only

set_lvs_detailed_log_file

Names the detailed log, which records each stage of extraction and matching. This is the first file to read when a run fails in a way the report does not explain.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_detailed_log_file ./out/lvs_detail.log
```

See also: set_lvs_log_file

set_lvs_log_file

Names the general LVS log file.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_log_file ./out/lvs.log
```

See also: set_lvs_detailed_log_file, set_lvs_verbose

set_lvs_marker_file

Names the marker file holding the layout coordinates of every LVS error, for display in the LVR graphics view or an external viewer.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_marker_file ./out/lvs.markers
```

See also: set_lvs_enable_marker_file, set_lvs_virtuoso_markers

set_lvs_csv_file

Names a CSV export of the results, for spreadsheet or script post-processing.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_csv_file ./out/lvs.csv
```

See also: set_lvs_enable_csv_export

set_lvs_db_file

Names the binary results database, which preserves full marker detail for later interactive review without re-running.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_db_file ./out/lvs.db
```

See also: set_lvs_enable_db_export

set_lvs_erc_report_file

Names the ERC report produced alongside LVS when set_perform_lvs_erc is enabled.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_erc_report_file ./out/erc.rpt
```

See also: set_perform_lvs_erc, set_lvs_report_erc

set_lvs_net_report_file

Names the per-net report listing every extracted net with its member geometry and connected devices.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_net_report_file ./out/nets.rpt
```

See also: dump_nets, set_lvs_report_net_names

set_lvs_device_report_file

Names the per-device report listing every extracted device with its type, dimensions and terminal nets. Compare this against the schematic device list when counts do not match.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_device_report_file ./out/devices.rpt
```

See also: set_lvs_report_device_counts

set_lvs_match_report_file

Names the matching report, which records how each layout instance was paired with its schematic counterpart and where pairing failed.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_match_report_file ./out/match.rpt
```

See also: set_lvs_match_report_unmatched

set_lvs_short_report_file

Names the short report, listing each detected short with the nets involved and, when tracing is enabled, the connecting path.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_short_report_file ./out/shorts.rpt
```

See also: set_lvs_short_report_nets

set_lvs_open_report_file

Names the open report, listing each detected open net.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_open_report_file ./out/opens.rpt
```

See also: set_lvs_open_report_path

set_lvs_extracted_netlist

Names the file the layout-extracted netlist is written to. This is the netlist the comparison actually sees from the layout side.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: LVS, MLVS

```
set_lvs_extracted_netlist ./out/extracted.cdl
```

See also: lvs_write_layout_from_db

set_mlvs_report_file

Names the macro LVS report.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: MLVS

```
set_mlvs_report_file ./out/mlvs.rpt
```

See also: mlvs, report_mlvs

set_svs_report_file

Names the schematic-versus-schematic report.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: SVS

```
set_svs_report_file ./out/svs.rpt
```

See also: run_svs, report_svs

18 LVS Supply and Global Nets

Supply declarations tell the extractor which nets are power and ground. This affects short detection, ERC, antenna accumulation and device bulk terminal resolution. The numbered variants exist for multi-supply designs; declare each domain separately rather than relying on name pattern matching.

set_lvs_power_pin

Declares the primary power net.

Argument	Type	Required	Default	Notes
name	string	required	none	Power net name

Applies to: LVS, MLVS, ERC

```
set_lvs_power_pin VPWR
```

See also: set_lvs_ground_pin, lvr_power, set_lvs_connect_power_global

set_lvs_power_pin1

Declares an additional power net in a multi-supply design.

Argument	Type	Required	Default	Notes
name	string	required	none	Power net name

Applies to: LVS, MLVS, ERC

```
set_lvs_power_pin1 VDDA
```

See also: set_lvs_power_pin

set_lvs_power_pin2

Declares an additional power net.

Argument	Type	Required	Default	Notes
name	string	required	none	Power net name

Applies to: LVS, MLVS, ERC

```
set_lvs_power_pin2 VDDIO
```

See also: set_lvs_power_pin

set_lvs_power_pin3

Declares an additional power net.

Argument	Type	Required	Default	Notes
name	string	required	none	Power net name

Applies to: LVS, MLVS, ERC

```
set_lvs_power_pin3 VDDPST
```

See also: set_lvs_power_pin

set_lvs_power_pin4

Declares an additional power net.

Argument	Type	Required	Default	Notes
name	string	required	none	Power net name

Applies to: LVS, MLVS, ERC

```
set_lvs_power_pin4 VDDQ
```

See also: set_lvs_power_pin

set_lvs_power_pin5

Declares an additional power net.

Argument	Type	Required	Default	Notes
name	string	required	none	Power net name

Applies to: LVS, MLVS, ERC

```
set_lvs_power_pin5 VDDR
```

See also: set_lvs_power_pin

set_lvs_power_pin6

Declares an additional power net.

Argument	Type	Required	Default	Notes
name	string	required	none	Power net name

Applies to: LVS, MLVS, ERC

```
set_lvs_power_pin6 VDDH
```

See also: set_lvs_power_pin

set_lvs_ground_pin

Declares the primary ground net.

Argument	Type	Required	Default	Notes
name	string	required	none	Ground net name

Applies to: LVS, MLVS, ERC

```
set_lvs_ground_pin VGND
```

See also: set_lvs_power_pin, lvr_ground

set_lvs_ground_pin1

Declares an additional ground net.

Argument	Type	Required	Default	Notes
name	string	required	none	Ground net name

Applies to: LVS, MLVS, ERC

```
set_lvs_ground_pin1 VSSA
```

See also: set_lvs_ground_pin

set_lvs_ground_pin2

Declares an additional ground net.

Argument	Type	Required	Default	Notes
name	string	required	none	Ground net name

Applies to: LVS, MLVS, ERC

```
set_lvs_ground_pin2 VSSIO
```

See also: set_lvs_ground_pin

set_lvs_ground_pin3

Declares an additional ground net.

Argument	Type	Required	Default	Notes
name	string	required	none	Ground net name

Applies to: LVS, MLVS, ERC

```
set_lvs_ground_pin3 VSSPST
```

See also: set_lvs_ground_pin

set_lvs_ground_pin4

Declares an additional ground net.

Argument	Type	Required	Default	Notes
name	string	required	none	Ground net name

Applies to: LVS, MLVS, ERC

```
set_lvs_ground_pin4 VSSQ
```

See also: set_lvs_ground_pin

set_lvs_alias_vpwr

Maps an alternative name onto the standard VPWR supply. Use where a library names its positive supply differently from the rest of the design.

Argument	Type	Required	Default	Notes
name	string	required	none	Alias for VPWR

Applies to: LVS, MLVS, ERC

```
set_lvs_alias_vpwr VDD
```

See also: set_lvs_alias_vgnd, split_cdl_sky130

set_lvs_alias_vgnd

Maps an alternative name onto the standard VGND supply.

Argument	Type	Required	Default	Notes
name	string	required	none	Alias for VGND

Applies to: LVS, MLVS, ERC

```
set_lvs_alias_vgnd VSS
```

See also: set_lvs_alias_vpwr

set_lvs_alias_vpb

Maps an alternative name onto the standard VPB p-bulk terminal, used by the four-terminal device conventions of SKY130 and similar PDKs.

Argument	Type	Required	Default	Notes
name	string	required	none	Alias for VPB

Applies to: LVS, MLVS, ERC

```
set_lvs_alias_vpb VNW
```

See also: set_lvs_alias_vnb, set_lvs_extract_bulk_terminal

set_lvs_alias_vnb

Maps an alternative name onto the standard VNB n-bulk terminal.

Argument	Type	Required	Default	Notes
name	string	required	none	Alias for VNB

Applies to: LVS, MLVS, ERC

```
set_lvs_alias_vnb VPW
```

See also: set_lvs_alias_vpb

set_lvs_global_net

Declares a net as global, so that occurrences of the name throughout the hierarchy are treated as one net without explicit port connections.

Argument	Type	Required	Default	Notes
name	string	required	none	Global net name

Applies to: LVS, MLVS, ERC

```
set_lvs_global_net VDD
```

See also: set_lvs_connect_power_global, lvr_power

set_lvs_internal_net_prefix

Sets the prefix used when naming nets that have no label in the layout. Choose a prefix that cannot collide with real net names.

Argument	Type	Required	Default	Notes
prefix	string	required	none	Prefix string

Applies to: LVS, MLVS

```
set_lvs_internal_net_prefix LVR_N
```

See also: set_lvs_assign_internal_nets

set_lvs_connect_layer

Names a layer to participate in connectivity extraction. Repeatable.

Argument	Type	Required	Default	Notes
layer	string	required	none	Layer name

Applies to: LVS, MLVS

```
set_lvs_connect_layer Metal1
```

See also: set_lvs_connect_all_metals, set_lvs_use_svrf_connect

19 LVS Hierarchy and Cell Selection

These commands decide what part of the design is compared and at what level of hierarchy. Hierarchical comparison is faster and gives more localized error messages; flattening is the fallback when the layout and schematic hierarchies do not correspond.

set_lvs_top_cell

Names the top cell for the comparison, overriding set_top for the LVS engine only.

Argument	Type	Required	Default	Notes
cell	string	required	none	Top cell name

Applies to: LVS, MLVS

```
set_lvs_top_cell DiffAmp_NOFILL
```

See also: set_top, set_lvs_check_top_only

set_lvs_cell_filter

Restricts the comparison to cells matching the given pattern. Wildcards are supported.

Argument	Type	Required	Default	Notes
pattern	string	required	none	Cell name pattern

Applies to: LVS, MLVS

```
set_lvs_cell_filter sg13g2_*
```

See also: set_lvs_only_cell, set_lvs_skip_cell

set_lvs_skip_cell

Excludes a named cell from the comparison. Repeatable.

Argument	Type	Required	Default	Notes
cell	string	required	none	Cell name

Applies to: LVS, MLVS

```
set_lvs_skip_cell sg13g2_fill_1
```

See also: set_lvs_only_cell, set_lvs_skip_fill_cells

set_lvs_only_cell

Restricts the comparison to a single named cell. Repeatable. The fastest way to isolate a failing block from a large design.

Argument	Type	Required	Default	Notes
cell	string	required	none	Cell name

Applies to: LVS, MLVS

```
set_lvs_only_cell DiffAmp
```

See also: set_lvs_skip_cell

set_lvs_macro_cell

Declares a cell as a hard macro, to be matched on its pin interface rather than its internal topology. Repeatable.

Argument	Type	Required	Default	Notes
cell	string	required	none	Macro cell name

Applies to: LVS, MLVS

```
set_lvs_macro_cell sram_1024x32
```

See also: mlvs, lvs_consider_macro_pins, lvs_check_macro

set_lvs_hierarchical

Compares layout and schematic hierarchically, cell by cell, rather than flattening both to transistor level. Faster on large designs and produces errors localized to the cell in which they occur, but requires the two hierarchies to correspond.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_hierarchical 1
```

See also: set_lvs_flatten_cells, set_lvs_cell_depth

set_lvs_check_all_cells

Compares every cell in the hierarchy rather than only the top. Gives complete coverage at the cost of runtime.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_check_all_cells 1
```

See also: set_lvs_check_top_only, set_lvs_report_pass_cells

set_lvs_check_top_only

Compares only the top cell, treating all sub-cells as already verified. Appropriate when sub-blocks have been signed off independently.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_check_top_only 1
```

See also: set_lvs_check_all_cells

set_lvs_flatten_cells

Flattens the hierarchy before comparison. Necessary when the layout and schematic hierarchies differ, for example after logic restructuring or when the layout merges cells the netlist keeps separate.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_flatten_cells 1
```

See also: set_lvs_hierarchical

set_lvs_skip_empty_cells

Skips cells containing no geometry, which are usually placeholders and produce noise rather than useful results.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_skip_empty_cells 1
```

See also: set_lvs_skip_fill_cells

set_lvs_skip_fill_cells

Skips fill and decap cells. These contribute no connectivity of interest and are often absent from the schematic entirely, so skipping them removes a large class of expected mismatches.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_skip_fill_cells 1
```

See also: set_lvs_skip_empty_cells, set_lvs_skip_cell

set_lvs_cell_depth

Limits how many levels of hierarchy are traversed. The default traverses the full depth. Use a shallow depth to get a fast top-level sanity result on a deep design.

Argument	Type	Required	Default	Notes
value	integer	optional	0	Maximum hierarchy depth; 0 means unlimited

Applies to: LVS, MLVS

```
set_lvs_cell_depth 3
```

See also: set_lvs_hierarchical, set_lvs_check_all_cells

20 LVS Device Extraction

These commands select which device types are recognized in the layout and how closely extracted dimensions must match the schematic. Enabling a device family that a technology does not use costs runtime; disabling one it does use produces a device count mismatch that looks like a layout error but is a setup error.

`set_lvs_extract_nmos`

Recognizes NMOS transistors during extraction.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_extract_nmos 1
```

See also: set_lvs_extract_pmos, set_lvs_use_svrf_device_rules

`set_lvs_extract_pmos`

Recognizes PMOS transistors during extraction.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_extract_pmos 1
```

See also: set_lvs_extract_nmos

`set_lvs_extract_resistor`

Recognizes resistors during extraction.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_extract_resistor 1
```

See also: set_lvs_extract_capacitor

set_lvs_extract_capacitor

Recognizes capacitors during extraction, including MIM and MOM structures where the rule file defines them.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_extract_capacitor 1
```

See also: set_lvs_extract_resistor

set_lvs_extract_diode

Recognizes diodes during extraction. Required for antenna diode matching.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_extract_diode 1
```

See also: set_perform_lvs_antenna

set_lvs_extract_bjt

Recognizes bipolar junction transistors during extraction.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_extract_bjt 1
```

See also: set_lvs_extract_diode

set_lvs_extract_ldd

Recognizes lightly doped drain device variants as distinct device types rather than folding them into the base MOS family.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_extract_ldd 1
```

See also: set_lvs_extract_nmos

set_lvs_extract_bulk_terminal

Extracts the bulk terminal as an explicit fourth device terminal. Required for PDKs whose netlists carry four-terminal devices; without it, bulk connectivity errors go undetected.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_extract_bulk_terminal 1
```

See also: set_lvs_match_ignore_bulk, set_lvs_alias_vpb

set_lvs_extract_nwell_terminal

Extracts the n-well terminal explicitly.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_extract_nwell_terminal 1
```

See also: set_lvs_extract_bulk_terminal, set_perform_lvs_well_check

set_lvs_extract_sub_terminal

Extracts the substrate terminal explicitly.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_extract_sub_terminal 1
```

See also: set_lvs_extract_bulk_terminal

set_lvs_use_svrf_device_rules

Uses the device recognition rules defined in the SVRF deck rather than the built-in defaults. This is the correct setting for any foundry deck, since only the deck knows the process device definitions.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_use_svrf_device_rules 1
```

See also: set_lvs_svrf_file, set_perform_svrf_device_parse

set_lvs_use_legacy_extraction

Uses the pre-2.0 extraction path. Provided so that an older result can be reproduced exactly; not recommended for new work.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_use_legacy_extraction 1
```

See also: set_lvs_use_svrf_device_rules

set_lvs_fin_pitch

Sets the fin pitch for FinFET technologies, used to convert a device's fin count into an effective width. Must match the process value or every FinFET width comparison will be wrong by a constant factor.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Fin pitch in microns

Applies to: LVS, MLVS

```
set_lvs_fin_pitch 0.048
```

See also: set_lvs_w_tolerance

set_lvs_w_tolerance

Sets the fractional tolerance allowed when comparing device widths. A small non-zero value absorbs rounding differences between the layout and the schematic without masking genuine sizing errors.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Width tolerance as a fraction

Applies to: LVS, MLVS

```
set_lvs_w_tolerance 0.01
```

See also: set_lvs_l_tolerance

set_lvs_l_tolerance

Sets the fractional tolerance allowed when comparing device lengths.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Length tolerance as a fraction

Applies to: LVS, MLVS

```
set_lvs_l_tolerance 0.01
```

See also: set_lvs_w_tolerance

21 LVS Matching

Matching pairs layout instances with schematic instances. Name-based matching is fast and precise when both sides use the same names; topology-based matching is the general fallback and is what makes LVS work on a netlist whose names bear no relation to the layout's.

set_lvs_match_by_name

Matches instances by name where names correspond on both sides. Much faster than topological matching and gives clearer error messages.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_match_by_name 1
```

See also: set_lvs_match_by_topology, set_lvs_match_case_sensitive

set_lvs_match_by_topology

Matches instances by graph topology, ignoring names. The general case, and the only option when the netlist was generated by a tool that did not preserve layout instance names.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_match_by_topology 1
```

See also: set_lvs_match_by_name, set_lvs_match_bipartite

set_lvs_match_case_sensitive

Treats instance and net names as case-sensitive. Disable when comparing a case-preserving format against a case-folding one.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_match_case_sensitive 1
```

See also: set_lvs_match_by_name

set_lvs_match_symmetric_sd

Treats source and drain as interchangeable, which is correct for symmetric MOS devices and prevents a large class of spurious mismatches arising purely from terminal ordering.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_match_symmetric_sd 1
```

See also: set_lvs_match_by_topology

set_lvs_match_ignore_bulk

Ignores bulk terminals when matching. Useful when the schematic omits bulk connections that the layout necessarily has.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_match_ignore_bulk 1
```

See also: set_lvs_extract_bulk_terminal

set_lvs_match_allow_prefix

Allows a common prefix difference between layout and schematic names, so that names differing only by a wrapper prefix still match.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_match_allow_prefix 1
```

See also: set_lvs_match_prefix

set_lvs_match_prefix

Sets the prefix to be allowed or stripped during name matching.

Argument	Type	Required	Default	Notes
prefix	string	required	none	Prefix string

Applies to: LVS, MLVS

```
set_lvs_match_prefix X_
```

See also: set_lvs_match_allow_prefix

set_lvs_match_report_unmatched

Reports every instance that could not be matched, rather than only summarizing the count. Essential during debug.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_match_report_unmatched 1
```

See also: set_lvs_error_on_unmatched_inst, set_lvs_match_report_file

set_lvs_match_min_percentage

Sets the minimum percentage of instances that must match for the comparison to be considered successful. A partial-match threshold is useful for tracking progress on a design under active repair; it must be 100 for sign-off.

Argument	Type	Required	Default	Notes
value	integer	optional	0	Minimum match percentage, 0 to 100

Applies to: LVS, MLVS

```
set_lvs_match_min_percentage 100
```

See also: set_lvs_report_match_percent

set_lvs_match_sort_based

Uses a sort-based matching algorithm, which is fast on designs with many identical instances such as memory arrays.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_match_sort_based 1
```

See also: set_lvs_match_bipartite

set_lvs_match_bipartite

Uses bipartite graph matching. More thorough than sort-based matching and better at resolving ambiguity in highly symmetric circuits, at greater cost.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_match_bipartite 1
```

See also: set_lvs_match_sort_based, set_lvs_match_by_topology

22 LVS Connectivity, Shorts and Opens

Short and open detection operate on the extracted connectivity graph. Both can be restricted to a layer or a net to keep a debug run tractable on a design with many violations.

set_lvs_short_check_all_layers

Checks for shorts on every layer rather than only the routing layers.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_short_check_all_layers 1
```

See also: set_lvs_short_layer

set_lvs_short_layer

Restricts short checking to a named layer. Repeatable.

Argument	Type	Required	Default	Notes
layer	string	required	none	Layer name

Applies to: LVS, MLVS

```
set_lvs_short_layer Metal2
```

See also: set_lvs_short_check_all_layers

set_lvs_short_use_rtree

Uses an R-tree spatial index to accelerate short detection. Recommended for any large design.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_short_use_rtree 1
```

See also: set_lvs_use_rtree, set_lvs_rtree_node_size

set_lvs_short_report_nets

Reports the identity of both nets involved in each short rather than only the location.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_short_report_nets 1
```

See also: set_lvs_short_report_nets

set_lvs_open_check_all_nets

Checks every net for opens rather than only signal nets.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_open_check_all_nets 1
```

See also: set_lvs_open_net

set_lvs_open_check_schematic

Also checks the schematic side for opens, catching netlist errors that would otherwise be attributed to the layout.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_open_check_schematic 1
```

See also: set_lvs_open_check_all_nets

set_lvs_open_use_via

Uses via connectivity when determining whether a net is continuous. Disable to find nets held together only by a missing via.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_open_use_via 1
```

See also: lvs_check_open

set_lvs_open_report_path

Reports the geometric path along which the open was detected, which localizes the break rather than only naming the net.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_open_report_path 1
```

See also: set_lvs_open_report_file

set_lvs_open_net

Restricts open checking to a named net. Repeatable.

Argument	Type	Required	Default	Notes
net	string	required	none	Net name

Applies to: LVS, MLVS

```
set_lvs_open_net clk
```

See also: set_lvs_open_check_all_nets

set_lvs_use_svrf_connect

Uses the connect statements from the SVRF deck to build connectivity. Correct for any foundry deck, since the deck defines which layers connect through which cut layers.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_use_svrf_connect 1
```

See also: set_lvs_connect_layer, set_lvs_svrf_file

set_lvs_use_pins_for_nets

Derives net names from pin geometry rather than text labels alone. Necessary for layouts that mark terminals with pin shapes instead of labels.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_use_pins_for_nets 1
```

See also: lvr_pinlayer, set_lvs_assign_internal_nets

set_lvs_assign_internal_nets

Assigns generated names to unlabelled internal nets so that they can be referenced in reports.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_assign_internal_nets 1
```

See also: set_lvs_internal_net_prefix

set_lvs_connect_all_metals

Treats all metal layers as connected through the normal via stack without requiring explicit connect statements. A convenience for quick checks; production runs should use the deck's own connect statements.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_connect_all_metals 1
```

See also: set_lvs_use_svrf_connect, set_lvs_connect_layer

set_lvs_connect_power_global

Connects power nets globally by name across the hierarchy without requiring explicit port connections at every level.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_connect_power_global 1
```

See also: set_lvs_global_net, lvr_power

23 LVS Reporting Detail

These switches control how much detail appears in LVS output. Turning everything on produces very large reports on a design with many errors; the usual practice is to enable coordinates and net names during debug and reduce to summary output for regression.

set_lvs_report_pass_cells

Lists cells that passed, not only those that failed. Useful for confirming coverage.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_pass_cells 1
```

See also: set_lvs_report_fail_cells

set_lvs_report_fail_cells

Lists cells that failed.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_fail_cells 1
```

See also: set_lvs_report_pass_cells

set_lvs_report_device_counts

Reports device counts by type for both layout and schematic. The first number to check when a comparison fails, since a count mismatch points at extraction rather than at matching.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_device_counts 1
```

See also: set_lvs_device_report_file

set_lvs_report_net_names

Includes net names in the report rather than internal identifiers.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_net_names 1
```

See also: set_lvs_net_report_file

set_lvs_report_coordinates

Includes layout coordinates for each error, which is what makes a report navigable in a viewer.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_coordinates 1
```

See also: set_lvs_marker_file

set_lvs_report_match_percent

Reports the percentage of instances matched.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_match_percent 1
```

See also: set_lvs_match_min_percentage

set_lvs_report_timing

Reports per-stage runtime, for profiling a slow run.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_timing 1
```

See also: set_lvs_detailed_log_file

set_lvs_report_erc

Includes ERC findings in the LVS report.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_erc 1
```

See also: set_perform_lvs_erc, set_lvs_erc_report_file

set_lvs_report_shorts

Includes short details in the report.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_shorts 1
```

See also: set_lvs_short_report_file

set_lvs_report_opens

Includes open details in the report.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_opens 1
```

See also: set_lvs_open_report_file

set_lvs_report_unconnected

Includes unconnected pin details in the report.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_unconnected 1
```

See also: lvs_check_unconnected_pins

set_lvs_report_all_errors

Reports every error rather than stopping at the configured maximum.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_report_all_errors 1
```

See also: set_lvs_max_error, set_lvs_summary_only

set_lvs_summary_only

Emits summary output only, suppressing per-error detail. Appropriate for a regression dashboard.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_summary_only 1
```

See also: set_lvs_summary_file

set_lvs_enable_marker_file

Writes the marker file named by set_lvs_marker_file.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_enable_marker_file 1
```

See also: set_lvs_marker_file

set_lvs_enable_csv_export

Writes the CSV export named by set_lvs_csv_file.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_enable_csv_export 1
```

See also: set_lvs_csv_file

set_lvs_enable_db_export

Writes the binary results database named by set_lvs_db_file.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_enable_db_export 1
```

See also: set_lvs_db_file

set_lvs_verbose

Increases log verbosity.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_verbose 1
```

See also: set_lvs_log_file, set_lvs_report_timing

24 LVS Waivers

A waiver suppresses a known, reviewed violation. LVR records each waiver against a SHA-256 hash of the geometry it was written for, so that a waiver stops applying automatically once that geometry changes. This is what prevents a waiver written for one revision from silently hiding a new violation in the next.

set_lvs_enable_waivers

Applies the waiver database named by set_lvs_waiver_file.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_enable_waivers 1
```

See also: set_lvs_waiver_file

set_lvs_waiver_expiry_check

Checks waiver expiry dates and reports waivers that have lapsed, so that a temporary waiver cannot quietly become permanent.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_waiver_expiry_check 1
```

See also: set_lvs_enable_waivers, set_lvs_waiver_owner

set_lvs_waiver_audit_trail

Records which waiver suppressed which violation, producing the audit record a sign-off review requires.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_waiver_audit_trail 1
```

See also: set_lvs_signature_file, set_lvs_waiver_owner

set_lvs_waiver_wildcard

Allows wildcard patterns in waiver definitions. Convenient, but a broad pattern can suppress violations nobody has reviewed, so prefer specific waivers for sign-off.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_waiver_wildcard 1
```

See also: set_lvs_enable_waivers

set_lvs_waiver_spatial_filter

Restricts waivers to the spatial region they were written for, so a waiver cannot travel to an unrelated part of the die.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_waiver_spatial_filter 1
```

See also: set_lvs_waiver_wildcard

set_lvs_waiver_owner

Records the owner of a waiver for the audit trail.

Argument	Type	Required	Default	Notes
name	string	required	none	Owner name or identifier

Applies to: LVS, MLVS

```
set_lvs_waiver_owner rpatel
```

See also: set_lvs_waiver_audit_trail

25 LVS Performance and Memory

These commands tune the LVS engine's use of threads, spatial indices and memory. The defaults are conservative; on a dedicated machine with a large design there is usually significant headroom.

set_lvs_num_threads

Sets the thread count for the LVS engine, independently of the global set_num_cores.

Argument	Type	Required	Default	Notes
value	integer	optional	1	Thread count

Applies to: LVS, MLVS

```
set_lvs_num_threads 16
```

See also: set_num_cores

set_lvs_use_rtree

Uses an R-tree spatial index for geometric queries during extraction. Recommended for any design of significant size.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_use_rtree 1
```

See also: set_lvs_rtree_node_size, set_lvs_short_use_rtree

set_lvs_rtree_node_size

Sets the R-tree node size. Larger nodes reduce tree depth and memory overhead but make each node test more expensive; the default suits most designs.

Argument	Type	Required	Default	Notes
value	integer	optional	0	Node size

Applies to: LVS, MLVS

```
set_lvs_rtree_node_size 16
```

See also: set_lvs_use_rtree

set_lvs_chunk_size

Sets the work chunk size handed to each thread. Smaller chunks balance load better on uneven designs; larger chunks reduce scheduling overhead.

Argument	Type	Required	Default	Notes
value	integer	optional	0	Chunk size

Applies to: LVS, MLVS

```
set_lvs_chunk_size 1000
```

See also: set_lvs_num_threads

set_lvs_use_cache

Caches intermediate extraction results so that repeated cells are extracted once. A large win on designs dominated by repeated standard cells.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_use_cache 1
```

See also: set_lvs_cache_size_mb

set_lvs_cache_size_mb

Sets the cache size in megabytes.

Argument	Type	Required	Default	Notes
value	integer	optional	0	Cache size in MB

Applies to: LVS, MLVS

```
set_lvs_cache_size_mb 4096
```

See also: set_lvs_use_cache

set_lvs_memory_limit_mb

Caps the memory the LVS engine may use. On reaching the cap the engine spills rather than being killed by the operating system, which is worth setting on a shared compute farm.

Argument	Type	Required	Default	Notes
value	integer	optional	0	Memory limit in MB

Applies to: LVS, MLVS

```
set_lvs_memory_limit_mb 65536
```

See also: set_lvs_cache_size_mb, use_max

`set_lvs_flush_time`

Sets the LVS log flush interval in seconds.

Argument	Type	Required	Default	Notes
`value`	integer	optional	`0`	Flush interval in seconds

Applies to: LVS, MLVS

```
set_lvs_flush_time 5
```

See also: set_log_flush_time

26 LVS Stage Selection

The set_perform_* family arms individual stages of the LVS pipeline. set_perform_lvs is the master switch; the remainder select which stages run. Disabling a stage removes its runtime cost and its findings, so a run with stages disabled must not be represented as a full LVS result.

set_perform_lvs

Master switch for the LVS pipeline.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs 1
```

See also: run_lvs

set_perform_lvs_device_extract

Runs device extraction from the layout. Required by every other LVS stage.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_device_extract 1
```

See also: set_lvs_extract_nmos, set_lvs_use_svrf_device_rules

set_perform_lvs_matching

Runs the instance matching stage.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_matching 1
```

See also: lvs_check_matching, set_lvs_match_by_topology

set_perform_lvs_shorts

Runs short detection.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_shorts 1
```

See also: lvs_check_shorts

set_perform_lvs_opens

Runs open detection.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_opens 1
```

See also: lvs_check_open

set_perform_lvs_macro

Runs macro-level checking.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_macro 1
```

See also: lvs_check_macro, mlvs

set_perform_lvs_erc

Runs ERC as part of the LVS pipeline, reusing the extracted connectivity rather than requiring a separate run.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_erc 1
```

See also: run_erc, set_lvs_erc_report_file

set_perform_lvs_soft_connect

Runs soft connection checking, which identifies nets connected only through high-resistance paths such as a well or substrate rather than through metal.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_soft_connect 1
```

See also: set_perform_lvs_well_check

set_perform_lvs_port_check

Checks top-level ports against the schematic interface.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_port_check 1
```

See also: lvs_enable_ports

set_perform_lvs_unconnected_pin

Runs unconnected pin detection.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_unconnected_pin 1
```

See also: lvs_check_unconnected_pins

set_perform_lvs_property_check

Compares device properties such as width, length and multiplier between layout and schematic. Without this a topologically correct but mis-sized layout will pass.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_property_check 1
```

See also: set_lvs_w_tolerance, set_lvs_l_tolerance

set_perform_lvs_parallel_reduce

Reduces parallel device chains to a single equivalent device before matching, so that a layout drawn as several parallel fingers matches a schematic drawn as one wide device.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_parallel_reduce 1
```

See also: set_perform_lvs_series_reduce

set_perform_lvs_series_reduce

Reduces series device chains to a single equivalent device before matching.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_series_reduce 1
```

See also: set_perform_lvs_parallel_reduce

set_perform_lvs_well_check

Checks well and bulk connectivity.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_well_check 1
```

See also: set_lvs_extract_nwell_terminal, set_perform_lvs_soft_connect

set_perform_lvs_antenna

Runs antenna checking within the LVS pipeline.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_antenna 1
```

See also: run_antenna, set_lvs_extract_diode

set_perform_lvs_power_check

Checks power net integrity.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_power_check 1
```

See also: set_lvs_power_pin, set_erc_check_undriven_power

set_perform_lvs_ground_check

Checks ground net integrity.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_ground_check 1
```

See also: set_lvs_ground_pin

set_perform_lvs_datc_flow

Runs the DATC benchmark-compliant flow.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_datc_flow 1
```

See also: lvs_set_datc_flow, lvs_use_datc_flow

set_perform_lvs_svs

Runs schematic-versus-schematic comparison within the LVS pipeline.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_lvs_svs 1
```

See also: run_svs

set_perform_svrf_device_parse

Parses device definitions from the SVRF deck. Required when device recognition comes from the foundry deck rather than built-in defaults.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_perform_svrf_device_parse 1
```

See also: set_lvs_use_svrf_device_rules, set_lvs_svrf_file

27 Electrical Rule Checks

The ERC family operates on extracted connectivity and reports electrical conditions that are legal geometrically but wrong electrically. Each check is independently armed. ERC requires the supply nets to have been declared, since most of the checks are defined relative to power and ground.

set_erc_check_short_to_power

Reports signal nets shorted to a power supply.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: ERC

```
set_erc_check_short_to_power 1
```

See also: set_lvs_power_pin, set_erc_check_short_to_ground

set_erc_check_short_to_ground

Reports signal nets shorted to ground.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: ERC

```
set_erc_check_short_to_ground 1
```

See also: set_lvs_ground_pin

set_erc_check_power_short

Reports shorts between distinct power domains, which in a multi-supply design is among the most damaging errors to reach silicon.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: ERC

```
set_erc_check_power_short 1
```

See also: set_lvs_power_pin1, set_erc_check_undriven_power

set_erc_check_undriven_power

Reports power nets with no driver, typically a supply that was routed but never connected to a pad or regulator.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: ERC

```
set_erc_check_undriven_power 1
```

See also: set_erc_check_power_short

set_erc_check_unconnected_input

Reports device inputs left unconnected, which float to an indeterminate level and can cause static current.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: ERC

```
set_erc_check_unconnected_input 1
```

See also: set_erc_check_floating_gate

set_erc_check_dangling_output

Reports device outputs that drive nothing.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: ERC

```
set_erc_check_dangling_output 1
```

See also: set_erc_check_unconnected_input

set_erc_check_multi_driver

Reports nets driven by more than one source, which risks contention unless the drivers are tristate or open-drain.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: ERC

```
set_erc_check_multi_driver 1
```

See also: set_erc_check_dangling_output

set_erc_check_floating_gate

Reports transistor gates with no connection. A floating gate is both a functional hazard and a reliability one.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: ERC

```
set_erc_check_floating_gate 1
```

See also: set_erc_check_unconnected_input

set_erc_check_floating_bulk

Reports device bulk terminals with no connection, which leaves the body at an undefined potential and risks latch-up.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: ERC

```
set_erc_check_floating_bulk 1
```

See also: set_erc_check_floating_gate, set_perform_lvs_well_check

set_erc_report_info

Includes informational messages alongside errors in the ERC report.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: ERC

```
set_erc_report_info 1
```

See also: report_erc, set_erc_max_error

28 Cadence Virtuoso Integration

These commands drive LVR's integration with Cadence Virtuoso. Markers are pushed into the Virtuoso canvas as dbCreateMarker overlays through a SKILL callback, so violations can be navigated in the layout editor the designer is already working in.

set_lvs_virtuoso_markers

Emits LVS markers in Virtuoso marker format for display as dbCreateMarker overlays on the layout canvas.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_virtuoso_markers 1
```

See also: set_lvs_skill_integration, set_lvs_marker_file

set_lvs_skill_integration

Enables the SKILL integration layer, allowing LVR to invoke SKILL procedures in a running Virtuoso session.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_skill_integration 1
```

See also: set_lvs_skill_callback, set_lvs_virtuoso_markers

set_lvs_gds_export_before_run

Exports the current Virtuoso cellview to GDS immediately before the run, so that the check operates on what is on screen rather than on a stale file.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_gds_export_before_run 1
```

See also: set_lvs_cdl_export_before_run, set_lvs_virtuoso_cell

set_lvs_cdl_export_before_run

Exports the schematic to CDL immediately before the run.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: LVS, MLVS

```
set_lvs_cdl_export_before_run 1
```

See also: set_lvs_gds_export_before_run

set_lvs_virtuoso_lib

Names the Virtuoso library.

Argument	Type	Required	Default	Notes
library	string	required	none	Library name

Applies to: LVS, MLVS

```
set_lvs_virtuoso_lib myLib
```

See also: set_lvs_virtuoso_cell

set_lvs_virtuoso_cell

Names the Virtuoso cell.

Argument	Type	Required	Default	Notes
cell	string	required	none	Cell name

Applies to: LVS, MLVS

```
set_lvs_virtuoso_cell DiffAmp
```

See also: set_lvs_virtuoso_view

set_lvs_virtuoso_view

Names the Virtuoso view, typically layout or schematic.

Argument	Type	Required	Default	Notes
view	string	required	none	View name

Applies to: LVS, MLVS

```
set_lvs_virtuoso_view layout
```

See also: set_lvs_virtuoso_cell

set_lvs_skill_callback

Names the SKILL procedure LVR calls when a marker is selected, allowing a site-specific action to be attached to marker navigation.

Argument	Type	Required	Default	Notes
procedure	string	required	none	SKILL procedure name

Applies to: LVS, MLVS

```
set_lvs_skill_callback lvrMarkerSelect
```

See also: set_lvs_skill_integration

29 Parasitic Extraction

The PEX engine extracts resistance and capacitance from the layout and writes SPEF, DSPF or SDF. Resistance is modelled by ladder segmentation of each conductor; capacitance is decomposed into ground, fringe and coupling components. IR drop and electromigration analysis run on the extracted network.

set_pex_run_extraction

Master switch for parasitic extraction.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_run_extraction 1
```

See also: pex

set_pex_extract_resistance

Extracts conductor resistance.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_extract_resistance 1
```

See also: set_pex_extract_via_resistance, set_pex_ladder_seg_len_um

set_pex_extract_via_resistance

Extracts via resistance. Significant in designs with long vertical paths through many cut layers.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_extract_via_resistance 1
```

See also: set_pex_extract_resistance

set_pex_extract_capacitance

Master switch for capacitance extraction.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_extract_capacitance 1
```

See also: set_pex_extract_ground_cap, set_pex_extract_coupling_cap

set_pex_extract_ground_cap

Extracts capacitance to the ground plane.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_extract_ground_cap 1
```

See also: set_pex_extract_capacitance

set_pex_extract_fringe_cap

Extracts fringing capacitance from conductor edges, which dominates for narrow wires at advanced nodes.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_extract_fringe_cap 1
```

See also: set_pex_extract_ground_cap

set_pex_extract_coupling_cap

Extracts coupling capacitance between neighbouring conductors. Required for any crosstalk or signal-integrity analysis.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_extract_coupling_cap 1
```

See also: set_pex_max_coupling_dist_um, set_pex_inter_net_coupling_only

set_pex_extract_bump

Extracts bump parasitics.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_extract_bump 1
```

See also: set_pex_extract_tap

set_pex_extract_tap

Extracts substrate and well tap parasitics.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_extract_tap 1
```

See also: set_pex_extract_bump

set_pex_inter_net_coupling_only

Extracts coupling only between different nets, discarding intra-net coupling. Substantially reduces SPEF size with no loss for crosstalk analysis.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_inter_net_coupling_only 1
```

See also: set_pex_extract_coupling_cap

set_pex_rc_reduction

Reduces the extracted RC network before writing, trading some accuracy for a much smaller and faster-to-simulate netlist.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_rc_reduction 1
```

See also: set_pex_min_resistance_ohm, set_pex_min_cap_ff

set_pex_skip_power_nets

Excludes power and ground nets from extraction. These dominate the parasitic count, so excluding them greatly shrinks a signal-only SPEF.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_skip_power_nets 1
```

See also: set_pex_exclude_nets, set_pex_run_ir_drop

set_pex_run_ir_drop

Runs IR drop analysis on the extracted resistive network, solved with a preconditioned conjugate gradient method.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_run_ir_drop 1
```

See also: set_pex_supply_voltage, set_pex_cg_max_iter

set_pex_run_em

Runs electromigration analysis, comparing current density in each segment against the process limit.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_run_em 1
```

See also: set_pex_em_violation_ratio

set_pex_dump_rc

Dumps the raw RC network before reduction, for debugging an unexpected extraction result.

Argument	Type	Required	Default	Notes
flag	integer	optional	0	1 enables, 0 disables

Applies to: PEX

```
set_pex_dump_rc 1
```

See also: set_pex_rc_reduction

set_pex_ladder_seg_len_um

Sets the segment length used when discretizing a conductor into a resistive ladder. Shorter segments model distributed resistance more accurately and produce a larger network.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Segment length in microns

Applies to: PEX

```
set_pex_ladder_seg_len_um 1.0
```

See also: set_pex_max_ladder_segs

set_pex_max_ladder_segs

Caps the number of ladder segments per conductor, bounding network size on very long wires.

Argument	Type	Required	Default	Notes
value	integer	optional	0	Maximum segments per net

Applies to: PEX

```
set_pex_max_ladder_segs 100
```

See also: set_pex_ladder_seg_len_um

set_pex_max_coupling_dist_um

Sets the maximum distance at which coupling capacitance is considered. Beyond this distance coupling is negligible and computing it is wasted work.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Maximum coupling distance in microns

Applies to: PEX

```
set_pex_max_coupling_dist_um 2.0
```

See also: set_pex_extract_coupling_cap

set_pex_min_coupling_cap_ff

Discards coupling capacitors below this value.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Minimum coupling capacitance in femtofarads

Applies to: PEX

```
set_pex_min_coupling_cap_ff 0.001
```

See also: set_pex_min_cap_ff

set_pex_min_resistance_ohm

Discards resistors below this value, shorting the segment instead.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Minimum resistance in ohms

Applies to: PEX

```
set_pex_min_resistance_ohm 0.01
```

See also: set_pex_rc_reduction

set_pex_min_cap_ff

Discards capacitors below this value.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Minimum capacitance in femtofarads

Applies to: PEX

```
set_pex_min_cap_ff 0.001
```

See also: set_pex_min_coupling_cap_ff

set_pex_supply_voltage

Sets the nominal supply voltage used as the reference for IR drop analysis.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Supply voltage in volts

Applies to: PEX

```
set_pex_supply_voltage 1.8
```

See also: set_pex_run_ir_drop, set_pex_ir_drop_threshold_mv

set_pex_ir_drop_threshold_mv

Sets the IR drop above which a node is reported as a violation.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Threshold in millivolts

Applies to: PEX

```
set_pex_ir_drop_threshold_mv 50.0
```

See also: set_pex_supply_voltage

set_pex_cg_max_iter

Caps the conjugate gradient solver iteration count. A run that hits this cap has not converged and its IR results should not be trusted.

Argument	Type	Required	Default	Notes
value	integer	optional	0	Maximum solver iterations

Applies to: PEX

```
set_pex_cg_max_iter 1000
```

See also: set_pex_cg_tolerance

set_pex_cg_tolerance

Sets the convergence tolerance of the conjugate gradient solver. Tighter tolerances cost iterations.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Convergence tolerance

Applies to: PEX

```
set_pex_cg_tolerance 1e-6
```

See also: set_pex_cg_max_iter

set_pex_em_violation_ratio

Sets the current density ratio, relative to the process limit, above which a segment is reported as an electromigration violation.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Violation ratio threshold

Applies to: PEX

```
set_pex_em_violation_ratio 1.0
```

See also: set_pex_run_em

set_pex_num_threads

Sets the thread count for the PEX engine.

Argument	Type	Required	Default	Notes
value	integer	optional	1	Thread count

Applies to: PEX

```
set_pex_num_threads 16
```

See also: set_num_cores

set_pex_dbu_to_um

Sets the conversion factor from database units to microns for the PEX engine. Must match the layout database or every extracted value will be scaled wrongly.

Argument	Type	Required	Default	Notes
value	double	optional	0.0	Database units per micron

Applies to: PEX

```
set_pex_dbu_to_um 1000.0
```

See also: set_lvs_dbu

set_pex_cap_unit

Sets the capacitance unit used in output, for example FF or PF.

Argument	Type	Required	Default	Notes
unit	string	required	none	Capacitance unit

Applies to: PEX

```
set_pex_cap_unit FF
```

See also: set_pex_res_unit, set_pex_time_unit

set_pex_res_unit

Sets the resistance unit used in output, for example OHM or KOHM.

Argument	Type	Required	Default	Notes
unit	string	required	none	Resistance unit

Applies to: PEX

```
set_pex_res_unit OHM
```

See also: set_pex_cap_unit

set_pex_time_unit

Sets the time unit used in output, for example PS or NS.

Argument	Type	Required	Default	Notes
unit	string	required	none	Time unit

Applies to: PEX

```
set_pex_time_unit PS
```

See also: set_pex_cap_unit

set_pex_report_file

Names the primary PEX report.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: PEX

```
set_pex_report_file ./out/pex.rpt
```

See also: set_pex_report_spef

set_pex_report_spef

Names the SPEF output file, the standard interchange format for parasitics.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: PEX

```
set_pex_report_spef ./out/design.spef
```

See also: set_pex_spef_vendor, compare_spefs

set_pex_report_dspf

Names the DSPF output file, which carries the parasitic network as a SPICE subcircuit for direct simulation.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: PEX

```
set_pex_report_dspf ./out/design.dspf
```

See also: set_pex_report_spef

set_pex_report_sdf

Names the SDF output file, carrying back-annotated delays.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: PEX

```
set_pex_report_sdf ./out/design.sdf
```

See also: set_pex_time_unit

set_pex_report_ir

Names the IR drop report.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: PEX

```
set_pex_report_ir ./out/ir.rpt
```

See also: set_pex_run_ir_drop

set_pex_report_em

Names the electromigration report.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: PEX

```
set_pex_report_em ./out/em.rpt
```

See also: set_pex_run_em

set_pex_report_markers

Names the PEX marker file, for viewing IR and EM violations at their layout coordinates.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: PEX

```
set_pex_report_markers ./out/pex.markers
```

See also: load_gui

set_pex_log_file

Names the PEX log file.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: PEX

```
set_pex_log_file ./out/pex.log
```

See also: set_pex_verbosity

set_pex_tech_file

Names the PEX technology file, which supplies the per-layer sheet resistance, thickness and dielectric constants extraction depends on.

Argument	Type	Required	Default	Notes
path	string	required	none	File path

Applies to: PEX

```
set_pex_tech_file ../tech/tech.pextech
```

See also: set_pex_run_extraction

set_pex_design_name

Sets the design name recorded in the SPEF and DSPF headers.

Argument	Type	Required	Default	Notes
name	string	required	none	Design name

Applies to: PEX

```
set_pex_design_name top_soc
```

See also: set_pex_report_spef

set_pex_spef_vendor

Sets the vendor string in the SPEF header. Some downstream timers key their parsing on this field.

Argument	Type	Required	Default	Notes
vendor	string	required	none	Vendor string

Applies to: PEX

```
set_pex_spef_vendor Ramtera
```

See also: set_pex_spef_version

set_pex_spef_version

Sets the SPEF format version in the header.

Argument	Type	Required	Default	Notes
version	string	required	none	SPEF version string

Applies to: PEX

```
set_pex_spef_version 1.0
```

See also: set_pex_spef_vendor

set_pex_net_filter

Restricts extraction to nets matching the given pattern.

Argument	Type	Required	Default	Notes
pattern	string	required	none	Net name pattern

Applies to: PEX

```
set_pex_net_filter clk*
```

See also: set_pex_exclude_nets

set_pex_exclude_nets

Excludes nets matching the given pattern from extraction.

Argument	Type	Required	Default	Notes
pattern	string	required	none	Net name pattern

Applies to: PEX

```
set_pex_exclude_nets VDD*
```

See also: set_pex_skip_power_nets, set_pex_net_filter

set_pex_include_layers

Restricts extraction to the named layers.

Argument	Type	Required	Default	Notes
layers	string	required	none	Layer list

Applies to: PEX

```
set_pex_include_layers Metal1,Metal2,Metal3
```

See also: set_pex_exclude_layers

set_pex_exclude_layers

Excludes the named layers from extraction.

Argument	Type	Required	Default	Notes
layers	string	required	none	Layer list

Applies to: PEX

```
set_pex_exclude_layers Metal8,Metal9
```

See also: set_pex_include_layers

set_pex_verbosity

Sets PEX log verbosity.

Argument	Type	Required	Default	Notes
value	integer	optional	0	Verbosity level

Applies to: PEX

```
set_pex_verbosity 2
```

See also: set_pex_log_file

set_pex_ir_verbose_nodes

Reports per-node IR detail rather than only violating nodes.

Argument	Type	Required	Default	Notes
value	integer	optional	0	1 enables verbose node output

Applies to: PEX

```
set_pex_ir_verbose_nodes 1
```

See also: set_pex_report_ir

set_pex_em_verbose_segments

Reports per-segment electromigration detail rather than only violating segments.

Argument	Type	Required	Default	Notes
value	integer	optional	0	1 enables verbose segment output

Applies to: PEX

```
set_pex_em_verbose_segments 1
```

See also: set_pex_report_em

set_pex_flush_time

Sets the PEX log flush interval in seconds.

Argument	Type	Required	Default	Notes
value	integer	optional	0	Flush interval in seconds

Applies to: PEX

```
set_pex_flush_time 5
```

See also: set_log_flush_time

30 Deprecated Commands

These commands are accepted so that existing scripts continue to parse, but have no effect in LVR 2.0. They should be removed from any script being brought forward.

`set_lrf_flow`

Deprecated in LVR 2.0 and has no effect. Rule-file format is now detected from the file read by read_lrf or read_svrf, so no explicit flow selection is needed. The command is still accepted, and still appears in older scripts, so that those scripts parse without modification. Remove it when revising a script.

Argument	Type	Required	Default	Notes
`flag`	integer	optional	`0`	Ignored

Applies to: —

```
# set_lrf_flow 1 <- remove; no effect in LVR 2.0
```

See also: read_lrf, read_svrf, set_gds_flow

Appendix A Commands by Operating Mode

Each column lists the commands that have effect in that init_tool_db mode. Commands marked *all* in their entry apply in every mode and are omitted here.

44 commands apply in all modes.

DRC (8 commands)

drc drc_max_error report_drc report_lefdef_drc run_drc run_qdrc set_report_file_drc set_report_file_lefdef_drc

LVS (226 commands)

dump_nets lvs lvs_check_macro lvs_check_matching lvs_check_open lvs_check_shorts lvs_check_unconnected_pins lvs_consider_macro_pins lvs_detect_open lvs_detect_short lvs_enable_ports lvs_ground_pin lvs_mapping_pin_to_terminal lvs_max_error lvs_option_max_error lvs_power_pin lvs_set_datc_flow lvs_set_macros_cdl_file_name lvs_set_schematic_file_name lvs_set_spice_file_name lvs_set_verilog_netlist_file_name lvs_timeout lvs_use_datc_flow lvs_waiver_macro_net lvs_waiver_netlist_net lvs_write_layout_from_db lvs_write_schematic_from_spice mlvs read_spice read_verilog report_lvs report_mlvs run_lvs set_lvs_abort_on_first_error set_lvs_alias_vgnd set_lvs_alias_vnb set_lvs_alias_vpb set_lvs_alias_vpwr set_lvs_assign_internal_nets set_lvs_cache_size_mb set_lvs_cdl_export_before_run set_lvs_cell_depth set_lvs_cell_filter set_lvs_check_all_cells set_lvs_check_top_only set_lvs_chunk_size set_lvs_connect_all_metals set_lvs_connect_layer set_lvs_connect_power_global set_lvs_csv_file set_lvs_db_file set_lvs_dbu set_lvs_def_file set_lvs_detailed_log_file set_lvs_device_report_file set_lvs_enable_csv_export set_lvs_enable_db_export set_lvs_enable_marker_file set_lvs_enable_waivers set_lvs_erc_report_file set_lvs_error_on_mismatch set_lvs_error_on_open set_lvs_error_on_short set_lvs_error_on_unmatched_inst set_lvs_extract_bjt set_lvs_extract_bulk_terminal set_lvs_extract_capacitor set_lvs_extract_diode set_lvs_extract_ldd set_lvs_extract_nmos set_lvs_extract_nwell_terminal set_lvs_extract_pmos set_lvs_extract_resistor set_lvs_extract_sub_terminal set_lvs_extracted_netlist set_lvs_fin_pitch set_lvs_flatten_cells set_lvs_flush_time set_lvs_gds_export_before_run set_lvs_gds_file set_lvs_global_net set_lvs_ground_pin set_lvs_ground_pin1 set_lvs_ground_pin2 set_lvs_ground_pin3 set_lvs_ground_pin4 set_lvs_hierarchical set_lvs_inst_match_timeout set_lvs_internal_net_prefix set_lvs_l_tolerance set_lvs_layer_map_file set_lvs_lef_file set_lvs_log_file set_lvs_lrf_file set_lvs_macro_cell set_lvs_macro_timeout set_lvs_macros_cdl_file set_lvs_marker_file set_lvs_match_allow_prefix set_lvs_match_bipartite set_lvs_match_by_name set_lvs_match_by_topology set_lvs_match_case_sensitive set_lvs_match_ignore_bulk set_lvs_match_min_percentage set_lvs_match_prefix set_lvs_match_report_file set_lvs_match_report_unmatched set_lvs_match_sort_based set_lvs_match_symmetric_sd set_lvs_max_error set_lvs_max_error_per_macro set_lvs_max_error_per_net set_lvs_memory_limit_mb set_lvs_net_report_file set_lvs_num_threads set_lvs_only_cell set_lvs_open_check_all_nets set_lvs_open_check_schematic set_lvs_open_max_per_net set_lvs_open_net set_lvs_open_report_file set_lvs_open_report_path set_lvs_open_use_via set_lvs_power_pin set_lvs_power_pin1 set_lvs_power_pin2 set_lvs_power_pin3 set_lvs_power_pin4 set_lvs_power_pin5 set_lvs_power_pin6 set_lvs_report_all_errors set_lvs_report_coordinates set_lvs_report_device_counts set_lvs_report_erc set_lvs_report_fail_cells set_lvs_report_file set_lvs_report_match_percent set_lvs_report_net_names set_lvs_report_opens set_lvs_report_pass_cells set_lvs_report_shorts set_lvs_report_timing set_lvs_report_unconnected set_lvs_rtree_node_size set_lvs_short_check_all_layers set_lvs_short_layer set_lvs_short_max_per_layer set_lvs_short_report_file set_lvs_short_report_nets set_lvs_short_use_rtree set_lvs_signature_file set_lvs_skill_callback set_lvs_skill_integration set_lvs_skip_cell set_lvs_skip_empty_cells set_lvs_skip_fill_cells set_lvs_spice_file set_lvs_spice_file1 set_lvs_spice_file2 set_lvs_summary_file set_lvs_summary_only set_lvs_svrf_file set_lvs_tech_file set_lvs_tech_node set_lvs_top_cdl_file set_lvs_top_cell set_lvs_total_timeout set_lvs_unit set_lvs_use_cache set_lvs_use_dbu set_lvs_use_legacy_extraction set_lvs_use_pins_for_nets set_lvs_use_rtree set_lvs_use_svrf_connect set_lvs_use_svrf_device_rules set_lvs_verbose set_lvs_verilog_file set_lvs_verilog_file1 set_lvs_verilog_file2 set_lvs_virtuoso_cell set_lvs_virtuoso_lib set_lvs_virtuoso_markers set_lvs_virtuoso_view set_lvs_w_tolerance set_lvs_waiver_audit_trail set_lvs_waiver_expiry_check set_lvs_waiver_file set_lvs_waiver_owner set_lvs_waiver_spatial_filter set_lvs_waiver_wildcard set_lvs_warning_as_e

rror set_mlvs_report_file set_perform_lvs set_perform_lvs_antenna set_perform_lvs_datc_flow s
et_perform_lvs_device_extract set_perform_lvs_erc set_perform_lvs_ground_check set_perform_lvs_
macro set_perform_lvs_matching set_perform_lvs_opens set_perform_lvs_parallel_reduce set_perfo
rm_lvs_port_check set_perform_lvs_power_check set_perform_lvs_property_check set_perform_lvs_se
ries_reduce set_perform_lvs_shorts set_perform_lvs_soft_connect set_perform_lvs_svs set_perfor
m_lvs_unconnected_pin set_perform_lvs_well_check set_perform_svrf_device_parse set_report_file_
lvs set_report_file_mlvs set_sch_file sky130_to_spice split_cdl_asap7 split_cdl_gpdk split_c
dl_ng15 split_cdl_ng45 split_cdl_saed32 split_cdl_sky130 split_cdl_xh018 test_lvs verilog2sc
h

MLVS (219 commands)

dump_nets lvs_check_macro lvs_check_matching lvs_check_open lvs_check_shorts lvs_check_unconn
ected_pins lvs_consider_macro_pins lvs_detect_open lvs_detect_short lvs_enable_ports lvs_grou
nd_pin lvs_mapping_pin_to_terminal lvs_power_pin lvs_set_datc_flow lvs_set_macros_cdl_file_nam
e lvs_set_schematic_file_name lvs_set_spice_file_name lvs_set_verilog_netlist_file_name lvs_ti
meout lvs_use_datc_flow lvs_waiver_macro_net lvs_waiver_netlist_net lvs_write_layout_from_db
lvs_write_schematic_from_spice mlvs read_spice read_verilog report_mlvs set_lvs_abort_on_firs
t_error set_lvs_alias_vgnd set_lvs_alias_vnb set_lvs_alias_vpb set_lvs_alias_vpwr set_lvs_ass
ign_internal_nets set_lvs_cache_size_mb set_lvs_cdl_export_before_run set_lvs_cell_depth set_l
vs_cell_filter set_lvs_check_all_cells set_lvs_check_top_only set_lvs_chunk_size set_lvs_conne
ct_all_metals set_lvs_connect_layer set_lvs_connect_power_global set_lvs_csv_file set_lvs_db_f
ile set_lvs_dbu set_lvs_def_file set_lvs_detailed_log_file set_lvs_device_report_file set_lvs
_enable_csv_export set_lvs_enable_db_export set_lvs_enable_marker_file set_lvs_enable_waivers
set_lvs_erc_report_file set_lvs_error_on_mismatch set_lvs_error_on_open set_lvs_error_on_short
set_lvs_error_on_unmatched_inst set_lvs_extract_bjt set_lvs_extract_bulk_terminal set_lvs_extr
act_capacitor set_lvs_extract_diode set_lvs_extract_ldd set_lvs_extract_nmos set_lvs_extract_n
well_terminal set_lvs_extract_pmos set_lvs_extract_resistor set_lvs_extract_sub_terminal set_l
vs_extracted_netlist set_lvs_fin_pitch set_lvs_flatten_cells set_lvs_flush_time set_lvs_gds_ex
port_before_run set_lvs_gds_file set_lvs_global_net set_lvs_ground_pin set_lvs_ground_pin1 se
t_lvs_ground_pin2 set_lvs_ground_pin3 set_lvs_ground_pin4 set_lvs_hierarchical set_lvs_inst_ma
tch_timeout set_lvs_internal_net_prefix set_lvs_l_tolerance set_lvs_layer_map_file set_lvs_lef
_file set_lvs_log_file set_lvs_lrf_file set_lvs_macro_cell set_lvs_macro_timeout set_lvs_macr
os_cdl_file set_lvs_marker_file set_lvs_match_allow_prefix set_lvs_match_bipartite set_lvs_mat
ch_by_name set_lvs_match_by_topology set_lvs_match_case_sensitive set_lvs_match_ignore_bulk se
t_lvs_match_min_percentage set_lvs_match_prefix set_lvs_match_report_file set_lvs_match_report_
unmatched set_lvs_match_sort_based set_lvs_match_symmetric_sd set_lvs_max_error set_lvs_max_er
ror_per_macro set_lvs_max_error_per_net set_lvs_memory_limit_mb set_lvs_net_report_file set_lv
s_num_threads set_lvs_only_cell set_lvs_open_check_all_nets set_lvs_open_check_schematic set_l
vs_open_max_per_net set_lvs_open_net set_lvs_open_report_file set_lvs_open_report_path set_lvs
_open_use_via set_lvs_power_pin set_lvs_power_pin1 set_lvs_power_pin2 set_lvs_power_pin3 set_
lvs_power_pin4 set_lvs_power_pin5 set_lvs_power_pin6 set_lvs_report_all_errors set_lvs_report_
coordinates set_lvs_report_device_counts set_lvs_report_erc set_lvs_report_fail_cells set_lvs_
report_file set_lvs_report_match_percent set_lvs_report_net_names set_lvs_report_opens set_lvs
_report_pass_cells set_lvs_report_shorts set_lvs_report_timing set_lvs_report_unconnected set_
lvs_rtree_node_size set_lvs_short_check_all_layers set_lvs_short_layer set_lvs_short_max_per_la
yer set_lvs_short_report_file set_lvs_short_report_nets set_lvs_short_use_rtree set_lvs_signat
ure_file set_lvs_skill_callback set_lvs_skill_integration set_lvs_skip_cell set_lvs_skip_empty
_cells set_lvs_skip_fill_cells set_lvs_spice_file set_lvs_spice_file1 set_lvs_spice_file2 set
_lvs_summary_file set_lvs_summary_only set_lvs_svrf_file set_lvs_tech_file set_lvs_tech_node
set_lvs_top_cdl_file set_lvs_top_cell set_lvs_total_timeout set_lvs_unit set_lvs_use_cache se
t_lvs_use_dbu set_lvs_use_legacy_extraction set_lvs_use_pins_for_nets set_lvs_use_rtree set_lv
s_use_svrf_connect set_lvs_use_svrf_device_rules set_lvs_verbose set_lvs_verilog_file set_lvs_
verilog_file1 set_lvs_verilog_file2 set_lvs_virtuoso_cell set_lvs_virtuoso_lib set_lvs_virtuos
o_markers set_lvs_virtuoso_view set_lvs_w_tolerance set_lvs_waiver_audit_trail set_lvs_waiver_
expiry_check set_lvs_waiver_file set_lvs_waiver_owner set_lvs_waiver_spatial_filter set_lvs_wa
iver_wildcard set_lvs_warning_as_error set_mlvs_report_file set_perform_lvs set_perform_lvs_an
tenna set_perform_lvs_datc_flow set_perform_lvs_device_extract set_perform_lvs_erc set_perform
_lvs_ground_check set_perform_lvs_macro set_perform_lvs_matching set_perform_lvs_opens set_per
form_lvs_parallel_reduce set_perform_lvs_port_check set_perform_lvs_power_check set_perform_lvs

_property_check set_perform_lvs_series_reduce set_perform_lvs_shorts set_perform_lvs_soft_connect set_perform_lvs_svs set_perform_lvs_unconnected_pin set_perform_lvs_well_check set_perform_svrf_device_parse set_report_file_mlvs set_sch_file sky130_to_spice split_cdl_asap7 split_cdl_gpdk split_cdl_ng15 split_cdl_ng45 split_cdl_saed32 split_cdl_sky130 split_cdl_xh018 verilog2sch

SVS (20 commands)

dump_nets lvs_enable_ports read_spice read_verilog report_svs run_svs set_lvs_dbu set_lvs_tech_node set_lvs_unit set_lvs_use_dbu set_report_file_svs set_sch_file set_svs1_spice set_svs1_verilog set_svs2_spice set_svs2_verilog set_svs_report_file sky130_to_spice svs verilog2sch

ERC (40 commands)

dump_nets erc lvs_ground_pin lvs_power_pin read_spice report_erc run_erc set_erc_check_dangling_output set_erc_check_floating_bulk set_erc_check_floating_gate set_erc_check_multi_driver set_erc_check_power_short set_erc_check_short_to_ground set_erc_check_short_to_power set_erc_check_unconnected_input set_erc_check_undriven_power set_erc_max_error set_erc_report_info set_lvs_alias_vgnd set_lvs_alias_vnb set_lvs_alias_vpb set_lvs_alias_vpwr set_lvs_dbu set_lvs_global_net set_lvs_ground_pin set_lvs_ground_pin1 set_lvs_ground_pin2 set_lvs_ground_pin3 set_lvs_ground_pin4 set_lvs_power_pin set_lvs_power_pin1 set_lvs_power_pin2 set_lvs_power_pin3 set_lvs_power_pin4 set_lvs_power_pin5 set_lvs_power_pin6 set_lvs_tech_node set_lvs_unit set_lvs_use_dbu set_report_file_erc

XOR (11 commands)

read_gds2 report_fastxor report_xor run_fastxor run_xor set_gui_xor_result set_report_file_fastxor set_report_file_xor set_xor1_gds set_xor2_gds xor_check

FASTXOR (7 commands)

read_gds2 report_fastxor run_fastxor set_gui_xor_result set_report_file_fastxor set_xor1_gds set_xor2_gds

DENSITY (8 commands)

density density_layer density_pixel density_window rep_density report_density run_density set_report_file_density

DFM (7 commands)

density_layer density_pixel density_window dfm report_dfm run_dfm set_report_file_dfm

ANTENNA (5 commands)

antenna rep_antenna report_antenna run_antenna set_report_file_antenna

PEX (57 commands)

compare_spefs pex set_pex_cap_unit set_pex_cg_max_iter set_pex_cg_tolerance set_pex_dbu_to_um set_pex_design_name set_pex_dump_rc set_pex_em_verbose_segments set_pex_em_violation_ratio set_pex_exclude_layers set_pex_exclude_nets set_pex_extract_bump set_pex_extract_capacitance set_pex_extract_coupling_cap set_pex_extract_fringe_cap set_pex_extract_ground_cap set_pex_extract_resistance set_pex_extract_tap set_pex_extract_via_resistance set_pex_flush_time set_pex_include_layers set_pex_inter_net_coupling_only set_pex_ir_drop_threshold_mv set_pex_ir_verbose_nodes set_pex_ladder_seg_len_um set_pex_log_file set_pex_max_coupling_dist_um set_pex_max_couplin

g_errors set_pex_max_em_errors set_pex_max_errors set_pex_max_ir_errors set_pex_max_ladder_segs set_pex_min_cap_ff set_pex_min_coupling_cap_ff set_pex_min_resistance_ohm set_pex_net_filter set_pex_num_threads set_pex_rc_reduction set_pex_report_dspf set_pex_report_em set_pex_report_file set_pex_report_ir set_pex_report_markers set_pex_report_sdf set_pex_report_spef set_pex_res_unit set_pex_run_em set_pex_run_extraction set_pex_run_ir_drop set_pex_skip_power_nets set_pex_spef_vendor set_pex_spef_version set_pex_supply_voltage set_pex_tech_file set_pex_time_unit set_pex_verbosity

LEF/DEF flows (53 commands)

check_allpins check_allrect check_allsegs check_allvias check_antenna check_antenna_cut check_area check_cut_enclosure check_cut_enclosureedge check_cut_enclosureedge_opposite check_cut_forbidden_spacing check_cut_on_centerline check_cut_short check_cut_spacing check_dangling_wire check_enclosure_to_joint check_eol_extn_spacing check_eol_keepout check_floating_patch check_invalid_cut check_loop check_metal_area_spacing check_metal_eol_spacing check_metal_forbidden_spacing check_metal_joint_corner_spacing check_metal_spacing check_min_cut check_min_step check_minimal_width check_nsmetal check_off_grid check_open_check check_out_of_die check_powersegs check_rect_only check_shorts check_span_length_table check_table_cut_spacing check_table_prl_spacing check_unconnpin check_via_in_pin check_widthtable check_wrongway load_def_db no_cells no_obs no_pwr no_rect no_vias read_def read_lef run_lefdef_drc set_lefdef_flow

Appendix B Minimal Scripts

The shortest script that produces a valid result for each engine. Add reporting and resource commands as required.

DRC

```
init_tool_db DRC
set_technode sky130
set_top gcd1_sky130hd
read_lrf ../rule/sky130.lrf
read_gds ../gds/gcd1_sky130hd.gds
load_db DRC
drc gcd1_sky130hd
set_report_file_drc ./out/drc.rpt
report_drc
```

LVS

```
init_tool_db LVS
set_technode sg13g2
set_top DiffAmp_NOFILL
read_svrf ../rule/sg13g2.extract.cal
read_gds ../gds/DiffAmp_NOFILL.gds
set_lvs_spice_file ../netlist/DiffAmp.cdl
set_lvs_power_pin VDD
set_lvs_ground_pin VSS
load_db LVS
lvs DiffAmp_NOFILL
set_lvs_report_file ./out/lvs.rpt
report_lvs
```

ERC

```
init_tool_db ERC
set_technode sky130
set_top gcd1_sky130hd
read_lrf ../rule/sky130.lrf
read_gds ../gds/gcd1_sky130hd.gds
set_lvs_power_pin VPWR
set_lvs_ground_pin VGND
set_erc_check_floating_gate 1
set_erc_check_short_to_power 1
load_db ERC
erc gcd1_sky130hd
set_report_file_erc ./out/erc.rpt
report_erc
```

XOR

```
init_tool_db XOR
set_technode sky130
set_top gcd1_sky130hd
set_xor1_gds ../gds/revA.gds
set_xor2_gds ../gds/revB.gds
load_db XOR
xor_check gcd1_sky130hd
set_report_file_xor ./out/xor.rpt
report_xor
```

DENSITY

```
init_tool_db DENSITY
set_technode generic22
set_top analog_core
read_svrf ../rule/foundry_deck/density.cal
read_gds ../gds/analog_core.gds
density_layer Metal1
density_layer Metal2
density_window 100000 100000 50000 50000 1
load_db DENSITY
density analog_core
set_report_file_density ./out/density.rpt
report_density
```

PEX

```
init_tool_db PEX
set_technode generic22
set_top top_soc
read_gds ../gds/top_soc.gds
set_pex_tech_file ../tech/tech.pextech
set_pex_run_extraction 1
set_pex_extract_resistance 1
set_pex_extract_capacitance 1
set_pex_extract_coupling_cap 1
set_pex_report_spef ./out/top_soc.spef
load_db PEX
pex top_soc
```

Appendix C LRF Rule Language Reference

LRF is LVR's native rule format, read by read_lrf. It is an alternative to SVRF, not a supplement: a single run uses one or the other, never both. LRF is more compact than SVRF for hand-written decks and is the format used by the shipped examples.

Every derivation operator accepts an optional -layerOut. Supplying it names the result as a persistent layer that later statements can reference; omitting it leaves the result in the evaluation pipeline for the next operator to consume, which is the usual form inside a rule.

Rule structure

An LRF file is a sequence of statements. Layer declarations and mappings come first, then derivations, then the rules themselves. A rule is introduced by drc rule or density rule, carries a caption that becomes the marker text, and contains one or more measurement operators.

`drc rule`

`drc rule "<name>"`

Opens a design rule. The name appears as the rule identifier on every marker the rule produces, and is what correlation scripts key on, so it should match the foundry design rule manual exactly.

```
drc rule "dnwell.2"
```

`caption`

`caption "<text>"`

Supplies the human-readable text carried on each marker. By convention this repeats the rule identifier and states the requirement, so that a marker is self-explanatory in a viewer without reference back to the deck.

```
caption "dnwell.2 : min. dnwell width : 3.0um"
```

`density rule`

`density rule "<name>"`

Opens a density rule. Density rules measure fill ratio within a window rather than distances between edges, and report windows falling outside the permitted range.

```
density rule "min fail"
density rule "max fail"
```

Layer declaration and mapping

These statements bind layer names to GDS layer numbers and datatypes. They must appear before any derivation or rule that references the layers they declare.

lvr layer def

`lvr layer def <name> <number>`

Declares a layer name and binds it to a GDS layer number. The declared name is what every later statement refers to.

```
lvr layer def BOUND 30000
```

lvr layer map

`lvr layer map <number> -datatype <n> <target>`

Maps a GDS layer number and datatype onto a target layer. Use the -texttype form for text records rather than geometry. Mapping is how a single GDS layer number carrying several datatypes is separated into distinct logical layers.

Option	Meaning
-datatype <n>	Match geometry records with this datatype
-texttype <n>	Match text records with this texttype

```
lvr layer map 235 -datatype 4 30000
lvr layer map 74 -texttype 5 30320
```

connect

`connect <layer1> <layer2>`

Declares that two layers are electrically connected. Connect statements build the connectivity graph that LVS and ERC operate on; a missing statement produces opens that are not in the layout, and a spurious one produces shorts that are not there.

```
connect MET1 MET2
```

lvs timeout

`lvs timeout <seconds>`

Bounds the LVS comparison from within the rule file rather than the run script. Useful where a deck is distributed with a known-safe limit.

```
lvs timeout 10
```

Boolean operators

Boolean operators derive new layers from existing ones. Every operator accepts an optional -layerOut; when it is omitted the result stays in the evaluation pipeline and is consumed by the next operator, which is the usual form inside a rule.

and

`and <layerA> <layerB> [-layerOut <out>]`

Boolean intersection. Produces the regions covered by both inputs. The standard way to derive a doped region from a diffusion layer and an implant layer.

Option	Meaning
`-layerOut &lt;out&gt;`	Name the result as a persistent layer

```
and DIFF NSDM -layerOut NDIFF
and DIFF NSDM
```

or

`or <layerA> <layerB> [-layerOut <out>]`

Boolean union. Produces the regions covered by either input. Overlapping shapes merge into single polygons, which matters when a later operator measures area or perimeter.

Option	Meaning
`-layerOut &lt;out&gt;`	Name the result as a persistent layer

```
or PSD NSD -layerOut MYDIFF
or PSD NSD
```

not

`not <layerA> <layerB> [-layerOut <out>]`

Boolean subtraction. Removes from the first operand every region covered by the second. Deriving the substrate region by subtracting the n-well from the bulk extent is the canonical use.

Option	Meaning
`-layerOut &lt;out&gt;`	Name the result as a persistent layer

```
not bulk NWELL -layerOut PWELL
not bulk NWELL
```

get intersecting

`get intersecting [-neg] <layerA> <layerB> [<layerC>] [-layerOut <out>]`

Selects whole shapes from the first layer that touch or overlap the others. Unlike and, which clips to the overlap region, this returns the complete selected shapes. -neg inverts the selection, returning shapes that do *not* intersect, which is how isolated or unlanded features are found.

Option	Meaning
`-neg`	Select shapes that do not intersect
`-layerOut &lt;out&gt;`	Name the result as a persistent layer

```
get intersecting LICON DIFF -layerOut diff_licon
get intersecting -neg npc poly licon
get intersecting -neg npc poly licon -layerOut unlanded
```

`get corners`

`get corners <layer> [-layerOut <out>]`

Extracts corner features from the input layer. Corners are subject to their own spacing and rounding rules in most processes, so isolating them is the first step in any corner-specific check.

Option	Meaning
`-layerOut <out>`	Name the result as a persistent layer

```
get corners DIFF -layerOut CD
get corners DIFF
```

Sizing

Sizing grows or shrinks geometry. It is the basis of keep-out construction, context envelopes and worst-case misalignment analysis.

`size`

`size (extent) -by <value> [-layerOut <out>]`

Offsets geometry by the given amount; positive grows, negative shrinks. In extent mode the operation applies to the minimal axis-aligned bounding rectangle covering all shapes on the layer, rather than to each polygon independently — which preserves the overall footprint while discarding interior detail. Used to build guard and keep-out regions and to enlarge search regions ahead of a Boolean operation.

Option	Meaning
`(extent)`	Operate on the global bounding extent rather than per polygon
`-by <value>`	Offset amount in microns; positive grows, negative shrinks
`-layerOut <out>`	Name the result as a persistent layer

```
size (extent) -by 1.0 -layerOut bulk
```

Measurement operators

Measurement operators compare a geometric quantity against a limit and emit a marker where the comparison fails. All share the same comparison suffixes.

edge width

`edge width <layer> -lt|-gt|-eq <value>`

Measures the width of features on a layer. -lt flags features narrower than the limit, which is the minimum-width form and by far the most common; -gt flags features wider than the limit, used for maximum-width rules on wide-metal and stress-sensitive layers.

Option	Meaning
-lt <value>	Flag measurements less than the value
-gt <value>	Flag measurements greater than the value
-eq <value>	Flag measurements equal to the value

```
edge width DNWELL -lt 3.0
edge width DNWELL -gt 3.0
```

edge length

`edge length <layer> -lt|-gt <value>`

Measures edge length rather than feature width. Used for end-of-line and minimum edge rules, where a short edge is difficult to resolve lithographically.

Option	Meaning
-lt <value>	Flag edges shorter than the value
-gt <value>	Flag edges longer than the value

```
edge length MF -lt 7.2
```

area

`area <layer> -lt|-gt <value> [-layerOut <out>]`

Measures polygon area in square microns. The minimum-area form catches metal islands too small to pattern reliably. With -layerOut, the failing shapes are captured as a layer for further processing rather than reported directly.

Option	Meaning
-lt <value>	Flag polygons smaller than the value
-gt <value>	Flag polygons larger than the value
-layerOut <out>	Capture failing shapes as a layer

```
area MET3 -lt 0.24
area MET3 -lt 0.24 -layerOut temp
```

spacing

```
spacing [-same layer] [-wider <w>] [-longer <l>] <layerA> <layerB> -lt <value>
```

Measures the distance between edges on two layers, or within one layer with -same layer. The -wider and -longer qualifiers express parallel-run-length rules, in which the required spacing increases once two edges are both wider than a threshold and run parallel for longer than a threshold. This conditional form is what most advanced-node metal spacing rules use.

Option	Meaning
-same layer	Measure between shapes on the same layer
-wider <w>	Apply only where the feature is wider than this
-longer <l>	Apply only where the parallel run exceeds this length
-lt <value>	Flag spacings less than the value

```
spacing -same layer DNWELL DNWELL -lt 6.3
spacing LVTN HVTP -lt 0.38
spacing -wider 1.5 -longer 4.0 -same layer metal8 metal8 -lt 1.5
```

extension

```
extension <layerA> <layerB> -lt|-eq <value>
```

Measures how far one layer extends beyond another. Used for gate extension beyond diffusion and similar overhang requirements. -eq 0 checks for exact coincidence, which is how a required flush alignment is expressed.

Option	Meaning
-lt <value>	Flag extensions less than the value
-eq <value>	Flag extensions equal to the value

```
extension DNWELL DNWELL -eq 0
extension DNWELL DNWELL -lt 0.1
```

enclosure

```
enclosure -all side|-opposite side <inner> <outer> -lt <value> [-layerOut <out>]
```

Measures how much the outer layer surrounds the inner one. -all side requires the enclosure on every side simultaneously; -opposite side requires it on opposing pairs, which is the form most via landing rules use because it permits an asymmetric landing pad. Choosing the wrong variant is a common source of both false and missed violations.

Option	Meaning
-all side	Require the enclosure on every side
-opposite side	Require the enclosure on opposing sides
-lt <value>	Flag enclosures less than the value
-layerOut <out>	Capture failing shapes as a layer

```
enclosure -all side VARAC NWELL -lt 0.15
enclosure -all side VARAC NWELL -lt 0.15 -layerOut temp
enclosure -opposite side licon poly POLY -lt 0.08
```

Appendix D Worked LRF Rule

A complete minimum-width rule, showing declaration, derivation, rule opening, caption and measurement in the order they must appear.

```
# ---- layer declarations ----------------------------------------
lvr layer def DNWELL 64
lvr layer def NWELL 20
lvr layer def DIFF 65
lvr layer def NSDM 93
lvr layer map 65 -datatype 20 DIFF

# ---- connectivity ----------------------------------------------
connect MET1 MET2

# ---- derivations -----------------------------------------------
size (extent) -by 1.0 -layerOut bulk
not bulk NWELL -layerOut PWELL
and DIFF NSDM -layerOut NDIFF

# ---- rules -----------------------------------------------------
drc rule "dnwell.2"
caption "dnwell.2 : min. dnwell width : 3.0um"
edge width DNWELL -lt 3.0

drc rule "dnwell.3"
caption "dnwell.3 : min. dnwell spacing : 6.3um"
spacing -same layer DNWELL DNWELL -lt 6.3

drc rule "met3.1"
caption "met3.1 : min. met3 area : 0.24um2"
area MET3 -lt 0.24
```