RAMTERA

PRODUCT DOCUMENTATION · PHYSICAL IMPLEMENTATION (PRE)

RAMTERA-PRE-CMD

PRE Command Reference

All 135 registered Tcl commands, grouped by flow stage — extracted from the shipping registration tables

August 2026 · Ramtera PRE engineering

How to read this reference

PRE's command surface follows industry-standard implementation-tool conventions: SDC and UPF files source directly (both are Tcl), option grammars match the documented forms, and options not yet honored by the engine are accepted and recorded with an explicit acknowledgement rather than rejected — the acknowledgement line always states what was honored. This reference lists every registered command with a one-line synopsis and the load-bearing options; full option grammars follow the industry-standard documented forms.

135	12	native	recorded
commands	flow-stage groups	SDC + UPF sourcing	unimplemented options acknowledged

Design data in / out (10)

Command	Synopsis
`read_lef`	technology + macros: layers, sites, pins, antenna properties
`read_def`	design: die, rows, tracks, components, pins, nets
`read_verilog`	structural gate-level netlist
`read_liberty`	timing library: cells, arcs, NLDM tables, pin caps
`spefIn`	IEEE 1481 parasitics; -rc_corner -spef_field -scaleRC -keep_star_node
`read_parasitics`	alias of spefIn
`write_def`	placement + routing snapshot out
`streamOut`	hierarchical GDSII; -mapFile -libName -structureName -units
`write_gds`	alias of streamOut
`loadTestDesign`	synthetic design generator for bring-up; -cells N

Floorplanning (33)

Command	Synopsis
`floorPlan`	die/core by size or -r aspect/utilization form
`setCoreUtilization`	target utilization for floorPlan -r
`setAspectRatio`	core aspect ratio
`setRectilinearCore`	non-rectangular core outline by point list
`addRows`	site rows over the core or a box
`flipRows`	row orientation parity
`addTracks`	routing tracks per layer, pitch and offset
`planDesign`	macro seeding + row generation in one step
`placeMacro`	place one macro at x y with orientation
`setMacroOrient`	orientation of a placed macro
`setMacroPadding`	keep-out padding around a macro
`addHaloToBlock`	placement halo around a block
`addBlockageHalo`	halo expressed as a blockage

Command	Synopsis
createPlaceBlockage	hard placement blockage box
createPartialBlockage	density-limited (partial) blockage
createRouteBlockage	routing blockage per layer
createFence	hard region: cells of a group must stay inside
createRegion	soft region preference
addPlaceGuide	placement guide box
definePartition	partition definition for hierarchical flows
assignPartitionPins	pin assignment on partition boundaries
editPin	IO pin position/layer/side edits
setPinConstraint	side/order/spacing constraints for IO placement
addPadRing	pad ring generation
planBump	bump array planning
addPowerSeed	seed points for power planning
relativeFPlan	relative floorplan constraints between objects
editFloorplan	interactive floorplan edit entry
snapFloorplan	snap floorplan objects to grid
checkFloorPlan	row/track/blockage consistency checks
saveFloorPlan	floorplan to file
loadFloorPlan	floorplan from file
clearFloorPlan	reset floorplan state

Placement (8)

Command	Synopsis
place_design	global place + legalize + detail (quadratic path)
place	alias of place_design
place_opt_design	full flow; nonlinear engine when setPlaceMode -nlp true
place_global_nlp	nonlinear global placement stage alone
setPlaceMode	-nlp true|false -place_global_max_density -quality
legalizePlacement	row legalization
refinePlace	incremental swap-based detail placement
checkPlace	legality + out-of-core + HPWL report

Clock tree synthesis (6)

Command	Synopsis
create_clock_tree_spec	tree spec from placed flops or -sinks
ccopt_design	H-tree synthesis; buffer sized from setCTSMode -bufferList
setCTSMode	-bufferList -routeClkNet -routeNonDefaultRule -route*PreferredLayer

Command	Synopsis
`create_route_rule`	-name -width_multiplier/-spacing_multiplier or -width {layer w}
`insert_clock_buffers`	tree buffers into the netlist; -buf overrides sizing
`clock_tree_debugger`	tree topology report

Routing (4)

Command	Synopsis
`globalRoute`	PathFinder global routing over GCell grid
`routeDesign`	detailed routing; -layers N; applies bound clock NDRs
`clearRoute`	remove routing
`reportRoute`	routing statistics

Timing constraints (SDC — sourced natively) (15)

Command	Synopsis
`create_clock`	-period -name on a source
`set_input_delay`	external arrival at inputs
`set_output_delay`	external required at outputs
`set_wire_rc`	-res -cap wire model for unannotated nets
`set_false_path`	exception: -from/-through/-to
`set_multicycle_path`	N -setup|-hold with documented priority
`set_max_delay`	path max-delay exception
`set_min_delay`	path min-delay exception (hold)
`reset_path`	remove matching exceptions
`report_path_exceptions`	applied/ignored exception table; -ignored
`get_ports`	object accessor (name echo)
`get_pins`	object accessor
`get_cells`	object accessor
`get_clocks`	object accessor
`get_nets`	object accessor

Timing analysis and optimization (7)

Command	Synopsis
`timeDesign`	STA; MMMC-aware; distributes when hosts configured
`report_timing`	worst paths with per-hop arrivals
`optDesign`	-postCTS [-hold]: setup fixing / hold fixing
`set_timing_derate`	early/late derates per view
`reset_timing_derate`	clear derates
`setAnalysisMode`	-checkType setup|hold

Command	Synopsis
write_sta_result	single-line result dump (distributed collection)

Multi-mode multi-corner (MMMC) (13)

Command	Synopsis
create_library_set	-name -timing {libs}
create_rc_corner	-name -postRoute_cap -postRoute_res scaling
create_delay_corner	-name -library_set -rc_corner
create_constraint_mode	-name -sdc_files {files}
create_analysis_view	-name -constraint_mode -delay_corner
set_analysis_view	-setup {views} -hold {views}
all_analysis_views	query
all_setup_analysis_views	query
all_hold_analysis_views	query
all_constraint_modes	query
all_delay_corners	query
all_rc_corners	query
all_library_sets	query

Multi-CPU and distributed (5)

Command	Synopsis
setMultiCpuUsage	-localCpu N -remoteHost K
getMultiCpuUsage	current settings
checkMultiCpuUsage	validation report
setDistributeHost	-local | -rsh -host {h..} | -lsf -queue Q; -setupScript -exec
reportThreads	worker pool status

Power intent (UPF — sourced natively) (15)

Command	Synopsis
create_power_domain	domain over element list
set_domain_area	domain fence box (drives placement fences)
set_domain_voltage	domain voltage
create_supply_net	supply net
create_supply_port	supply port
connect_supply_net	net-to-domain connection
set_isolation	isolation strategy on a domain
set_isolation_control	enable net/sense for a strategy
map_isolation_cell	strategy to library cell

Command	Synopsis
`set_level_shifter_cell`	level shifter selection
`insert_isolation`	execute isolation insertion at crossings
`optPower`	leakage recovery by downsizing, re-time + revert guard
`report_power`	leakage + switching estimate
`report_upf`	power-intent state
`clear_upf`	reset power intent

Power grid and rail analysis (10)

Command	Synopsis
`addRing`	core power rings
`addStripe`	stripe arrays per layer
`sroute`	follow-pin standard-cell rails
`addPGPad`	power pad markers (solver boundary)
`commitPowerPlan`	publish PG geometry
`set_rail_rc`	sheet resistances / via resistance
`analyzeRail`	static IR solve per net
`report_rail`	worst drop / bounce table
`verify_power_em`	current-density check vs limits
`clearRail`	reset PG state

Physical completion and signoff prep (9)

Command	Synopsis
`setEndCapMode`	-leftCell -rightCell
`addEndCap`	row end caps (fixed)
`addWellTap`	-cellInterval -checkerBoard -inRowOffset -avoidAbutment
`setFillerMode`	-fitGap -markFixed and filler list
`addFiller`	gap fill, largest-first; exact-area accounting
`deleteFiller`	remove physical-only cells by -prefix
`verifyProcessAntenna`	cumulative charge-ratio check; -detailed -report
`attachDiode`	protection diode beside a pin; -diodeCell -pin
`fixAntenna`	-layerJump and/or -diodeCell repair loop

PROVENANCE — This reference is generated from the source registration tables, so the count and the shipping binary cannot disagree. Aliases (place/place_design, write_gds/streamOut, read_parasitics/spefIn) are listed individually because scripts use both names.

About Ramtera

Ramtera Inc. (Delaware, USA), with R&D at Ramtera Design Automation India Pvt Ltd (Bangalore), builds electronic design automation software for physical verification and physical implementation. LVR is Ramtera's physical verification engine (DRC, LVS, ERC, XOR, parasitic extraction); PRE is Ramtera's physical implementation environment (floorplanning, placement, clock tree synthesis, routing, timing analysis and optimization), built on industry-standard command conventions so existing flow scripts and operator skills transfer directly. The company holds granted patents and registered copyrights covering its engines.

Contact

Web	www.ramtera.com
Inquiries	sameer@ramtera.com
Engineering	Bangalore, Karnataka, India

This document reports measurements from Ramtera's engineering environment (Ubuntu 24.04, GCC 13.3, Tcl 8.6) unless stated otherwise. Product behavior and figures are subject to change; no compatibility, certification, or endorsement by any third party is stated or implied.