

Virtuoso Liberate Reference Manual

Product Version 16.1

March 2017

© 2006–2017 Cadence Design Systems, Inc. All rights reserved.

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Trademarks: Trademarks and service marks of Cadence Design Systems, Inc. contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence's trademarks, contact the corporate legal department at the address shown above or call 800.862.4522. All other trademarks are the property of their respective holders

Restricted Permission: This publication is protected by copyright law and international treaties and contains trade secrets and proprietary information owned by Cadence. Unauthorized reproduction or distribution of this publication, or any portion of it, may result in civil and criminal penalties. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. Unless otherwise agreed to by Cadence in writing, this statement grants Cadence customers permission to print one (1) hard copy of this publication subject to the following conditions:

1. The publication may be used only in accordance with a written agreement between Cadence and its customer.
2. The publication may not be modified in any way.
3. Any authorized copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement.
4. The information contained in this document cannot be used in the development of like products or software, whether for internal or external use, and shall not be used for the benefit of any other party, whether or not for consideration.

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

Restricted Rights: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

Contents

| | |
|---|----|
| Preface | 31 |
| <u>Introduction to Characterization</u> | 31 |
| <u>The Role and Importance of Libraries</u> | 31 |
| <u>A Growing Problem</u> | 31 |
| <u>Virtuoso Characterization Suite</u> | 33 |
| <u>System Requirements</u> | 34 |
| <u>Software and Licensing Requirements</u> | 35 |
| <u>About This Manual</u> | 35 |
| <u>Audience Profile</u> | 35 |
| <u>Additional Documents for Reference</u> | 36 |
| <u>Rapid Adoption Kits</u> | 36 |
| <u>Typographic and Syntax Conventions</u> | 37 |
| <u>Customer Support</u> | 37 |
| <u>Feedback about Documentation</u> | 38 |
| 1 | |
| Introduction | 39 |
| <u>What is Liberate?</u> | 39 |
| <u>Performance is Key</u> | 39 |
| 2 | |
| Getting Started | 41 |
| <u>Environment Variables</u> | 41 |
| <u>Path to Executable</u> | 41 |
| <u>Shell Environment Variables for Controlling Licensing Checks</u> | 42 |
| <u>Managing Licenses</u> | 42 |
| <u>Wait for Available License</u> | 43 |
| <u>Invoking Liberate</u> | 44 |
| <u>System Libraries</u> | 44 |

| | |
|---|----|
| <u>Preparing for Characterization</u> | 45 |
| <u>Extracted Cell Netlist</u> | 45 |
| <u>Device Models</u> | 45 |
| <u>Tcl Command File</u> | 45 |
| <u>Running Liberate</u> | 47 |
| <u>Using Spectre</u> | 48 |

3

Parallel Processing

| | |
|--|----|
| <u>Multi-threading</u> | 51 |
| <u>Distributed Processing</u> | 51 |
| <u>Using a Queuing System</u> | 51 |
| <u>Packet Mode</u> | 52 |
| <u>Arc Packet Flow</u> | 52 |
| <u>Enabling the Arc Packet Flow</u> | 53 |
| <u>Interpreting the logs2xlsx Results</u> | 56 |
| <u>File Organization for the Arc Packet Flow</u> | 57 |
| <u>Recovery Flow</u> | 57 |
| <u>Frequently Asked Questions</u> | 58 |
| <u>set_client Mode (Non-Packet Mode)</u> | 58 |
| <u>Spectre Kernel Interface</u> | 59 |
| <u>Enabling SKI</u> | 59 |

4

Liberate Commands

| | |
|---------------------------------------|----|
| <u>add_cell_attribute</u> | 64 |
| <u>add_lib_attribute</u> | 65 |
| <u>add_margin</u> | 66 |
| <u>add_pin_attribute</u> | 70 |
| <u>append_library</u> | 71 |
| <u>char_library</u> | 73 |
| <u>check_delay_monotonicity</u> | 80 |
| <u>check_lvf_merge_indices</u> | 82 |
| <u>compare_ccs_nldm</u> | 83 |
| <u>compare_library</u> | 86 |

Virtuoso Liberate Reference Manual

| | |
|--|-----|
| <u>copy_arc</u> | 101 |
| <u>define_arc</u> | 105 |
| <u>define_bundle_pins</u> | 121 |
| <u>define_bus</u> | 123 |
| <u>define_cell</u> | 125 |
| <u>define_cell_leakage</u> | 134 |
| <u>define_duplicate_cell</u> | 136 |
| <u>define_duplicate_pins</u> | 137 |
| <u>define_group</u> | 138 |
| <u>define_index</u> | 139 |
| <u>define_input_waveform</u> | 141 |
| <u>define_leafcell</u> | 144 |
| <u>define_leakage</u> | 148 |
| <u>define_map</u> | 152 |
| <u>define_max_capacitance_attr_limit</u> | 153 |
| <u>define_max_capacitance_limit</u> | 154 |
| <u>define_max_transition</u> | 155 |
| <u>define_min_transition</u> | 156 |
| <u>define_out_to_out_arc</u> | 157 |
| <u>define_pin_load</u> | 158 |
| <u>define_pulse_generator_arc</u> | 160 |
| <u>define_template</u> | 161 |
| <u>delete_arc</u> | 165 |
| <u>em_buffer_cell_pin_bounds</u> | 167 |
| <u>em_follow_hidden_power</u> | 168 |
| <u>get_cells</u> | 169 |
| <u>get_var</u> | 172 |
| <u>ibis_define_component</u> | 173 |
| <u>ibis_define_header</u> | 175 |
| <u>ibis_define_model</u> | 176 |
| <u>ibis_define_model_selector</u> | 178 |
| <u>ibis_define_pin</u> | 179 |
| <u>ibis_define_waveform_template</u> | 181 |
| <u>merge_library</u> | 185 |
| <u>one_cold</u> | 190 |
| <u>one_hot</u> | 191 |

| | |
|------------------------------------|-----|
| <u>packet slave cells</u> | 192 |
| <u>printvars</u> | 193 |
| <u>read ldb</u> | 194 |
| <u>read library</u> | 197 |
| <u>read spice</u> | 199 |
| <u>read truth table</u> | 200 |
| <u>read vdb</u> | 204 |
| <u>remove pin attribute</u> | 205 |
| <u>reset defaults</u> | 206 |
| <u>select index</u> | 207 |
| <u>set aging criteria</u> | 208 |
| <u>set attribute</u> | 211 |
| <u>set client</u> | 215 |
| <u>set conditional</u> | 216 |
| <u>set constraint</u> | 217 |
| <u>set constraint criteria</u> | 222 |
| <u>set default group</u> | 229 |
| <u>set dependent load</u> | 232 |
| <u>set driver cell</u> | 233 |
| <u>set driver waveforms file</u> | 236 |
| <u>set em skip monitor</u> | 237 |
| <u>set gnd / set vdd</u> | 239 |
| <u>set input voltage</u> | 242 |
| <u>set logic condition</u> | 243 |
| <u>set max fanout</u> | 244 |
| <u>set network port</u> | 245 |
| <u>set operating condition</u> | 246 |
| <u>set output voltage</u> | 247 |
| <u>set pin attribute</u> | 248 |
| <u>set pin capacitance</u> | 250 |
| <u>set pin delay threshold</u> | 254 |
| <u>set pin gnd</u> | 255 |
| <u>set pin slew threshold</u> | 256 |
| <u>set pin vdd</u> | 257 |
| <u>set receiver cap thresholds</u> | 260 |
| <u>set rsh cmd</u> | 262 |

| | |
|-------------------------------------|-----|
| <u>set_sim_init_condition</u> | 263 |
| <u>set_simultaneous_switch</u> | 265 |
| <u>set_three_state</u> | 266 |
| <u>set_units</u> | 267 |
| <u>set_var</u> | 268 |
| <u>set_vdd</u> | 269 |
| <u>write_datasheet</u> | 270 |
| <u>write_ibis_file</u> | 273 |
| <u>write_ldb</u> | 275 |
| <u>write_library</u> | 276 |
| <u>write_template</u> | 289 |
| <u>write_top_netlist</u> | 299 |
| <u>write_userdata_library</u> | 300 |
| <u>write_vdb</u> | 303 |
| <u>write_verilog</u> | 305 |
| <u>write_vital</u> | 313 |
| | |
| 5 | |
| <u>Liberate Parameters</u> | 319 |
| <u>adjust_tristate_load</u> | 331 |
| <u>adjust_tristate_load_ccsp</u> | 332 |
| <u>alspice_diode</u> | 332 |
| <u>alspice_leakage_option</u> | 333 |
| <u>alspice_option</u> | 333 |
| <u>auto_index_distinct_risefall</u> | 334 |
| <u>auto_index_input_slew</u> | 334 |
| <u>auto_index_weak_driver_mode</u> | 334 |
| <u>binning_detail</u> | 335 |
| <u>bisection_info</u> | 335 |
| <u>bundle_when</u> | 336 |
| <u>bundle_arc_mode</u> | 337 |
| <u>bus_syntax</u> | 338 |
| <u>capacitance_attr_mode</u> | 339 |
| <u>capacitance_pin_rollup_k</u> | 339 |
| <u>capacitance_pin_rollup_mode</u> | 339 |

Virtuoso Liberate Reference Manual

| | |
|--|-----|
| <u>capacitance range mode</u> | 341 |
| <u>capacitance save mode</u> | 341 |
| <u>ccs abs tol</u> | 342 |
| <u>ccs base curve points</u> | 342 |
| <u>ccs base curve share mode</u> | 342 |
| <u>ccs cap duplicate risefall</u> | 343 |
| <u>ccs cap hidden pin</u> | 344 |
| <u>ccs cap hidden pin mode</u> | 344 |
| <u>ccs cap is propagating</u> | 345 |
| <u>ccs cap mode add missing</u> | 345 |
| <u>ccs cap use input transition</u> | 345 |
| <u>ccs cap use input transition tristate</u> | 346 |
| <u>ccs correct current by area</u> | 347 |
| <u>ccs current model pin load</u> | 347 |
| <u>ccs enable sawtooth out</u> | 347 |
| <u>ccsn floating init mode</u> | 348 |
| <u>ccs force grid delay</u> | 348 |
| <u>ccs infer output dir</u> | 349 |
| <u>ccs init voltage comp thresh</u> | 349 |
| <u>ccs max current thresh</u> | 350 |
| <u>ccs max pts</u> | 350 |
| <u>ccs multiple switching output mode</u> | 350 |
| <u>ccs rel tol</u> | 351 |
| <u>ccs segmentation effort</u> | 351 |
| <u>ccs smooth lower rise</u> | 352 |
| <u>ccs smooth upper fall</u> | 353 |
| <u>ccs voltage smooth thresh</u> | 353 |
| <u>ccs voltage tail tol</u> | 354 |
| <u>ccs voltage tail tol mode</u> | 354 |
| <u>ccs voltage tail trim tol</u> | 355 |
| <u>ccs warn negative rcvr caps</u> | 356 |
| <u>ccs waveform min time step</u> | 356 |
| <u>ccs waveform smooth mode</u> | 356 |
| <u>ccsn active ccr recognition mode</u> | 357 |
| <u>ccsn allow duplicate condition</u> | 357 |
| <u>ccsn allow multiple input switching</u> | 357 |

| | |
|---|-----|
| <u>ccsn_allow_overlap_when</u> | 358 |
| <u>ccsn_allow_partial_voltage_swing</u> | 358 |
| <u>ccsn_arc_channel_check</u> | 359 |
| <u>ccsn_arc_consistent_cut</u> | 359 |
| <u>ccsn_arc_high_effort</u> | 359 |
| <u>ccsn_bus_holder_mode</u> | 360 |
| <u>ccsn_channel_inputs_high_effort</u> | 360 |
| <u>ccsn_compatibility_mode</u> | 361 |
| <u>ccsn_consistent_side_inputs</u> | 361 |
| <u>ccsn_controlling_path_check</u> | 361 |
| <u>ccsn_dc_static_check</u> | 362 |
| <u>ccsn_dc_static_check_mode</u> | 362 |
| <u>ccsn_dc_static_check_thresh</u> | 363 |
| <u>ccsn_dc_template_size</u> | 363 |
| <u>ccsn_default_group_add_when</u> | 363 |
| <u>ccsn_default_group_criteria_mode</u> | 364 |
| <u>ccsn_dual_tie_enable</u> | 364 |
| <u>ccsn_extra_default_stages</u> | 365 |
| <u>ccsn_fanout_select_mode</u> | 365 |
| <u>ccsn_include_passgate_attr</u> | 365 |
| <u>ccsn_input_xfr_probe_mode</u> | 365 |
| <u>ccsn_io_allow_multiples</u> | 366 |
| <u>ccsn_io_mode</u> | 366 |
| <u>ccsn_io_mode_enable</u> | 367 |
| <u>ccsn_miller_init_mode</u> | 368 |
| <u>ccsn_miller_init_vin_thresh</u> | 368 |
| <u>ccsn_miller_vout_delta_variation</u> | 368 |
| <u>ccsn_model_unbuffered_output</u> | 369 |
| <u>ccsn_no_input_dc_current_mode</u> | 369 |
| <u>ccsn_one_sided_tristate</u> | 369 |
| <u>ccsn_pin_criteria_mode</u> | 370 |
| <u>ccsn_pin_high_effort</u> | 370 |
| <u>ccsn_pin_stage_lshift</u> | 370 |
| <u>ccsn_pin_stage_merge_mode</u> | 371 |
| <u>ccsn_pin_unconditional</u> | 372 |
| <u>ccsn_pin_voltage_level_attrib</u> | 372 |

| | |
|---|-----|
| <u>ccsn_prefer_min_vt_probe</u> | 372 |
| <u>ccsn_prefer_two_sided_stages</u> | 373 |
| <u>ccsn_print_is_needed_if_false_attr_value</u> | 373 |
| <u>ccsn_prop_noise_peak_mode</u> | 374 |
| <u>ccsn_prop_retry_duration_incr</u> | 374 |
| <u>ccsn_prop_retry_peak_incr</u> | 374 |
| <u>ccsn_probe_mode</u> | 375 |
| <u>ccsn_prune_last_stage</u> | 375 |
| <u>ccsn_simultaneous_switch_save_vecdata</u> | 376 |
| <u>ccsn_sort_merge_hidden_mode</u> | 376 |
| <u>ccsn_switch_cell_partition_mode</u> | 376 |
| <u>ccsn_tempus_promote_mode</u> | 377 |
| <u>ccsn_use_io_ccb_format</u> | 378 |
| <u>ccsn_xfr_ccc_probe_mode</u> | 378 |
| <u>ccsp_base_curve_points</u> | 379 |
| <u>ccsp_default_group</u> | 379 |
| <u>ccsp_leakage_current_abstol</u> | 380 |
| <u>ccsp_leakage_current_compensation_mode</u> | 380 |
| <u>ccsp_min_pts</u> | 381 |
| <u>ccsp_pin_direction_post_default</u> | 381 |
| <u>ccsp_prune_factor</u> | 381 |
| <u>ccsp_prune_second_tol</u> | 382 |
| <u>ccsp_prune_start_tol</u> | 382 |
| <u>ccsp_quantization_num_steps</u> | 383 |
| <u>ccsp_rel_tol</u> | 383 |
| <u>ccsp_related_pin_mode</u> | 383 |
| <u>ccsp_segmentation_effort</u> | 384 |
| <u>ccsp_table_reduction</u> | 384 |
| <u>ccsp_tail_tol</u> | 385 |
| <u>cell_port_case</u> | 385 |
| <u>char_mos_term_cap</u> | 386 |
| <u>cleanup_tmpdir</u> | 387 |
| <u>combinational_out_to_out_arc</u> | 387 |
| <u>combinational_risefall</u> | 388 |
| <u>conditional_arc</u> | 388 |
| <u>conditional_cap_hidden_pin</u> | 389 |

| | |
|--|-----|
| <u>conditional_cap_hidden_pin_mode</u> | 389 |
| <u>conditional_cap_hidden_pin_thresh</u> | 390 |
| <u>conditional_constraint</u> | 391 |
| <u>conditional_expression</u> | 391 |
| <u>conditional_hidden_power</u> | 392 |
| <u>conditional_immunity</u> | 392 |
| <u>conditional_include_constant</u> | 393 |
| <u>conditional_include_output</u> | 393 |
| <u>conditional_leakage</u> | 394 |
| <u>conditional_min_period</u> | 394 |
| <u>conditional_mpw</u> | 395 |
| <u>conditional_rcvr_cap_select_criteria</u> | 396 |
| <u>constraint_async_probe_internal</u> | 397 |
| <u>constraint_bisection_mode</u> | 397 |
| <u>constraint_check_final_state</u> | 398 |
| <u>constraint_check_final_state_threshold</u> | 399 |
| <u>constraint_check_rebound</u> | 400 |
| <u>constraint_clock_gater</u> | 400 |
| <u>constraint_combinational</u> | 401 |
| <u>constraint_combinational_step_limit</u> | 401 |
| <u>constraint_combinational_step_size</u> | 402 |
| <u>constraint_delay_degrade</u> | 402 |
| <u>constraint_delay_degrade_abstol</u> | 402 |
| <u>constraint_delay_degrade_abstol_max</u> | 403 |
| <u>constraint_delay_degrade_minimize_dtoq</u> | 404 |
| <u>constraint_delay_degrade_minimize_dtoq_clock_only</u> | 405 |
| <u>constraint_delay_degrade_minimize_dtoq_mode</u> | 405 |
| <u>constraint_delay_degrade_minimize_dtoq_tol</u> | 406 |
| <u>constraint_delay_min_check</u> | 406 |
| <u>constraint_dependent_nominal</u> | 407 |
| <u>constraint_dependent_recrem</u> | 407 |
| <u>constraint_dependent_setuphold</u> | 408 |
| <u>constraint_dependent_setuphold_input_threshold</u> | 409 |
| <u>constraint_dependent_setuphold_margin</u> | 409 |
| <u>constraint_dependent_setuphold_margin_ratio</u> | 410 |
| <u>constraint_dependent_setuphold_pessimism</u> | 410 |

| | |
|--|-----|
| <u>constraint failed value</u> | 410 |
| <u>constraint glitch hold</u> | 412 |
| <u>constraint glitch peak</u> | 412 |
| <u>constraint glitch peak internal</u> | 413 |
| <u>constraint glitch peak max</u> | 413 |
| <u>constraint glitch peak mode</u> | 414 |
| <u>constraint glitch peak report inherent</u> | 415 |
| <u>constraint hold probe</u> | 415 |
| <u>constraint info</u> | 416 |
| <u>constraint info pass fail</u> | 416 |
| <u>constraint linear waveform</u> | 417 |
| <u>constraint margin</u> | 417 |
| <u>constraint merge state</u> | 417 |
| <u>constraint output load</u> | 417 |
| <u>constraint output pin</u> | 418 |
| <u>constraint output pin mode</u> | 419 |
| <u>constraint probe internal</u> | 420 |
| <u>constraint probe lower fall</u> | 420 |
| <u>constraint probe lower rise</u> | 420 |
| <u>constraint probe mode</u> | 420 |
| <u>constraint probe multiple</u> | 422 |
| <u>constraint probe upper fall</u> | 422 |
| <u>constraint probe upper rise</u> | 423 |
| <u>constraint search bound</u> | 423 |
| <u>constraint search bound bisection mode</u> | 423 |
| <u>constraint search bound estimation mode</u> | 424 |
| <u>constraint search bound probe mode</u> | 425 |
| <u>constraint search iteration limit</u> | 425 |
| <u>constraint search time abstol</u> | 426 |
| <u>constraint slew degrade</u> | 426 |
| <u>constraint snap to bound</u> | 427 |
| <u>constraint sweep pulse detection mode</u> | 427 |
| <u>constraint sweep pulse width max</u> | 428 |
| <u>constraint tran end extend</u> | 428 |
| <u>constraint tran end extend retry</u> | 429 |
| <u>constraint tran end mode</u> | 429 |

| | |
|--|-----|
| <u>constraint user defined probe mode</u> | 430 |
| <u>constraint vector equivalence mode</u> | 430 |
| <u>constraint vector mode</u> | 431 |
| <u>constraint width degrade</u> | 432 |
| <u>constraint width degrade abstol</u> | 432 |
| <u>constraint width degrade abstol max</u> | 432 |
| <u>constraint worst vector abstol</u> | 433 |
| <u>debug flow</u> | 433 |
| <u>def arc drive side bidi</u> | 433 |
| <u>def arc msg level</u> | 434 |
| <u>def arc vector consistency check</u> | 434 |
| <u>default power avg mode</u> | 435 |
| <u>define arc ignore mode</u> | 435 |
| <u>define arc preserve when string</u> | 436 |
| <u>define duplicate cap mode</u> | 436 |
| <u>delay constrained by setup recovery</u> | 437 |
| <u>delay inp fall</u> | 437 |
| <u>delay inp rise</u> | 438 |
| <u>delay out fall</u> | 438 |
| <u>delay out rise</u> | 438 |
| <u>disable method</u> | 439 |
| <u>discard timing sense after merge</u> | 441 |
| <u>disk wait time</u> | 441 |
| <u>driver cell acc mode</u> | 441 |
| <u>driver cell all inputs</u> | 442 |
| <u>driver cell info</u> | 442 |
| <u>driver cell load all outputs</u> | 442 |
| <u>driver cell load ldb cmd</u> | 443 |
| <u>driver waveform arcs only</u> | 443 |
| <u>driver waveform output precision</u> | 444 |
| <u>driver waveform pulse mode</u> | 444 |
| <u>driver waveform wildcard mode</u> | 445 |
| <u>duplicate pin attr mode</u> | 445 |
| <u>duplicate risefall power</u> | 446 |
| <u>duplicate risefall power ccsp</u> | 446 |
| <u>ecsm_arctype_enable</u> | 447 |

| | |
|--|-----|
| <u>ecsm_cap_hidden_pin</u> | 448 |
| <u>ecsm_cap_input_slew_mode</u> | 448 |
| <u>ecsm_cap_load_effect_tol</u> | 448 |
| <u>ecsm_cap_mode</u> | 449 |
| <u>ecsm_cap_style</u> | 450 |
| <u>ecsm_cap_use_input_transition</u> | 450 |
| <u>ecsm_capacitance_factor</u> | 451 |
| <u>ecsm_capacitance_precision</u> | 451 |
| <u>ecsm_factor_mode</u> | 452 |
| <u>ecsm_invert_gnd_current</u> | 452 |
| <u>ecsm_measure_output_range</u> | 453 |
| <u>ecsm_version</u> | 454 |
| <u>ecsm_waveform_for_bidi_pin</u> | 454 |
| <u>ecsm_waveform_style</u> | 455 |
| <u>ecsm_waveform_time_factor</u> | 455 |
| <u>ecsm_waveform_time_precision</u> | 455 |
| <u>ecsmn_loadcap_mode</u> | 456 |
| <u>ecsmn_mode</u> | 456 |
| <u>ecsmn_vtol_mode</u> | 456 |
| <u>em_calculation_include_input</u> | 457 |
| <u>em_calculation_monitor_rails</u> | 457 |
| <u>em_calculation_monitor_rails_skip_layer</u> | 458 |
| <u>em_char_arcs_mode</u> | 458 |
| <u>em_clock_freq</u> | 459 |
| <u>em_current_type</u> | 459 |
| <u>em_data_file</u> | 460 |
| <u>em_freq_mode</u> | 460 |
| <u>em_iacpeak_mode</u> | 460 |
| <u>em_include_string</u> | 461 |
| <u>em_maxcap</u> | 461 |
| <u>em_maxcap_type</u> | 462 |
| <u>em_maxcap_frequency</u> | 462 |
| <u>em_report_data_usage_mode</u> | 462 |
| <u>em_tech_file</u> | 463 |
| <u>em_user_string</u> | 463 |
| <u>em_window_estimate_mode</u> | 463 |

| | |
|--|-----|
| <u>enable command history</u> | 464 |
| <u>extsim ccs option</u> | 464 |
| <u>extsim cells use nodeset for io pad</u> | 465 |
| <u>extsim cmd</u> | 465 |
| <u>extsim cmd option</u> | 466 |
| <u>extsim constraint option</u> | 467 |
| <u>extsim deck dir</u> | 467 |
| <u>extsim deck header</u> | 468 |
| <u>extsim deck style</u> | 469 |
| <u>extsim exclusive</u> | 469 |
| <u>extsim flatten netlist</u> | 470 |
| <u>extsim immunity option</u> | 471 |
| <u>extsim leakage option</u> | 471 |
| <u>extsim lic keep</u> | 472 |
| <u>extsim line length limit</u> | 472 |
| <u>extsim model include</u> | 473 |
| <u>extsim model include leakage</u> | 474 |
| <u>extsim model include mode</u> | 474 |
| <u>extsim monitor deck dir</u> | 475 |
| <u>extsim monitor enable</u> | 476 |
| <u>extsim monitor timeout</u> | 476 |
| <u>extsim mpw option</u> | 477 |
| <u>extsim node name prefix</u> | 477 |
| <u>extsim option</u> | 478 |
| <u>extsim option presim</u> | 478 |
| <u>extsim reuse ic</u> | 479 |
| <u>extsim sanitize param name</u> | 480 |
| <u>extsim save failed</u> | 480 |
| <u>extsim save passed</u> | 481 |
| <u>extsim save verify</u> | 482 |
| <u>extsim tar cmd</u> | 483 |
| <u>extsim timestep</u> | 484 |
| <u>extsim tran append</u> | 484 |
| <u>extsim use node name</u> | 484 |
| <u>floating_channel_bias</u> | 485 |
| <u>floating_channel_mode</u> | 485 |

| | |
|--|-----|
| <u>floating_node_initialize_mode</u> | 486 |
| <u>force_avg_default_select_order</u> | 486 |
| <u>force_condition</u> | 487 |
| <u>force_default_group</u> | 488 |
| <u>force_edge_timing_type</u> | 488 |
| <u>force_leakage_if_no_pg_pin</u> | 489 |
| <u>force_related_power_pin</u> | 490 |
| <u>force_unconnected_pg_pin</u> | 490 |
| <u>group_attribute</u> | 491 |
| <u>heartbeat_initial_timeout</u> | 491 |
| <u>heartbeat_timeout</u> | 492 |
| <u>hidden_power</u> | 492 |
| <u>ibis_compensate_odt</u> | 493 |
| <u>ibis_has_weak_hold</u> | 493 |
| <u>ibis_iv_max_step_factor</u> | 493 |
| <u>ibis_iv_mode</u> | 494 |
| <u>ibis_iv_step</u> | 494 |
| <u>ibis_odt_min_current</u> | 495 |
| <u>ibis_sim_duration</u> | 495 |
| <u>ibis_t2b_cmd</u> | 495 |
| <u>ibis_tend_factor</u> | 496 |
| <u>ibis_vt_max_num_pts</u> | 496 |
| <u>ibis_vt_min_num_pts</u> | 497 |
| <u>immunity_glitch_peak</u> | 497 |
| <u>immunity_noise_skew_ratio</u> | 498 |
| <u>init_clock_period_mode</u> | 498 |
| <u>init_comb_num_cycles</u> | 499 |
| <u>init_comb_related_pin_period</u> | 499 |
| <u>init_constraint_period</u> | 499 |
| <u>init_constraint_period_binning_mode</u> | 500 |
| <u>init_constraint_period_check_mode</u> | 500 |
| <u>init_delay_period</u> | 501 |
| <u>init_pin_hidden_period</u> | 501 |
| <u>init_pin_hidden_num_cycles</u> | 501 |
| <u>init_pin_hidden_period_mode</u> | 502 |
| <u>input_noise</u> | 502 |

| | |
|---|-----|
| <u>input output voltage</u> | 503 |
| <u>keep dcap leakage</u> | 503 |
| <u>keep default leakage group</u> | 504 |
| <u>keep empty cells</u> | 504 |
| <u>keep user defined arc failed data</u> | 505 |
| <u>ldb checkpoint dir</u> | 505 |
| <u>ldb precision</u> | 506 |
| <u>ldb save all cells</u> | 506 |
| <u>leakage add input pin</u> | 507 |
| <u>leakage add missing group</u> | 507 |
| <u>leakage cell attribute</u> | 507 |
| <u>leakage expand state</u> | 508 |
| <u>leakage float internal supply</u> | 508 |
| <u>leakage force tristate pin</u> | 509 |
| <u>leakage merge state</u> | 510 |
| <u>leakage mode</u> | 510 |
| <u>leakage model internal pin</u> | 511 |
| <u>leakage precision</u> | 512 |
| <u>leakage ramp vsrc</u> | 512 |
| <u>leakage sim duration</u> | 512 |
| <u>library copyright</u> | 513 |
| <u>library revision</u> | 513 |
| <u>library revision mode</u> | 513 |
| <u>lic max timeout</u> | 514 |
| <u>lic queue timeout</u> | 515 |
| <u>logic and</u> | 515 |
| <u>logic not</u> | 515 |
| <u>logic or</u> | 515 |
| <u>mac address query timeout</u> | 516 |
| <u>mark failed data</u> | 516 |
| <u>mark failed data replacement</u> | 517 |
| <u>max capacitance attr limit</u> | 517 |
| <u>max capacitance attr mode</u> | 517 |
| <u>max capacitance derive limit maxload</u> | 518 |
| <u>max capacitance factor</u> | 518 |
| <u>max capacitance limit</u> | 518 |

| | |
|---|-----|
| <u>max_hidden_vector</u> | 519 |
| <u>max_leakage_vector</u> | 519 |
| <u>max_noise_width</u> | 520 |
| <u>max_transition</u> | 520 |
| <u>max_transition_attr_limit</u> | 520 |
| <u>max_transition_factor</u> | 521 |
| <u>max_transition_for_outputs</u> | 521 |
| <u>max_transition_include_power</u> | 521 |
| <u>measure_cap_active_driver_mode</u> | 522 |
| <u>measure_cap_lower_fall</u> | 522 |
| <u>measure_cap_lower_rise</u> | 522 |
| <u>measure_cap_upper_fall</u> | 523 |
| <u>measure_cap_upper_rise</u> | 523 |
| <u>measure_ccs_cap_lower_rise</u> | 523 |
| <u>measure_ccs_cap_upper_fall</u> | 523 |
| <u>measure_em_target_occurrence</u> | 524 |
| <u>measure_output_range</u> | 524 |
| <u>measure_output_range_abstol</u> | 525 |
| <u>measure_slew_lower_fall</u> | 525 |
| <u>measure_slew_lower_rise</u> | 525 |
| <u>measure_slew_upper_fall</u> | 525 |
| <u>measure_slew_upper_rise</u> | 525 |
| <u>measure_target_occurrence</u> | 526 |
| <u>mega_bundle_mode</u> | 526 |
| <u>mega_enable</u> | 527 |
| <u>mega_mode_constraint</u> | 527 |
| <u>mega_mode_delay</u> | 528 |
| <u>mega_mode_hidden</u> | 529 |
| <u>merge_related_preset_clear</u> | 530 |
| <u>min_capacitance_for_outputs</u> | 531 |
| <u>min_output_cap</u> | 531 |
| <u>min_period</u> | 532 |
| <u>min_period_when</u> | 532 |
| <u>min_transition</u> | 533 |
| <u>min_transition_attr_limit</u> | 533 |
| <u>min_transition_factor</u> | 533 |

| | |
|---|-----|
| <u>min_transition_for_outputs</u> | 534 |
| <u>min_transition_include_power</u> | 534 |
| <u>mpw_criteria</u> | 534 |
| <u>mpw_glitch_peak</u> | 535 |
| <u>mpw_input_threshold</u> | 535 |
| <u>mpw_linear_waveform</u> | 535 |
| <u>mpw_search_bound</u> | 536 |
| <u>mpw_search_mode</u> | 536 |
| <u>mpw_skew_factor</u> | 536 |
| <u>mpw_slew</u> | 537 |
| <u>mpw_slew_clock_factor</u> | 537 |
| <u>mpw_table</u> | 538 |
| <u>mpw_vector_bin_mode</u> | 538 |
| <u>msg_level</u> | 538 |
| <u>msg_level_user_data_override</u> | 539 |
| <u>msg_limit_per_type_per_cell</u> | 539 |
| <u>net_batch_mode</u> | 539 |
| <u>nochange_mode</u> | 539 |
| <u>nochange_value</u> | 540 |
| <u>normalized_driver_waveform</u> | 540 |
| <u>non_seq_copy_dst_pin</u> | 542 |
| <u>non_seq_copy_src_pin</u> | 542 |
| <u>non_seq_pin_swap</u> | 542 |
| <u>nonseq_as_recrem</u> | 543 |
| <u>output_internal_pin</u> | 543 |
| <u>packet_arc_notification_interval</u> | 543 |
| <u>packet_arc_notification_limit</u> | 544 |
| <u>packet_arc_notification_list</u> | 544 |
| <u>packet_arc_optimize_idle_clients</u> | 545 |
| <u>packet_arcs_per_thread</u> | 545 |
| <u>packet_arcs_per_thread_auto_adjust</u> | 546 |
| <u>packet_require_spectre_char_opt</u> | 546 |
| <u>packet_cell_max_fets</u> | 547 |
| <u>packet_clients</u> | 548 |
| <u>packet_client_idle_count</u> | 548 |
| <u>packet_client_resubmit_count</u> | 548 |

| | |
|--|-----|
| <u>packet_client_timeout</u> | 549 |
| <u>packet_client_timeout_action</u> | 549 |
| <u>packet_log_filename</u> | 549 |
| <u>packet_mode</u> | 550 |
| <u>packet_rdb_mode</u> | 550 |
| <u>packet_rsh_mode</u> | 551 |
| <u>parenthesize_not</u> | 551 |
| <u>parenthesize_sdf_cond</u> | 551 |
| <u>parse_auto_define_leafcell</u> | 552 |
| <u>parse_space_bang_is_comment</u> | 552 |
| <u>pin_based_leakage</u> | 553 |
| <u>pin_based_power</u> | 553 |
| <u>pin_based_signal_level_mode</u> | 554 |
| <u>pin_capacitance_matching_mode</u> | 555 |
| <u>pin_type_order</u> | 555 |
| <u>pin_vdd_supply_style</u> | 556 |
| <u>power_add_input_pin</u> | 556 |
| <u>power_adjust_for_pin_load</u> | 557 |
| <u>power_binate_arc</u> | 557 |
| <u>power_combinational_include_output</u> | 557 |
| <u>power_divide_num_switching_mode</u> | 558 |
| <u>power_info</u> | 558 |
| <u>power_info_filename</u> | 559 |
| <u>power_minimize_switching</u> | 560 |
| <u>power_model_gnd_waveform_data_mode</u> | 560 |
| <u>power_multi_output_binning_mode</u> | 561 |
| <u>power_sequential_include_complementary_output</u> | 561 |
| <u>power_sim_estimate_duration</u> | 562 |
| <u>power_subtract_leakage</u> | 562 |
| <u>power_subtract_leakage_msg_level</u> | 563 |
| <u>power_subtract_leakage_mode</u> | 563 |
| <u>power_subtract_output_load</u> | 564 |
| <u>power_subtract_output_load_mode</u> | 565 |
| <u>power_tend_match_tran</u> | 565 |
| <u>predriver_waveform</u> | 566 |
| <u>predriver_waveform_mode</u> | 567 |

| | |
|---|-----|
| <u>predriver waveform npts</u> | 567 |
| <u>predriver waveform ratio</u> | 567 |
| <u>preserve user function</u> | 568 |
| <u>prevector period</u> | 568 |
| <u>prevector slew</u> | 568 |
| <u>prevector voltage waveform mode</u> | 570 |
| <u>process match pins to ports</u> | 570 |
| <u>process node</u> | 571 |
| <u>ramp vsrc</u> | 571 |
| <u>rc floating cap mode</u> | 571 |
| <u>rcp cmd</u> | 572 |
| <u>rdb checkpoint dir</u> | 573 |
| <u>rdb exit if source differ</u> | 573 |
| <u>rechar checksum</u> | 574 |
| <u>removal glitch peak</u> | 574 |
| <u>res merge</u> | 575 |
| <u>res open tol</u> | 575 |
| <u>res tol</u> | 575 |
| <u>reset leakage current mode</u> | 576 |
| <u>reset negative leakage power value</u> | 576 |
| <u>reset negative constraint</u> | 576 |
| <u>reset negative delay</u> | 577 |
| <u>reset negative leakage power</u> | 577 |
| <u>reset negative power</u> | 577 |
| <u>resolve collision</u> | 578 |
| <u>retry count</u> | 579 |
| <u>retry count file operation</u> | 579 |
| <u>rsh cmd</u> | 579 |
| <u>scale load by template</u> | 580 |
| <u>scale tran by template</u> | 580 |
| <u>scan dummy include leakage power</u> | 581 |
| <u>sdf cond equals</u> | 581 |
| <u>sdf cond postfix</u> | 582 |
| <u>sdf cond prefix</u> | 583 |
| <u>sdf cond style</u> | 583 |
| <u>sdf cond variable map</u> | 584 |

| | |
|--|-----|
| <u>sdf logic and</u> | 584 |
| <u>sdf logic not</u> | 584 |
| <u>sdf logic or</u> | 585 |
| <u>server timeout</u> | 585 |
| <u>set var failure action</u> | 585 |
| <u>sim default engine</u> | 586 |
| <u>sim duration</u> | 586 |
| <u>sim estimate duration</u> | 587 |
| <u>sim init condition</u> | 587 |
| <u>sim init condition estimation mode</u> | 587 |
| <u>sim init duration</u> | 588 |
| <u>sim power duration extend</u> | 588 |
| <u>sim use init duration</u> | 589 |
| <u>simultaneous switch</u> | 589 |
| <u>simultaneous switch from cell when</u> | 590 |
| <u>simultaneous switch offset</u> | 591 |
| <u>simultaneous switch use arc when</u> | 591 |
| <u>simultaneous switch worst vector</u> | 592 |
| <u>ski alter mode</u> | 592 |
| <u>ski clean mode</u> | 593 |
| <u>ski compatibility mode</u> | 594 |
| <u>ski enable</u> | 595 |
| <u>ski_mdlthreshold_exact</u> | 595 |
| <u>ski_power_subtract_output_load_match_extsim</u> | 596 |
| <u>ski_reset_cnt</u> | 596 |
| <u>ski_use_large_memory</u> | 597 |
| <u>skip_nfs_sync</u> | 597 |
| <u>slew_lower_fall</u> | 598 |
| <u>slew_lower_rise</u> | 598 |
| <u>slew_normalize</u> | 598 |
| <u>slew_upper_fall</u> | 598 |
| <u>slew_upper_rise</u> | 599 |
| <u>sort_cells</u> | 599 |
| <u>sort_groups_under_pin</u> | 600 |
| <u>sort_pins</u> | 600 |
| <u>sort_pins_under_when</u> | 601 |

| | |
|--|-----|
| <u>spectre dash log</u> | 601 |
| <u>spectre use char opt license</u> | 601 |
| <u>spice character map / spectre character map</u> | 602 |
| <u>spice delimiter</u> | 603 |
| <u>spice delimiter replacement</u> | 603 |
| <u>spice instance name require x prefix</u> | 604 |
| <u>spice logical netname mode</u> | 604 |
| <u>subtract hidden power</u> | 605 |
| <u>subtract hidden power consider all supplies</u> | 605 |
| <u>subtract hidden power use default</u> | 605 |
| <u>supply define mode</u> | 606 |
| <u>switch cell bounded dc current</u> | 607 |
| <u>switch cell dc current</u> | 607 |
| <u>switch cell dc current output offset</u> | 608 |
| <u>switch cell internal node timing arc</u> | 608 |
| <u>switch cell powerdown function</u> | 609 |
| <u>switch function attr mode</u> | 609 |
| <u>template unique power mode</u> | 610 |
| <u>test cell at end</u> | 610 |
| <u>timing group unateness</u> | 611 |
| <u>tmpdir</u> | 611 |
| <u>toggle leakage state</u> | 612 |
| <u>tran tend estimation mode</u> | 612 |
| <u>tristate disable transition</u> | 612 |
| <u>tristate pin cap always on res mode</u> | 613 |
| <u>unique pin data</u> | 613 |
| <u>use pid tmpdir</u> | 614 |
| <u>user data attr order</u> | 614 |
| <u>user data override</u> | 615 |
| <u>user data quote attributes</u> | 615 |
| <u>user data quote simple attr</u> | 616 |
| <u>variation mean nominal model mode</u> | 616 |
| <u>vector check mode</u> | 617 |
| <u>vector estimate dump</u> | 617 |
| <u>vector side input</u> | 618 |
| <u>verilog cg filter edge</u> | 618 |

| | |
|--|-----|
| <u>voltage_map</u> | 618 |
| <u>vsrc_slope_mode</u> | 620 |
| <u>waveform_report</u> | 621 |
| <u>wnflag</u> | 621 |
| <u>write_library_is_unbuffered</u> | 621 |
| <u>write_library_mode</u> | 622 |
| <u>write_library_sync_ldb</u> | 622 |
| <u>write_logic_function</u> | 623 |
| <u>write_logic_function_group_at_end</u> | 623 |
| <u>write_logic_function_mode</u> | 623 |
| <u>write_logic_function_statetable_limit</u> | 624 |
| <u>write_logic_function_statetable_mode</u> | 624 |
| <u>write_min_transition_attr</u> | 624 |
| <u>write_template_force_power</u> | 625 |
| | |
| 6 | |
| Library Comparisons | 627 |
| <u>Panel Buttons</u> | 628 |
| <u>Pull-Down menus</u> | 628 |
| Graphical Library Comparisons | 629 |
| <u>Style: X/Y</u> | 629 |
| <u>Style: Accuracy</u> | 630 |
| <u>Style: ErrorBound</u> | 631 |
| Data Selection | 633 |
| <u>Cell Type Selection</u> | 634 |
| <u>Data Type Selection</u> | 634 |
| | |
| 7 | |
| Liberate Details | 637 |
| Delay Models | 639 |
| <u>NLDM</u> | 639 |
| <u>CCS</u> | 640 |
| <u>ECSM</u> | 641 |
| Pin Capacitance | 642 |
| <u>NLDM Capacitance</u> | 642 |

Virtuoso Liberate Reference Manual

| | |
|---|-----|
| <u>CCS Receiver Capacitance</u> | 643 |
| <u>ECSM Capacitance</u> | 643 |
| <u>Timing Constraints</u> | 643 |
| <u>Setup and Hold</u> | 643 |
| <u>Dependant constraint characterization</u> | 646 |
| <u>Recovery and Removal</u> | 647 |
| <u>Non-sequential Setup and Hold</u> | 647 |
| <u>Min Pulse Width</u> | 648 |
| <u>No Change Arcs</u> | 649 |
| <u>Power Models</u> | 653 |
| <u>Leakage Power</u> | 653 |
| <u>Switching and Hidden Power</u> | 655 |
| <u>Power Subtraction</u> | 656 |
| <u>Power Validation</u> | 657 |
| <u>Common Usage Modes for Power and Leakage</u> | 657 |
| <u>Signal Integrity Models</u> | 658 |
| <u>Steady State Current</u> | 659 |
| <u>Noise Immunity Curves</u> | 659 |
| <u>Hyperbolic Input Noise, DC Noise Margin</u> | 660 |
| <u>Composite Current Source Noise (CCSN) Models</u> | 660 |
| <u>CCSN DC Current</u> | 660 |
| <u>CCSN Output Voltage</u> | 661 |
| <u>CCSN Miller Capacitance</u> | 662 |
| <u>CCSN Propagated Noise</u> | 662 |
| <u>IBIS Models</u> | 663 |
| <u>IBIS model content</u> | 663 |
| <u>IBIS modeling flow</u> | 665 |
| <u>Truth Table Example for IBIS</u> | 672 |
| <u>Electromigration Models</u> | 674 |
| <u>Electromigration Model Content</u> | 674 |
| <u>Electromigration Modeling Flow</u> | 674 |
| <u>Data Table Index Determination</u> | 677 |

A

| | |
|---|-----|
| <u>Truth Table Format</u> | 679 |
| <u>Truth Table Format – Basic</u> | 679 |
| <u>Liberate Truth Table Format</u> | 687 |
| <u>Truth Table Format – IBIS</u> | 689 |
| <u>Notes on command syntax</u> | 689 |
| <u>IBIS Truth Table Commands</u> | 690 |
| <u>Creating an IBIS file with multiple [Model] sections for a single device</u> | 700 |

B

| | |
|--|-----|
| <u>Constraint-related Delay and Clock Path Measurement</u> | 703 |
|--|-----|

C

| | |
|--|-----|
| <u>External Simulator Options and Settings</u> | 705 |
| <u>Spectre</u> | 705 |
| <u>General Spectre Settings for Accuracy and Performance</u> | 705 |
| <u>Spectre Kernel Interface (SKI)</u> | 706 |
| <u>Correlation between stand-alone Spectre & SKI</u> | 707 |
| <u>Special Licensing</u> | 708 |
| <u>HSPICE</u> | 708 |
| <u>Accuracy settings for 28nm and below</u> | 708 |
| <u>ECSM Accuracy and Correlation Settings for 20nm and Below</u> | 709 |
| <u>Driver Cell</u> | 709 |
| <u>Recommended Liberate Settings for ECSM</u> | 709 |
| <u>CCS Accuracy and Correlation Settings</u> | 710 |
| <u>Driver Waveform</u> | 710 |
| <u>Recommended Liberate Settings for CCS</u> | 710 |

D

| | |
|---|-----|
| <u>Deprecated and Legacy Commands and Variables</u> | 711 |
| <u>Deprecated Commands</u> | 711 |
| <u>add_lib_attribute</u> | 711 |
| <u>add_cell_attribute</u> | 711 |

Virtuoso Liberate Reference Manual

| | |
|--|-----|
| <u>add_pin_attribute</u> | 711 |
| <u>define_arc</u> | 712 |
| <u>define_em</u> | 713 |
| <u>remove_pin_attribute</u> | 713 |
| <u>set_ccs_retry_thresholds</u> | 714 |
| <u>set_client</u> | 715 |
| <u>set_em_imax</u> | 716 |
| <u>Deprecated Variables</u> | 717 |
| <u>bundle_count</u> | 717 |
| <u>ccsn_allow_probe_driven_by_feedback</u> | 717 |
| <u>ccsn_overlap_ccr_include_mode</u> | 718 |
| <u>cell_use_both_ff_latch_groups</u> | 718 |
| <u>default_capacitance</u> | 719 |
| <u>default_group_method</u> | 719 |
| <u>default_leakage</u> | 720 |
| <u>default_power</u> | 720 |
| <u>default_timing</u> | 720 |
| <u>default_unateness</u> | 721 |
| <u>em_period</u> | 722 |
| <u>extsim_interactive</u> | 722 |
| <u>leakage_accuracy_mode</u> | 723 |
| <u>max_capacitance_auto_mode</u> | 723 |
| <u>mpw_delay_use_active_edge</u> | 724 |
| <u>packet_arc_licensing_mode</u> | 724 |
| <u>packet_arc_require_spectre_char_opt</u> | 725 |
| <u>rc_sort_mode</u> | 727 |
| <u>Backward Compatibility Variables</u> | 728 |
| <u>capacitance_force_hidden</u> | 728 |
| <u>ccsn_active_ccr_recognition_mode</u> | 728 |
| <u>ccsn_check_valid_noise_prop</u> | 728 |
| <u>ccsn_compatibility_mode</u> | 729 |
| <u>ccsn_compatibility_multi_corners</u> | 729 |
| <u>ccsn_dc_estimate_mode</u> | 730 |
| <u>ccsn_dc_sweep_mode</u> | 730 |
| <u>ccsn_fanout_select_mode</u> | 730 |
| <u>ccsn_io_skip_channel_inputs</u> | 731 |

| | |
|---|-----|
| <u>ccsn input xfr probe mode</u> | 731 |
| <u>ccsn mega mbit hidden mode</u> | 732 |
| <u>ccsn model unbuffered output</u> | 732 |
| <u>ccsn pin unconditional</u> | 733 |
| <u>ccsn prefer two sided stages</u> | 733 |
| <u>ccsn probe mode</u> | 733 |
| <u>ccsn prop retry duration incr</u> | 734 |
| <u>ccsn redundant pin stages</u> | 734 |
| <u>ccsn skip mod vdata</u> | 735 |
| <u>cell leakage power legacy mode</u> | 735 |
| <u>char mos term cap ski mode</u> | 736 |
| <u>conditional expression accuracy</u> | 736 |
| <u>constraint search time reltol mode</u> | 736 |
| <u>default non unate rcvr cap adjust</u> | 737 |
| <u>default power subtract hidden mode</u> | 737 |
| <u>default timing tristate enable</u> | 738 |
| <u>def arc delay metric mode</u> | 739 |
| <u>disable current measure effort</u> | 739 |
| <u>driver waveform lib mode</u> | 740 |
| <u>ecsmp invert gnd current</u> | 740 |
| <u>extsim model include multi vector mode</u> | 740 |
| <u>extsim save driver</u> | 741 |
| <u>floating node consistency check</u> | 741 |
| <u>ignore dummy fets</u> | 742 |
| <u>non seq probe mode</u> | 743 |
| <u>power multi vector mode</u> | 743 |
| <u>power subtract leakage tran mode</u> | 743 |
| <u>reset negative power mode</u> | 744 |
| <u>set pin slew threshold mode</u> | 744 |
| <u>switch cell infer unateness from ldb</u> | 745 |
| <u>tristate pin cap use arc</u> | 745 |
| <u>user data keep simple attr quotes</u> | 746 |
| <u>write logic function async mode</u> | 746 |
| <u>voltage map ldb char mode</u> | 747 |
| <u>Legacy Debug Variables</u> | 748 |
| <u>ccs_retry_info</u> | 748 |

| | |
|---|-----|
| <u>ccs_retry_mode</u> | 748 |
| <u>ccs_retry_multi_peak_tol</u> | 749 |
| <u>ccs_retry_voltage_tail_tol</u> | 750 |
| <u>extsim_ccs_retry_option</u> | 750 |
| <u>extsim_ccs_retry_tran_append</u> | 751 |

E

| | |
|--|-----|
| <u>Variables to Use When Qualifying and Migrating Between Versions</u> | 753 |
|--|-----|

F

| | |
|-----------------------|-----|
| <u>Glossary</u> | 761 |
|-----------------------|-----|

Virtuoso Liberate Reference Manual

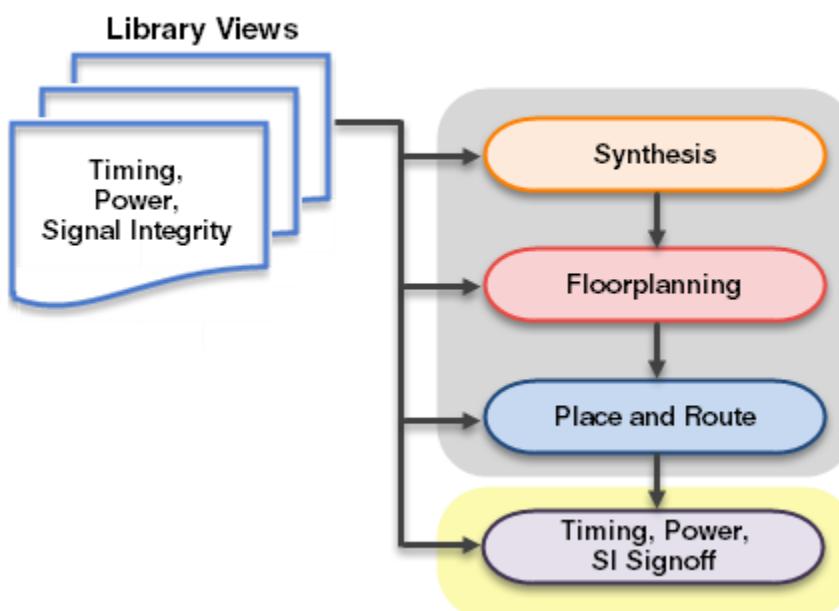
Preface

Introduction to Characterization

The Role and Importance of Libraries

Creation of electrical views is a pre-requisite for any digital design flow. The electrical information stored in the library views is used throughout design implementation from logic synthesis, through design optimization to final signoff verification. Accurate library view creation is essential to ensure close correlation between the design intent and the final silicon.

Digital Implementation Flow

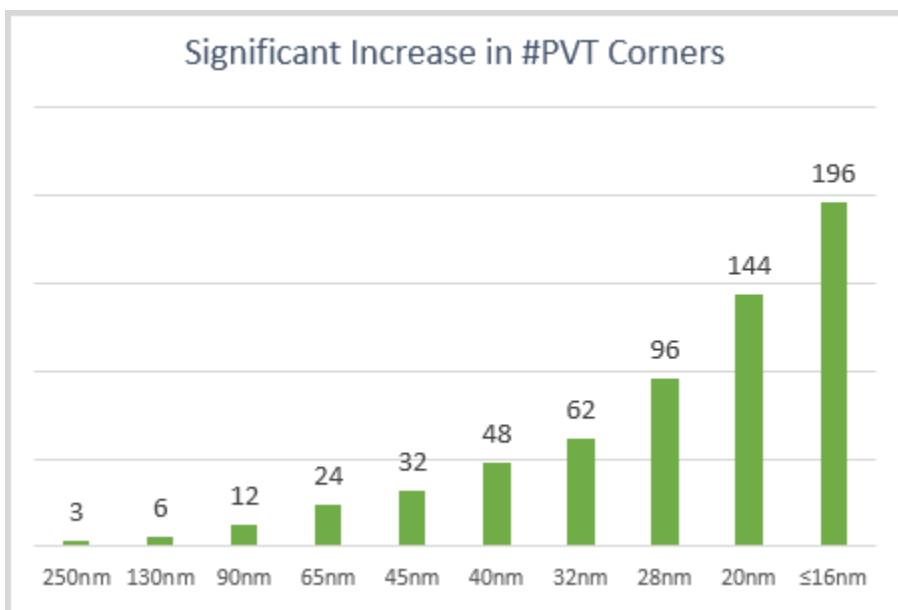


A Growing Problem

In nanometer geometries (65nm or below), the required number of library views is growing dramatically because of issues related to power leakage and process variation. To minimize

power leakage at deep submicron nodes, we see process variations such as LVT, RVT, and HVT (low/regular/high voltage) being utilized. For example, to manage power at 65nm, it is common to have library cells with two or three different threshold values (high threshold to reduce leakage power, lower thresholds to improve performance), and to use two or more on-chip supply voltages. In this scenario, the number of views needed for 65nm will be six times greater than what is needed for 130nm.

The figure below shows the growing trend that requires PVT corners to accurately model the circuit behavior:

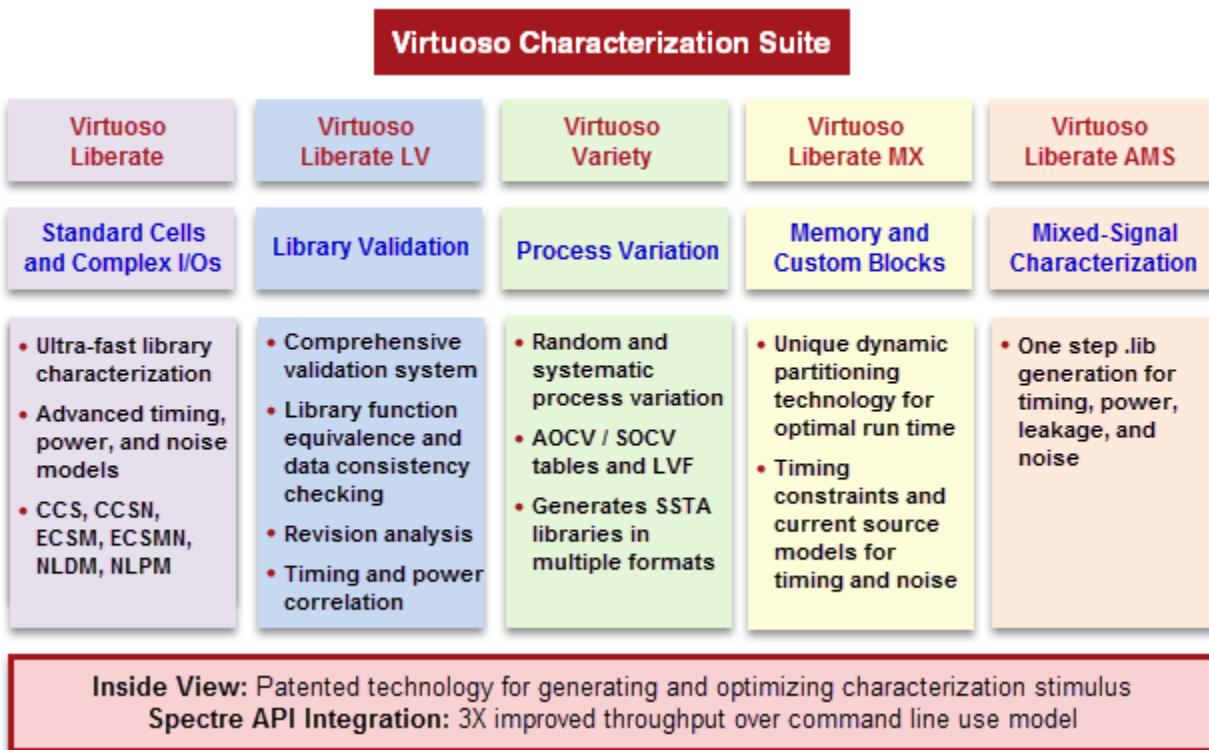


In addition, library views require more advanced models like:

- Current source models CCS and ECSM
- Statistical models – AOCV/SOCV/LVF
- Netlist extraction at various temperatures for Nanometer Process Nodes
- Support multiple foundries to assure flexibility for yield issues
- Support for many more functional designs – 1000+ STD cell, I/O, custom datapath, memory and Analog IP

Virtuoso Characterization Suite

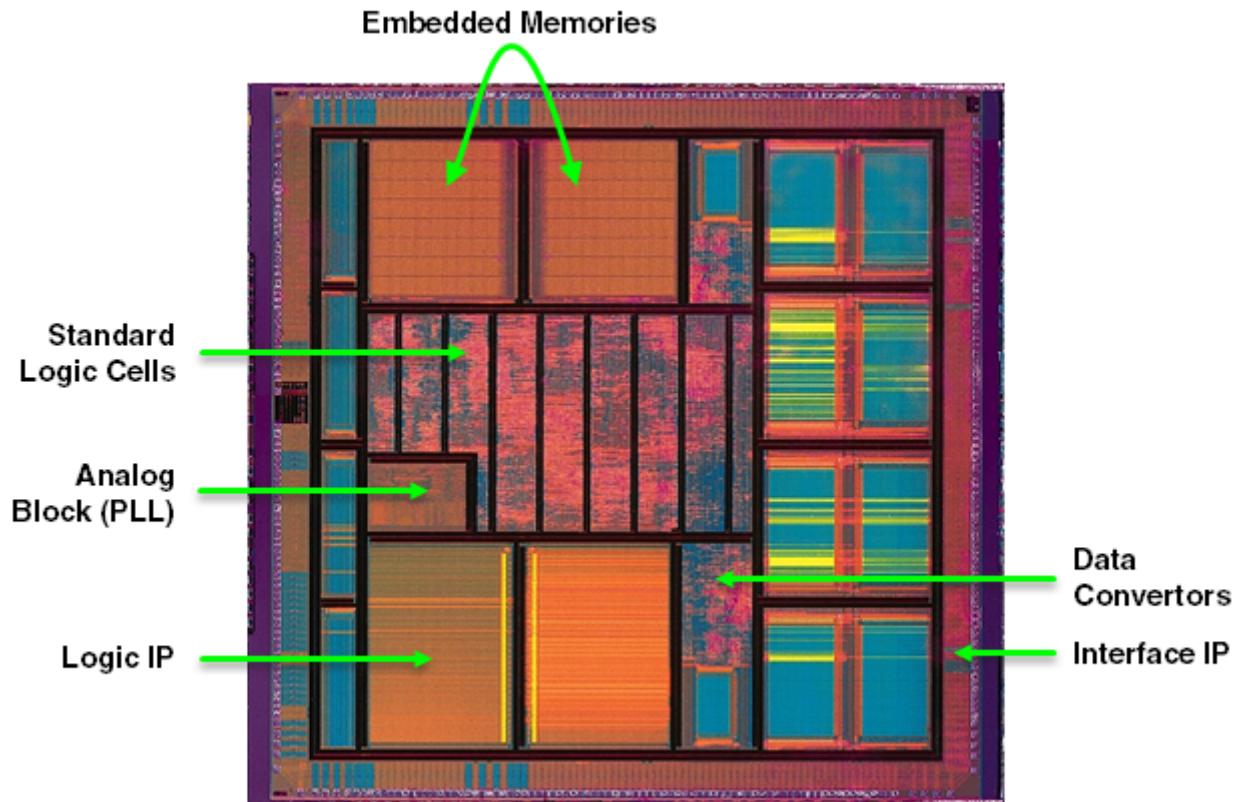
To address all the challenges, Cadence offers Virtuoso Characterization Suite that covers the complete portfolio of characterization solutions given below:



The Virtuoso Characterization Suite intends to provide highly efficient and automated electrical view creation and validation for all IP blocks that including the following:

- Logic and I/O cells (GPIO, PCI, SSTL, PECL, and so on)
- Embedded Memory (SRAM, ROM, Register files, CAM, and so on)
- Custom digital blocks (custom cells, datapath, cores, and so on)

- Interface IP and analog blocks (USB, Serdes, DDR, and so on)



System Requirements

Liberate, Variety, Liberate MX, Liberate LV, and Liberate AMS run exclusively on Linux operating system. The following table lists the supported platforms:

| Architecture | Development OS | Supported Environments |
|----------------|----------------|------------------------|
| x86_64 (32/64) | RHEL 5.5 | RHEL 6 |
| | | SLES10 |
| | | SLES11 |

For detailed information about the requirements, see [Computing Platforms](#).

Software and Licensing Requirements

■ LIBERATE 15.1

The following table lists the required server and client product numbers for each product in the Virtuoso Characterization Suite:

| Product Name | Server Product Number | Client Product Number |
|--------------|-----------------------|-----------------------|
| Liberate | ALT110 | ALT111 |
| Variety | ALT210 | ALT211 |
| Liberate MX | ALT410 | ALT411 |
| Liberate LV | ALT610 | ALT611 |
| Liberate AMS | ALT810 | ALT811 or ALT812 |

■ MMSIM 15.1

| Product Name | Product Number |
|--------------|---|
| Spectre® XPS | 91600 or 90004 |
| Spectre® APS | 3500 (restricted for characterization), 91050, or 90004 |

About This Manual

The *Virtuoso Liberate Reference Manual* describes the Cadence® Virtuoso® Liberate tool. The manual includes opening chapters that describe what Liberate LV does and how to get started with the tool. Later chapters discuss the commands and variables that can be used with Liberate.

Audience Profile

This manual is aimed at developers and designers who want to work on to create electrical views in industry standard formats such as Synopsys Liberty (.lib) format. It assumes that you are familiar with:

■ SPICE simulations

- Basic expected behavior of the design being used

Additional Documents for Reference

For information about known problems and solutions, see [Virtuoso Characterization Suite Known Problems and Solutions](#).

For a list of new features in a release, see [Virtuoso Characterization Suite What's New](#).

For information about other products in Virtuoso Characterization Suite, refer to the following manuals:

- [Virtuoso Liberate LV Reference Manual](#) describes the Liberate LV library validator—a tool that provides a collection of capabilities used to validate and verify the data consistency, accuracy, and completeness of cell libraries.
- [Virtuoso Variety Reference Manual](#) describes the Virtuoso Variety process variation cell characterizer—a tool that characterizes process variation aware timing models and generates libraries for multiple statistical static timing analyzers (SSTA) without requiring re-characterization for each unique format.
- [Virtuoso Liberate MX Reference Manual](#) describes Liberate MX—a tool that provides library creation capabilities to cover memory cores.
- [Virtuoso Liberate AMS Reference Manual](#) describes Liberate AMS—a tool that provides library creation capabilities for Analog Mixed Signal (AMS) macro blocks.
- [ALAPI Reference Manual](#) describes a Tcl interface that allows access to the Liberate characterized Library DataBase (LDB).

Rapid Adoption Kits

Cadence provides [Rapid Adoption Kits](#) that demonstrate how to use Virtuoso applications in your design flows. These kits contain design databases and instructions on how to run the design flow.

Typographic and Syntax Conventions

This section describes the typographic and syntax conventions used in this manual.

| | |
|-----------------|--|
| literal | Non-italic words indicate keywords that you must enter literally. These keywords represent command or variable names. |
| <i>argument</i> | Words in italics indicate text that you must replace with an appropriate value. |
| < > | Angle brackets indicate text that you must replace with a single appropriate value. When used with vertical bars, they enclose a list of choices from which you must choose one. |
| | Vertical bars separate a choice of values. They take precedence over any other character. |
| - | Hyphens denote arguments of commands or variables. Usually arguments denoted in this way are optional but, as noted in the syntax, some are required. The hyphen is part of the name and must be included when the argument is used. |
| { } | Braces indicate values that must be denoted as a list. When used with vertical bars, braces enclose a set of values from which you must choose one or more. |
| | When you specify a list, the values must be enclosed by either quotation marks or braces. For example, {val1 val2 val3} and "val1 val2 val3" are legal lists. |

Some arguments are positional and must be used in the order they are shown. Any positional arguments that are used must be given after any arguments denoted with hyphens.

Customer Support

For assistance with Cadence products:

- Contact Cadence Customer Support

Cadence is committed to keeping your design teams productive by providing answers to technical questions and to any queries about the latest software updates and training needs. For more information, visit: <https://www.cadence.com/support>

- Log on to Cadence Online Support

Customers with a maintenance contract with Cadence can obtain the latest information about various tools at: <https://support.cadence.com>

Feedback about Documentation

You can contact Cadence Customer Support to open a service request if you:

- Find erroneous information in a product manual
- Cannot find in a product manual the information you are looking for
- Face an issue while accessing documentation by using Cadence Help

You can also submit feedback by using the following methods:

- In the Cadence Help window, click the *Feedback* button and follow instructions.
- On the Cadence Online Support Product Manuals page, select the required product and submit your feedback by using the *Provide Feedback* box.

Introduction

The *Virtuoso Liberate Reference Manual* describes the Cadence® Virtuoso® Liberate solution. The document includes opening chapters that describe what Liberate does and how to get started with it. Later chapters discuss the commands and variables that can be used with Liberate.

What is Liberate?

Liberate is an accurate, highly efficient and easy-to-use library characterizer that creates electrical views (timing, power and signal integrity) in industry standard formats such as Synopsys Liberty (.lib) format. It requires only the foundry device models and the extracted cell netlists (in SPICE format) from which it will create all the required electrical views. By automating the process for generating views, Liberate ensures that the library's functional, timing, power and signal integrity values are both accurate and complete thus avoiding potential chip failures caused by missing or bad library data.

Performance is Key

Given this increase in views it is paramount that the characterization for nanometer technologies is very efficient. Liberate deploys a number of techniques to improve turnaround time for library creation, including use of a built-in circuit simulator, **Alspice**, that is specially optimized for the simulation of digital circuits. As the creation of a typical standard cell library view often requires over a million simulations, using a simulator which is optimized for characterization can significantly reduce runtimes. In addition, Liberate uses intelligent techniques to reduce the number of simulations required by eliminating unnecessary vector sequences and deploying smart searching techniques to characterize timing constraints (setup, hold etc.) within sequential cells, which is typically the major bottleneck in the characterization process.

To further improve turnaround times, Liberate supports both multi-threaded and distributed parallel processing. The distribution occurs at a very fine grained level so that the characterization effort is optimally distributed amongst all the available processors.

In addition to using **Alspice**, Liberate supports using external SPICE simulators such as **HSPICE®**, and **Spectre®**. Liberate can utilize its internal simulator to reduce the work load of the external simulator. This maintains consistency with 'golden' circuit simulators while leveraging much of Liberate's speed advantage.

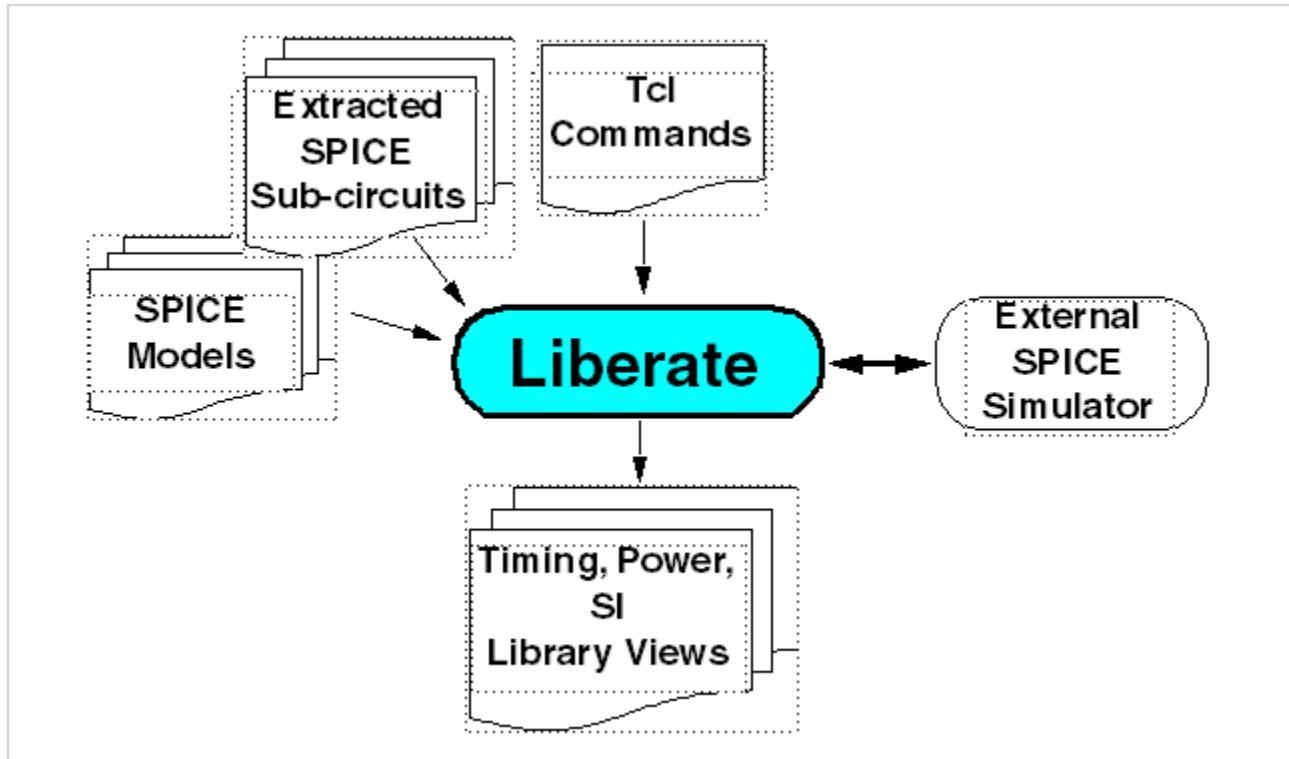


Figure 3: Using Liberate with an external SPICE simulator

Liberate includes the ability to graphically compare libraries. This can be used to verify the results of Liberate against an existing 'golden' library. It also provides early feedback on how foundry model updates will impact a library's electrical characteristics.

Getting Started

This chapter describes how to start using Liberate.

Before using Liberate, make sure that it is installed correctly and that all the necessary prerequisite data is available. (See the **Cadence Installation Guide**, and **Cadence License Manager** manuals.)

Environment Variables

Path to Executable

Set the following environment variables to include Liberate in your executable path:

```
% setenv ALTOSHOME <install_dir>/<liberate_release_name>
% set path=($path $ALTOSHOME/bin)
```

Set the following to include integrated Spectre in your executable path:

```
% set path ($path $ALTOSHOME/tools.lnx86/spectre/bin)
```

64-bit Machine Support

Liberate also ships with support for 64-bit machines. To use a 64-bit port, set the **ALTOS_64** environment variable before running Liberate, as shown below.

```
% setenv ALTOS_64 1
```

If you are using Spectre Kernel Interface (SKI), the Liberate and Spectre binaries must be compatible. This is defined as the same style of binary (32- vs 64 bit) and version. For the version of MMSIM (also known as, Spectre) that is tested for compatibility with Liberate, see the README which ships with the Liberate release. Do not use a version of MMSIM older than the tested compatible version. To enable 64-bit support in MMSIM:

```
setenv CDS_AUTO_64BIT 1
```

Shell Environment Variables for Controlling Licensing Checks

- **ALTOS_LIC_MAX_TIMEOUT**

```
setenv ALTOS_LIC_MAX_TIMEOUT <value>
```

where,

value is time in seconds

This shell environment variable specifies for how long Liberate (both server and client) will wait to obtain a license.

For a server process, if the ALTOS_QUEUE variable is enabled, Liberate will attempt to check out 1 server license. If the max timeout is reached, and no server license has been checked out, then Liberate will reset the timer and loop back to continue waiting for a license. For a client, when the max timeout is reached and at least one license was checked out, then the Liberate client will start to run with the licenses it has. No additional licenses are checked out.

- **ALTOS_LIC_CHECK_ALT_TIMEOUT**

```
setenv ALTOS_LIC_CHECK_ALT_TIMEOUT <value>
```

where,

value is time in seconds

Some Cadence characterization products can run using more than one product license. This shell environment variable controls both the server and client timeout before trying to check out an alternative license feature if there are any such licenses in the license pool.

Managing Licenses

Liberate uses a server/client licensing scheme. A server license is used for invoking and monitoring the characterization run on the server machine while the client licenses are used for running characterization on the client machines and for any post-processing of the library database. Each Liberate server can access all the available client licenses. For example, with two server licenses and forty client licenses the following configurations are all valid:

- A single characterization run using 40 client processes
- Two simultaneous characterization runs, each with 20 client processes
- Two simultaneous characterization runs, one with 30 client processes and one with 10 client processes

Note: It is important not to request more client licenses than are available because doing so can increase the wall-clock time required to complete the characterization.



Ensure that the license daemon (`cds1md`) and the license server (`lmgrd`) have the same version and that this version is the same as that required for a release. For example, v11.11.1 is required for the Liberate 15.1 release. If a mismatch is detected, unexpected license behavior might be observed. For example, the license search path can be reset to `<none>` after a failed license check out request. This can result in incorrect license checking process.

On a 64-bit license host, the 64-bit `cds1md` and `lmgrd` must be used instead of the default 32-bit ones.

Wait for Available License

When a Liberate job is submitted, a request is made for a license. To request that Liberate wait until a license becomes available, it is necessary to set the following CSH environment variable:

```
setenv ALTOS_QUEUE 1
```

Liberate clients run using different types of client license features, depending on the product names. Some product licenses can be mixed and matched together. Liberate clients can run using the following product licenses: Liberate_Client, Liberate_LX_Client, and Variety_LX_Client. For example, to run Liberate with 4 threads, two Liberate_Client and 2 Liberate_LX_Client features could be utilized.

When a Liberate server starts, it checks out 1 server license – Liberate_Server, Variety_LX_Server, or Liberate_LX_Server. Later, when simulations are ready to begin, Liberate tries to check out N clients, where N is the number of threads specified in the `char_library -thread` command option. Liberate tries to check out a combination of:

Liberate_Client + Liberate_LX_Client + Variety_LX_Client

- If Liberate acquires all N licenses, then it starts simulations using N threads.
- If Liberate acquires M licenses, $0 < M < N$, and `ALTOS_QUEUE` is not set to 1, then it starts M threads of jobs.
- If Liberate does not acquire a license and `ALTOS_QUEUE` is not set to 1, then it quits.
- If Liberate acquires fewer than N licenses and `ALTOS_QUEUE` is set to 1, then it waits for up to the value of the `lic_max_timeout` variable, trying to get all N licenses.

After `lic_max_timeout` is reached, if Liberate acquires `M` licenses and `M > 0`, then it starts `M` threads.

- ❑ If `M` is 0, then it again waits for another `lic_max_timeout` seconds to acquire client licenses. This process is repeated until at least one client license can be checked out (since `ALTOS_QUEUE` is set to 1).

Invoking Liberate

Users may do a quick version check of Liberate by executing this from the command line:

```
% liberate -v
```

This will print the current version of Liberate and exit.

Liberate utilizes `stdout` and `stderr` for all messages. By default, no log file is created. To invoke Liberate while creating a log file:

```
% liberate my.tcl |& tee my.log
```

It is possible to pass arbitrary strings into Liberate. This is done by added any strings to the Liberate command after the filename. Example:

```
% liberate my.tcl /home/user/liberate/my_run_dir |& tee my.log
```

Then to access the strings in a Liberate run:

```
#puts "Number of Tcl arguments = $argc"
#puts "Tcl Command line = $argv0 $argv"
#for {set i 0} {$i < $argc} {incr i} {
#    set curr_arg [lindex $argv $i]
#    puts "arg $i = $curr_arg"
#}
if {$argc == 0} {
    set run_dir [exec pwd]
} else {
    set run_dir [lindex $argv 0]
}
puts "Set run_dir to $run_dir"
```

System Libraries

Liberate is shipped enabled with dynamic linked system libraries. To verify **Liberate** is capable of running on your system, just try executing it. If Liberate fails to start properly, it may be possible that you have an old system and that there are missing or incorrect system libraries. If this occurs, and you have already checked your environment setup is correct, you can try using static linked binaries by setting the following environment variable. Example:

```
setenv ALTOS_USE_STATIC_BINARIES 1
```

Preparing for Characterization

Three pieces of data are required to run Liberate. These are:

1. Extracted standard cell netlists in SPICE format
2. Foundry device models in SPICE format
3. A Liberate command file in Tcl format

Extracted Cell Netlist

The transistors, diodes, resistors, capacitors and extracted parasitic elements (RCs) that comprise the cell are passed to Liberate in SPICE format. Extracted SPICE netlists can be created directly from the cell layout by device and interconnect parameter extraction tools. Standard SPICE and Hspice as well as some Spectre netlist formats are currently supported. Multiple cells can be specified in a single file or as a group of files. Each cell to be characterized must have a **.subckt** definition in the files passed to Liberate. To specify the cell netlists use the **read_spice** Tcl command.

```
read_spice {nand2x4.spi nor2x2.spi}
```

Device Models

The device models are supplied by the foundry and represent the electrical parameters of the target process. The device models include models from transistors (P and N channel), diodes, capacitors, and resistors. Most device model files include different parameters for different process corners such as a typical corner, fast corner, and a slow corner.

To specify the voltage and temperature to use for characterization, use the **set_operating_condition** command. The operating conditions, including the temperature and voltages should be specified before reading in the netlist.

```
set_operating_condition -voltage 1.2 -temp 25
```

To read device models into Liberate, use the **read_spice** Tcl command:

```
read_spice {models.spi nand2x4.spi nor2x2.spi}
```

Tcl Command File

Liberate uses the Tcl scripting language to control the characterization process. The Tcl script is used to specify the cell netlists, SPICE models and operating conditions. In addition, the Tcl script defines the range of data that the characterization is to be performed over, such as input slew and output-loading conditions. Liberate will simulate and measure each cell using each of the specified input slews and loads and will generate the appropriate delay tables,

timing checks (setup, hold etc) and power information (switching power, hidden power, state-dependent leakage). Liberate can also generate library information for industry recognized formats such as the Composite Current Source (CCS) model and the Effective Current Source Model (ECSM), among others. The Tcl commands available for controlling Liberate are given in Chapter 3.

A sample Tcl script for running Liberate is shown below. This script will characterize the cells NAND2x4, NOR2x2 and DFFX1:

```
# Define templates for characterization.
# Delay template for 3 input slews and 3 loads
define_template -type delay \
    -index_1 {0.025 0.1 0.25} \
    -index_2 {0.0010 0.015 0.100} \
    delay_3x3

# Power template for 3 input slews and 3 loads
define_template -type power \
    -index_1 {0.025 0.1 0.25} \
    -index_2 {0.0010 0.015 0.100} \
    power_3x3

# Timing constraint template for 3 input slews
define_template -type constraint \
    -index_1 {0.025 0.1 0.25} \
    -index_2 {0.025 0.1 0.25} \
    constraint_3x3

# Specify the PVT for this characterization run
set_operating_condition -voltage 1.2 -temp 25

# Read in the SPICE subckts and models
read_spice {models.spi nand2x4.spi nor2x2.spi dffx1.spi}

# Define how to characterize each group of cells
define_cell \
    -input {A1 A2 D} \
    -output {Z Q QN} \
    -clock {CK} \
    -async {SN} \
    -delay delay_3x3 \
    -power power_3x3 \
    -constraint constraint_3x3 \
    {NAND2X4 NOR2X2 DFFX1}

# Perform characterization and write out the library
char_library
write_library tt_1p2_25.lib
```

Liberate can automatically create a list of template and cell definitions from an existing library. An example Tcl file for template creation is show below:

```
# Read in an existing library to create templates
read_library existing.lib
write_template liberate_templates
```

The above will create a file called `liberate_templates.tcl`. This file can be used in a subsequent Liberate characterization run. For example:

Virtuoso Liberate Reference Manual

Getting Started

```
# Read templates and cell definitions for characterization
source liberate_templates.tcl

# Specify the PVT for this characterization run
set_operating_condition -voltage 1.2 -temp 25

# Read in the SPICE subckts and models
read_spice {models.spi nand2x4.spi nor2x2.spi dffx1.spi}

# Perform characterization and write out the library
char_library
write_library tt_1p2_25.lib
```

Liberate can also be used to re-characterize an existing library to update the models or for a different PVT condition. To do this, use the **verbose** option to **write_template**. This will create a template with additional **define_arc** commands that adhere to the existing library structure. An example is shown below:

```
# Read in an existing library to create verbose templates
read_library existing.lib
write_template -verbose existing_templates
```

The above will create a file called `existing_templates.tcl`. This file can be used in a subsequent Liberate characterization run. For example:

```
# Read templates and cell definitions for characterization
source existing_templates.tcl

# Specify the PVT for this characterization run
set_operating_condition -voltage 1.2 -temp 75

# Read in the spice subckts and models
read_spice {models.spi nand2x4.spi nor2x2.spi dffx1.spi}

# Perform characterization and write out the library
char_library
write_library tt_1p2_75.lib
```

The re-characterization flow shown above will honor all of the arc conditions that pre-exist in a given library. The re-characterization flow is useful for updating an existing Liberate created library with a new PVT (process/temperature/voltage). However, if using a non-Liberate created library, be aware that the input library may not cover all the logical states that Liberate derives from the transistor level description. It is recommended instead to use an existing library to only create the cell and template definitions to be used in a Liberate run. This is to ensure consistent and complete logic-state coverage.

Running Liberate

To perform a characterization, simply type **liberate** followed by the Tcl command file. A trial run of Liberate can be performed as follows:

```
% cd $ALTOSHOME/examples/liberate
% liberate char.tcl |& tee char.log
% vi example.lib
```

The example library will contain delay, power and leakage power constructs for a few example cells. To re-characterize this library at a different temperature use:

```
% liberate gen_template.tcl |& tee gen_template.log  
% liberate char.tcl |& tee char.log
```

To compare the two libraries use:

```
% echo "compare_library -gui example.cmp.gui \  
-report example.cmp.txt ref.lib example.lib" > comp.tcl  
% liberate comp.tcl |& tee comp.log  
% vi example.cmp.txt  
% lcplot example.cmp.gui
```

It is recommended that library characterization and model generation are distinguished as two separate Liberate runs. This allows for separate optimization of Liberate settings between characterization and model generation.

By default, Liberate keeps all the characterized library data in memory for improved efficiency. However this limits the size of the cell library (~1000 cells) that can be characterized in a single run using 32-bit machines within a reasonable amount of time. The alternative is to use only 64-bit machines or to distribute the work onto multiple hosts. Liberate can automate the distribution using the packet mode. The packet mode is discussed in Chapter 3.

The final model generation step (**write_library**, **write_verilog**, and so on.) should then be performed on a 64-bit machine (use CDS_AUTO_64BIT environment variable) using the **read_Idb** command. The **read_Idb** command requires a value. This value can be an ldb file or a directory. The **read_Idb** command automatically adjusts and handles the data. There is no option needed (or available) for **read_Idb** to specify if it is loading a file or a directory.

When generating models, the LDB must have all necessary settings stored inside it or the run script that is used to generate the models must load the LDB and provide all necessary settings. To store all commands into the LDB, set the following variable:

```
set_var enable_command_history 1
```

Using Spectre

Liberate now ships with the version of Spectre that has been qualified for use with the release. See \${ALTOSHOMEROOT}/README for details on the qualified release of Spectre. To use this version of Spectre, make sure your environment is clean and does not include any reference to other versions of MMSIM/Spectre. This includes the PATH and the LD_LIBRARY_PATH settings.

To use the qualified version of Spectre with Liberate:

- Add \${ALTOSHOMEROOT}/tools.lnx86/spectre/bin to your environment PATH.



Caution

Do NOT include other versions of MMSIM/Spectre in the PATH or LD_LIBRARY_PATH settings.

To use any other specific version of Spectre:

1. Add the `bin` directory to your environment `PATH`.
2. Update your environment settings as needed.

Note: Liberate is designed to allow you to easily change the version of Spectre used for characterization. It is up to you to ensure that the environment is clean and does not mix different versions.

Virtuoso Liberate Reference Manual
Getting Started

Parallel Processing

This chapter describes how to use Liberate across multiple CPUs.

Cell libraries today typically contain more than 1000 cells with some having as many as 5000 (or more) cells. The cells range from simple inverters to very complex AOI and OAI cells to sequential cells to multibit flop arrays. To characterize these libraries, a server farm is often utilized. In general, the greater the number of CPUs utilized, the faster the run time. Liberate supports concurrent application of both multi-threading and distributed processing.

Multi-threading

In multi-threading, multiple CPU cores residing on the same physical machine operate on the same memory image. This is the simplest way to use parallel processing with Liberate. The `-thread` argument of the **char_library** command specifies how many multi-threaded CPUs Liberate can use. The default value of the `-thread` argument is 0, which allows Liberate to use all of the CPUs on each machine. Liberate does not observe the machine loading and can easily overload a machine when the number of threads is allowed to default to 0.

Distributed Processing

Distributed processing occurs when a program distributes the work across many hosts. Liberate partitions the characterization task into a group of related simulations to be performed on each of the available CPUs. Liberate supports the following two forms of distributed processing: Client and Packet.

Using a Queuing System

For starting remote jobs on a client machine, use the `rsh_cmd` (default `ssh`) parameter to specify the shell command.

Packet Mode

Packet mode pre-processes only those cells that will be characterized on each client machine. This mode is more suitable for a large number of cells.

Arc Packet Flow

When Liberate is called in the Arc Packet flow, a characterization job (the server) is initiated. The server analyzes each cell to sort the cells to be characterized from largest to smallest. Then it sends a request to the queuing system to initialize each client. Set the packet_clients parameter to the total number of clients Liberate can submit. The client does the actual simulation work. As each client starts, it will notify the server when it is initialized and ready. The server optimally divides the cells to be characterized between the available clients. This means some of the cells will be characterized on a single client while others might be spread over multiple clients. The threshold at which a cell is divided across multiple clients is controlled by the following formula:

```
packet_arcs_per_thread * threads
```

where:

packet_arcs_per_thread is a parameter.

threads is the value of the -thread argument supplied to the char_library command.

The benefits of the Arc Packet flow are as following:

- Scalability

Instead of dividing the characterization run into cells, it is now divided into jobs (groups of arcs). Each cell in a library can contain 10 to 200+ arcs. Therefore, we have the choice of running 200 simulations on a single client with 4 CPUs (50 arcs per CPU) or we can run 25 clients with 4 CPUs each (2 arcs per CPU). The simulation wall clock time might be reduced by 25x. However, due to the overhead in preprocessing, there might be a small increase in the CPU time.

- Load balancing

As each cell is now divided into its component arcs when doing job distribution, the server can balance work better between all available clients.

- Fault tolerance

The Arc Packet flow has numerous redundancy built-in features, such as following, to handle the potential machine issues:

- ❑ All clients communicate regularly with the server and send statistics about load and free memory and so on.
- ❑ An unfinished job can be reassigned to a different client.
- ❑ A stalled or killed client will be resubmitted to the queue.
- ❑ Both the server and the clients will wait when an NFS disk is too slow or busy. The server forces NFS synchronizations for distributed jobs to make sure that files are visible to clients even if the NFS disk system is slow.
- ❑ The server regularly monitors the status of all client hosts and prints warning messages if the host is overloaded or about to run out of memory.

Enabling the Arc Packet Flow

The steps to enable the Arc Packet flow are as following:

1. Set up a basic run.

Run Liberate on a handful of cells without using any batch queue, Arc Packet flow, or SKI. Think of this run as a pipe cleaner run to test the interface to the simulator and the basic characterization settings such as power supplies, templates, netlists, and models.

When using Spectre, here are some commonly recommended production settings. Your production settings can be different due to changes over time in the recommended settings, and your specific simulation needs.

```
# Set the path to the simulation binary
set_var extsim_cmd "${ALTOSHOME}/tools.lnx86/spectre/bin/spectre"

# Specify the simulator command line arguments
set_var extsim_cmd_option "+spice"

# Enable the reuse of initial conditions to improve run time
set_var extsim_reuse_ic 3

# The Deck Header is used to include arbitrary Spectre syntax commands.
set_var extsim_deck_header simulator lang=spectre\nSetOption1 options
    reltol=1e-4\nsimulator lang=spice"

## Simulator Options
set_var extsim_leakage_option "method=gear save=nooutput gmin=1e-15
    redefinedparams=ignore"
set_var extsim_option "method=gear save=nooutput gmin=1e-15
    redefinedparams=ignore"

## Tran Append
set_var extsim_tran_append "lteratio=10"
```

Note: See the README file in the release directory for the compatible version of Spectre. If the Spectre version that you are using is different from the qualified release,

a message advising which version is qualified will be printed in the log file.

2. Enable Arc Packet flow.

```
# When using an active driver, enable reuse of the driver waveform
# (see set_driver_cell)
set_driver_waveforms_file driver_waveforms.wave

# For optimal throughput, always use multi-threading.
set THREADS 2

# Specify the maximum number of batch queue submissions (for example, LSF bsub
# commands)
# The total number of Liberate_Client and Simulator licenses required is
# packet_clients * THREADS = 40 for this run
set_var packet_clients 20
set_var packet_mode arc

# Specify the batch submission command.
# The number of threads should be specified if greater than 1. All threads must
# also run on the same host.
set_var rshcmd "bsub -q qname -R \"span\[hosts=1\]\\" -n ${THREADS} -o %B/log
-e %B/log"
char_library -thread $THREADS -cells $cells
```

3. Analyze the run-time statistics by using the logs2xlsx utility.

The logs2xlsx utility analyzes the run-time logs from an arc or cell packet characterization run for overall CPU utilization. A report is generated to map the packet run time. This utility takes the following two arguments:

- The path to ldb directory. This can be a full or relative path.
- The path and name of the .xlsx file where the results will be saved.

For example:

```
 ${ALTOSHOMEROOT}/bin/logs2xlsx ${rundir}/my.ldb.gz ${rundir}/runtime.xlsx
```

The logs2xlsx utility automatically detects if the ldb directory contains cell-packet or arc-packet log files, parses them, and saves the data into the named .xlsx file. The .xlsx file can be opened with Microsoft Excel (2007 or later) or OpenOffice Calc on Linux machines. The .xlsx file has the following three workbooks (tabs):

Virtuoso Liberate Reference Manual

Parallel Processing

- **Summary:** The first section on this tab includes important settings used for arc-packet or cell-packet flow. The second section below it includes a summary for each group of arcs or cell in the arc-packet flow or each cell in the cell-packet flow.

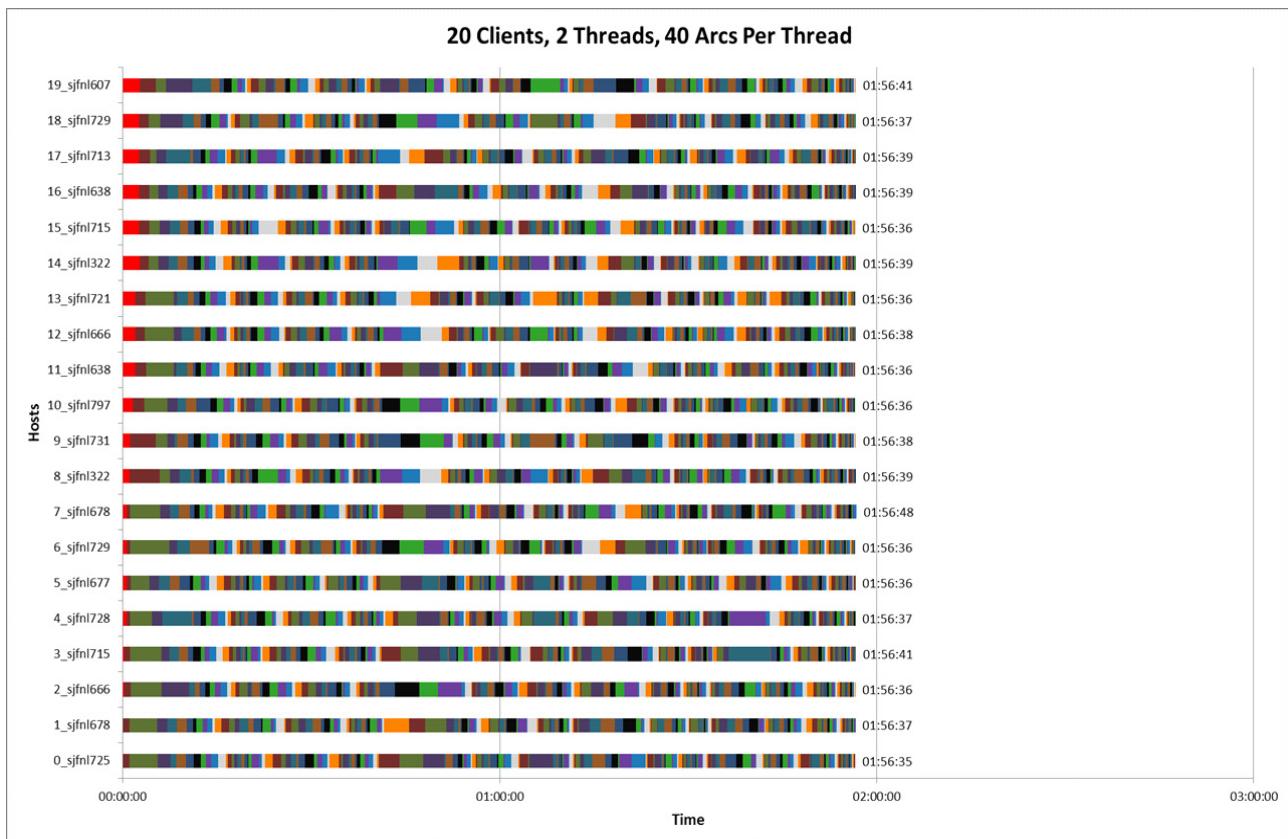
| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T |
|--------------------------|---------|--------|---------|-----------------------------|---------------------|-----------|-------|-----------------|-----------------|------------|--------------|------------|----------|----------|------------|------------|--------------|----------------|--------------|
| Cell Name | Host | Client | Task ID | Start Date/Time | Finish Date/Time | Wall Time | Error | Constraint Arcs | Transition Arcs | Power Arcs | Leakage Arcs | Noise Arcs | Cap Arcs | MPW Arcs | Other Arcs | Total Arcs | Sim CPU Time | Total CPU Time | Task Latency |
| 1 Summary for run at: | | | | ./..:/path/to/run.ldb.gz | | | | | | | | | | | | | | | |
| 2 ALTOSHOME | | | | /path/to/LIBERATE14.1/lnx86 | | | | | | | | | | | | | | | |
| 3 packet_clients | 20 | Hosts | 20 | | | | | | | | | | | | | | | | |
| 4 threads | 2 | Cells | 1227 | | | | | | | | | | | | | | | | |
| 5 packet_arcs_per_thread | 40 | | | | | | | | | | | | | | | | | | |
| 6 Release | 14.1 | | | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | | | | |
| 8 | | | | | | | | | | | | | | | | | | | |
| 9 CELL_NAME_1 | sjfn713 | 17 | 1 | 09/15/2014 14:24:05 | 09/15/2014 14:24:21 | 00:00:16 | 0 | 0 | 10 | 16 | 4 | 0 | 0 | 0 | 0 | 30 | 00:00:18 | 00:00:19 | 00:00:00 |
| 10 CELL_NAME_2 | sjfn728 | 4 | 1 | 09/15/2014 14:24:05 | 09/15/2014 14:24:24 | 00:00:19 | 0 | 0 | 10 | 16 | 4 | 0 | 0 | 0 | 0 | 30 | 00:00:19 | 00:00:20 | 00:00:00 |
| 11 CELL_NAME_3 | sjfn677 | 5 | 1 | 09/15/2014 14:24:08 | 09/15/2014 14:24:26 | 00:00:19 | 0 | 0 | 10 | 16 | 4 | 0 | 0 | 0 | 0 | 30 | 00:00:22 | 00:00:23 | 00:00:01 |
| 12 CELL_NAME_4 | sjfn678 | 7 | 1 | 09/15/2014 14:24:08 | 09/15/2014 14:24:32 | 00:00:24 | 0 | 0 | 10 | 16 | 4 | 0 | 0 | 0 | 0 | 30 | 00:00:30 | 00:00:31 | 00:00:00 |
| 13 CELL_NAME_5 | sjfn666 | 12 | 1 | 09/15/2014 14:24:34 | 09/15/2014 14:24:48 | 00:00:16 | 0 | 0 | 4 | 10 | 4 | 0 | 0 | 0 | 0 | 18 | 00:00:11 | 00:00:11 | 00:00:02 |
| 14 CELL_NAME_6 | sjfn638 | 16 | 1 | 09/15/2014 14:24:34 | 09/15/2014 14:24:47 | 00:00:15 | 0 | 0 | 4 | 10 | 4 | 0 | 0 | 0 | 0 | 18 | 00:00:12 | 00:00:12 | 00:00:02 |
| 15 CELL_NAME_7 | sjfn638 | 11 | 1 | 09/15/2014 14:24:34 | 09/15/2014 14:24:46 | 00:00:12 | 0 | 0 | 4 | 10 | 4 | 0 | 0 | 0 | 0 | 18 | 00:00:08 | 00:00:09 | 00:00:00 |
| 16 CELL_NAME_8 | sjfn797 | 10 | 1 | 09/15/2014 14:24:39 | 09/15/2014 14:24:51 | 00:00:12 | 0 | 0 | 4 | 10 | 4 | 0 | 0 | 0 | 0 | 18 | 00:00:09 | 00:00:10 | 00:00:00 |
| 17 CELL_NAME_9 | sjfn715 | 15 | 1 | 09/15/2014 14:24:40 | 09/15/2014 14:24:51 | 00:00:11 | 0 | 0 | 4 | 10 | 4 | 0 | 0 | 0 | 0 | 18 | 00:00:10 | 00:00:11 | 00:00:00 |
| 18 CELL_NAME_10 | sjfn322 | 8 | 1 | 09/15/2014 14:24:41 | 09/15/2014 14:24:55 | 00:00:14 | 0 | 0 | 4 | 10 | 4 | 0 | 0 | 0 | 0 | 18 | 00:00:12 | 00:00:12 | 00:00:00 |
| 19 CELL_NAME_11 | sjfn721 | 13 | 1 | 09/15/2014 14:24:41 | 09/15/2014 14:24:53 | 00:00:12 | 0 | 0 | 4 | 10 | 4 | 0 | 0 | 0 | 0 | 18 | 00:00:08 | 00:00:09 | 00:00:00 |
| 20 CELL_NAME_12 | sjfn729 | 6 | 1 | 09/15/2014 14:24:42 | 09/15/2014 14:24:54 | 00:00:12 | 0 | 0 | 4 | 10 | 4 | 0 | 0 | 0 | 0 | 18 | 00:00:09 | 00:00:10 | 00:00:00 |
| 21 CELL_NAME_13 | sjfn729 | 18 | 1 | 09/15/2014 14:24:42 | 09/15/2014 14:24:55 | 00:00:13 | 0 | 0 | 4 | 10 | 4 | 0 | 0 | 0 | 0 | 18 | 00:00:09 | 00:00:10 | 00:00:00 |
| 22 CELL_NAME_14 | sjfn728 | 4 | 1 | 09/15/2014 14:24:45 | 09/15/2014 14:24:58 | 00:00:14 | 0 | 0 | 4 | 10 | 4 | 0 | 0 | 0 | 0 | 18 | 00:00:10 | 00:00:11 | 00:00:01 |
| 23 CELL_NAME_15 | sjfn797 | 10 | 1 | 09/15/2014 14:19:01 | 09/15/2014 14:19:25 | 00:00:25 | 0 | 0 | 14 | 30 | 8 | 0 | 0 | 0 | 0 | 52 | 00:00:30 | 00:00:31 | 00:00:01 |
| 24 CELL_NAME_16 | sjfn728 | 4 | 1 | 09/15/2014 14:19:11 | 09/15/2014 14:19:35 | 00:00:27 | 0 | 0 | 14 | 30 | 8 | 0 | 0 | 0 | 0 | 52 | 00:00:33 | 00:00:34 | 00:00:03 |
| 25 CELL_NAME_17 | sjfn729 | 6 | 1 | 09/15/2014 14:19:11 | 09/15/2014 14:19:40 | 00:00:34 | 0 | 0 | 14 | 30 | 8 | 0 | 0 | 0 | 0 | 52 | 00:00:40 | 00:00:42 | 00:00:05 |
| 26 CELL_NAME_18 | sjfn322 | 8 | 1 | 09/15/2014 14:19:11 | 09/15/2014 14:19:43 | 00:00:32 | 0 | 0 | 14 | 30 | 8 | 0 | 0 | 0 | 0 | 52 | 00:00:48 | 00:00:49 | 00:00:00 |
| 27 CELL_NAME_19 | sjfn322 | 14 | 1 | 09/15/2014 14:22:03 | 09/15/2014 14:22:20 | 00:00:17 | 0 | 0 | 6 | 18 | 8 | 0 | 0 | 0 | 0 | 32 | 00:00:17 | 00:00:18 | 00:00:00 |
| 28 CELL_NAME_20 | sjfn678 | 1 | 1 | 09/15/2014 14:22:04 | 09/15/2014 14:22:21 | 00:00:18 | 0 | 0 | 6 | 16 | 8 | 0 | 0 | 0 | 0 | 30 | 00:00:17 | 00:00:18 | 00:00:01 |

- **ChartData:** This tab contains the data that is used to create the chart in the next workbook.

Virtuoso Liberate Reference Manual

Parallel Processing

- **Chart:** This tab has a bar or column chart depending on the number of machines and total run time. It shows the wall clock time used for each host machine.



Interpreting the logs2xlsx Results

- A run might benefit by changing the `packet_arcs_per_thread` variable from the default setting.
 - When many hosts are idle at the end, a smaller value might be better.
 - If the colored regions are very small in comparison to the total run time, a larger value might improve the wall clock time.
- The red on the left indicates dead time where the client is waiting to be assigned to a host. If there is significant startup time for each host, there might be significant latency in the queuing system.

File Organization for the Arc Packet Flow

In the Arc Packet flow, the files are organized as described below.

■ LDB files

- The LDB files in the Arc Packet flow are stored in a directory called `altos.<ID>.ldb.gz`. The ID is a Liberate-generated code that is primarily based on the timestamp and the process ID. At the end of the run, use the `write_lldb` command to rename the temporary directory to a name of your choice.
- The temporary directory can contain one or more actual LDB files (one per cell). Each LDB file is named in the format `<CELL>.ldb.gz` where `CELL` is the name of the cell whose characterized data is stored in that particular file.

■ Log files

The client log files in Arc Packet flow capture the `stdout` and `stderr` from a particular client and are named in the format `client_<id>.log` where `id` is a number that goes from 0 to `n-1` (`n` is the number of `packet_clients`). As there might also be messages from the queuing system, the LSF log is stored in the LDB in a file called `log.<N>` when requested by the user setting of `-o` or `-e` in the `bsub` command. It should be a copy of `client_<N>.log`, but can also include LSF messages.

■ RDB files

The LDB directory can also contain some RDB data that is used to facilitate distributed characterization of cells. The RDB files get removed after the assembly job completes.

Recovery Flow

If an Arc Packet flow-based run fails to complete, the run can be restarted. To do this, locate the LDB file from the run. Load the LDB file using the `read_lldb` command before the `char_library` command and restart the run.

The LDB filename can be controlled using:

```
set_var ldb_filename_prefix MyDB
```

The LDB location can be controlled using:

```
set_var ldb_checkpoint_dir <path>/ldbDir
```

To restart an Arc Packet flow:

```
read_lldb <path>/ldbDir/MyDB.ldb.gz
```

Frequently Asked Questions

- **INFO (PRL-36):** The time out for a remote job to respond is 3600 seconds.
Answer: This means that Liberate will wait for 3600 seconds before warning the user that their clients are not getting assigned to a host. If no hosts are being assigned, it might be due to incorrect settings in the `rsh_cmd` parameter. If a few hosts are getting assigned, the root cause might be an overloaded server farm.
- Performance statistics for each cell show only CPU time. How can one know the wall clock time for each cell?
Answer: Wall clock time is only relevant for the full run in the Arc Packet flow. Given that all cells "start" at the beginning of the run, the wall clock time for any particular cell is meaningless. You should look at the total wall clock time for all your cells.

set_client Mode (Non-Packet Mode)

The **set_client** mode pre-processes all cells in each client machine. This mode allows Liberate to manage multiple hosts in a distributed processing flow without using a queuing system. However, this mode is not suitable for large libraries because it keeps all characterization data in memory possibly requiring 64-bit software and hosts with a significant amount of memory and swap.

In this mode, the `set_client` commands specify the names of client host machines to be used. For each machine, a directory in which Liberate can temporarily store data must also be specified.

The `rcp_cmd` (default `scp`) parameter can be used to specify the command for copying files from the host to the client machines. Before starting a parallel-processing job based on a **set_client** command, ensure that the following commands can be run to access the host machines without requiring any password or passphrase:

- `ssh` or `rsh` from the server to the client (For more information, see the `rsh_cmd` parameter.)
- `scp` or `rcp` a file from the server to the client (For more information, see the `rcp_cmd` parameter.)

Following is an example of **set_client** mode:

```
# Specify three client machines to use for job distribution
# Use /tmp/liberate_%N to store intermediate files
set_client -dir /tmp/liberate_%N  LinuxHost1
set_client -dir /tmp/liberate_%N  LinuxHost2
```

```
set_client -dir /tmp/liberate_%N  LinuxHost2
set_var rsh_cmd rsh
set_var rcp_cmd rcp
char_library
```

Spectre Kernel Interface

The Spectre Kernel Interface (SKI) is used to improve the run time. With SKI, there is a significantly less disk I/O and better license management.

Enabling SKI

Liberate must be run with a compatible version of Spectre. To know which version of Spectre has been qualified for use with the Liberate release that you are using, refer to the README file in the release documents. Do not use a version of Spectre that is older than the qualified version. If an incompatible version of Spectre is in use, a warning message will be printed in the log file and SKI will be disabled automatically.



The set_client command cannot be used with SKI.

After checking the Spectre version, run the following steps to enable SKI with Liberate:

1. Run Spectre in standalone mode. A lot of basic issues can be prevented by just getting Liberate working with standalone Spectre.
2. Configure the following environment settings:

- Both Liberate and Spectre must use the same type of binary (32- or 64-bit). Set either both of the following environment variables or neither of them:

```
setenv CDS_AUTO_64BIT all
setenv ALTOS_64 1
```

- Use the following Tcl settings:

```
## Spectre-SKI
set_var ski_enable 1
set_var ski_alter_mode 3
set_var ski_clean_mode 2
set_var ski_mdlthreshold_exact 1
set_var ski_power_subtract_output_load_match_extsim 1
set_var extsim_use_node_name 1
```

Virtuoso Liberate Reference Manual

Parallel Processing

```
set_var extsim_save_passed none ; # Do not save the decks with SKI
as the disk storage requirement is too high
set ::env(TMPDIR) ${rundir}/ski_tmpdir ; # SKI must be
run in $CWD, not /dev/shm or /tmp
```

Note: The recommendations can and will change from time to time.

Liberate Commands

This chapter describes the Tcl commands that control library creation.

The command options that are prefixed with a hyphen (-) are optional except where explicitly indicated. All commands have a `-help` option. When this option is included with a command name, a help message is printed out that lists all currently available options for that command. There might be options that get printed out when help is used but are not officially supported. The only supported options are those that are documented in this manual. When `-help` is used, all other command options are ignored.

| a... | |
|---|--|
| <u>add_cell_attribute</u> | <u>add_pin_attribute</u> |
| <u>add_lib_attribute</u> | <u>append_library</u> |
| <u>add_margin</u> | |
| c... | |
| <u>char_library</u> | <u>compare_ccs_nldm</u> |
| <u>check_delay_monotonicity</u> | <u>compare_library</u> |
| <u>check_lvf_merge_indices</u> | <u>copy_arc</u> |
| d... | |
| <u>define_arc</u> | <u>define_leakage</u> |
| <u>define_bundle_pins</u> | <u>define_map</u> |
| <u>define_bus</u> | <u>define_max_capacitance_attr_limit</u> |
| <u>define_cell</u> | <u>define_max_capacitance_limit</u> |
| <u>define_cell_leakage</u> | <u>define_max_transition</u> |
| <u>define_duplicate_cell</u> | <u>define_min_transition</u> |
| <u>define_duplicate_pins</u> | <u>define_out_to_out_arc</u> |
| <u>define_group</u> | <u>define_pin_load</u> |

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|----------------------------------|--------------------------------------|
| <u>define index</u> | <u>define pulse generator arc</u> |
| <u>define input waveform</u> | <u>define template</u> |
| <u>define leafcell</u> | <u>delete arc</u> |
| e... | |
| <u>em buffer cell pin bounds</u> | <u>em follow hidden power</u> |
| g... | |
| <u>get cells</u> | <u>get var</u> |
| i... | |
| <u>ibis define component</u> | <u>ibis define model selector</u> |
| <u>ibis define header</u> | <u>ibis define pin</u> |
| <u>ibis define model</u> | <u>ibis define waveform template</u> |
| m.... | |
| <u>merge library</u> | |
| o... | |
| <u>one cold</u> | <u>one hot</u> |
| p... | |
| <u>packet slave cells</u> | <u>printvars</u> |
| r... | |
| <u>read ldb</u> | <u>read vdb</u> |
| <u>read library</u> | <u>remove pin attribute</u> |
| <u>read spice</u> | <u>reset defaults</u> |
| <u>read truth table</u> | |
| s... | |
| <u>select index</u> | <u>set operating condition</u> |
| <u>set aging criteria</u> | <u>set output voltage</u> |
| <u>set attribute</u> | <u>set pin attribute</u> |
| <u>set client</u> | <u>set pin capacitance</u> |
| <u>set conditional</u> | <u>set pin delay threshold</u> |
| <u>set constraint</u> | <u>set pin gnd</u> |

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|----------------------------------|------------------------------------|
| <u>set constraint criteria</u> | <u>set pin slew threshold</u> |
| <u>set default group</u> | <u>set pin vdd</u> |
| <u>set dependent load</u> | <u>set receiver cap thresholds</u> |
| <u>set driver cell</u> | <u>set rsh cmd</u> |
| <u>set driver waveforms file</u> | <u>set sim init condition</u> |
| <u>set em skip monitor</u> | <u>set simultaneous switch</u> |
| <u>set gnd / set vdd</u> | <u>set three state</u> |
| <u>set input voltage</u> | <u>set units</u> |
| <u>set logic condition</u> | <u>set var</u> |
| <u>set max fanout</u> | <u>set vdd</u> |
| <u>set network port</u> | |
| W... | |
| <u>write datasheet</u> | <u>write top netlist</u> |
| <u>write ibis file</u> | <u>write userdata library</u> |
| <u>write ldb</u> | <u>write vdb</u> |
| <u>write library</u> | <u>write verilog</u> |
| <u>write template</u> | <u>write vital</u> |

add_cell_attribute

Provides the ability to add user-specified cell attributes to the LDB. The attributes might be applied to a cell and can be simple or complex.

Options

{cells} {attributes}

Specify attribute(s) to add to cell(s) in a library.

Example

The following illustrates how to add attributes to a cell (notice the use of wildcards for cell names):

```
add_cell_attribute {AN??D* AO3*} {
    test_cell () {
        ff (IQ,IQN) {
            clocked_on : "cp";
            next_state : "d";
            clear : "rn";
        }
    }
}
```

add lib attribute

Provides the ability to add user-specified library attributes to the LDB. The attributes might be applied to a library and can be simple or complex.

Options

{attributes} Specify attribute(s) to add to a library.

Example

The following illustrates how to add attributes to a library:

```
    add_lib_attribute  {
        wire_load ("5000") {
            resistance : 2.500000e-03;
            capacitance : 0.001000;
            area : 3.000000;
            fanout_length("1.0000000", "5.0000000");
        }
    }
```

add_margin

Adds margin (padding) to values in the library. The margin is added to all cells in the library with two exceptions: power (all types of power) and hidden (input pin power) which can be added to specific cells.

Options

| | |
|--|---|
| -abs <value> | Specify an absolute margin, that is, the amount of margin to add in standard units. Default: 0 . 0 (no margin) |
| -cells {list} | List of cells. |
| -direction <rise fall both> | Specify direction of data to add margin. Default: "both" |
| -ocv | Use this option when generating a .lib file with LVF data (see <code>write_library -sensitivity_file</code>) to apply margin to the ocv_sigma data in the output library. If <code>-sigma_type</code> is not specified, the margin is applied to <code>ocv_sigma_early</code> and <code>ocv_sigma_late</code> . To apply a different margin to early and late, separate Liberate sessions must be used. Use the <code>-rel</code> and <code>-abs</code> options to specify the margin. When used with <code>-ocv</code> , the <code>-type</code> option supports only <code>delay</code> , <code>setup</code> , and <code>hold</code> . |
| -index_1 {list} | Specifies <code>index_1</code> points. Default: all points. |
| -index_2 {list} | Specifies <code>index_2</code> points. Default: all points. |
| -pin {list} | List of pins, bus syntax. For example, <code>A[7:0]</code> |
| -related {list} | List of related pins, bus syntax. |
| -rel <value> | Specify a relative margin, that is, a relative ratio amount of margin to add. Default: 0 . 0 (0%) If set to a positive number, the resulting library values becomes larger where larger is defined as more positive and smaller is more negative. If set to a negative number, the resulting library values is smaller if positive and larger if negative. For example, 0.05 = 5% margin. |

Virtuoso Liberate Reference Manual

Liberate Commands

`-sensitivity_file <filename>`

Specifies the sensitivity file to be loaded. The sensitivity file is created by Variety using the `write_variation` command with the `-format "sensitivity"` option. The `-sensitivity_file` option can be used only with one of the following `add_margin -type` options: `constraint`, `delay`, `hold`, `non_seq_hold`, `non_seq_setup`, `recovery`, `removal`, `setup`, and `trans`. Only one sensitivity file can be used in a single Liberate run. If multiple sensitivity files are required, each with different characterized sensitivity values, these files must be merged together into a single file using the `merge_library` command.

Note: Both the `add_margin -sensitivity_file` and `write_library -sensitivity_file` options can be used in the same run, but it is not recommended as special care must be taken to ensure that double counting of sensitivity data does not result.

`-sensitivity_normalize`

Specifies to Liberate that the sensitivity data needs to be normalized to the Liberate LDB corner.

Note: This option can only be used when the `-sensitivity_file` option is used. In addition, the sensitivity file must have been created using the `write_variation` command with the `-format sensitivity_plus_nom` option.

`-sigma_factor <value>`

Specifies a sigma scale factor to apply to the `ocv_sigma_*` data before it is summed to the corresponding nominal data of the specified by `-type`. Default: `1.0`

`-sigma_type {early | late | max | min | avg}`

Specifies a list of name-value pairs for sigma scale factor. Only the following names are supported: `constraint`, `delay`, `hold`, `recovery`, `removal`, `setup`, and `trans`. If this option is not specified, the value defaults to `1.0` for all supported sensitivity types.

-type {list} Specifies the type of data to be modified. Valid types include: cap, constraint, delay, delay_ccs, hidden, hold, leakage, minimum_period, mpw, nochange, non_seq_setup, non_seq_hold, power, recovery, removal, retain, retain_ccs, retain_trans, setup, and trans. Default: apply margin to all types.

If the **-type** option is not specified, the requested margin is applied to all data types.

You can specify the type nochange to apply a margin to nochange arcs. These arcs are a subset of the **-type** constraint.

Note: The **define_arc** command must be specified to enable nochange arcs to be modeled.

The constraint type applies the same margin to all constraint types, that is, setup, hold, recovery, removal, and mpw.

If you use the types power and hidden, you can also use the -cells option to apply the margin to specific cells. However, all other types can only be applied globally.

```
add_margin \
    -type {power|hidden} \
    -cells {celllists} \
    -pin {pin lists} \
    -related {related_pin list} \
    -when "when_condition" \
    -rel rel \
    -abs abs
```

-when "string" State dependent arc.

This command can be used after **char_library**, **read_libraries**, and **read_library**, but it must be used before model generation such as with **write_library**. Multiple **add_margin** commands can be specified.

Examples

Example 1:

```
read_libraries test.ldb.gz
write_library no_margin.lib
```

Virtuoso Liberate Reference Manual

Liberate Commands

Example 2:

```
# Add 10% to power
add_margin -type power -rel 0.1

# Add 50ps to delay
add_margin -type delay -abs 50e-12
write_library margin.lib
```

Example 3:

```
read_ldb my.ldb
# Can also use read_library or char_library
add_margin -type setup -sigma_factor 3 -sensitivity_file MySensitivity.dat {
Cell1 }
write_library -filename my.lib slow
```

Example 4:

```
read_ldb DATA/char.ldb.gz
add_margin -sensitivity_file "DATA/sens.lib" -type "non_seq_setup"
-sigma_factor 0.1
add_margin -sensitivity_file "DATA/sens.lib" -type "delay" -sigma_factor 0.5
write_library -filename new_data/test.lib final
```

Example 5:

```
add_margin -ocv -rel 0.1 -type delay
write_library -sensitivity_file myLib
```

add_pin_attribute

Provides the ability to add user-specified pin attributes to the LDB. The attributes might be applied to a pin and can be simple or complex.

Options

```
{cells} {pins} {attributes}
```

Specify attribute(s) to add to the pin(s) of cell(s).

Example

The following illustrates how to add attributes to a library:

Example of adding attributes to a cell: (notice the use of wild cards for cell names)

```
add_cell_attribute {AN??D* AO3*} {  
    test_cell () {  
        ff (IQ,IQN) {  
            clocked_on : "cp";  
            next_state : "d";  
            clear : "rn";  
        }  
    }  
}
```

Example of adding attributes to a pin: (wild cards used to match cell groups and pin names)

```
add_pin_attribute lv_buffer* a* {  
    level_shifter_data_pin : true;  
    input_signal_level : "dvdd2v8to1v8";  
}
```

Sample usage:

```
# Read in ldb file  
read_ldb test.ldb.gz  
  
# Apply user data  
add_lib_attribute {myAttribute1}  
  
# Write .lib  
write_library -ccs -overwrite test.lib
```

append_library

Appends the cells from separate libraries into the output library. The cells should be unique in each library. The action is executed in the following manner:

- ❑ Appends the cell-level data "as-is".
- ❑ Combines the library-level data (attributes and groups) from multiple libraries including the templates and driver waveforms into a single library header.

Note: The command does not merge data from a cell in one library to a cell with the same name in another library. To merge library data, use the merge_library command.

Options

| | |
|------------------------------------|---|
| -allow_conflict | Allow libraries to be appended even if there are conflicts. |
| -filename <i><filename></i> | Output file name. Default: <libname>.lib |
| -overwrite | Overwrite existing .lib file. Default: do not overwrite and create a unique file name. |
| -user_data <i><filename></i> | Use this option to add additional attributes and groups to the library-level data of the final library. No attributes are added to the cell-level data. |
| {libraries} | List of libraries to append. |
| <libname> | Library name. |

Handling Driver Waveform Conflicts

When appending libraries with driver waveforms any conflicts in driver waveforms (normalized_driver_waveform groups with the same driver_waveform_name in different libraries) are treated as follows:

- **Same index_2 (normalized_voltage) but different slews (input_net_transition index_1):** If the conflicting normalized driver waveform refers to a different set of slews to the original normalized driver waveform, those slews are merged with the original to form a new normalized waveform group using the original driver_waveform_name name. For example:

If the original normalized driver waveform group with `driver_waveform_name` 'active_driver' has slews (`index_1` entries): **0.1 0.3 0.5**

...and the conflicting waveform group with `driver_waveform_name` 'active_driver' has slews: **0.2 0.4 0.6**

...the output normalized driver waveform group 'active_driver' will include slews:
0.1 0.2 0.3 0.4 0.5 0.6.

- **Same index_2** (normalized_voltage) **with same slews** (input_net_transition `index_1`): If the conflicting waveform has some of the same slews as the original waveform, then the waveform values must match with the original waveform within a 1ps tolerance. If all the normalized_driver_waveform values match then no change is made to the original waveform and the conflicting normalized waveform group is not written to the output library.
- **Different values:** If the conflicting waveform has different values (one or more values differ by more than 1ps from the original) then the name of the conflicting waveform is changed to `original_waveform_#<num>` when `<num>` represents the number of the library being appended. For example, if the conflict occurs in the 2nd library being appended (3rd in the list passed to append library) then `<num>` will be 2). All references to the conflicting templates will be renamed to use the new waveform name. A message like this is output:

```
INFO (append_library): Conflicting definition of a driver waveform template found in library 'test.lib'. Changing all occurrences of 'ACTIVE-WAVEFORM_INV:rise' to 'ACTIVEWAVEFORM_INV:rise_#1' for the cells appended from this library.
```

`append_library` permits driver waveform lookup templates to have different index sizes and will output the first definition found, all others ignored.

The libraries that are not appended under these conditions are:

- **Other template conflicts:** If there are other conflicts (a lookup template that uses the same name but refers to a different number of index entries) the library with the conflicting template is not appended. A warning message is output.
- **Voltage map conflicts:** `append_library` checks `voltage_map` attributes and omits libraries that have voltage map attributes in conflict with the first definition.

The `-allow_conflict` option permits libraries with conflicts to be appended. However, resulting library *will have errors* that require fixing before being used by downstream tools.

Example

```
append_library -filename allCells.lib {a_Cells.lib b_Cells.lib} allCells
```

char_library

Performs library characterization. Each cell listed in a `define_cell` command is characterized if the SPICE subcircuit definition for that cell is defined in the netlists passed to the `read_spice` command. Only one `char_library` command is allowed per Liberate run.

In addition to pin capacitance and state dependent leakage, basic non-linear table lookup models for timing (including effective current source delay models (ECSM), constraints and power are characterized by default.

Options

`-auto_index`

Instructs Liberate to automatically create the indices for all constructs (except `si_immunity`) while overriding the values specified in the given templates. The number of entries for each index is taken for the appropriate pre-defined template. This feature uses the `max_transition` parameter to determine the range of output loads for each cell. To automatically generate `si_immunity` indices set the `max_noise_width` parameter. **Important:** The `auto_index` option utilizes the `inside_view` algorithm and cannot be used with the `-io` option (since the `-io` option disables the `inside_view`). See [Data Table Index Determination](#) for detailed information on determining data table index values.

Important

If your script includes the `define_index` command, it is applied after `auto_index` completes. If `define_index` is successful (correctly specified with cell/pin etc), it will override the index values determined by `auto_index`.

The `auto_index` option of `char_library` now requires `min_transition` and `min_output_cap` defined if packet mode is being used. If these are not defined, Liberate will generate an Error.

Note: In non-Packet Mode, `min_transition` and `min_output_cap` do not need to be defined, however, Liberate will output a Warning encouraging the user to set these variables.

Virtuoso Liberate Reference Manual

Liberate Commands

`-auto_max_capacitance`

Instructs Liberate for the explicit computation of the pin-based attribute *max_capacitance* using the same method as the `-auto_index` option. This option incurs the same run-time increase as if `char_library -auto_index` is enabled without modifying the actual `index_*` values. Only the *max_capacitance* related attributes are updated. Further, if both `auto_max_capacitance` and `auto_index` are enabled at the same time, then `auto_index` supersedes.

`-ccs`

Instructs Liberate to characterize composite current source (CCS) delay data.

`-ccsn`

Instructs Liberate to characterize composite current source noise (CCSN) data.

`-ccsp`

Instructs Liberate to characterize composite current source power (CCSP) data. This option is required when advanced power constructs are needed.

`-cells {cell_names}`

Lists the names of cells that need to be characterized. By default, Liberate characterizes all the cells defined by `define_cell` commands. When used with the `-exclude` option, Liberate excludes the cells specified in the list from characterization.

`-client_postproc "string"`

Instructs Liberate to execute the given shell command string after a client finishes running in "distributed" mode. For example, this can be used to free up external SPICE licenses as soon as a client finishes rather than waiting until all the clients finish.

`-ecsm`

Enables characterization of ECSM timing data. This option is on by default and is provided for script readability.

`-ecsmn`

Enables characterization of the effective current source model noise (ECSMN) data.

`-ecsmpl`

Enables characterization of the effective current source model power (ECSMP) data.

Note: The `-ecsmpl` and `-ccsp` options enable the same characterization. Therefore, only one of these two options is required.

Virtuoso Liberate Reference Manual

Liberate Commands

`-em`

Instructs Liberate to characterize electromigration models. This is a standalone characterization that cannot be combined with other data formats such as `-ccs`, `-ccsn`, `-ccsp`, `-ecsm`, `-ecsmn`, and `-ecsmpl`. For this feature, you need to use Spectre with APS (see the `-extsim` option of the [char_library](#) command and the variable [extsim_cmd_option](#)).

For more information, see the [Electromigration Models](#) section in the Liberate Details chapter.

`-exclude`

Excludes the cells specified with the `-cells` option from characterization.

`-extsim <simulator_name>`

Instructs Liberate to use the specified external SPICE simulator for characterization instead of Alspice. Alspice is the default built-in SPICE simulator. The license for the external simulator must be available. Currently, the following external simulators are supported: HSPICE, Spectre, and SKI.

Important

The simulator set using this `-extsim` option must correspond to the setting for the [extsim_cmd](#) variable.

When the `-extsim` option is specified, temporary run directories named `altos.<unique_id>.0`, `altos.<unique_id>.1`, and so on will be created to store the external simulation run-time files based on the thread number (0, 1, and so on). The `<unique_id>` is a unique ID based on the date, time, and the Liberate process ID.

If distributed processing is requested using the [set_client](#) command, these temporary run directories will be created in the directory specified by the `-dir` option of the [set_client](#) command.

If distributed processing is requested using the [packet_log_filename](#) parameter, the `TMPDIR` environment variable and the `tmpdir` parameter determine where the temporary simulation files will be stored. If the specified directory does not exist, Liberate tries to create it using the `mkdir -p` system command.

Virtuoso Liberate Reference Manual

Liberate Commands

`-extsim_leakage <simulator_name>`

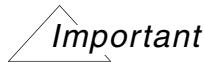
Instructs Liberate to use the specified external simulator for leakage calculation. This is used for leakage simulations only – all other simulations use the default simulator, Alspice. You can also set the variable, `extsim_model include leakage`, to specify a separate set of models for leakage calculations. If this is not set, Liberate uses the same set of models for all simulations.

`-ibis < 0 | 1 | 2 | typ | min | max >`

Enables IBIS model characterization for IO cells. The value tells Liberate which corner is to be characterized for IBIS format data.

`-io`

Enables characterization of cells without using "*Inside View*". This is useful for cells that contain analog or that are too large for digital vector analysis using *Inside View*. Using this option disables the automatic arc-determination and vector-generation of Liberate. For I/O cells, each of the arcs and associated logic conditions must be expressed explicitly using the `define_arc` and `define_leakage` commands.



In `io` mode, the `-probe` option is required for all constraint `define_arc` commands, including MPW.

`-server_thread <value>`

Specifies the number of threads (multithreaded CPUs) to be made available for the server to use. The extra threads are used to characterize the active driver (see `set_driver_cell`) waveforms. Default: use the same value as that of the `-thread` option.

Note: If the Server is submitted to a queuing system, the `-server_thread` value and the number of threads requested from the queuing systems should be the same positive integer value (`!=0`). Otherwise, the server can get overloaded.

`-si`

Instructs Liberate to create signal integrity data. At least a template of type `si_iv_curve` must be pre-defined using the `define_template` command. If a template exists for `si immunity`, the noise immunity-rejection curves will also be characterized.

Virtuoso Liberate Reference Manual

Liberate Commands

```
-skip {cin | constraint | delay | hold | hold_only | leakage | mpw  
| nochange | non_seq_hold | non_seq_setup | power | recovery |  
removal | setup | setup_only}
```

Instructs Liberate to skip characterization of the selected categories. Default: Do not skip any data types

Disables characterization of specific categories of data. A list containing multiple categories is supported. It is recommended that this option should be used only to improve run time while studying the characterization output from Liberate for a specific category. For example, while tuning the setup for constraint characterization, use `-skip {delay power leakage cin hold}` to speed up the run time for constraints.

Important

Skipping a category can have undesirable affects. For example, skipping `delay` and `power` will impact `cin` by reducing the number of vectors used to measure `cin`. We do **not** recommend skipping any categories while generating a production library since this can lead to an incomplete library. The value `setup` will skip `setup`, `recovery`, `non_seq_setup`. The value `hold` will skip `hold`, `removal`, `non_seq_hold`.

Virtuoso Liberate Reference Manual

Liberate Commands

`-skip_list {mycell {list_of_pins} {items_to_skip} <...> }`

Specifies a list of items to skip for a list of cells, or pin-wise for a list of cells.

If more granularity is needed, you can specify a list-of-items to skip on a list-of-pins on a list-of-cells. The skip list can be any item the `-skip` option supports. Examples:

To skip items on cells:

```
char_library -skip_list {mycell {items to skip} <...> }
```

To skip items on pins on cells:

```
char_library -skip_list {mycell {list of pins} {items to skip} <...> }
```

The `{list of pins}` supports * (asterisk) for a wildcard. Regular expressions such as `"*1"` are not supported at this time. Liberate automatically recognizes if the group after the cell is a list of skip items or a list of pins.

`-thread <number>`

Defines the maximum number of threads to use on the current machine. Even if the `-thread` option is not specified Liberate will automatically use multiple threads based on the available CPUs. Running on two or more threads will provide a significant reduction in characterization time.

`-trial`

Generates a dummy database that supports NLDM format library data.

This option instructs Liberate to run all of the preprocessing without running the actual simulations. The database will consist of NLDM format library data with proper structure but dummy data values. All other output formats are disabled including: `ccs`, `ccsn`, `ccsp`, `ecsm`, `ecsmn`, `ecsp`. When combined with a `write_ldb` and `write_library` command, this will result in a library file that is structurally valid. This library can be used with commands such as `write_template` and `write_verilog`.

`-user_arcs_only` Specifies to characterize only the user-specified arcs. The "inside view" of Liberate skips the automatic addition of arcs. To use this option, you must provide all the required arcs using the `define_arc` command. This option is used to ensure that the `write_template` verbose flow matches the reference library structure more closely.

The `char_library` command should never be called after the API is initialized using `ALAPI_init`. All commands that create models call `ALAPI_init`.

Examples

```
# Characterize for CCS and SI
char_library -ccs -si

# Use Spectre for characterization of ECSM and CCSN and CCSP
char_library -ecsm -ccsn -ccsp -extsim Spectre

# Only characterize DFFX1 and INVX1
char_library -cells {INVX1 DFFX1}

# Free up the Hspice license once each client exits
char_library -client_postproc "hspice -C -K"
```

check_delay_monotonicity

Checks `cell_rise` and `cell_fall` delay data to ensure that all the delay entries are monotonically increasing with respect to output load (index_2). That is, the table is checked for monotonicity for all loads for each slew. The checks are performed as the library is being written out using `write_library`.

Options

| | |
|-----------------------------|--|
| <code>-adjust</code> | Amount to adjust the delay or transition by when fixing non-monotonic data. Default: 0.001ps |
| <code>-ecsm</code> | Check ECSM waveform data. |
| <code>-exit</code> | Exit if non-monotonic delay data (by load) is found. This option causes <code>write_library</code> to exit if an error is found such that the library is incomplete. The warnings or errors are written to the screen and indicate the bad table entry, the values involved and the arc type including the <i>when</i> condition. |
| <code>-fix</code> | Fix the non-monotonic NLDM data. Use this option to repair any delay and rise/fall transition monotonicity problems (with respect to the output load) by making the non-monotonic table entry equal to the previous entry plus one added to the least significant digit. When non-monotonic data is fixed, it may become out of sync with ECSM and CCS waveform data since only the NLDM data gets fixed. This is especially true when the <code>-slew</code> option is used to fix non-monotonic data by slew. |
| | Note: The <code>-exit</code> option overrides the <code>-fix</code> option. |
| <code>-fix_ccs_delay</code> | Fix any monotonicity problem caused by output load and/or transition for CCS delay. This option controls whether the CCS delay data should be adjusted. When the library contains both NLDM and CCS timing data, use this option in combination with the <code>-fix</code> option. This can help to avoid mismatches between NLDM and CCS timing. |
| <code>-ocv</code> | Checks for OCV delay sensitivity if only the <code>-ocv</code> option is specified. If the <code>-transition</code> option is specified along with the <code>-ocv</code> option, the transition sensitivity is also checked. |

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|-------------|---|
| -slew | Check data monotonicity based on the changing input slew. This means that for a given load, the data corresponding to consecutive all slews are checked. While delay and transition usually slows down with increasing load for a given slew, it is not necessarily correct that there is a correlation between increasing slew and monotonic increasing data for a given load. |
| -transition | Check the rise and fall transitions data. |

This command can be used after char_library, read_ldb, and read_library. It should be used before model generation such as with write_library.

Example

```
read_ldb my.ldb
check_delay_monotonicity -ecsm -transition -fix
write_library my.lib
```

Warnings and errors will look like:

```
*Warning* (write_library): Non-monotonic (by load) rise_transition values: (3, 4) 0.35 < 0.37 for DFFX1:CLK->Q
*Error* (write_library): Non-monotonic (by load) cell_fall values: (2, 5) 0.254 < 0.257 for DFFX1:CLK->Q
```

check_lvf_merge_indices

Checks the indexes in the Liberate LDB against the indexes in the sensitivity file for each arc for which both sets of data exist. When merging LVF data into the output library (see [write_library -sensitivity_file](#)), the indexes must match between the NLDM and LVF data. If the relative and absolute tolerances are exceeded, a message is output. In addition, if the `-action` is set to `error`, Liberate exits without writing out the library.

Options

`-action <"warn" | "error">`

Specifies the action to take when the sensitivity LVF and NLDM indexes do not match within the specified tolerance.

Default: "warn"

`-slew_abstol <float>`

Specifies the slew (`index_1`) absolute tolerance applied when comparing the index values while merging LVF into the output library. Default: `1e-12` (in seconds)

`-load_abstol <float>`

Specifies the load (`index_2`) absolute tolerance applied when comparing the index values while merging LVF into the output library. Default: `1e-15` (in farads)

`-slew_reltol <float>`

Specifies the slew (`index_1`) relative tolerance applied when comparing the index values while merging LVF into the output library. Default: `0.01` (1%)

`-load_reltol <float>`

Specifies the load (`index_2`) relative tolerance applied when comparing the index values while merging LVF into the output library. Default: `0.01` (1%)

This command must be used before the [write_library](#) command is run.

compare_ccs_nldm

Compares the CCS data to the NLDM data in a single library and reports the differences that exceed the defined tolerances.

Options

- absolute_average Report absolute average. Default: *report relative average*
- abstol <value> Sets the absolute tolerance limit for the comparison between CCS and NDLM errors. Default: $0.002 \times \text{time_unit}$ (typically 2ns)

Note: The `-abstol` and `-reltol` options define absolute and relative tolerance limits for each comparison. Any comparison that exceeds both these tolerances is considered an outlier and will be reported.
- cells {*cell_names*} Specifies a list of cells to compare. Default: *all cells*

Note: This option supports the use of a wildcard. If the `-exclude` option is used, the list of cells will be excluded from the comparison.
- exclude Exclude the list of cells from the comparison.
- format < txt | xls | htm > Specifies the format for the output report, that is, text, Microsoft Excel®, or HTML. Default: `txt`

The default (`txt`) is a standard textual format. The `xls` value can be used to get an output format that is more suitable for import into Microsoft Excel. The `htm` value can be used to request an HTML output format. The default directory name is `"./html"` and can be changed using the `-group` option. A one page comparison will be generated for each cell group. Open the file `index.html` in a web browser to view the report.

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|--------------------|--|
| -group <dirname> | Specifies the directory name to store cell comparisons for each cell group. Default: <i>all cells in a single report</i> |
| | The <code>-group</code> option requests a group by group comparison, storing the results in the given directory name. A cell group is determined by the define_group command or by the <i>footprint</i> attribute. The comparison report for each group is stored in the file <code><dir_name>/<group_name>.cmp.txt</code> |
| -gui <filename> | Defines the name of an intermediate file that can be used for graphical comparisons of data with the <code>lcplot</code> utility. |
| -lcplot | Starts the <code>lcplot</code> utility for viewing the comparison results graphically. When using this option, the <code>-gui</code> option is not required because a comparison data file called <code><library_lib>.gui</code> will be created automatically. |
| -nworst <number> | List the number of top <i>nworst</i> cells with outliers per data type. Default: 5 |
| | This option defines the number of failing cells to report for each data type in the report summary. For the top <i>nworst</i> cells, the worst absolute and relative outlier is reported. |
| -percent_max_diff | Report the percent error of the max difference. Default: <i>report the maximum percent difference</i> |
| -reltol <value> | Percentage tolerance for the limit for the comparison between CCS and NDLM errors. Default: 0.02 |
| | Note: The <code>-abstol</code> and <code>-reltol</code> options define absolute and relative tolerance limits for each comparison. Any comparison that exceeds both these tolerances is considered an outlier and will be reported. |
| -report <filename> | Output comparison file name. Default: <code><library_name>.cmp.txt</code> An overall comparison summary is also written to the standard output. |
| -verbose | Report comparison of all data in the library, regardless of tolerances. The generated report shows every comparison including those that did not exceed a tolerance. |
| <library_name> | Library name. |

This command should be run by itself in a separate Liberate run.

Example

```
# Set relative tolerance to 1%, delay tolerance to 1ps
compare_ccs_nldm -reltol 0.01 -abstol "delay 1e-12" comp.lib
```

compare_library

Compares data found in the reference library (`ref_lib`) to the matching data found in the compare library (`comp_lib`) and reports the differences that exceed the defined tolerances. The report includes the comparison of attributes, capacitance, leakage, delay, transition, power, timing constraints, and comparison of advanced model data such as ECSM, CCS, Electromigration (EM), Liberty Variation Format (LVF), and Normalized Driver Waveform (NDW). For CCS, the current waveforms are converted to voltage waveforms and the comparisons performed using delay and slew thresholds rather than for each current measurement. If the table indices in the comparison library are different from the reference library, bi-linear interpolation will be used prior to performing the comparison. For CCSN, the following data types are supported: `ccsn_dc`, `ccsn_vout` and `miller_cap` (propagation tables are not yet implemented). For `ccsn_dc` and `ecsm`, 5 points of the dc current data are compared: the first point, the last point and 3 intermediate points.

The output will report when reference and comparison values are zero (including `cap`, `max_tran`, `max_cap` etc.). If the reference value is zero and the comparison value is not zero then the percent difference is reported using a question mark ("?") This data point is not included in the computation of the overall average but it is counted as an outlier.

Options

`-absolute_average` Report absolute average. Default: *report relative average*

`-abstol <value | {list}>`

Specifies the absolute difference tolerance. The `abstol` value should be given in standard units and instead of library units, that is, "delay 5e-12" to set the `abstol` for delay to 5ps.

Default: $1e-3 \times \text{data_type_unit}$ (0.001 times the default unit for each data type)

For example, if the `time_unit` is in nS, the `abstol` for delay will default to 0.001nS or 1ps.

Note: The `-abstol` and `-reltol` options define absolute and relative tolerance limits for each comparison. These options accept a single value or a paired list of type and value. Any comparison that exceeds both these tolerances is considered an outlier and will be reported. Individual tolerances can be set for each different data type by assigning values to the following compare types:

`all, cap, ccs, ccs_cap, ccsn_dc, ccsn_vout, constraint, delay, ecssm, ecssm_cap, hyper, leakage, max_cap, max_trans, miller_cap, noise, power, siv, trans, timing, capacitance, voltage, current`

If the option has only a single value, the type for that value is assumed to be `all`.

`-cells {cell_names}` Specifies a list of cells to compare. Default: `all cells`

Note: This option supports the use of a wildcard. If the `-exclude` option is used, the specified list of cells will be excluded from the comparison.

`-cellmap {list}` Specifies a list of pairs of `ref_lib` and `comp_lib` cells to compare. Default: Compare all matching cell names.

The new option is used to control how `compare_library` chooses which cells to compare. The rules for priorities pertaining to cell mapping are as follows.

- If both `-cells` and `-cellmap` options are specified, then each cell in the `-cells` list is compared to the `cellmap` list. If a cell in the `-cells` list maps to a valid pair in the `cellmap` then the mapped reference cell and comparison cell is compared. One-to-many and many-to-one mapping is allowed and comparison is done for all valid combinations. If a cell in the `-cells` list is not present in the `cellmap`, then the comparison is done for the same cell in both libraries.
- If only `-cellmap` is provided but not `-cells`, then each valid cell pair from the `cellmap` is compared.
- If only `-cells` is provided but not `-cellmap`, then all cells in the `-cells` list that exist in both libraries are compared.
- If neither `-cells` and `-cellmap` are provided, then all the cells that are present in both the reference and comparison libraries are compared.

Virtuoso Liberate Reference Manual

Liberate Commands

`-comp_adjust_tristate_load <value>`

Controls if pin capacitance should be added to the load indices on tristate pins and adjusts the tristate load of comparison library before comparing. Default: -1

Accepted values are:

- | | |
|----|---|
| -1 | Follow setting of the <u>adjust_tristate_load</u> variable. |
| 0 | Do not adjust the load indices. |
| 1 | Add the <code>rise_capacitance</code> and <code>fall_capacitance</code> of the tristate pin to the associated load indices. |
| 2 | The same as 1 except that the pin attribute capacitance will be added to the load indices. |
| 21 | The same as 1 except that only timing arc loads will be adjusted (not power arc loads.) |
| 22 | The same as 2 except that only timing arc loads will be adjusted (not power arc loads.) |

`-exact_match` Only compare arcs with identical logic ('when') conditions.

`-exclude` Exclude the list of cells from the comparison.

`-format < txt | xls | htm >`

Specifies the format for the output report, that is, text, Microsoft Excel®, or HTML. Default: `txt`

The default (`txt`) is a standard textual format. The `xls` value can be used to get an output format that is more suitable for import into Microsoft Excel. The `htm` value can be used to request an HTML output format. The default directory name is `"./html"` and can be changed using the `-group` option. A one page comparison will be generated for each cell group. Open the file `index.html` in a web browser to view the report.

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|-----------------------|--|
| -group <dirname> | Specifies the directory name to store cell comparisons for each cell group. Default: <i>all cells in a single report</i> |
| | The <code>-group</code> option requests a group by group comparison, storing the results in the given directory name. A cell group is determined by the define_group command or by the <i>footprint</i> attribute. The comparison report for each group is stored in the file <code><dir_name>/<group_name>.cmp.txt</code> |
| -gui <filename> | Defines the name of an intermediate file that can be used for graphical comparisons of data with the <code>lcplot</code> utility. |
| -index1_range <range> | <i>index1</i> range to use for comparison. Default: <i>use all indices</i> |
| -index2_range <range> | <i>index2</i> range to use for comparison. Default: <i>use all indices</i> |
| | The <code>-index1_range</code> and <code>-index2_range</code> options can be used to limit the comparison to a range of <i>index1</i> or <i>index2</i> values. The range is either two values separated by a <code>"-"</code> (for example, <code>"1-3"</code> to compare the first three indices) or a single value (For example, <code>"2"</code> to compare the second index only). |
| -lcplot | Starts the <code>lcplot</code> utility for viewing the comparison results graphically. When using this option, the <code>-gui</code> option is not required because a comparison data file called <code><comp.lib>.gui</code> will be created automatically. |
| -lib <abs rel> | Request a Liberty formatted report with absolute or relative data differences. |
| | The <code>-lib</code> option can be used to request an output report formatted like the <code>comp.lib</code> , where the values in the data table represent the absolute or relative differences between the two libraries. The output report will be named <code><comp.lib>_<abs / rel>.cmp</code> . |
| -multiple_matches | Report comparison of all arcs that have functional overlap with a reference arc |
| -nldm_only | Requests comparison of only the NLDM data. Comparison of the following data is ignored: |
| | <ul style="list-style-type: none">■ CCS and ECSV timing■ Noise and power constructs |

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|-------------------------------------|--|
| <code>-no_interpolation</code> | Disables the comparison of data groups that have mismatched numbers of indices. No interpolation between index points will be performed. By default, if the number of index values is different, the comparison values will be interpolated. |
| <code>-nworst <number></code> | List the number of top <i>nworst</i> cells with outliers per data type. Default: 5 This option defines the number of failing cells to report for each data type in the report summary. For the top <i>nworst</i> cells, the worst absolute and relative outlier is reported. |
| <code>-ocv_include_nominal</code> | Compares the (nominal+sigma) and (nominal-sigma) values. Default: compare <code>ocv_sigma</code> values. The LVF (Liberty Variation Format) <code>ocv_sigma_*</code> table values can be very small. Comparing these values directly can lead to a significant number of outliers. Use this option to include the nominal delay in the comparison. This will reduce the number of outliers. |
| <code>-padding</code> | Pad delay, transitions and constraints by $\frac{1}{2}$ input slew. Pad power by an additional $\frac{1}{2}CV^2$. The <code>-padding</code> option can be useful when comparing very small or even negative delay values. The reference and comparison delay, transition, and constraint data is padded by a $\frac{1}{2}$ input slew before comparison. In addition, the power values are now padded by an additional $\frac{1}{2}CV^2$ (where C =output capacitance, V =Vdd for that pin) to the power numbers before performing the comparison. This will not apply to hidden power because the output is not toggling. |
| <code>-padding_index</code> | Specify which slew index to use when adding padding. Use one of the following values to specify this: <code>same</code> , <code>mid</code> , and <code>end</code> . The default is to use the <code>same</code> slew index as for delay. |
| <code>-percent_max_diff</code> | Report the percent error of the max difference. Default: <i>report max percent difference</i> |

Virtuoso Liberate Reference Manual

Liberate Commands

`-ref_adjust_tristate_load <value>`

Controls if pin capacitance should be added to the load indices on tristate pins and adjusts the tristate load of reference library before comparing. Default: -1

Accepted values are:

- | | |
|----|---|
| -1 | Follow setting of the <u>adjust_tristate_load</u> variable. |
| 0 | Do not adjust the load indices. |
| 1 | Add the <code>rise_capacitance</code> and <code>fall_capacitance</code> of the tristate pin to the associated load indices. |
| 2 | The same as 1 except that the pin attribute capacitance will be added to the load indices. |
| 21 | The same as 1 except that only timing arc loads will be adjusted (not power arc loads.) |
| 22 | The same as 2 except that only timing arc loads will be adjusted (not power arc loads.) |

`-reltol <value | {list}>`

Specifies the relative tolerance limit for each comparison.
Default: 0.01 (1%)

For more details, see the explanation of the `-abstol` option above.

`-report <filename>` Output report file name. Default: `<comp_lib>.cmp.txt`

An overall comparison summary is also written to the standard output.

Virtuoso Liberate Reference Manual

Liberate Commands

-skip {list}

Specifies a list of data comparison types to skip. Default: *none* (do not skip any comparison types)

Valid comparison types are:

```
attributes, cap, ccs, ccs_cap, ccs_retain, ccsn_dc,
ccsn_prop, ccsn_vout, ccsp, ccsp_cap, ccsp_dc,
ccsp_lc, ccsp_res, clear, delay, delay_variation,
ecsm, ecsm_cap, ecsm_cap_variation, ecsm_variation,
em, em_maxcap, hidden_power, hold, hyper, in_cap,
leakage, max_cap, max_trans, miller_cap, mpw, noise,
nonseq_hold, nonseq_setup, power, preset, recovery,
removal, retain, retain_slw, setup, siv, three_state,
three_state_disable, three_state_enable, time_const,
trans, trans_variation, tristate
```

In addition, a small collection of "macro-types" are available. These are convenient groupings of some of the basic types:

```
all      (this is the default)
capacitance = {cap ccs_cap ccs_retain ecsm_cap
ecsm_cap_variation in_cap max_cap miller_cap}
constraint = {setup hold recovery removal mpw
nonseq_setup nonseq_hold}
current = {ccs ccsn_dc ccsp siv}
timing = {delay delay_variation ecsm ecsm_variation
max_trans time_const retain retain_slw trans
trans_variation}
voltage = {hyper noise ccsn_vout}
```

The **-skip** and **-type** options separate dynamic power from hidden power. For example, specifying **-skip {power}** skips the dynamic power, while specifying **-skip {hidden_power}** skips the hidden power, and **-skip {power hidden_power}** skips both.

-type {list}

Specifies a list of data comparison types to include. Default: *all*

-unmatched

Report unmatched data entries.

-upscale

When the data for a particular arc have different data dimensions in two different libraries (for example, 7x1 vs. 7x7), the data dimension of the smaller table is scaled up to match the data dimension of the larger table.

-report <filename>

Output comparison file name. Default: *<library_name>.cmp.txt*

An overall comparison summary is also written to the standard output.

Virtuoso Liberate Reference Manual

Liberate Commands

`-verbose`

Report all comparisons regardless of tolerances.

The generated report shows every comparison including those that did not exceed a tolerance.

`<ref_lib>`

Reference library.

`<comp_lib>`

Comparison library.

When comparing libraries, the data entries must have equivalent conditions. Two entries are deemed as equivalent if they have the same or overlapping logic conditions, related pins, and data type. In some instances not all the data in the reference library will have an equivalent in the comparison library. To report these entries use the `-unmatched` option. To compare entries only when there is an exact match in the *when* conditions, use the `-exact_match` option. If comparing libraries with different cell names, use the `define_map` command to map the names in the comparison library to the reference library. Note that all the pin names must match. Specify the `-multiple_matches` option to report comparison of all arcs that have functional overlap with a reference arc. The default is to report the table that gives the best match. Multiple arcs will be shown in the output file as (N of M) after the "when :" line. For example:

```
| when : !M1 Vs (! (M1) * !(M2)) (1 of 2), Timing : combinational
```

Note: The `-exact_match` option overrides the `-multiple_match` option.

When comparing two libraries that have different index values, slew thresholds and units, the values in `comp_lib` will be scaled accordingly before comparison. The following characters are used to indicate that some form of data manipulation has occurred before the comparison:

- * : scaling due to slew thresholds or units
- ^ : input slews extrapolated
- ~ : output loads extrapolated
- ! : the indices were switched
- + : both the `ref_lib` and `comp_lib` values were padded.

When comparing libraries that have different *when* conditions, the data groups that have overlapping conditions will be compared. If the number of indices (dimensions) differs between two data groups then the data in the smaller dimension table is expanded to fit the larger dimension table. For example, if comparing delay data based only on input slew versus delay data based on slew and load, the 1-D slew table will be expanded to a 2-D slew/load table by using the first value of the load indices from the 2-D table.

When the reference and comparison library values are 0 (including for `cap max_tran`, `max_cap` etc.) a report will be generated. If the reference value is zero and the comparison

Virtuoso Liberate Reference Manual

Liberate Commands

value is non-zero, then the percent difference is reported as a "/0". This point is not included in the overall average equation but it is counted as an outlier.

Example

```
# Set all relative tolerances to 2%, constraint tolerance
# to 3%, power tolerance to 5%. Set absolute tolerance
# values for constraint, transition, leakage, and power
compare_library \
    -reltol { all 0.02 constraint 0.03 power 0.05 } \
    -abstol { constraint 5e-12
               trans 5.0e-12
               leakage 2.e-15
               power 3e-15 } \
    ref.lib \
    comp.lib
```

Sample Output Report

Legend : < outlier, * scaled, ! indices switched, ^ slews extrapolated, ~ loads extrapolated, + padding added, /0 divide by zero Legend : / slews interpolated, # loads interpolated

```
*** BEGIN INVX1 COMPARISON ***

INVX1 Delay Comparison in ns
+-----+-----+-----+-----+-----+-----+
--+-----+-----+-----+
| Row # |           Pin Name | Ref Value | Comp Value |      Diff |      Diff
% |      Type | Index_1 | Index_2 |           |
+-----+-----+-----+-----+-----+-----+
--+-----+-----+-----+
| 1 |           INVX1:A->ON FR | 0.181790 | 0.171756 | -0.010034 | -
5.52% | delay | 0.304 | 0.058 |
+-----+-----+-----+
| 2 |           INVX1:A->ON FR | 0.239880 | 0.227162 | -0.012718 | -
5.30% | delay | 0.612 | 0.058 |
+-----+-----+-----+
| 3 |           INVX1:A->ON RF | 0.149020 | 0.138183 | -0.010837 | -
7.27% | delay | 0.612 | 0.058 |
+-----+-----+-----+
--+-----+-----+-----+
```

Virtuoso Liberate Reference Manual

Liberate Commands

INVX1 Delay SUMMARY

| Diff | Data Type | Entries | Avg Diff | Avg Diff% | Sigma% | Max |
|---|-----------|----------|----------|-----------|--------|-----|
| Diff | Max Diff% | Outliers | | | | |
| 0.01272 | delay(ns) | 98 | -0.00166 | -2.30% | 4.27% | - |
| | | -7.27% | 3 | | | |
| Worst delay outlier: Max Abs: -0.01272, Row # : 2; Max Rel: -7.27%, Row # : 3 | | | | | | |

INVX1 Transition Comparison in ns

| % | Row # | Pin Name | Ref Value | Comp Value | Diff | Diff |
|-------|-------|----------------|-----------|------------|-----------|------|
| % | Row # | Type | Index_1 | Index_2 | | |
| 8.15% | 1 | INVX1:A->ON FR | 0.219420 | 0.201528 | -0.017892 | - |
| | 2 | INVX1:A->ON FR | 0.219220 | 0.201360 | -0.017860 | - |
| 8.15% | 3 | INVX1:A->ON FR | 0.219550 | 0.201632 | -0.017918 | - |
| 8.16% | 4 | INVX1:A->ON FR | 0.219330 | 0.201455 | -0.017875 | - |
| | 5 | INVX1:A->ON FR | 0.219460 | 0.202950 | -0.016510 | - |
| 7.52% | 6 | INVX1:A->ON FR | 0.238460 | 0.225337 | -0.013123 | - |
| 5.50% | 7 | INVX1:A->ON FR | 0.316770 | 0.301474 | -0.015296 | - |
| 4.83% | | | 0.058 | | | |

INVX1 Transition SUMMARY

| Diff | Data Type | Entries | Avg Diff | Avg Diff% | Sigma% | Max |
|---------|-----------|----------|----------|-----------|--------|-----|
| Diff | Max Diff% | Outliers | | | | |
| 0.01792 | trans(ns) | 98 | -0.00242 | -2.1% | 2.94% | - |
| | | -8.16% | 7 | | | |

Virtuoso Liberate Reference Manual

Liberate Commands

```
+-----+-----+-----+-----+-----+  
-----+-----+-----+  
Worst trans outlier: Max Abs: -0.01792, Row # : 3; Max Rel: -8.16%,  
Row # : 3
```

*** END INVX1 COMPARISON ***

Overall LIBRARY SUMMARY

| Diff | Data Type | Entries | Avg Diff | Avg Diff% | Sigma% | Max |
|------|-------------|----------|----------|-----------|--------|---------|
| | Max Diff% | Outliers | | | | |
| | leakage(nW) | 2 | 0.00000 | 0.00% | 0.00% | 0.00000 |
| | 0.00% | 0 | | | | |

| Diff | Data Type | Entries | Avg Diff | Avg Diff% | Sigma% | Max |
|------|-----------|----------|----------|-----------|--------|---------|
| | Max Diff% | Outliers | | | | |
| | cap(pf) | 2 | 0.00000 | 0.00% | 0.00% | 0.00000 |
| | 0.00% | 0 | | | | |

| Diff | Data Type | Entries | Avg Diff | Avg Diff% | Sigma% | Max |
|---------|-----------|----------|----------|-----------|--------|-----|
| | Max Diff% | Outliers | | | | |
| 0.01272 | delay(ns) | 98 | -0.00166 | -2.30% | 4.27% | - |
| | -7.27% | 3 | | | | |

Worst delay outlier (one per cell):

```
+-----+-----+-----+-----+-----+  
-----+-----+-----+  
-----+-----+-----+-----+-----+
```

Virtuoso Liberate Reference Manual

Liberate Commands

| # | Row # | Cell | Max Diff | Row # | Cell | Max Diff% | |
|---------|-----------|-----------|----------|----------|-----------|-----------|-----|
| 7.27% | 3 | INVX1 | -0.01272 | 2 | INVX1 | - | |
| Diff | Max Diff% | Data Type | Entries | Avg Diff | Avg Diff% | Sigma% | Max |
| 0.01792 | -8.16% | trans(ns) | 98 | -0.00242 | -2.1% | 2.94% | - |

Worst trans outlier (one per cell):

| # | Row # | Cell | Max Diff | Row # | Cell | Max Diff% | |
|---------|-----------|-----------|----------|----------|-----------|-----------|---------|
| 8.16% | 3 | INVX1 | -0.01792 | 3 | INVX1 | - | |
| Diff | Max Diff% | Data Type | Entries | Avg Diff | Avg Diff% | Sigma% | Max |
| 0.00% | 0 | power(pJ) | 98 | 0.00003 | 1.99% | 6.54% | 0.00000 |
| Entries | Avg Diff% | | | | | | |
| | | Sigma% | Outliers | | | | |

Virtuoso Liberate Reference Manual

Liberate Commands

| | | | | | | | | |
|---|--------|--------|--------|--------|--------|--------|--------|--------|
| | 298 | | -0.82% | | 5.18% | | 10 | |
| + | -----+ | -----+ | -----+ | -----+ | -----+ | -----+ | -----+ | -----+ |

```
*** LIBRARY Comparison of comp.lib to ref.lib completed on Wed May 31 14:39:41 PDT
2006
```

Explanation of the Output Report Terminology

- Ref Library: The first library listed in the compare_library command comp Library: The second library listed at the end of the compare_library command.
- Library and Cell Attribute Comparison groups:
 - Level: The level in the library where the attribute was found, either library level or particular to a cell.
 - Attribute: The name of the attribute.
 - Ref Value: The attribute value from the reference library Comp Value: The attribute value from the comparison library
- Data Comparison Table
 - Row #: The row index number.
 - Pin Name: This field identifies the arc being compared. It can contain a pin, related_pin and their directions. If a single pin is listed then this is a hidden power or an MPW arc.
 - Ref Value: The data value as extracted from the ref (reference) library.
 - Comp Value: The data value as extracted from the comp (comparison) library
 - Diff: Ref_Value - Comp_Value
 - Diff %: Diff / Ref_Value
 - Type: The type of the data. Example: leakage, delay, rising, falling and power.
 - Index_1: The value of the index_1 in the Ref Library
 - Index_2: The value of the index_2 in the Ref Library
- Note: If the index_1 and index_2 values are not exactly matching, then the right side of the row will show a ^, ~, / or # to indicate either extrapolation or interpolation occurred. See the Legend at the top of the report for a detailed description.
- Data Comparison Summary:

Virtuoso Liberate Reference Manual

Liberate Commands

- Data Type: The type of the data.
- Entries: The number of entries compared.
- Avg Diff: The sum of the Diff values in the table divided by the number of entries in the table.
- Avg Diff%: The sum of the Diff% values in the table divided by the number of entries in the table.
- Sigma %: $\text{Sqrt}(\text{Abs}((\text{Sum of Diff\%}^2) / \# \text{Entries}) - (\text{Avg Diff\%}^2)))$
- Max Diff: The largest Diff value
- Max Diff%: The Diff% related to the Max Diff.
- Outliers: This is the total number of outliers. An outlier is a data comparison where both diff > abstol and diff% > reltol for the data type. Each outlier in the report is indicated by a "<" character at the end of the row.

Sample Summary Report

Overall LIBRARY SUMMARY

| Diff | Data Type | Entries | Avg Diff | Avg Diff% | Sigma% | Max |
|---------|----------------|----------|----------|-----------|--------|---------|
| | Max Diff% | Outliers | | | | |
| | leakage(nW) | 2 | 0.00000 | 0.00% | 0.00% | 0.00000 |
| | 0.00% | 0 | | | | |
| | cap(pf) | 2 | 0.00000 | 0.00% | 0.00% | 0.00000 |
| | 0.00% | 0 | | | | |
| 0.01272 | delay(ns) | 98 | -0.00166 | -2.30% | 4.27% | -7.27% |
| | | 3 | | | | |
| 0.01792 | trans(ns) | 98 | -0.00242 | -2.18% | 2.94% | -8.16% |
| | | 7 | | | | |
| | constraint(ns) | 0 | 0.00000 | 0.00% | 0.00% | 0.00000 |
| | 0.00% | 0 | | | | |

Virtuoso Liberate Reference Manual

Liberate Commands

| | | | | | |
|------------|-----------|---------|----------|-------|---------|
| power (pJ) | 98 | 0.00003 | 1.99% | 6.54% | 0.00000 |
| 0.00% | 0 | | | | |
| <hr/> | | | | | |
| Entries | Avg Diff% | Sigma% | Outliers | | |
| 298 | -0.82% | 5.18% | 10 | | |
| <hr/> | | | | | |

Explanation of the Summary Report Terminology:

- Entries: The total number of entries that were compared.
- Avg Diff%: See above
- Sigma % See above
- Pass%: (#Entries - Outliers) / #Entries
- Outliers: See Above

copy_arc

Copies the data from one arc to another arc. This command is executed during model generation (see [write_library](#)) and affects the output library. Therefore, this command must be issued prior to model generation with [write_library](#).

Options

- from_cell <name>** Specifies the cell name to copy data from.
- from_dir <name>** Specifies the data direction to copy from. For example, "rise" for `cell_rise` and "fall" for `cell_fall`. Default: " " (copy both rise and fall data).
- from_library <name>** Specifies the library name to copy the arc from. Default: the first library read.
- from_pin <name>** Specifies the pin name in the specified `-from_cell` to copy data from.
- from_related <name>** Specifies the related_pin timing to copy data from under the specified `-from_cell` and `-from_pin`. Default: "*" (copy arcs regardless of related pin)
- from_related_pg <name>** Specifies the related power/ground pin to copy data from. Default: "*" (copy arcs regardless of related power/ground pin)
- from_timing_sense <value>** Specifies the timing sense to copy data from. (Example: `positive_unate`)
Default: " " (Copy regardless of timing sense)
- from_timing_types {list}** Specifies a list of timing types to copy data from. Default: {} (copy regardless of timing type)
- from_subgroup_type <name>** Specifies the name of a subgroup type to copy data from. For example: "cell_rise" or "fall_transition" for `-type delay`. Default: " " (copy all sub-groups).

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|--|--|
| -from_when <value> | Specifies the when condition of the arc to copy data from. Default: "*" (copy all arcs regardless of the when condition) |
| -margin {list} | Specifies the absolute margin (in Seconds) to add to the copied data. You can use one value for all subgroups or a list of <subgroup> <value> pairs. Default: 0.0 (no margin) |
| | A single value will be applied to all tables in a timing group. If multiple values are specified there must be one value for each table in the timing group for the "from" arc. (Example: for a delay table, specify margin to add in this order: rise_delay, fall_delay, rise_transition, fall_transition.) |
| -method < replace append augment > | Specifies the copy method to use. The supported methods are: "replace", "append", and "augment". Default: "replace" (replace the existing matching arc) |
| -rel_margin {list} | Specifies the relative margin to add to the copied data. You can use one value for all subgroups or a list of <subgroup> <value> pairs. For example, {rise_constraint 0.1 fall_constraint 0.2} or use 0.05 to represent a 5% relative margin. Default: 0.0 (0%) With the -rel_margin option, you can specify a single value or a list of values (similar to margin.) |
| -swap_direction | Specifies to swap rise data with fall data and conversely after copying. |
| -to_cell <name> | Specifies the cell name to copy the data to. Default: see -from_cell |

Virtuoso Liberate Reference Manual

Liberate Commands

-to_pin <name> Specifies the pin name in the **-to_cell** to copy the data to. This option accepts a wildcard. Default: see **-from_pin**

For example, the following command enables a *one-to-many* *copy* operation by copying the relevant data selected using the **-pin**, **-from_related**, and so on options to all pins that match the wildcard name:

```
copy_arc \
  -type {delay} \
  -method replace \
  -from_cell $cell \
  -from_pin SLEEPOUTN \
  -from_related SLEEPSN \
  -from_timing_sense positive_unate \
  -to_cell $cell \
  -to_pin SLEEPSN \
  -to_pin *_altos* \
  -to_timing_sense negative_unate
```

-to_related <name> Specifies the **related_pin** to copy the data to. Default: see **-from_related**

-to_related_pg <name>

Specifies the related power/ground pin to copy the data to. Default: see **-from_related_pg**

-to_subgroup_type <name>

Specifies the name of subgroup type to copy the data to. For example: "cell_rise" or "fall_transition" for **-type delay**. Default: "" (see **-from_subgroup_type**)

-to_sdf_cond <string>

Specifies the **sdf_cond** string to use after copying the arc. Default: Create the **sdf_cond** from the **-to_when** condition.

-to_timing_sense <value>

Specifies the timing sense to copy the data to. Default: " " (see **-from_timing_sense**)

-to_timing_types {list}

Specifies the timing type(s) to copy the data to. Default: {} (see **-from_timing_types**)

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|-------------------------------------|---|
| <code>-to_when <value></code> | Specifies the <code>when</code> condition to use after copying the arc. Default: see <code>-from_when</code> |
| <code>-type {list}</code> | Specifies the type of data to be copied. Default: copy all data types. The supported types are: <code>cap</code> , <code>ccs</code> , <code>ccsn</code> , <code>ccs_retain</code> , <code>constraint</code> , <code>delay</code> , <code>hold</code> , <code>leakage</code> , <code>mpw</code> , <code>power</code> , <code>recovery</code> , <code>removal</code> , <code>retain</code> , <code>setup</code> , and <code>timing</code> . The <code>cap</code> value allows copying all the capacitance related data from one pin to another. This includes capacitance attributes such as <code>capacitance</code> , <code>rise_capacitance</code> , <code>fall_capacitance</code> as well as any CCS or ECSV receiver capacitance. For example: <pre>copy_arc \ -type cap \ -from_pin "my_D" -to_pin "your_D\"</pre> |

Note: The `copy_arc` command does not currently support `write_socv`.

define_arc

Specifies a user-defined arc to override automatic arc determination by Liberate. An "arc" represents library data between a given pin and a related pin. Typically this command is only required for the characterization of complex I/O cells. When used without the `-ignore` option, the pin directions must be specified using a vector or the appropriate `define_arc` options such as `-pin_dir` and `-related_pin_dir`.

Options

- `-attribute {list}` List of user-defined attribute/value pairs. These attribute and values pairs are added to the .lib for the specific arc.
- `-ccsn_stage <arc | pin>`
Specifies the CCSN data type. Default: `pin`
- `-ccsn_probe <node>` Specifies the CCSN internal probe node name used as the ccsn stage input/output.
- `-delay_threshold {in_rise in_fall out_rise out_fall}`
Defines a list of four delay percentage measurement points (a ratio of VDD normalized to between 0 and 1) for the arc. For the `-delay_threshold` option, these values represent the following measurement thresholds in the exact order as given here: `input_rise_delay, input_fall_delay, output_rise_delay, output_fall_delay`.
If not specified, all delays are measured at the values defined by the `delay_*_*` parameters.
- `-dependent_load {<pin load_value>...}`
List of pin-load pairs to add to dependent side pins. (See [set_dependent_load](#))
This option provides user control over the load applied to side outputs that impact the arc. These are specified as a list of pin-load pairs, that is, `{pin1 load1 pin2 load2 ...}`. Dependent loads in the `define_arc` command will supersede those specified by the `set_dependent_load` command.

Virtuoso Liberate Reference Manual

Liberate Commands

`-dual_dir <U | D | B>`

Specifies the switching direction of dual pin (Up, Down or Both) used to set load direction. This option can have one of the following values: U (up), D (down) or B (both).

Note: The `-dual_dir` option is the equivalent of the `-load_dir` option as it defines the direction of the load circuitry to apply to the `dual_pin` of this `define_arc` command.

`-dual_pin <name>` Specifies the name of the other pin in a differential output pair.

`-dual_related <name>`

Specifies the name of the other pin in a pair of differential input pins.

When differential pairs are specified for inputs using `-dual_related` or for outputs using `-dual_pin`, the delay measurements can be made using the voltage crossover between the differential signals. To request that the delay measurements use the crossover point, the `-delay_threshold` option must be specified with a value of cross instead of a ratio. For example:

```
-delay_threshold { 0.5 0.5 cross cross }
```

`-extsim_deck_header`

Allows to provide external simulator commands directly to the external simulator on an individual arc basis without using the Liberate process or reviewing them. This option is intended to be used when an external simulator is used (refer to the `-extsim` option of the `char_library` command). It is a local arc specific version of the `extsim_deck_header` variable. As Liberate does not parse the string specified by this option, ensure that the contents are valid and consistent with the arc simulation. The value string can contain the return character ("\n"). The value string is included at the top of simulation deck. For example:

```
define_arc -extsim_deck_header ".ic n128 0" -related_pin
ck -pin Q ...
```

`-ic <"ic list">`

Defines the initial conditions to be applied during simulation. There should be one voltage value for each pin in the pinlist.

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|--------------------------|---|
| -ignore | Prevents characterization of all arcs originating from the related_pin and ending at the pin. When this option is used, only the pin and related_pin options are required. All other options, including the -vector option are not required and will be ignored. This option can be used to disable the internal view in Liberate from analyzing the specified arc. |
| -load_dir <U D B> | Output load direction: U (pullUp), D (pullDown), or B (Both) |
| -logic_condition <value> | Provides the logic states for static side pins. This option uses the same syntax and affects vector selection in the same manner as the -when option. However, unlike the -when option, it does not appear in the output library. This option can be used with or without the -vector option. |
| | Note: The -when, -logic_condition, and -vector options must be consistent; otherwise, the define_arc command will be ignored. This means that a given pin must have either 0 / X or 1 / X in all three options. |
| -margin | Allows a user-specified backoff margin to be added on a per-arc basis. This only applies to timing constraint arcs characterized using the path-delay method, such as define_arc commands that use the -pin_probe and -related_probe options and set_constraint commands with the -pin_probe_factor, -related_probe_factor, -min_margin, and -max_margin options. |

Virtuoso Liberate Reference Manual

Liberate Commands

-metric {list} Specifies a list of one or more metric types for measuring timing constraints. For related details, see [Using the -metric option](#).

Note: When a list of metrics is used in conjunction with a list of metric_thresh values, the order and number of the variables and values must match between these lists.

For example:

```
define_arc \
    -metric {
        delay
        constraint_delay_degrade_abstol
        constraint_delay_degrade_abstol_max } \
    -metric_thresh {
        0.1
        1e-12
        1e-12 } \
    -pin D \
    -related_pin clock \
    -vector { RR ... } \
    myCell
```

-metric_thresh {list}

Accepts a list of threshold values. When used in combination with the **-metric** option, overwrites the values specified by **set_constraint_criteria**, or by the corresponding global variable, **constraint_slew_degrade**, for this arc. This list must have one value corresponding to each criteria in the metric list.

Note: When a list of metrics is used in conjunction with a list of metric_thresh values, the order and number of the variables and values must match between these lists.

-pin {pins}

List of destination pins for the arc (typically, output pins for combinational arcs, input pins for timing constraint, or hidden power arcs). (REQUIRED)

-pin_dir <R | F> Transition direction of pin(s).

-pin_gnd {pin voltage ...}

Specifies a list of arc-specific input gnd pin and voltage pairs.

Note: This option can be used without the **-vector** option.

Virtuoso Liberate Reference Manual

Liberate Commands

`-pin_load <template>`

Specifies additional circuitry to be applied to all the destination pins of the `define_arc` command. This option refers to a template that defines the loading circuitry to be placed prior to the loading capacitance for the pin. The loading template must be pre-defined using the `define_pin_load` command. The additional circuitry can include pull-up and pull-down resistances and series resistance. The `load_dir` option defines whether one or both of the pullup or pulldown resistances should be applied to this arc. Setting it to `U` (up) will include the pullup resistance, setting it to `D` (down) will include the pulldown resistance, while setting it to `B` (both) means both pullup and pulldown resistors are included.

`-pin_load_dir {pin_name load_template direction}`

Applies specific pin loads to specific output pins. This provides more granularity of control than the `-pin_load` option. This option takes a triplet of arguments: `pin_name`, `load_template`, and `direction`. It can also be specified as a list of triplets. Here, the `load_template` argument is the name of the loading configuration defined with the `define_pin_load` command, and `direction` is specified with `U` (pull-up resistance), `D` (pull-down resistance), or `B` (both pull-up and pull-down resistances).

Note: `-pin_load_dir` overrides any `-pin_load` that might be present. For example:

```
-pin_load_dir {pin1 load_template1 U pin2 load_template2 D}
```

`-pin_probe {pins}` List of pin monitor node names

`-pin_probe_dir {pins}`
List of pin monitor node directions [RIF].

`-pin_probe_threshold {pins}`
List of pin monitor node thresholds.

`-pin_vdd {pin voltage ...}`
Specifies a list of arc-specific input vdd pin and voltage pairs.
Note: This option can be used without the `-vector` option.

Virtuoso Liberate Reference Manual

Liberate Commands

`-pinlist {list_of_pins}`

List of pins corresponding to vector.

`-pg_pin <value>` Assign a value to this power/ground pin only

`-prevector_pinlist {list}`

User-specified pin list for pre-vectors.

`-prevector_slew`

Specifies the slew rate to apply to the input and bidi pins in the `-prevector_pinlist`. Supported values are: `<value>`, `min`, `mid`, `max`, `index_n`. For more information, see [prevector slew](#).

`-prevector "<vector ... >"`

User specified initialization vectors. Default: no prevectors

For related information, see [Using the -prevector option](#).

`-probe <{names} | altos_internal>`

Specifies a list of names of nodes to monitor for sequential cells constraint characterization or keyword.

The `-probe` option is used for timing constraints. It defines the node(s) to monitor when determining the constraint. It can be an external pin, such as, the `Q` pin in a flip-flop or an internal node name.

Use the `-probe altos_internal` option when a constraint can be measured at both an internal node and an output pin. This instructs Liberate to use the internal probe node. If the `probe` argument is not specified, the pin defined by the `constraint_output_pin` variable is probed.

`-probe_dir <R | F>` Specifies the node direction that the probe nodes will be transitioning. The acceptable values are: `R` and `F`.

One direction can be provided for each probe node.

Note: The `-probe_dir` option is used with the `-probe` option.

`-related_pin {pins}`

List of related pin names (typically input pins for combinational arcs, clock pins for timing constraint arcs).

`-related_pin_dir <R | F>`

Transition direction of related pin(s)

`-related_probe {pins}`

List of related monitor node names.

Note: The `-pin_probe` and `-related_probe` options are used to specify the measurement target nodes.

`-related_probe_dir {R | F}`

List of related_probe monitor node directions [R|F]

`-related_probe_threshold {pins}`

List of related monitor node thresholds.

`-sdf_cond <"function">`

Defines logic conditions of the other pins of the cell to enable the arc using SDF compatible syntax.

Note: If `sdf_cond` is specified by the user using the `define_arc -sdf_cond` command, Liberate will leave it intact and perform no replacement. A warning message will be displayed indicating that the SDF conditions have been overwritten by user when the `write_library` command is run. Any special characters (that is, " +!&*") will be kept.

Sample message:

Warning (write_library): The `sdf_cond` has been overridden by user supplied values (see `define_arc -sdf_cond`) in one or more timing groups. These user supplied values are not checked for consistency by Liberate.

If `msg_level >0`, Liberate will dump the details for all cells that contain user-defined arcs with `sdf_cond`.

`-slew_threshold { lower_rise upper_rise lower_fall upper_fall }`

Defines a list of four slew percentage measurement points (a ratio of VDD normalized to between 0 and 1) for the arc. For the `-slew_threshold` option, the values in the list represent the following measurement thresholds in the exact order as given here: `lower_rise_slew`, `upper_rise_slew`, `lower_fall_slew`, `upper_fall_slew`.

If not specified, all slews are measured at the values defined by the `measure_slew_*` parameters.

Virtuoso Liberate Reference Manual

Liberate Commands

```
-type < async | ccsn_first | ccsn_last | combinational | disable |
edge | enable | hidden | hold | mpw | nochange_high_high|
nochange_high_low | nochange_low_high | nochange_low_low |
non_seq_hold | non_seq_setup | power | recovery | removal | setup >
```

Type of arc (Default: *combinational*)

```
-value <value | {list}>
```

Overrides the characterized values. Default: use the characterized values.

The value(s) override the characterized results and forces a value (or list of values) for all entries into the data table for the specified arc. This option accepts a single value or a list of values. If a single value is provided, then the table gets a scalar type of data. However, if a list is specified, there must be one entry for each point in the data table. For example, a 7x7 table would require 49 values. For any time-based arc, the value is in seconds (i.e.: 5e-9 = 5ns).

```
-value_trans <value | {list}>
```

Overrides the values in the transition table. Default: use the characterized values.

The `-value_trans` option works similarly to the `-value` option except that it applies to transition table. See the `-value` option for more information.

```
-vector <"stimulus">
```

Vector stimulus used to simulate the specified arc, each bit can be one of: R, F, X, 1, or 0. For related details, see [Using the -vector option](#).

```
-when <"function">
```

Defines the logic conditions of the other pins of the cell to enable this arc using the Liberty `when` syntax. It corresponds to the Liberty `when` attribute.

```
{cell_names}
```

List of cells.

The `define_arc` command can be applied to a single cell or a list of cell names. The templates to use for each arc will default to the template defined for the cell unless a `define_index` command is specified for that particular arc.

Liberate can measure the path from the `pin` to a user-defined `pin_probe` and from the `related_pin` to a user-defined `related_probe`. The `pin_probe` and `related_probe`

options can be used to specify the measurement target nodes. The `-pin_probe_dir`, and `-related_probe_dir` options can be used to specify the transition direction of the `pin_probe` and `related_probe` nodes. The `-pin_probe_threshold` and `-related_probe_threshold` options specify a ratio of supply and can be used to specify the measurement thresholds for the `pin_probe` and `related_probe` nodes. See [Constraint-related Delay and Clock Path Measurement](#) for more information and an example.

The `define_arc` command can be used to guide the CCSN characterization. Liberate attempts to automatically identify internal probe nodes. Use the `ccsn_probe` option to identify a particular node to use as the probe. When arc based ccsn is specified, the same node must be used as the probe node so a `ccsn_first_stage` and `ccsn_last_stage` are consistent and represent an arc from the input to the output pin. For example:

```
define_arc \
    -related_pin A \
    -type ccsn_first \
    -ccsn_stage pin \
    myCell
```

This command must be used before [char_library](#).

Using the `-metric` option

The `-metric` option is used for setting constraint criteria for timing constraint arcs and can contain a list that includes the following keywords corresponding to Liberate variables:

```
delay | delay_both_edges | constraint_delay_degrade
glitch | constraint_glitch_peak
slew | constraint_slew_degrade
width | constraint_width_degrade
constraint_delay_degrade_abstol
constraint_delay_degrade_abstol_max
constraint_delay_degrade_minimize_dtoq_tol
constraint_delay_degrade_minimize_dtoq
path_delta
removal_glitch_peak
```

Note: `path_delta` cannot be used together with other metrics in the same `define_arc` command.

- ❑ When set to `delay` the arc will be in violation when a delay change at the probed pin exceeds `constraint_delay_degrade` parameter.
- ❑ When set to `slew`, the arc will be in violation when the slew change at the probed pin exceeds either: the `constraint_slew_degrade` parameter; the threshold set

with option `metric_thresh` (below); or the threshold set with the `set_constraint_criteria` command.

- ❑ When set to `delay_both_edges`, the metric applies to constraints where the probe node transitions in response to both the related and the constrained pins. The constraint is tested with delays from both the constrained and related pins to the associated transition on the probed node. Degradation failure on either edge indicates a failure of the constraint.
- ❑ When set to `glitch` the arc will be in violation when the glitch-peak at the probed pin exceeds `constraint_glitch_peak` parameter.
- ❑ When set to `width`, the arc will be in violation when the output pulse width degrades beyond the percentage set in either: the `constraint_width_degrade` variable; the threshold set with option `metric_thresh` (below); or the threshold set with the `set_constraint_criteria` command.
- ❑ If both delay and glitch criteria is set for hold time, the glitch crteria is used.
- ❑ The `metric_path_delta` causes Liberate to measure the data path, (path from the `pin` to the `pin_probe`) and the clock path (the path from the `related_pin` to the `related_probe`) and report the difference between these two paths as the constraint value. For more information, see [Appendix B, “Constraint-related Delay and Clock Path Measurement”](#).

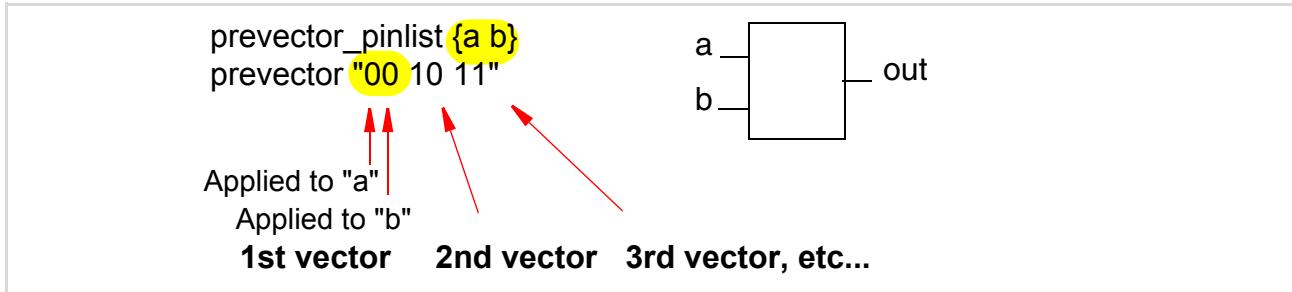
Example:

```
define_arc \
-type setup \
-metric path_delta \
-pin D -pin_dir R \
-pin_probe n1 -pin_probe_dir R \
-related_pin clk -relate_pin_dir R \
-relate_probe n2 -related_probe_dir F \
{ myFF }
```

Using the -prevector option

The `-prevector` option specifies an arbitrary simulation stimulus to be applied before the actual vector where the characterization will be measured. Pre-vectors are used to put a device in a user-defined state before proceeding with characterization. Prevector values must be 0 or 1. See related variables: `prevector_period` and `prevector_slew`.

This option must be used in conjunction with `-prevector_pinlist`. The following illustrates how prevectors are specified with the `-prevector_pinlist` option:



 **Important**

1. The last state of the prevector must match the beginning state of the vector for a given pin, otherwise an unintended transition will occur.
2. When `prevector` is specified, Liberate disables Inside View for this arc and operates as if it is in `io` mode for this arc. In `io` mode, `probe` is required for all constraint arcs, including `mpw`. (See [char library](#))

Example 1:

```
# Define the IOCELL
define_cell \
  -input { D } \
  -output { Q } \
  -clock { CK } \
  -pinlist { D CK Q } \
  -delay delay_template_3x3 \
  -power power_template_3x3 \
  MYCELL

define_arc \
  -prevector_pinlist {D CK} \
  -prevector "00 01 00 10" \
  -vector      "1RR" \
  -related_pin CK \
  -pin Q \
  MYCELL
```

Example 2:

```
# constraint arcs from CK => CK using prevector min_pulse_width
```

```
define_arc \
  -type min_pulse_width \
  -prevector_pinlist {D CK Q} \
-prevector {100 111 001} \
-vector {0RF} \
  -related_pin CK \
  -pin CK \
  DFF

# constraint arcs from CK => CK using prevector min_pulse_width
define_arc \
  -type min_pulse_width \
  -prevector_pinlist {D CK Q} \
-prevector {100 111 011} \
-vector {0FF} \
  -related_pin CK \
  -pin CK \
  DFF
```

Using the -type option

The `-type` option defines the type of arc. The arc can be a combinational path from input pins (`related_pin`) to output pins (`pin`) for combinational cells. It can also be a timing constraint of type *setup*, *hold*, *recovery*, *removal* and *mpw* between data (`pin`) and a clock (`related_pin`) for sequential cells. Other valid types include `async`, `combinational`, `disable`, `edge`, `enable`, `hidden`, `non_seq_hold`, `non_seq_setup` and `power`. An `async` arc corresponds to a preset or clear transition. An `edge` arc between an input and an output pin will be an edge-triggered transition. The `enable` and `disable` types are used for specifying arcs that enable and disable tristate gates. A `hidden` arc specifies an `internal_power` group found under the `input` or `bidi` pin that does not cause any output to transition and is used to characterize the hidden power for that pin. The `non_seq_hold` and `non_seq_setup` types are used for specifying setup and hold arcs between a pin and a non-clock related pin. The `power` arc specifies an `internal_power` group found under the `output` or `bidi` pin where one or more outputs transition.

Specify the `define_arc` commands for `nochange` arcs whenever they are required to be modeled in the output library. These arcs will either have a user-defined value or be characterized by Liberate. See the variables `nochange_mode` and `nochange_value` for more details. If `nochange_*_*` arcs are specified without `-pin_dir` or `-vector` and `nochange_mode` is set to 1, both `rise_constraint` and `fall_constraint` are characterized. If `-pin_dir` or `-vector` are specified only the specified `rise_constraint` and `fall_constraint` are generated. The `nochange` pull-in constraints use the same metric settings as `delay_constraint_delay_degrade`, `constraint_delay_degrade_abstol`,

constraint_delay_degrade_abstol_max. Using abstol makes the most sense for nochange arcs because the object is to preserve the clock shape within some tolerance and not to locate the metastable region. All values should be positive.

When using a glitch metric for nochange arcs, the standard glitch metric settings apply.

If there is no define_arc -type nochange_*, no such arc is written to the output library. Nochange arcs are only generated if a define_arc exists.

Important

The define_arc command must be specified if Liberate is to characterize and model nochange arcs.

Example:

```
# Generate all 4 nochange arc constraints between EN and CKI with the correct
# metric (delay, pull-in, or glitch) for the specified cell (ANDX1 in this example).
set_var nochange_mode 1
define_arc -type nochange_low_high -pin EN -related_pin CKI ANDX1
define_arc -type nochange_high_high -pin EN -related_pin CKI ANDX1
```

Using the -vector option

The -vector option defines the stimulus to simulate this arc. It is defined as a string of bits (digits) where each bit corresponds to one string in the pinlist. Each bit can have the values R (rising), F (falling), X (don't care), 1 (logic high), 0 (logic low). The order of the bits must correspond one-to-one to the order of pin list defined by using the -pinlist option of the define_cell or define_arc command. Blank spaces are permitted in the vector for readability. The vector value for a pin must be logically consistent with the -when and -constraint options. Else, the define_arc command is rejected. If a side input is specified as X, it is overridden by the state of the pin as specified in the -when or -constraint option. If there are buses in the pinlist, there should be one bit in the vector for each bus. The bit value is applied to all elements in a bus. If different logical values are required for each bit in the bus, the bus bits must be separately enumerated in the pinlist.

All stimulus vectors that satisfy the when condition will be enumerated and simulated.



Do not define a vector that drives a node such that it cannot be resolved (that is, a given node is driven both high and low.) This is called a "collision", resulting in the node voltage becoming unknown and causing high leakage. If this occurs, Liberate will ignore the define_arc command that caused this condition and output a warning message.

Examples:

```
define_arc \
  -pinlist { A B C[5:0] OUT } \
  -vector { R 0 1 F } \
  -related_pin A \
  -pin OUT \
  myCell

define_arc \
  -pinlist { A B C[5] C[4] C[3] C[2] C[1] C[0] OUT } \
  -vector { R 0 101110 F } \
  -related_pin A \
  -pin OUT \
  myCell
```

Examples

```
# Define the IOCELL
define_cell \
  -input {IN OEN} \
  -output {OUT} \
  -bidi {PAD} \
  -pinlist {IN OEN PAD OUT} \
  -delay delay_template_3x3 \
  -power power_template_3x3 \
  IOCELL
```

```
define_arc \
  -vector {XXRR} \
  -related_pin PAD \
  -pin OUT \
  IOCELL
define_arc \
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
-vector {XXFF} \
-related_pin PAD \
-pin OUT \
IOCELL

define_arc \
-type hidden \
-when "OEN * !PAD" \
-vector {R10X} \
-pin IN \
IOCELL

define_arc \
-type hidden \
-when "OEN * !PAD" \
-vector {F10X} \
-pin IN \
IOCELL

# Define additional loading for the PAD pin
define_pin_load \
-pullup_voltage 3.3 \
-pullup_resistance 1000 \
-pulldown_resistance 1000 \
-series_resistance 50 \
load_template

define_arc \
-vector {R0RX} \
-pin_load load_template \
-related_pin IN \
-pin PAD \
IOCELL

define_arc \
-vector {F0FX} \
-pin_load load_template \
-related_pin IN \
-pin PAD \
IOCELL

define_arc \
-type enable \
-vector {1FRX} \
-pin_load load_template \
-related_pin OEN \
-pin PAD \
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
IOCELL
define_arc \
  -type enable \
  -vector {0FFX} \
  -pin_load load_template \
  -related_pin OEN \
  -pin PAD \
  IOCELL
define_arc \
  -type disable \
  -vector {0RRX} \
  -pin_load load_template \
  -related_pin OEN \
  -pin PAD \
  IOCELL
define_arc \
  -type disable \
  -vector {1RFX} \
  -pin_load load_template \
  -related_pin OEN \
  -pin PAD \
  IOCELL

# For differential output signals into a cell
# I -> PAD
# Pinlist : I PAD PADN
define_arc \
  -vector "FFR" \
  -delay_threshold {0.5 0.5 CROSS CROSS} \
  -related_pin I \
  -dual_pin PADN \
  -pin PAD \
  IOCELL_diff
```

define_bundle_pins

Treats the pins as if they were from equivalent latches or flip-flops in a bank instead of including each of the input and/or output pins. For example, if pins SE1 and SE2 are from separate flip-flops that are identical, the bundle syntax merges them together in a single bundle with shared timing and power models. The resulting model is more compact than one in which all pins are uniquely specified. By using this method, all cells share a single model. It should only be used if each cell in the bank is identical.

Options

`-criteria {<delay | power> <min | avg | max>}`

Specifies how the characterized data is to be modeled when the `-use_pin` option is selected. Using the `-patterns` option restricts the bundle patterns considered for the selection. Without `-use_pin` all the bundle pins are measured separately. The `-criteria` option then controls how the individual pin models are combined into the bundle model. It does not affect the characterized vectors.

`-no_model`

Disables modeling of bundles in the output library. This option allows a bundle to be defined which is not modeled in the Library. This allows `-patterns` to be associated with the bundle pins to restrict characterization vectors for multi-bit cells without affecting the structure of the resulting library.

-patterns { list } List of string patterns. This option specifies a list of patterns consisting of 0 and 1 to be assigned side pins that are members of the bundle or the string `all`. Side pins are the pins that do not logically impact the cone of logic identified by mega-mode. These patterns reduce the number of vectors to be simulated by restricting the settings of nodes which are not critical to a particular measurement. The default patterns are {0 1} which will create 2 vectors with all 0's and all 1's assigned to side pins in the bundle. Setting the patterns to `all` does not restrict the bundle pins in any way. Setting the patterns to {01 10} would use two patterns with alternating 0's and 1's. The pattern is repeated as many times as necessary to cover the number of bits in the bundle so it is not necessary to devise a different pattern set for each bundle. As many patterns can be supplied as desired but increases the number of vectors. When `-patterns` is not set to `all`, the when conditions generated for the library does not include bundle pins. See ([mega_mode_constraint](#), [mega_mode_delay](#), [mega_mode_hidden](#), and [mega_enable](#)).

-use_pin <pin_name>

Specifies the name of the pin used to acquire all timing and power models related to a bundle. This saves considerable cell analysis and simulation time, because arcs involving all other pins in the bundle will be ignored.

Note: If this option is used with `define_bundle_pins`, the command must be used [before char_library](#).

If this option is not used, `define_bundle_pins` can be used after `char_library`, but before [write_library](#).

{cellNames}

List of cell name (Required). Default: {}

<bundle_name>

The name of the bundle (Required). Default: " "

{pins}

List of pins in the bundle (Required). Default: {}

Examples

```
define_bundle_pins -use_pin D1 $cellname D {D4 D3 D2 D1}
define_bundle_pins -use_pin Q1 $cellname Q {Q4 Q3 Q2 Q1}
```

define_bus

Groups the bus bits into a bus in the output library. It is not required if buses are explicitly defined in [define_cell](#).

Options

| | |
|------------------------------------|---|
| <code>-by <integer></code> | Specifies the bus bit increment. Default: 1 |
| <code>-dont_overwrite</code> | Specifies not to overwrite the previous bus definition. Default: overwrite (multiple <code>define_bus</code> commands for the same bus will be merged) |
| <code>-from <integer></code> | Specifies where the bus starts. Default: 0 Note: The value specified with the <code>-from</code> option can be higher or lower than the <code>-to</code> option, but not equal. These option define the bus start and ending range. |
| <code>-patterns { list }</code> | List of string patterns. This option specifies a list of patterns consisting of 0 and 1 to be assigned to the side pins that are members of the bundle or to the string <code>all</code> . Side pins are the pins that do not logically impact the cone of logic identified by the mega mode. These patterns reduce the number of vectors to be simulated by restricting the settings of nodes that are not critical to a particular measurement. The default patterns are <code>{0 1}</code> that creates 2 vectors with all 0s and all 1s assigned to side pins in the bundle. Setting the patterns to <code>all</code> does not restrict the bundle pins in any way. Setting the patterns to <code>{01 10}</code> would use two patterns with alternating 0s and 1s. The pattern is repeated as many times as necessary to cover the number of bits in the bundle, so it is not necessary to devise a different pattern set for each bundle. As many patterns can be supplied as desired, but it increases the number of vectors. When <code>-patterns</code> is not set to <code>all</code> , the when conditions generated for the library do not include bundle pins. See <u>mega_mode_constraint</u> , <u>mega_mode_delay</u> , <u>mega_mode_hidden</u> , and <u>mega_enable</u> . |
| <code>-to <integer></code> | Specifies where the bus ends. Default: 0 |
| <code>-use_bit</code> | Specifies the bus bit whose characterized data will be applied to the entire bus. Default: the value of the <code>-from</code> option |
| <code><cellName></code> | List of cell names (Required). Default: {} |

Virtuoso Liberate Reference Manual

Liberate Commands

<bus_name> Specifies the bus name (Required). Default: ""

Note: The `bus_name` option should be the name of the pins minus the bus index.

This command must be used before [write_library](#) and can also be used after [char_library](#) or [read_ldb](#).

define_cell

Defines how a cell is to be characterized. Each cell can have a unique `define_cell` command or a `define_cell` command can be shared amongst a group of cells.

Options

| | |
|---------------------------------------|---|
| <code>-async {pin_names}</code> | List of asynchronous pin names. |
| <code>-bidi {pin_names}</code> | List of bi-directional pin names. |
| <code>-clock {pin_names}</code> | List of clock pin names |
| <code>-constraint <name></code> | Name of template for constraint tables. The specified template is used for characterizing each library construct. If a template is specified, the appropriate construct is characterized for the given set of cells. If a template is omitted, the construct is not characterized. The <code>-constraint</code> option enables characterization of timing constraints (setup, hold, recovery, removal). The range of input slews to use for the data and clock signals is defined by the given template name where the template is pre-defined using the <code>define_template</code> command. |
| <code>-delay <name></code> | Name of template for delay tables. The specified template is used for characterizing each library construct. If a template is specified, the appropriate construct is characterized for the given set of cells. If a template is omitted, the construct is not characterized. The <code>-delay</code> option enables characterization of cell delay and output slew for the non-linear delay model (NLDM). The range of input slews and output loads to use for this construct is defined by the given template name where the template is pre-defined using the <code>define_template</code> command. |

Virtuoso Liberate Reference Manual

Liberate Commands

`-harness <load_subckt_name>`

Name of subckt containing custom load.

The `-harness` option allows you to specify an arbitrary SPICE format circuit to "wrap" around the cell that is being characterized. You need to provide a subcircuit containing an arbitrary input circuit and/or an arbitrary output load. The harness sub-circuit must have been loaded with `read_spice`. The harness sub-circuit must be defined with pins that have the identical names as the port names of the cell being characterized since the harness pin names are name mapped to the subcircuit being characterized. The following table indicates a naming convention used to refer to connections from the original characterization setup that should be shifted to new connections in the harness (see example):

| Harness Parameter Description | Harness Name |
|--------------------------------------|--|
| Input stimulus | <code><subckt_pin>_altos_stim</code> |
| Cell input port | <code><subckt_pin>_altos_in</code> |
| Cell output port | <code><subckt_pin>_altos_out</code> |
| Output load | <code><subckt_pin>_cap</code> |
| Supply voltage | <code><subckt_pin>_voltage</code> |

`-ignore_input_for_auto_cap {pin_names}`

List of input/bidi pins. Any timing arcs that originate at the pins specified in this list will not be considered for `auto_index` and `auto_max_capacitance` calculations. Instead, the values specified in the templates (see [define_template](#)) for this cell will be used. This option should not be used with `scale_load_by_template`, `scale_tran_by_template`, and `-auto_index` option of the [write_template](#) command because the template values will not represent correct values.

-ignore_output_for_auto_cap {pin_names}

List of bidi/output pins or an asterisk (*) to match all output pins.

Any timing arcs that terminate at the pins specified in this list will not be considered for `auto_index` and `auto_max_capacitance` calculations. Instead, the values specified in the templates (see `define_template`) for this cell will be used. This option should not be used with `scale_load_by_template`, `scale_tran_by_template`, and `-auto_index` option of the `write_template` command because the template values will not represent the correct values.

-ignore_pin_for_ccsn {pin_names}

List of input/bidi pins. No CCSN data is characterized for the specified pins.

-input {pin_names} List of input pin names

-internal {pin_names}

Defines a list of pin that are internal to the cell, that is, they are not part of the port list of the top level subckt definition for the cell. Such pins must be defined when they are used as `-pin` or `-rel_pins` option of the `define_arc` command.

Note: This option is available only in Liberate AMS.

-internal_supply {supply_names}

List of switched supply pin names.

The `-internal_supply` option is used for cells such as power switch cells to identify output pins that are to be treated as switched power nets. The `internal_supply` net must be a port in the subckt definition of the cell. When this option is used, Liberate will characterize additional `dc_current` data for CCCs connected to the specific internal supply pin(s), post process the `dc_current()` groups into the correct format recognized by LC, and output the powerdown function attribute. All `internal_supply` pins must also be identified as a supply using the `set_gnd` / `set_vdd` commands.

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|--|---|
| <code>-mpw <name></code> | Name of template for MPW tables. |
| | The <code>-mpw</code> option enables characterization of a two dimensional MPW table. The range of input slews to use for the data and clock signals is defined by the given template name where the template is pre-defined using the define_template command. |
| <code>-output {pin_names}</code> | List of output pin names. |
| <code>-pinlist {pin_names}</code> | Defines the pin order and is used by the <code>-vector</code> option of the define_arc command when specifying a user-defined timing arc. The pin list can contain internal pins as well as input, inout and output pins. |
| <code>-power <name></code> | Name of template for power tables. The specified template is used for characterizing each library construct. If a template is specified, the appropriate construct is characterized for the given set of cells. If a template is omitted, the construct is not characterized. |
| | The <code>-power</code> option enables characterization of switching power and hidden power (power dissipated when the output doesn't switch). The range of input slews and output loads to use for this construct is defined by the given template name where the template is pre-defined using the define_template command. |
| <code>-scan {pin_names}</code> | List of scan related pin names |
| <code>-scan_cell_postfix "string"</code> | String added to end of dummy scan cell and footprint names |
| <code>-scan_cell_prefix "string"</code> | String added to beginning of dummy scan cell and footprint names |
| <code>-scan_disable {pin value, ...}</code> | List of pin/value pairs that will disable scan mode |
| <code>-scan_scale_power_factor <factor></code> | |

Virtuoso Liberate Reference Manual

Liberate Commands

Scale applied to power in scan dummy cell. Default:
number_of_total_pins / number_of_non-scan_pins

-si_immunity <name>

Name of template for signal integrity immunity tables. The defined template is used for characterization of noise immunity rejection curves. The range of input noise widths and output loads to use are defined by the given template name where the template is pre-defined using the `define_template` command. The `-si` option of the `char_library` command must also be set for signal integrity characterization.

-type <normal | mega | io>

The type of cell. Default: `normal`

Note: In the `define_cell` command, the `-type uda` option has been renamed to `io`. Both `io` and `uda` are accepted and execute the same code. This means that they are synonymous.

Liberate uses a unique algorithm for automatic vector generation of larger (`mega`) cells that involves a more detailed pre-characterization analysis than is required for typical cells (`normal`). Mega cells are cells with a large number of pins, a large number of transistors, or both. Liberate will use the `normal` cell algorithm by default. If the `mega` cell algorithm is chosen, a message will appear in the log file. The `-type` option can be used to override the built-in selection criteria. The default is to let Liberate automatically choose which algorithm to use when characterizing each cell. Using `mega` mode will typically reduce both pre-processing and characterization time for large cells. Note that `mega` mode is currently in beta phase, as the primary focus is on timing and not power. Using `uda` disables the automatic vector generation of Liberate and characterizes only the user-defined arcs (`uda`).

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|---------------------------------------|---|
| <code>-user_arcs_only</code> | Skips the automatic addition of arcs by the <i>Inside View</i> of Liberate. This option has been provided to support the <code>char_library -user_arcs_only</code> functionality for a specific cell list. It affects only the cells listed in the <code>define_cell</code> command. You must provide all required arcs by using the <code>define_arc</code> command. |
| | Note: This option is often used with the <code>write_template -verbose</code> command to ensure that the new library exactly matches the reference library structure. |
| <code>-when <"function"></code> | Specifies user-defined cell-level logic constraints using the Liberty format <code>when</code> syntax, constraining the automatic vector generation of Liberate for the given cell. The <code>define_cell -when</code> logical condition applies only to steady state signals such as leakage states and side input states that are non-switching. Liberate does not automatically infer simultaneous switching inputs based on the logical condition. You can use <code>define_arc</code> to specify simultaneous switching inputs, or specify a truth table to be translated automatically. |
| <code>{cell_names}</code> | List of cell names to be characterized. |

The `-input`, `-bidi`, `-output`, `-clock`, and `-async` options define the pin type for the given list of pin names. All pins of a cell must have a defined pin type. If a pin name or pin type does not apply to a particular cell it will be ignored. For example, combinatorial cells such as NOR or NAND gates might not have `clock` or `async` pins. Therefore, any definition for these pins will be ignored. Likewise, if a pin name is specified but not used by a particular cell it will be ignored by that cell. The same pin name cannot appear in multiple pin types within a single `define_cell` command. For example, if one cell has an input Y and another has an output Y, they must be defined uniquely with separate `define_cell` commands.

The `-scan_*` options are required when a dummy cell (one with the scan pins removed) is required. The `-scan` option specifies the names of the scan-related pins that are to be removed. The `-scan_disable` option provides the state information that will disable scan mode. If the application of the `-scan_disable` constants results in 2 or more data groups with the same states, then the group data will be merged according to the rules specified by the `default_leakage`, `default_power`, and `default_timing` parameters. (If the default group is turned off, then the group data will be merged using the default setting of `max`) The `-scan_cell_prefix` and `-scan_cell_postfix` options provide a string to attach to the beginning or end of the cell name when the scan pins are removed. If either of these options is specified, the library will contain both the original cell and the scan dummy cell with the modified name. When the `write_library -scan_dummy_scale_power` option is enabled, the power data in the scan dummy will be scaled according to the factor specified

by the `-scan_scale_power_factor` option. This option specifies a scale to be applied to the remaining power tables when the scan pins are removed. The default power scaling factor is computed as the number of total pins divided by the number of non-scan pins.

This command must be used before [char library](#).

Examples

```
define_cell -input {A1 A2} -output {Z} \
-delay delay_3x3 -power power_3x3 \
{NAND2X4 NOR2X2}

define_cell \
    -input {D} \
    -output {Q QN} \
    -clock {CK} \
    -async {SN} \
-delay delay_5x5 \
-power power_5x5 \
-constraint constraint_3x3 \
{DFFX1}

define_cell \
    -input {A1 A2 A3 A4 SLP} \
    -output {Y} \
    -pinlist {A1 A2 A3 A4 SLP Y} \
-delay delay_5x5 \
-power power_5x5 \
-when "!SLP" \
{MTAND2 MTAND3 MTAND4}

# The following example attaches a 100 Ohm resistor between
# the PAD and PADN pins in addition to the load caps
# (from index_2) added to PAD and PADN.
=====template.tcl=====
define_cell \
    -input {I} -output {PAD PADN} \
    -pinlist {I PAD PADN} \
    -harness "Obuff_load" \
    -delay delay_template_DL \
    -power power_template_DL \
    OBUFF
=====char.tcl=====
lappend spicefiles /home/work/custom_load/testload
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
=====testload=====
.subckt Obuff_load PAD_altos_out PADN_altos_out
C1 PADN_altos_out 0 'PADN_cap'
C2 PAD_altos_out 0 'PAD_cap'
R1 PAD_altos_out PADN_altos_out 100
.ends

# Here is an example of a mux testcase with an
# un-buffered input and one-hot data selectors that use
# the define_cell -harness capabilities.
=====template.tcl===
define_cell \
    -input { S0 S0_B S1 S1_B I0 I1 } \
    -output { X } \
    -delay delay_template_3x3 \
    -power power_template_3x3 \
    -when "((S0 !S1) + (!S0 S1)) & \
        (S0 ^ S0_B) & \
        (S1 ^ S1_B)" \
    -pinlist {I0 I1 S0 S0_B S1 S1_B N X} \
    -harness mux_harness \
    { MUXI21 }

=====char.tcl=====
lappend spicefiles mux_harness.spi
=====harness subckt=====
.subckt mux_harness
+ I0_altos_in I0_altos_stim I1_altos_in
+ I1_altos_stim X_altos_out S0_altos_in
+ S0_altos_stim S0_B_altos_in
.inc 'INV_1.spx'
* driver for pin I0,
* I0_altos_stim -> I0_altos_tmp -> I0_altos_in
Xdriver0_1 I0_altos_tmp I0_altos_stim vdd vss INV_1
Xdriver0_2 I0_altos_in I0_altos_tmp vdd vss INV_1
* driver for pin I1,
* I1_altos_stim -> I1_altos_tmp -> I1_altos_in
Xdriver1_1 I1_altos_tmp I1_altos_stim vdd vss INV_1
Xdriver1_2 I1_altos_in I1_altos_tmp vdd vss INV_1
* driver for pin S0 and S0_B,
* S0_altos_stim -> S0_B_altos_in -> S0_altos_in
Xdrivers01 S0_B_altos_in S0_altos_stim vdd vss INV_1
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
Xdrivers02 S0_altos_in S0_B_altos_in vdd vss INV_1
* VDD_voltage is defined as a parameter by
* Liberate, <supply_net>_voltage
* .global is not supported to avoid net name
* conflicts with the cell being characterized
* and messing up power/leakage measurements
Vdd vdd 0 VDD_voltage
Vss vss 0 VSS_voltage
* X_cap is defined as a parameter by Liberate:
*      <port_name>_cap
* Here we are defining a PI load based on the
* index_2 value.
.param c_near=X_cap/2
.param c_far=X_cap/2
.param rshield=5
Cnear X_altos_out 0 c_near
Rsh X_altos_out far_end rshield
Cfar far_end 0 c_far

.ends
```

define_cell_leakage

Creates the `define_leakage` states. This command supports the complete iterative expansion of a list of pins while holding a separate list of pins at a static (DC) level.

Use this command to expand a list of pins into separate `define_leakage` commands. This command supports the complete iterative expansion of a list of pins in the `active_pinlist` while holding a separate list of pins specified in the `static_pinlist` at the values specified in the `static_vector`. In addition, different prevector sequences can be provided using the `prevector_pinlist` and `prevector_list`. The `prevector_when_list` associates specific prevector sequences from the `prevector_list` with a specific “when”.

Options

| | |
|---|--|
| <code>-echo</code> | Echo each generated <code>define_leakage</code> command into the log file. |
| <code>-active_pinlist {list}</code> | List of pin names to expand into different states. |
| <code>-prevector_list {list_of_values}</code> | List of prevectors. For example, {"0 1 0" "1 0 1"} |
| <code>-prevector_pinlist {list}</code> | List of pins corresponding to <code>-prevector</code> . |
| <code>-prevector_when_list {list}</code> | List of 'when' conditions corresponding to each prevector. For example, {"CK" "!CK"} |
| <code>-static_pinlist {list}</code> | List of pins that remain static (do not toggle). |
| <code>-static_vector {list}</code> | Vector string containing a value for each pin in the static pinlist. |
| <code>{cell_names}</code> | Required list of cell names to which the <code>define_leakage</code> commands are applied. |

This command must be specified after the `define_cell` command for the cells listed in the `define_cell_leakage` command and before the `char_library` command.

Examples

Example 1

Generate **define_leakage commands** by expanding active pin "b" while holding pin "a" as static.

```
define_cell -input { a b } -output y -pinlist {a b y} na2
define_cell_leakage \
    -echo \
    -active_pinlist "b"
    -static_pinlist "a" \
    -static_vector "1" \
    na2
```

The following commands will be echo'd into the log file and executed:

```
define_leakage -when "a*!b" -vector "10x" na2
define_leakage -when "a*b" -vector "11x" na2
```

Example 2

Generate **define_leakage commands** for two active pins and one static pin with prevectors associated with the state of `clk`.

```
define_cell -input { d en } -clock { clk } -output { q } -pinlist { d clk en q } dff
define_cell_leakage \
    -echo \
    -prevector_pinlist {clk} \
    -prevector_list {"0 1 0" "1 0 1"} \
    -prevector_when_list { "!clk" "clk" } \
    -active_pinlist "d clk" \
    -static_pinlist "en" \
    -static_vector "1" \
    { dff }
```

The following commands will be echo'd into the log file and executed:

```
define_leakage -when "en*!d*!clk" -prevector_pinlist "clk" -prevector "0 1 0" -vector "001x" dff
define_leakage -when "en*!d*clk" -prevector_pinlist "clk" -prevector "1 0 1" -vector "011x" dff
define_leakage -when "en*d*!clk" -prevector_pinlist "clk" -prevector "0 1 0" -vector "101x" dff
define_leakage -when "en*d*clk" -prevector_pinlist "clk" -prevector "1 0 1" -vector "111x" dff
```

define_duplicate_cell

Specifies a list of cells that will be duplicated using the characterization data. The characterization data must exist for the master cell. The duplicated cells are not characterized. If the `-cells` option of the [write library](#) command is used, the duplicated cells must be specified with this option to be written to the output library.

Options

| | |
|---------------------------|--|
| <code><cell></code> | The name of the master cell to be duplicated. |
| <code>{ cellname }</code> | List of cell names duplicated from the master. |

This command can be specified before [write library](#). If specified before [char library](#), the duplicated cells are not characterized.

Examples:

```
define_duplicate_cell INVX1 {INVX2 INVX3}  
define_duplicate_cell INVX1 {INVX4}
```

define_duplicate_pins

Specifies a list of pins for a cell that will not be directly characterized, but instead will be given duplicate data from a characterized pin. This command copies all the pin data from the *<pin>* to each of the duplicated pins including any data where the *<pin>* is a related_pin. If the pin is an input pin, the duplicate input pin will include pin cap, hidden power, constraints, and ccsn (CCS noise). In addition, all input to output arcs where the pin is a related_pin will be duplicated for each duplicate_pin. For example:

```
define_duplicate_pin mux A { B C D E F G H I }
```

Options

<cellname> The cell name to apply the duplication to.

<pin> The pin to be duplicated.

{ *duplicate_pins* } List of pin names to be duplicated.

This command also supports duplicating a complete bus. For example:

```
define_duplicate_pins myCell addrA addrB
```

...where *addrA* and *addrB* are buses. The following restrictions apply:

- ❑ *addrA* and *addrB* must have the same "direction" (cannot duplicate an input to an output.)
- ❑ *addrB* cannot be of a lower range than *addrA*, but *addrB* can be of a larger range than *addrA*. In this case, only the 1st *n* bits of *addrB* will be duplicates of *A* and the rest will remain unique. (*In other words, if *addrA* is 16 bits, and *addrB* is 8 bits, you can clone *addrA*[7:0] to *addrB*[7:0]. But if *addrA* is 8 bits, and *addrB* is 16 bits, you cannot clone anything into *addrB*[15:8] from *addrA*, because *addrA* doesn't contain that bit range.*)
- ❑ *addrB* does not need to exist - it can be created on the fly during *write_library*.
- ❑ Duplicating bundles is not supported.

define_group

Defines a cell group and a text description for that group. This information is used by the datasheet generator ([write_datasheet](#)), `compare_library -group`, `compare_structure` (only available in `Liberate_LV`), and `write_template` to group cells with similar functionality. The `define_group` command must occur after a [char_library](#), [read_Idb](#), or [read_library](#) command. To save the group definitions in the library database (`Idb`) a subsequent [write_Idb](#) command must be used. If `define_group` is not used, cells are grouped based on their footprint. If no footprint information is available then all cells belong to their own unique group.

The grouping information is accessible through two API functions, `ALAPI_cellgroups` and `ALAPI_description`. The API function `ALAPI_cellgroups` generates a list of groups from the library, while `ALAPI_description` returns the text description of a given cell. For more information on these API functions, see the [Virtuoso Liberate API Reference Manual](#).

Options

| | |
|------------------------------------|--|
| <code>< group ></code> | Group name. |
| <code><"description"></code> | Text describing the group. |
| <code>{ cell_names }</code> | List of cells belonging to this group. |

Example

```
# Define a group of OR gates
define_group OR2_gates "2 input OR" {OR2_1 OR2_2 OR2_4}
```

define_index

Overrides the indices specified in the templates referenced by [define_cell](#), or created using the `-auto_index` option of [char_library](#), for all the arcs between the `-related_pin` list and the `-pin` list for the given type. Valid table types are: `constraint`, `delay`, `mpw`, `power`, and `si_immunity`. The overrides apply only to the cells listed in the `define_index` command.

Options

`-index_1 {indices}` List indices to use as `index_1`

`-index_2 {indices}` List indices to use as `index_2`

`-pin {pins}` List of pin names (REQUIRED).

Note: The `-pin` option accepts the asterisk "*" wildcard. For example:

```
define_index -pin D*  # All pins beginning with "D"  
define_index -pin *  # All pins
```

`-related_pin {pins}`

List of related pin names. Accepts wildcards.

The `-related_pin` option is required for most arcs, but is not required for arcs that need only one pin, such as hidden power arcs and `mpw` or `min_period` arcs.

Note: The `-related_pin` option accepts the asterisk "*" wildcard.

`-type {data_types}` List of data types `constraint`, `ccsn_dc`, `delay`, `mpw`, `power`, or `si_immunity`.

The `ccsn_dc` template type is used for composite current source DC noise model characterization. For more information on the `ccsn_dc` template, see the [define_template](#) command.

`-when <"function">` Logic state of side inputs.

`{cell_names}` List of cell names. Accepts wildcards.

Important Points to Note

- ❑ The `-pin`, `-related_pin`, and at least one of `-index_1` or `-index_2` options must be specified.
- ❑ The size of the `-index_1` and the `-index_2` lists must be equal to the equivalent template type specified by `define_cell`. The `-when` option provides for defining unique indexes using the Liberty when syntax.
- ❑ Multiple `define_index` commands can be used to specify different overrides for different arcs for the same set of cells, or for different cells.
- ❑ The `define_index` command must follow the `define_cell` command and it must precede the `char_library` command.

Note: If with the `char_library` command you used the `-auto_index` option, the `define_index` command is applied *after* `auto_index` completes. If `define_index` is successful (correctly specified with cell/pin etc), it will override the index values determined by `-auto_index`.

Examples

```
define_template \
  -type delay \
  -index_1 {0.025 0.1 0.25} \
  -index_2 {0.0010 0.015 0.100} delay_3x3

define_cell \
  -input {A1 A2} \
  -output {Z} \
  -delay delay_3x3 {NAND2X4 NOR2X2}

# Define different output loads for A1 to Z arcs
define_index \
  -pin {Z} \
  -related_pin {A1} \
  -type delay \
  -index_2 {0.010 0.050 0.500} \
  {NAND2X4 NOR2X2}
```

define_input_waveform

Specifies a piece-wise linear waveform to drive the input during characterization.



All input transition values must be specified as PWL for a given index of slews. It is not valid to specify one index value as PWL and have the others default to another alternative input waveform method.

Options

`-direction < rise | fall >`

Specifies a logical direction of the given input waveform. The valid direction values are `rise` and `fall`. (REQUIRED)

`-gnd_val <voltage>` Specifies the input gnd voltage value of the associated input pin(s) in volts. (REQUIRED)

`-pinlist {cell pin <cell pin> ...}`

Specifies a list of cell-pin pairs to apply the waveforms. The cell and pins can be specified using wildcards. Examples are: Wildcards (*) are supported in place of explicit cell/pin names. User Inputs for a cell/pin is searched for in the following order: `cell:pin`, `cell:*`, `*:pin`, and `* : *`. If Normalized Driver Waveform (NDW) data needs to be written in the output library, use of wildcards is not recommended in the pin names. For more information about use of wildcards and NDW, see [driver_waveformWildcard_mode](#).

`-pwl {time voltage <time voltage> ...}`

List of paired values of time and voltage in MKS units. (REQUIRED)

`-scale`

Scale the normalized voltages in the PWL by vdd. (*Reserved for write_template flow - not a user option.*)

`-slew_index <slew_index>`

Specifies the index transition value that the pwl waveform is linked with. The slew from the index is specified in LDB or .lib units. (REQUIRED)

-vdd_val <voltage> Specifies the input vdd voltage value of the associated input pin(s) in volts. (REQUIRED)

This command must be used before char_library.

Example

This sample shows a template file that has slew thresholds set at 10% and 90%, and slew index values set at 0.02 and 0.555:

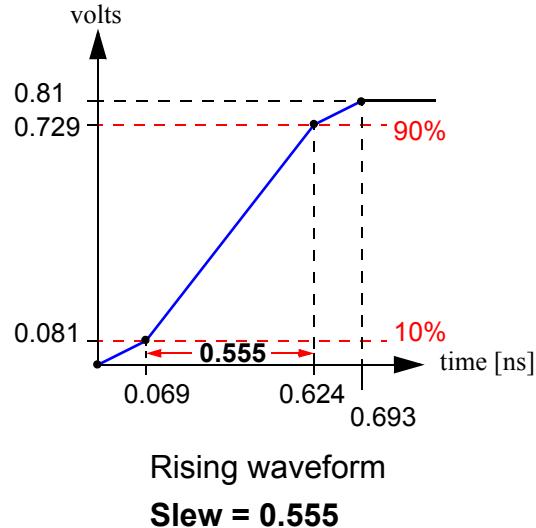
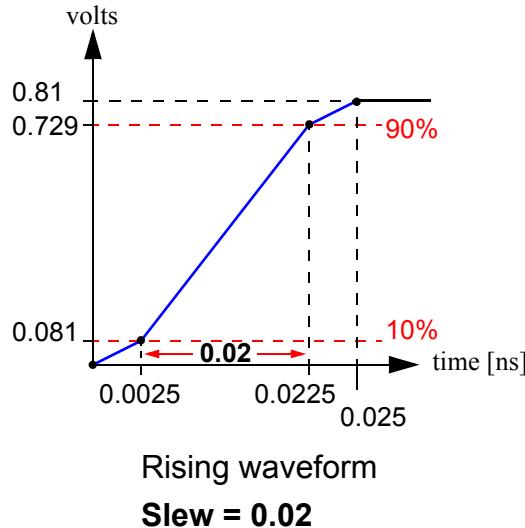
```
set_var slew_lower_rise 0.1
set_var slew_upper_rise 0.9
set_var slew_lower_fall 0.1
set_var slew_upper_fall 0.9
set_var measure_slew_lower_rise 0.1
set_var measure_slew_lower_fall 0.1
set_var measure_slew_upper_rise 0.9
set_var measure_slew_upper_fall 0.9

define_template -type delay \
    -index_1 {0.02 0.555} \
    -index_2 {0.08 0.71} \
    delay_template_2x2
```

Liberate checks that the PWL waveform supplied matches the `slew_index`. In this example, the PWL waveform must be defined so the slew time measured from the `measure_slew_lower_rise` to the `measure_slew_upper_rise` thresholds matches the `slew_index` value. If the `measure_slew_*` and the `slew_*` variables are not the same, then the slew time derived from the waveform must be derated before it is checked against the `slew_index`. The derate factor is computed as follows:

```
slew_derate = (measure_slew_upper_rise - measure_slew_lower_rise) / (slew_upper_rise
- slew_lower_rise)."
```

Both rising and falling waveforms must be defined. (*Only rising waveforms shown in this example.*)



```
# Definitions for rising PWL waveforms. Falling waveforms are similar.
define_input_waveform \
    -direction rise \
    -pwl {0.0 0.0 0.0025e-9 0.081 0.0225e-9 0.729 0.025e-9 0.81} \
    -vdd_val 0.81 \
    -gnd_val 0.0 \
    -slew_index 0.02 \
    -pinlist {INV_X1 A}

define_input_waveform \
    -direction rise \
    -pwl {0.0 0.0 0.069e-9 0.081 0.624e-9 0.729 0.693e-9 0.81} \
    -vdd_val 0.81 \
    -gnd_val 0.0 \
    -slew_index 0.555 \
    -pinlist {INV_X1 A}
```

define_leafcell

Specifies the bottom level (leaf) of the hierarchy in a cell level netlist. It is the boundary between the cell netlist and the model file. It also specifies the device type of the leaf devices so that the Inside-View algorithm can correctly identify the circuit functionality. This allows Liberate to correctly identify devices in the cell netlist even when the process model file cannot be parsed by Liberate but can be parsed by the external circuit simulator. This command can be used in combination with the `extsim_model_include` variable to enable external simulation (see `-extsim` option of the `char_library` command) with the process models and the compiled netlist. This command supports identification of mosfets, diodes, resistors, capacitors, and other device types.

Options

`-area "parameter_name"`

Specifies the name of the diode area parameter in the cell.
Default: "area"

`-element`

Enables circuit element-based leaf cells. Model name(s) must be specified.

Without this option, the `define_leafcell` command applies only to instances in the netlist. An instance in SPICE format has a name beginning with "X". When this option is enabled, the `define_leafcell` command is applied only to the circuit elements (such as R, C, M) instead of instances. This option supports circuit elements with a model, and a pin count greater than 2 (for example, 3-terminal R's and C's). Use this option when the netlist contains elements such as "M" (Mosfet), "D" (Diode), and so on.

Note: If the netlist has instances (preceding "X"), this option should not be used.

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|--|--|
| <code>-extsim_model</code> | Allows for partial <code>include</code> and partial <code>read_spice</code> . The <code>-extsim_model</code> option loads the model files for the leafcell(s). If this option is used, the leafcell being defined also needs to have <code>extsim_deck_header</code> insert a <code>".inc'<path>/modelfile.inc'"</code> to load a model for this cell – most likely a Verilog model. If one or more leafcells does not have the <code>-extsim_model</code> option nor the <code>extsim_model_include</code> variable present, the tool will output an error requesting use of the <code>extsim_model_include</code> variable/parameter and quits. |
| <code>-length "parameter_name"</code> | Specifies the name of the mos Length parameter in the cell. Default: '1' |
| <code>-multiple "parameter_name"</code> | Specifies the name of mos Multiple parameter. Default: 'm' |
| <code>-nfin "parameter_name"</code> | Specifies the name of the nfin parameter in the netlist instance (or element) that maps to the number of fingers of a FinFet process. Default: 'NFIN' |
| <code>-pin_position {list_of_pin_positions}</code> | Maps the pins of the leafcell in the netlist to the pins of the corresponding subcircuit or model in the model file. There should be only one number for each pin in the leafcell. The first pin is designated by 0. (REQUIRED) Pin positions usually start from 0 <drain gate source [bulk(s)]> <terminal_p terminal_n [bulks]> |
| <code>-pj "parameter_name"</code> | Specifies the name of the pj diode parameter in the cell. Default: 'pj' |
| <code>-scale "value"</code> | Specifies the mos parameter scale factor in the cell. This scale factor is used only by the Liberate "Inside View" to determine device sizes, and is not applied to the device sizes in the simulation netlist. Default: '1.0' |

```
-type < nmos | nmos_soi | pmos | diode | r | c | nmos_stk | pmos_stk  
| pmos_soi | npn | pnp | black_box >
```

Type of cell. For related details, see [Using the -type option](#).

```
-width "parameter_name"
```

Specifies the name of the mos Width parameter in the cell.

Default: 'w'

```
{leaf_cell_names} List of leaf cell names
```

This command must be used before `read_spice`. If the `-extsim_model` option is used with this command, it must be used before [char_library](#) and [write_library](#) as well.

If the `define_leafcell` commands in the characterization script have the `-extsim_model` option included, there are two ways to load model files for these leafcells – by using the `extsim_model_include` variable or using the `extsim_deck_header` variable. All other device models can be loaded by using `read_spice`.

Using the -type option

The `-type` option specifies the type of the cell. Supported settings are `nmos`, `pmos`, `diode`, `r`, `c`, `nmos_stk`, `pmos_stk`, `npn`, `pnp`, and `black_box`. The `nmos_stk` and `pmos_stk` types support 5 pin stacked NMOS/PMOS transistors. For 7 pin stacked MOS, the extra 2 pins are internal pins. Note that the `pin_position` for stacked MOS is: 'd g1 g2 s b'.

The `npn`, and `pnp` types identify circuit instances that are Bipolar npn or pnp transistors. These types support a `pin_position` with 3 positions (0, 1, and 2).

The `black_box` instance type supports as many pins in `-pin_position` as are required. The Liberate internal simulator, Alspice does not support BJT devices or black boxes. If the circuit has any of these devices, then an external simulator must be used with `-io` (see [char_library](#) `-extsim-io`). In addition, `extsim_model_include`, `define_leafcell`, `extsim_flatten_netlist (=0)` and `define_arc` must be used.

Note: Each type requires a minimum number of pins/element in the `pin_position` option.

The `nmos_soi` and `pmos_soi` types are used to identify Silicon on Insulator (SOI) leafcell mosfets. The supported pinout of the `nmos_soi` and `pmos_soi` related types is:

```
MXX d g s e [p] [b] [t] mname [L=val] [W=val] ...
```

where:

`MXX` is the instance name.

`d` is the drain

```
g is the gate
s is the source
e is the back gate (or substrate),
p is the external bulk
b is the bulk
t is the temperature node.
mname is the model name.
```

The drain, gate, source and back gate nodes are required. Up to 7 terminal nodes are supported.

Examples

Example 1

```
# Define the cell NCH_MAC as a leafcell
define_leafcell \
    -type pmos \
    -pin_position { 0 1 2 3 } \
    PCH_MAC

# Define the cell PCH_map as a leafcell.
# first node (gate) in netlist must be swapped with the
# second node (drain) to match drain,gate,source,bulk order
define_leafcell \
    -type pmos \
    -pin_position { 1 0 2 3 } \
    PCH_map
```

Example 2

```
set_var extsim_deck_header ".hdl /support/diode.va"
define_leafcell -extsim_model -type diode\
    -pin_position {0 1} {diodeva}
set spicefiles "netlist.sp"
lappend spicefiles "/support/sp_models.inc"
# Read in spectre netlists
read_spice -format spectre $spicefiles
```

define_leakage

Defines the logic conditions to use for calculating leakage power for the given cells. Using this command, you can disable the automatic determination of leakage conditions by Liberate for the listed cells.

Options

-extsim_deck_header

Allows to provide external simulator commands directly to the external simulator on an individual arc basis without having Liberate process or review them. This option is intended to be used when an external simulator is used (refer to the `-extsim` option of the [char_library](#) command). It is a local arc specific version of the variable `extsim_deck_header`. As Liberate does not parse the string specified by this option, ensure that the contents are valid and consistent with the arc simulation. The value string can contain the return character ("\\n"). The value string is included in the top of simulation deck. For example:

```
define_leakage -extsim_deck_header ".ic n128 0"  
-related_pin ck -pin Q ..
```

-ignore_for_default

Instructs that the specified leakage should not be considered in any calculations when computing default `leakage_power` groups or attributes. (See [set_default_group](#))

For example:

```
...  
read_ldb My.ldb.gz  
define_leakage -when "a*b" -ignore_for_default MyCell  
write_library My.lib
```

Note: This option must be set in a `define_leakage` command before the [write_library](#) command.

-load_dir <U | D | B>

Output load direction (pullUp, pullDown or Both).

-pinlist {list_of_pins}

List of pins corresponding to vector.

`-pin_load <template>`

Predefined pin loading. (See [define_pin_load](#))

`-prevector <"vector">`

User-specified initialization vectors. Default: no prevectors

`-prevector_pinlist {list}`

User-specified pin list for pre-vectors. For related details, see [Using the -prevector option](#).

`-when <"function">` Specifies the logic conditions using the Liberty *when* format syntax. (REQUIRED)

This option should be used when characterizing I/O cells that do not use automatic vector generation (`char_library -io`).

Note: Enhanced leakage characterization is also achievable by the tool honoring the `-when` condition specified in the [define_cell](#) command.

`-vector <"stimulus">`

Vector stimulus used to simulate the leakage, where each bit can be one of the following: X, 1, or 0.

The `-vector` option allows for partial `when` conditions for leakage in the output library, while having secondary pins set to a mix of 0s and 1s.

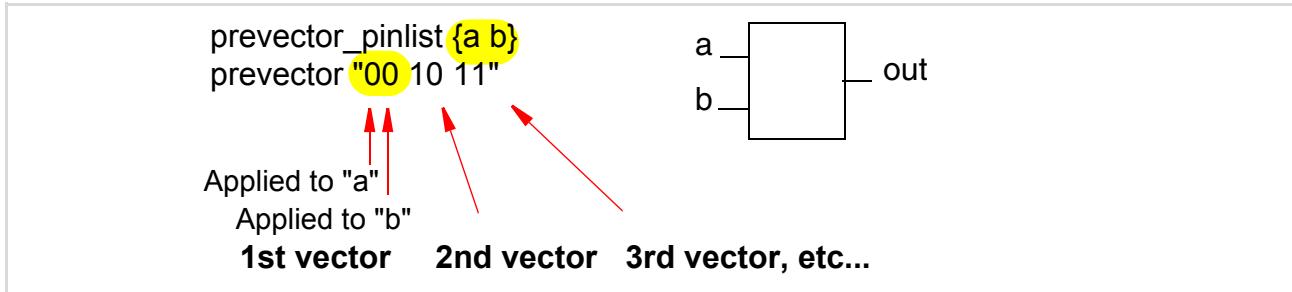
`{cell_names}` List of cell names.

This command must be used before `read_spice`, [char_library](#), and [write_library](#).

Using the -prevector option

The `-prevector` option specifies an arbitrary simulation stimulus to be applied before the actual vector where the characterization will be measured. Pre-vectors are used to put a device in a user-defined state before proceeding with characterization. Prevector values must be 0 or 1. See related variables: [prevector_period](#), and [prevector_slew](#).

This option must be used in conjunction with `prevector_pinlist`. The following illustrates how prevectors are specified with `prevector_pinlist`:



 **Important**

1. The last state of the prevector must match the beginning state of the vector for a given pin, otherwise an unintended transition will occur.
2. When `prevector` is specified, Liberate disables Inside View for this arc and operates as if it is in `io` mode for this arc. In `io` mode, `probe` is required for all constraint arcs, including `mpw`. (See [char library](#))

Example 1:

```
# Define the IOCELL
define_cell \
  -input { D } \
  -output { Q } \
  -clock { CK } \
  -pinlist { D CK Q } \
  -delay delay_template_3x3 \
  -power power_template_3x3 \
  MYCELL

define_leakage \
  -prevector_pinlist {D CK} \
  -prevector "00 01 00 10" \
  -vector      "1RR" \
  -related_pin CK \
  -pin Q \
  MYCELL
```

Example 2:

```
# constraint arcs from CK => CK using prevector min_pulse_width
```

```
define_leakage \
  -type min_pulse_width \
  -prevector_pinlist {D CK Q} \
-prevector {100 111 001} \
-vector {0RF} \
  -related_pin CK \
  -pin CK \
  DFF

# constraint arcs from CK => CK using prevector min_pulse_width
define_leakage \
  -type min_pulse_width \
  -prevector_pinlist {D CK Q} \
-prevector {100 111 011} \
-vector {0FF} \
  -related_pin CK \
  -pin CK \
  DFF
```

Examples

```
# Define the leakage conditions of an IO cell
define_leakage -when " D * !EN"           IO_cell
define_leakage -when " !D * !EN"           IO_cell
define_leakage -when " D *  EN *  PAD"    IO_cell
define_leakage -when " D *  EN * !PAD"   IO_cell
define_leakage -when " !D *  EN *  PAD"   IO_cell
define_leakage -when " !D *  EN * !PAD"  IO_cell
```

define_map

Defines a file for mapping cell names prior to writing out the template, library, Verilog, VITAL, or datasheet files. It can also be used to map cell names when doing a library comparison using the [**compare library**](#) command. It also changes the cells name(s) returned by the API functions: ALAPI_inputs, ALAPI_outputs, ALAPI_inouts, ALAPI_internals, ALAPI_clocks, ALAPI_pinnames, ALAPI_name, ALAPI_cellnames, and ALAPI_cellgroups.

Options

<map_filename> Name of the map file

If the specified file contains only cell name mapping, this command can be used before model generation (see [write_library](#), [write_verilog](#), and [write_vital](#)) and before the [**write_template**](#) command. However, if the specified file contains pin mapping, it must be used before the [read_lDb](#) and [read_library](#) commands.

The specified file should contain separate lines of one of the following formats:

- *<original_cell_name> <new_cell_name>*
- *<original_cell_name:pin_name> <new_pin_name>*

Example

Define a mapping file before writing the library

```
read_lDb      my.ldb.gz
define_map   my_cell.map
write_library my_mapped.lib
```

The map_file would contain the following information:

```
cell_1      cell1_new
cell_1:ck  CLK
```

Liberate maps the cell named `cell1` to `cell1_new` and the pin named `ck` in `cell_1` to `CLK`.

define_max_capacitance_attr_limit

Sets a pin-specific maximum capacitance attribute limit.

Options

| | |
|------------------------------|---|
| <code>-cells {list}</code> | List of cells. |
| <code>-pinlist {list}</code> | List of pins. |
| <code>< limit ></code> | The <code>max_capacitance</code> attribute limit value (in Farads). |

The values set by `define_max_capacitance_attr_limit` override the calculated `max_capacitance` attribute when the limit is exceeded. Multiple `define_max_capacitance_attr_limit` commands can be specified. A wildcard * is supported to allow the cell name to reference all cells. Only cells with the given pin name(s) will be effected. A wildcard cannot be used for the pin.

This command must be used before model generation ([write_library](#)).

Example

```
# Set the max capacitance for pin Y of the cell AND1
define_max_capacitance_attr_limit \
    -cell AND -pinlist Y 100e-15
```

define_max_capacitance_limit

Sets a pin-specific maximum capacitance.

Options

<value> Maximum allowable capacitance (in Farads).
{<cell> <pin> ...} List of cell pin pairs

This command has effect only when you use the `-auto_index` option with the `char_library` command. The values set by `define_max_capacitance_limit` override the calculated `max_capacitance` when the limit is exceeded. Multiple `define_max_capacitance_limit` commands can be specified. A wildcard `*` is supported to allow the cell name to reference all cells. Only cells with the given pin name will be effected. A wildcard cannot be used for the pin.

This command must be used before `char_library`.

Example

```
# Set the max capacitance for pin Y of the cell AND1
define_max_capacitance_limit 100e-15 { AND Y }
```

define_max_transition

Sets a pin-specific maximum transition.

Options

<value> Maximum allowable transition time (in seconds).

{*<cell> <pin> ...*} List of cell/pin pairs.

This command has effect only when using the `-auto_index` option with the `char_library` command. The values set by `define_max_transition` override the global value set by the `max_transition` parameter. Multiple `define_max_transition` commands can be specified. The wildcard asterisk (*) is supported to allow the cell name to reference all cells. Only cells with the given pin name will be effected. A wildcard cannot be used for the pin.



If this command is used with the VDB flow, care should be taken to not apply double-scaling. For example, the write_vdb command should not use -auto_index. Instead, your characterization script (i.e.: char.tcl) should have a read_vdb -> char_library -auto_index flow. If both scripts use -auto_index then the resulting library will have max_transition scaling applied twice.

This command must be used before `char_library`.

Example

```
# Set the default maximum transition time
set_var max_transition 1e-9
# Set maximum transition time for some clock pins
define_max_transition 0.5e-9 {DFFX1 CK LTX1 LCK}
define_max_transition 0.75e-9 {GATER CLKIN * CLK}
char_library -auto_index
```

define_min_transition

Sets a pin-specific minimum transition.

Options

`<value>` Minimum allowable transition time (in seconds).

`{<cell> <pin> ...}` List of cell/pin pairs.

This command has effect only when you use the `-auto_index` option with the `char_library` command. The values set by `define_min_transition` override the global value set by the `min_transition` variable. Multiple `define_min_transition` commands can be specified. A wildcard `*` is supported to allow the cell name to reference all cells. Only cells with the given pin name will be effected. A wildcard cannot be used for the pin.



If this command is used with the VDB flow, care should be taken to not apply double-scaling. For example, the write_vdb command should not use -auto_index. Instead, your characterization script (that is, char.tcl) should have a read_vdb -> char_library -auto_index flow. If both scripts use -auto_index then the resulting library will have min_transition scaling applied twice.

This command must be used before `char_library`.

define_out_to_out_arc

Instructs Liberate to characterize an arc from an output pin to an output pin. This command will merge a characterized arc A->X with arc A->Y into arc X->Y.

Options

| | |
|---------------------------------------|--|
| <code>-related <pin></code> | The related_pin for both forward arcs. This option specifies the name of the input pin whose transition triggers the arc. |
| <code>-related_out <pin></code> | The intermediate transitioning output. This option specifies the first output to switch and becomes the <code>related_pin</code> for the out-to-out arc. |
| <code>-out <pin></code> | The last transitioning output. This option specifies the output pin for both forward arcs and is the pin in the output library. |
| <code>-cells {list_of_cells}</code> | List of cells. |

This command must be used before [write_library](#).

Example

```
define_out_to_out_arc -related A -related_out X -out Y -cells { mycell }
```

define_pin_load

Defines additional loading that can be applied to a particular pin prior to the output load.

Options

-pullup_voltage <value>

Pullup voltage.

-pulldown_voltage <value>

Pulldown voltage.

-pullup_resistance <value>

Pullup load resistance in ohms.

-pulldown_resistance <value>

Pulldown load resistance in ohms.

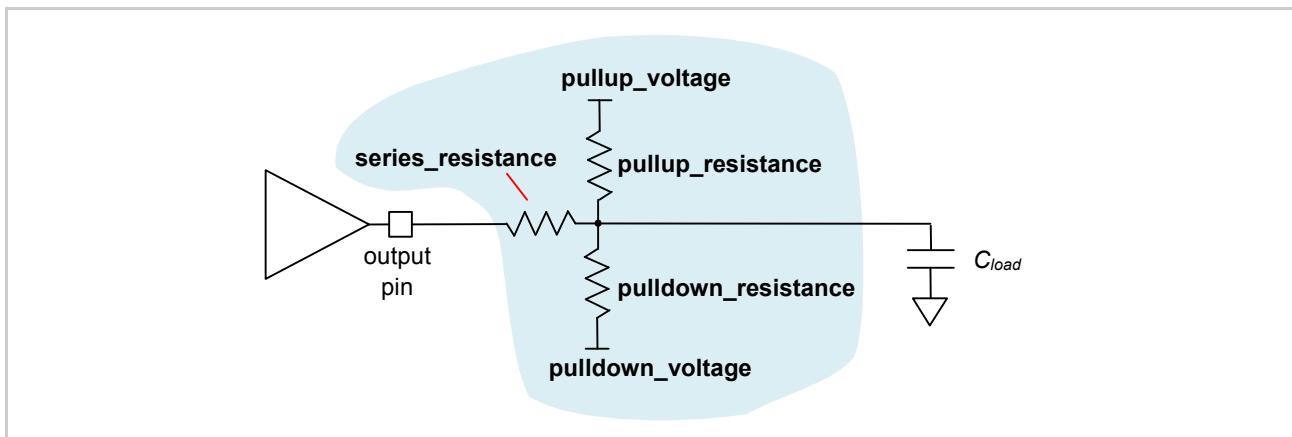
-series_resistance <value>

Series load resistance in ohms.

<pin_load_name>

Name of pin load definition. (REQUIRED)

The loading of the pin can consist of a pullup voltage source via a pullup resistance, a series resistance and a pulldown resistance tied to a pulldown voltage source. Resistances must be specified in ohms.



The load definition can be referenced by the define_arc command to specify additional loading to be applied to a specific arc. This can be particularly useful for characterizing I/O cells.

This command must be used before char_library.

Example

```
# Define additional loading for a pin
define_pin_load \
    -pullup_voltage 3.3 \
    -pullup_resistance 4000 \
    -pulldown_resistance 4000 \
    -series_resistance 25 \
    pin_load_template
```

define_pulse_generator_arc

Identifies arcs that generate a pulse on an output pin.

Options

-cells {cell_names}

Specify a required list of cells. The cell name accepts wildcard.
Default: " " (none specified)

-pin <pin_name> Specify the pin name (required). Default: " " (none specified)

-related_pin <pin_name>}

Specify the related pin name (required). Default: " " (none specified)

-when <"function"> Specify the logic condition of the side inputs. Default: " " (none specified)

This command can be executed multiple times, once for each pulse generator arc. For the specified arcs Liberate will subtract CV^2 from the fall power. Then, divide the fall power by two and copy the fall power into the rise power. This is done because a single transition on the input generates two transitions on the output, one for the falling arc power and one for the rising arc power and we do not want the downstream power tool to double count the power.

If the **-when** option is not specified, this command is applied to all arcs that match the specified pin and **related_pin**. If **-when** is specified, this command is only applied to those arcs that exactly match the specified **when** condition. An exact match occurs when all pins are present and are in the same state. The order of the pins is not considered.

This command must be used before char_library.

define_template

Defines a template to be used for characterization.

Options

`-type {delay | power | ccs | ccsn_dc | constraint | ecsm | mpw | si_iv_curve | si_immunity }`

Specifies the type of template being defined. (REQUIRED)

`-index_1 {values}` List values to be used as the first index. (REQUIRED)

`-index_2 {values}` List values to be used as the second index.

`-index_3 {values}` List values to be used as the third index.

`<template>` Name of template.

Note: All `-index_*` options for all the library constructs should be monotonically increasing.

This command must be used before char_library.

Using the -type option

The `-type` option specifies the type of template being defined. How each cell is to be characterized is defined by associating the defined template with the appropriate option of the define_cell command. Multiple `define_cell` commands can reference a single template.

Note: Internally, `define_template` uses a fixed set of units (listed below). These cannot be changed. *However*, units can be changed when writing a library with the set_units command.

current: 1mA (milliamps)

power: 1nW (nano watts)

resistance: 1kohm (kilohm)

time: 1ns (nano seconds)

voltage: 1V (volts)

capacitance: 1pf (pico farad)

Virtuoso Liberate Reference Manual

Liberate Commands

The `delay` template type is used for delay characterization using input slew and output load. It requires both `index_1` and `index_2` to be specified, where `index_1` represents the range of input slews and `index_2` represents the range of output loads.

The `power` template type is used for switching and hidden (internal) power characterization using input slew and output load. It requires both `index_1` and `index_2` to be specified, where `index_1` represents the range of input slews and `index_2` represents the range of output loads.

The `ccs` template type can be used for composite current source model (CCS) delay characterization. It requires `index_1` to be specified where `index_1` represents the range of the normalized voltage values to measure.

The `ccsn_dc` template type can be used for composite current source DC noise model characterization. It requires `index_1` and `index_2` to be specified where they represent a range of input/output voltages. If not specified Liberate uses a range of 29 voltage points from -Vdd to 2*Vdd. DC simulations are very fast, especially on a small CCB group of transistors extracted for noise stage simulations and therefore does not use much CPU time compared to the transient CCS noise models. It usually is not necessary to change the size of these tables from the default 29x29. It can be useful to optimize the size of the tables to avoid non-convergence errors at the extremes of the voltage range. The `ccsn_dc` template is global to all cells and is not included in the `define_cell` command.

The `constraint` template type can be used for timing constraint (setup, hold, removal, recovery) characterization. It requires both `index_1` and `index_2` to be specified, where `index_1` represents the range of input slews of the data signal and `index_2` represents the range of input slews of the reference signal (clock, reset etc.).

The `ecsm` template type can be used for effective current source model (ECSM) characterization. It requires `index_1` to be specified, where `index_1` represents the range of the normalized voltage values to measure.

The `mpw` template type is used when a two dimensional mpw table is required. The user must provide both `index_1` and `index_2`. In addition, the `define_cell -mpw` option must reference the `mpw` template. By default, if you do not provide an `mpw` template, the `mpw` timing constraint will be characterized as a single attribute. If the `mpw_table` variable is set to a variable, Liberate will output a one dimensional mpw table using the `define_cell` constraint variable list of values. Use the `define_template -type mpw` command only when a two-dimensional MPW table is required.

The `si_iv_curve` template type is used for steady state I/V characterization. It requires `index_1` to be specified, where `index_1` defines the number of sample voltage points between supply and ground for steady state high and between ground and supply for steady state low. Each sample point is equally spaced within the total voltage range.

The `si_immunity` template type is used for noise immunity rejection curve characterization. It requires both `index_1` and `index_2` to be specified, where `index_1` represents the range of input noise widths and `index_2` represents the range of output loads.

Examples

```
# Delay template for 3 input slews, 3 output loads
define_template -type delay \
-index_1 {0.025 0.1 0.25} \
-index_2 {0.0010 0.015 0.100} \
delay_3x3

# Power Template for 3 input slews, 3 output loads
define_template -type power \
-index_1 {0.025 0.1 0.25} \
-index_2 {0.0010 0.015 0.100} \
power_3x3

# Timing constraint template for 2 input slews
define_template -type constraint \
-index_1 {0.025 0.25} \
-index_2 {0.025 0.25} \
constraint_2x2

# ECSM template for 5 intervals
define_template -type ecssm \
-index_1 {0.05 0.2 0.5 0.8 .95} \
ecsm_5

# si_iv_curve template for 11 intervals
define_template -type si_iv_curve \
-index_1 {0 1 2 3 4 5 6 7 8 9 10} \
Si_iv_11

# si_immunity template for 3 noise widths, 3 loads
define_template -type si_immunity \
-index_1 {0.100 0.50 3.00} \
-index_2 {0.0010 0.015 0.100} \
Si_immunity_3x3

# CCS Noise DC Curve with 11 input and 11 output voltages.
define_template -type ccsn_dc \
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
-index_1 {-1.0 -0.5 -0.2 -0.1 0.0 \
0.1 0.2 0.5 1.0 1.2 1.5} \
-index_2 {-1.0 -0.5 -0.2 -0.1 0.0 \
0.1 0.2 0.5 1.0 1.2 1.5} \
ccsn_dc_template
```

delete_arc

Deletes one or more arcs from a library.

Options

| | |
|-------------------------------------|---|
| <code>-cell {cell_name}</code> | Specifies the cell name from which the arc is to be deleted. (Required) |
| <code>-pin "name"</code> | Specifies the pin of the arc to be deleted. (Required) |
| <code>-related "name"</code> | Specifies the related_pin of the arc to be deleted. Default: "*" (delete all arcs of the given type regardless of the related pin) |
| <code>-timing_sense "string"</code> | Specifies the 'timing_sense' of the arcs to be deleted. Default: " " (delete all arcs of the given type regardless of the timing_sense) |
| <code>-timing_types {list}</code> | Specifies a list of the 'timing_type'(s) of the arcs to be deleted. Default: {} (delete all arcs of the given type regardless of 'timing_type') |
| <code>-type {list}</code> | Supported types are: ccs, ccs_retain, constraint, delay, hold, mpw, power, recovery, removal, retain, setup, timing. Default: delete all data types. |
| <code>-when "value"</code> | Specifies the when of arc to be deleted, Default: "*" (delete all arcs of the given type regardless of the when condition) |

Use this command to specify the arcs to be deleted from the output library. This command does not affect characterization. It operates in the programming interface (ALAPI) level post characterization. Therefore, any function or command that uses ALAPI respects this command. Any function or command written in C language does not respect this command. Some commands, such as `write_ldb`, `write_socv` and `write_variation_table`, do not support `delete_arc`.

Example

```
delete_arc \
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
-pin {CEN} \
-type {setup hold} \
-cell cell_name
write_library -filename my.lib  myLib
```

This command must be used before model creation.

em_buffer_cell_pin_bounds

Specifies a list of cells to buffer (see [em_window_estimate_mode](#)) that are known to have failed window estimation before.

Options

| | |
|----------------------|--|
| -cells | Specifies a list of the cells to buffer. Wildcards are supported. Default: * (all cells) |
| -pins | Specifies a list of the pins in the cells to buffer. Wildcards are accepted. Default: * (all pins) |
| -indices (-1.0 -1.0) | Specifies a list of the pairs of slew (nS) and load (pF) to buffer. |

This command must be specified before [char_library](#).

Example

```
em_buffer_cell_pin_bounds  \
  -cells {SDFQOPTKS5D4BWP360H10P66PDP2ULVT} \
  -pins {Q} \
  -indices {0.0001 0.000002}
```

em_follow_hidden_power

Directs Liberate to characterize electromigration (EM) models for the specified pins. By default, Liberate does not characterize EM data for input pins.

Options

| | |
|----------------------------|---|
| <code>-cells {list}</code> | List of cell names. Default: * (all cells) |
| <code>-pins {list}</code> | List of pin names. Default: * (all cell pins) |

Use this command to specify input pins to characterize for EM using hidden power arcs. For this command to have any effect, EM characterization must be enabled (see the `-em` option of the [char_library](#) command and the [Electromigration Models](#) section of the Liberate Details chapter in this manual).

This command must be used prior to `char_library`.

get_cells

Returns the list of cells in a session.

Options

-type <netlist | leafcell | template | char | model | all>

Specifies the type of cells to be filtered and returned. Valid types include: netlist, leafcell, template, char, model, and all. Default: all

List of cells for which a netlist has been read through `read_spice`. Returns an empty list prior to `read_spice`. If multiple `read_spice` commands are run, returns a superset of all netlist subckts read. This can include subckts that are hierarchical and will not be characterized.

netlist List of cells for which a netlist has been read through `read_spice`. Returns an empty list prior to `read_spice`. If multiple `read_spice` commands are run, returns a superset of all netlist subckts read. This can include subckts that are hierarchical and will not be characterized.

leafcell List of cells marked as leaf cells. The list includes cells identified as leaf cells using the `define_leafcell` command and auto-generated leaf cells during netlist parsing. In order to obtain "auto-generated" leaf cells when using Variety, all `read_spice` commands must precede the call to the `get_cells` command. For all other programs, the `get_cells` command must follow `char_library` (Liberate) and `char_macro` (Liberate MX).

template List of cells from all `define_cell` commands that have a template (`define_cell` with a template), but ignores the driver-only cells (no template).

Virtuoso Liberate Reference Manual

Liberate Commands

| | | |
|---------------------------------|--------|--|
| | char | List of cells that were characterized. Returns an empty list before the <code>char_library</code> command. Includes all cells that were read using <code>read_lDb</code> and all cells that are characterized in the current session. |
| | model | List of cells that were written into a <code>.lib</code> file in the last <code>write_library</code> command. |
| | all | List of cells including all of above (default). |
| -status <passed failed all> | | Specifies the status of the cells to be filtered and returned. Valid status include: <code>passed</code> , <code>failed</code> , and <code>all</code> . Default: <code>all</code> |
| | passed | List of cells that were preprocessed successfully. |
| | failed | List of cells that were not preprocessed successfully. |
| | all | List of all cells. Do not filter based on passed/failed status. (Default) |
| <regexp> | | Specifies the pattern to be used when matching cell name. Pattern is any regexp pattern. Default: no pattern, match all cells. |

Examples

#Example 1:

```
set cell_all [get_cells]
puts "$cell_all"
```

Output:

```
DFFX1 INVX1 NOR2X1
```

#Example 2:

```
set cell_netlist [get_cells -type netlist]
puts "$cell_netlist"
```

Output:

```
DFFX1 INVX1 NOR2X1
```

#Example 3:

```
set cell_template [get_cells -type template]
puts "$cell_template"
```

Virtuoso Liberate Reference Manual

Liberate Commands

Output:

DFFX1 NOR2X1

get_var

Returns the current value of a Liberate variable, either the default value or the value set using [set_var](#). A list of Liberate variables can be generated using the [printvars](#) command.

Options

<variable_name> Liberate variable name

Example

```
# Get the value of default_timing
get_var default_timing
```

ibis_define_component

Provides a means for specifying the component definitions that are used for Liberate IBIS characterization.

Options

-harness_pins {values}

Specifies IBIS pins that have a harness attached.

Note: The harness can be used to specify a special pull-up/pull-down resistance/voltage setup for differential output pins.

-manufacturer "value"

Specifies the name of the component's manufacturer in the IBIS output file.

-net_typ_vdd {values}

Specifies a list of pairs of nets and typical vdd values in the following format:{net typ_vdd}

-net_typ_vss {values}

Specifies a list of pairs of nets and typical vss values in the following format:{net typ_vss} pairs. For example, if you are using an IBIS truth table and use the SET_NET_GND command in the truth table to specify the gnd/vss values for the IBIS pin, such as:

SET_NET_GND={ PAD: 0, 0, 0 PADN: 0, 0, 0 }

then, the following entry is automatically generated:

-net_type_vss { PAD 0 PADN 0 }

-package_C {values}

A list of package C condition. This option requires a list with three values representing the typical, minimum, and maximum values in that order.

-package_L {values}

A list of package L condition. This option requires a list with three values representing the typical, minimum, and maximum values in that order.

-package_R {values}

A list of package R condition. This option requires a list with three values representing the typical, minimum, and maximum values in that order.

-sim_fall "value" Simulation transition fall time in seconds. Default: 1n

-sim_rise "value" Simulation transition rise time in seconds. Default: 1n

-waveforms {values}

Specifies a list of pull-up/pull-down waveform templates. The templates must be defined by the [ibis_define_component](#) command.

<components> Specifies a list of IBIS components this command will apply to.

This command must be used before [char_library](#).

Example:

```
ibis_define_component \
  -manufacturer "Vendor" \
  -package_R { 0 0 0 } \
  -package_L { 0 0 0 } \
  -package_C { 0 0 0 } \
  -waveforms $wf_templates \
  -net_typ_vdd {A 1.1 PAD 3.3} \
  -net_typ_vss {A 0.0 PAD 0.0} \
  {IO_cell1_1 IO_cell1_2}
```

ibis_define_header

Provides a means for specifying the content of each section of the IBIS header at the top of the IBIS model file.

Options

```
-comment_char <value>           IBIS comment character. Default: '|'  
-copyright <value>             User-specified copyright text.  
-disclaimer <value>            User-specified disclaimer text.  
-file_rev <value>              IBIS file version. Default: '0.1'  
-filename <value>              IBIS output file name.  
-ibis_ver <value>              IBIS version. Default: '4.2'  
-notes <value>                User-specified notes.  
-source <value>                User-specified source description.  
<cellname>                   IBIS cell name.
```

This command must be used before [char_library](#).

Example

```
ibis_define_header -ibis_ver 4.2 \  
    -source "Vendor" \  
    -disclaimer "Property of Vendor. IBIS models are provided as a service to  
    our customers. No liability is assumed for accuracy or proper functioning in  
    any application." \  
    -copyright "Property of Vendor. Unauthorized reproduction and/or  
    distribution is strictly prohibited. Copyright (C) 2008, Vendor, all rights  
    reserved." \  
    -comment_char "|" \  
    -file_rev "1.0" \  
    IO_cell_1.ibs
```

ibis_define_model

Defines global parameters that are used to acquire and write model data. This command is required for each IBIS model characterization setup.

Options

| | |
|--------------------------------|--|
| <code>-cfix</code> | Rising waveform C_fixture. |
| <code>-component</code> | This model is defined for these component(s) list. |
| <code>-cref</code> | Timing specification test load capacitance. |
| <code>-cref_diff</code> | Timing specification differential capacitance. |
| <code>-enable</code> | "Active-High" or "Active-Low". |
| <code>-lfix</code> | Rising waveform L_fixture. |
| <code>-model_type</code> | Pin model type: Input, Output, I/O, 3-state. |
| <code>-polarity</code> | "Non-Inverting" or "Inverting". |
| <code>-r_load</code> | Loading resistor for [Ramp] measurement. |
| <code>-rfix</code> | Rising waveform R_fixture. |
| <code>-rref</code> | Timing specification test load resistance. |
| <code>-rref_diff</code> | Timing specification differential resistance. |
| <code>-vfix</code> | Rising waveform V_fixture. |
| <code>-vfix_max</code> | Rising waveform V_fixture_max. |
| <code>-vfix_min</code> | Rising waveform V_fixture_min. |
| <code>-vinh</code> | Minimum upper threshold voltage. |
| <code>-vinl</code> | Maximum lower threshold voltage. |
| <code>-vmeas</code> | Reference voltage for timing measurements. |
| <code>-vref</code> | Timing specification test load voltage. |
| <code>-when</code> | When condition to match waveform's "when" logic. |
| <code><modelName></code> | IBIS model name. |

This command must be used before [char_library](#).

Example

```
set cells {IO_cell_1}
# When 0.8V and 2.00V are measurement thresholds, set Vinl=0.8V and Vinh=2.00V.
ibis_define_model \
    -model_type 3-state \
    -polarity Non-Inverting \
    -enable Active-Low \
    -vinl 0.8 \
    -vinh 2.0 \
    -vmeas 1.65 \
    -vref 0 \
    -rref 100.00M \
    -cref 15p \
    -component $cells \
    IO_cell_1_model
```

ibis_define_model_selector

Specifies the set of user-defined model types should be included in a specific IBIS model.

Options

| | |
|-------------|--|
| -component | This model is defined for these component(s) list. |
| -model_desc | List of model and description in pair(s). |
| <name> | Model selector name. |

This command must be used before [char_library](#).

Example

```
set cells {MyOBuff}

set mslist { \
    LOW_VOLT_PUPD      "Low Voltage Pull_up enabled" \
    LOW_VOLT_!PUPD     "Low Voltage Pull_down enabled" \
}
ibis_define_model_selector \
    -model_desc $mslist \
    -component $cells \
    LVDS_OUT
```

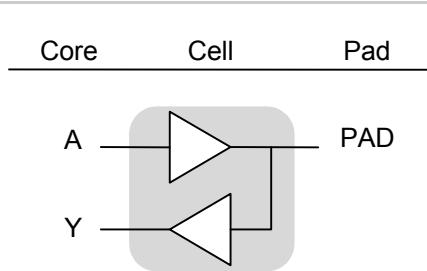
ibis_define_pin

Defines the pins that are required to generate electromigration models.

Options

| | |
|-------------------------|---|
| -net | Pin net/signal name from Spice netlist. |
| -model | Pin IBIS model name defined in the <code>ibis_define_model</code> command. |
| -enable | Enable pin signal (port) name from Spice netlist. |
| -function | 'when' condition to set pin to logic 1 (high) state. |
| -inv_pin | The corresponding inverting pin name for I/O output. |
| -inv_pin_net | The inverting pin signal (port) name. |
| -vdiff | The differential receiver threshold voltage from pin to <code>inv_pin</code> . |
| -tdelay_typ | The <code>typ</code> launch delays of the <code>inv_pin</code> relative to the pin. |
| -tdelay_min | The <code>min</code> launch delays of the <code>inv_pin</code> relative to the pin. |
| -tdelay_max | The <code>max</code> launch delays of the <code>inv_pin</code> relative to the pin. |
| -R_pin | Pin package R value. |
| -L_pin | Pin package L value. |
| -C_pin | Pin package C value. |
| -components | IBIS component name(s) list which use this pin definition. |
| -cinpin | Specify the core-input-pin related to this pad pin: core input pin (cell output pin) signal (port) name For example: pin Y of PAD->Y arc |
| -coutpin | Specify the core-output-pin related to this pad pin: core output pin (cell input pin) signal (port) name For example: pin A of A->PAD arc |
| <pin_name> | Pin name. |

Consider the following:



If PAD is a cell output pin and with arc A->PAD, then A is a core-output-pin (cell input pin). The signal goes from core to IO cell through pin A and then passes to the PAD pin. If PAD is the cell input pin with arc PAD->Y, then Y is the core-input-pin (cell output pin). The signal goes from PAD to (through IO cell) pin Y and then passes into core.

This command must be used before char library.

Examples

```
ibis_define_pin \
  -net VDDS \
  -model POWER \
  -component $cells \
  VDDS
```

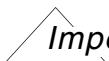
```
ibis_define_pin \
  -net PAD \
  -model IO_cell_1 \
  -enable "GZ" \
  -function "A" \
  -component $cells \
  PAD
```

ibis_define_waveform_template

Defines the waveforms used by the **ibis_define_component** command.

Options

```
-type <R | F>           Output waveform direction
-pull_dir <U | D | B>
                         Resistive load connected direction.
-vfix <value>          Voltage of the fixture
-rfix <value>          Resistance of the fixture
-lfix <value>           Inductance of the fixture
-cfix <value>           Capacitance of the fixture
-pins {list_of_pins}
                         List of pins to apply this waveform.
-nets {list_of_nets}
                         Specifies a list of net names of the corresponding IBIS pin
                         names.
name
                         Waveform template name. Suggested names are:
                         pullup_on_data, pullup_off_data,
                         pulldown_on_data, and pulldown_off_data.
```

 *Important*

If the **-type** and **-pull_dir** options are not specified, the waveform template name has to be one of: **pullup_on_data**, **pullup_off_data**, **pulldown_on_data**, or **pulldown_off_data**.

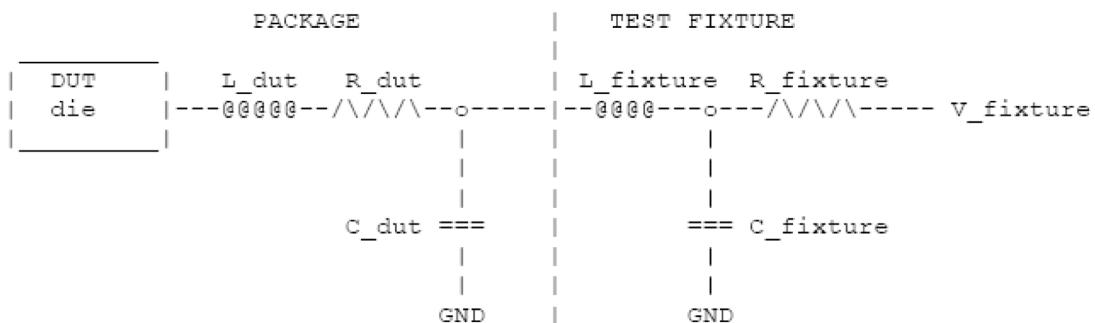
This command must be used before **char_library**.

Virtuoso Liberate Reference Manual

Liberate Commands

Here are details of the fixture parameters, taken from the IBIS 4.2 specification:

The 'fixture' subparameters specify the loading conditions under which the waveform is taken. The `R_dut`, `C_dut`, and `L_dut` subparameters are analogous to the package parameters `R_pkg`, `C_pkg`, and `L_pkg` and are used if the waveform includes the effects of pin inductance/capacitance. The diagram below shows the interconnection of these elements.



NOTE: The use of L_dut, R_dut, and C_dut is strongly discouraged in developing waveform data from simulation models. Some simulators may ignore these parameters because they may introduce numerical time constant artifacts.

Only the `R_fixture` and `V_fixture` subparameters are required, the rest of the subparameters are optional. If a subparameter is not used, its value defaults to zero. The subparameters must appear in the text after the keyword and before the first row of the waveform table.

Example

Here, two waveform templates are defined, and both are used by the `ibis define component` command.

The `vfix` value should be set differently for individual corners.

```
# In the typ corner (char_typ.tcl), you should have:  
set PULLUP_V_PAD 3.3  
  
# In the min corner (char_min.tcl), you should have:  
set PULLUP_V_PAD 3.0  
  
# In the max corner (char_max.tcl), you should have:  
set PULLUP_V_PAD 3.6
```

The `template_ibis.tcl` that contains all IBIS related commands (like the following `ibis define waveform template`) is sourced by all char scripts.

```
# pulldown_on_data
ibis define waveform template \
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
-type F \
-pull_dir U \
-vfix $PULLUP_V_PAD \
-rfix 50 \
-pins "P1" \
-nets "PAD" \
waveform_PAD_F_U_50

ibis_define_waveform_template \
-type F \
-pull_dir D \
-vfix $PULLDN_V_DOUT \
-rfix 50 \
-pins "P3" \
-nets "DOUT" \
waveform_DOUT_F_D_50

ibis_define_waveform_template \
-type R \
-pull_dir D \
-vfix $PULLDN_V_PAD \
-rfix 50 \
-pins "P2" \
-nets "PAD" \
waveform_PAD_R_D_50

set wf_templates {
    waveform_DOUT_R_D_50    <not shown above>
    waveform_DOUT_F_U_50    <not shown above>
    waveform_DOUT_R_U_50    <not shown above>
    waveform_DOUT_F_D_50
    waveform_PAD_R_D_50
    waveform_PAD_F_U_50    <not shown above>
    waveform_PAD_R_U_50    <not shown above>
    waveform_PAD_F_D_50    <not shown above>
}

ibis_define_component \
-manufacture "Cadence Design Systems, Inc." \
-package_R { 0.24 221m 0.35 } \
-package_L { 2.1nH 4.2nH 3.2nH } \
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
-package_C { 2.4pF 2.1pF 2.2pF } \
-waveforms $wf_templates \
-sim_rise 1.1 \
-sim_fall 1.1 \
-net_typ_vdd { P3 1.32 P2 3.6 } \
$ibisCellName
```

merge_library

Takes two (or more) libraries and merges them to make a new library.



Merging libraries should be performed as a separate Liberate run.

Options

-arcs Specifies that arc data should also be merged. By default, all library, cell, and pin data is merged.

-cells {cell_names}
Specifies a list of cells from the `from_lib` which should be merged into the `to_lib`. Default: all the cells in the `from_lib` should be merged.

Note: This option supports the use of a wildcard. If `-exclude` is used, the cells specified in the `-cells` list will be excluded and all the other cells included in the merge process.

-driver_waveform Preserves the driver waveform associations present in the “from” library.

Note: The `merge_library` command does not preserve any driver waveforms in the "to" library. By default, all references to driver waveforms are dropped from the output library.

-exclude Controls whether to exclude the cells specified with the **-cells** option from the second library.

Note: This option supports the use of a wildcard.

-filename <filename>

Output file name of merged library. Default: <to_lib>.m

-indent <number> Specifies the number of characters (spaces) to use for indentation in the merged library.

-libname <lib_name>

Specifies the name of the merged (resultant) library.

-method {method data_type}

Specifies the method for merging data into a library. Default: "append" (Append data missing from the target library.)

By default, `merge_library` appends only data that is missing in the first library from the second library. With the `-method` option, you can select the `min`, `max`, `avg` (average), `append`, and `sum` of data for any arcs that are present in both libraries to create the merged library. For example, "max" or "delay avg hold max".

The options are specified as a list of method and type pairs. Legal data types for this method of merging are `delay`, `retain`, `power`, `setup`, `hold`, `recovery`, and `removal`.

The following example shows how to merge two libraries and use the `max` data for the hold time:

```
merge_library -method "max hold" -filename mlib lib1 lib2
```

-no_extra_pins

Disables merging into the `to_lib` any pins that do not already exist in the `to_lib`. This means that during the merge operation, the extra pins are excluded from the second library.

-type {list}

Merges the specified types. Supported types are:

`attributes`, `cap`, `ccs`, `ccs_cap`, `ccs_retain`, `ccsn`, `ccsp`, `constraint`, `delay`, `ecsm`, `ecsm_cap`, `ecsmn`, `ecsmpl`, `em`, `hold`, `leakage`, `max_cap`, `max_trans`, `nochange`, `noise`, `min_pulse_width`, `power`, `recovery`, `removal`, `retain`, `setup`, and `minimum_period`. Default: Merge all types.

-unique_pin_data

Outputs unique data, such as timing and power, for each bus bit or bundle member (similar to `-unique_pin_data` for `write_library`). It specifies that original pin names are to be used inside the `when` condition string without going through the post-processing of changing pin names to bundle names.

Virtuoso Liberate Reference Manual

Liberate Commands

Without unique_pin_data With unique_pin_data

| | |
|--|--|
| bundle (QB) { pin (Q0B) { ... } pin (Q1B) { ... } timing () { ... } } | bundle (QB) { pin (Q0B) { ... } timing () { ... } pin (Q1B) { ... } timing () { ... } } |
|--|--|

-voltage <voltage>

Instructs Liberate to create a new library that contains values calculated by performing voltage scaling from two existing libraries. All characterized timing data in the library is scaled including delay, transition, constraints, retain, ccs, ccs_retain, and ecsm. In addition, internal power, leakage, and capacitance (pin cap and receiver capacitance (ccs, ecsm)) values will be scaled. Current-based power (ccsp or ecsmp) and noise data (si, ccsn, or ecsmn) will *not* be scaled and will be copied from the first library.

A comment is inserted in the output library to indicate that merging with voltage scaling was performed.

For example:

```
merge_library -voltage 1.0 -filename newLib_1.0.lib  
libA_0.9.lib libB_1.1.lib
```

<to_lib>

Specifies the library that will form the initial data in the new library. This is the "target" library into which data from the **from_lib** will be merged.

Virtuoso Liberate Reference Manual

Liberate Commands

{<*from_lib*>}

Specifies one or more libraries from which data will be taken and merged into the "target" (*to_lib*). If more than one library is specified, the merging takes place sequentially, in the order that libraries are specified.

The libraries in the *from_lib* list can contain wildcards in the cell name if surrounded by double quotes. This can be used, for example, to merge the footprint attribute "AND" to all of the cells whose names are prefixed with AND. The *to_lib* does not support the use of wildcards. An example of wildcard usage in a *from_lib* library is as follows:

```
cell ("AND*") { footprint : "AND" }
```

Note: If merging data from complete libraries (e.g. no changes are expected), it is recommended to set *adjust_tristate_load* to 0 prior to merging.

To merge multiple libraries, the *merge_library* command starts with the *to_lib*, and merges data from the *from_lib* into it.

Note: The *to_lib* will *not* be overwritten; the merge process takes place in memory and the results of the merge will be written to the file specified with the *filename* option.

All the data in the *to_lib* will be preserved and any extra data in the *from_lib* will be added. If a list of *from_lib* libraries is provided, the libraries will be merged into the output library, one library at a time. For example, any cells in the *from_lib* that are missing from the *to_lib* will be combined into the merged library and the merged output library will be sorted alphabetically. The *merge_library* command also supports bus structures in the *lib* file, meaning that Liberate will keep buses in the merged library if the source library has buses.

Normally, *merge_library* does not replace data; it only appends it. (See option *method* for an exception to this.) For example, if a *to_lib* has NLDM data and a *from_lib* has NLDM+CCS data, only the CCS data in the *from_lib* will be merged; the NLDM data in the *to_lib* will not be changed. Similarly, if a list of *from_lib* are given with overlapping data sets, only the *first* instance will be merged.

The *merge_library* command can also be used to add any missing attributes and groups, such as CCS, CCSN, and ECSV data, to the equivalent cells in the *to_lib*. For libraries that are not created by Liberate, *merge_library* will attempt to find an equivalent arc in the *to_lib*. An equivalent arc will have the same pins, data type and attributes (e.g. timing sense, timing type etc.) and an overlapping or equivalent logic state. If no match is found a warning is generated, indicating that the data could not be merged with the target library.

Example

```
# Merge CCS into an existing library
merge_library -filename orig_ccs.lib orig.lib ccs.lib
```

one_cold

Accepts a list of pins and returns a string that can be used to specify the one-cold relationship of the pins. For example, `one_cold { a b c }` will be converted to `((!a b c)+(a !b c)+(a b !c))`.

Note: This is a Tcl function.

Options

`{ list }` Specify a list of one-cold pins (Default: none)

This command must be used before [char_library](#).

Example

```
define_cell -when [ one_cold { a b c } ] ...
```

one_hot

Accepts a list of pins and returns a string that can be used to specify the one-hot relationship of the pins. For example, `one_hot { a b c }` will be converted to `((a !b !c) + (!a b !c) + (!a !b c))`.

Note: This is a Tcl function.

Options

`{ list }` Specify a list of one-hot pins. Default: `none`

This command must be used before [char_library](#).

Example

```
define_cell -when [ one_hot { a b c } ] ...
```

packet_slave_cells

Returns a unique list of cells assigned to the client, also known as, slave.

Note: This is a Tcl function.

When called from the server, an empty list is returned. See [Parallel Processing](#).

Example

```
# Set the constraint_delay_degrade global variable to 8% for myCell in the client
only
if { [packet_slave_cells] == "myCell" } {
    set_var constraint_delay_degrade 0.08
}
```

printvars

Lists the current values of all of Liberate's command variables.

Options

None.

Example

```
# List Liberate's variables
printvars

conditional_constraint = 0
constraint_delay_degrade = 0.1
constraint_glitch_peak = 0.1
constraint_glitch_hold = 0

...
slew_lower_rise = 0.3
slew_upper_rise = 0.7
slew_lower_fall = 0.3
slew_upper_fall = 0.7
spice_delimiter = /
toggle_leakage_state = 0
```

read_ldb

Reads an existing library database (ldb) created by the [write_library](#) command. The library database can then be used for formatting the library data, for example, creating a datasheet.

Options

`-check_driver_waveforms`

Checks both pre-driver and active-driver generated waveforms in ldb to verify that they have the correct slews.

`-check_spice`

Checks netlist and model for changes and recharacterize cell if any changes are detected.

`-check_var`

Checks variables and commands for changes. If a change is detected, that change will map to a specific `remove_type`, and that type (or types) will be recharacterized. This works only if `-incremental` is set.

`-incremental`

Specifies incremental capability. This option instructs Liberate to examine the ldb and performs characterization for any data that is missing from the ldb.

Note: The `-incremental` flow is currently supported when running a single cell and is not supported with any packet flow (see [packet_clients](#)).

`-remove {list_of_cells}`

Specifies a list of cells to remove from the ldb. Default: (none)

By default, Liberate will not characterize any cell that has already been loaded from an ldb. A cell that is removed from the ldb during loading will be re-characterized.

`-remove_failed`

Specifies to automatically remove cells that are marked as failed in the ldb. In most flows, this can be used in place of "`read_ldb -remove { list of failing cells }`".

`-remove_type {list_of_types}`

Specifies a list of data types to be removed from the ldb, so they will be re-characterized. Supported types are: `ccs`, `ccsn`, `ccsp`, `cin`, `constraint`, `delay`, `hold`, `leakage`, `mpw`, `power`, and `setup`. Default: `none`

`<filename>`

A library database file in ldb format.

Liberate can also use an existing ldb to recover from any characterization run that didn't complete successfully. As each cell is characterized it is saved to an ldb in the current directory, named *altos.ldb.<#>*. This temporary ldb can subsequently be read by Liberate (using `read_ldb`) to complete the characterization. A complete ldb that contains all the cells can then be saved using `write_ldb`.



Important
You may have a given variable stored in the ldb and also specified in your Tcl script. Whichever occurs *last* in your script will determine the value of the variable (i.e.: the last setting of the variable will override an earlier setting.)

The `read_ldb` command will automatically recognize if the input file is gzipped during loading. The GNU gzip utility must be in the search path if the input file is gzipped.

Using the `-check_spice` option

check_spice enables checking if the spice netlist and/or the spice model has changed. The Spice netlist check is based on the parsed X/M/D/R/C circuit elements, which generates a checksum, along with a count and total value of each element type that is stored in the ldb. The Spice model check is based on the top-level model path name & time stamp, which is stored in the ldb. If a change in the netlist or models is detected, the cell will be removed and completely recharacterized. Required for this flow are:

- set option `-incremental`
- set variable `enable_command_history`
- set variable `rechar_chksum` (specifies the linux routine to generate a checksum.)

Suggested uses for incremental capability:

- Adding or deleting arcs; only the affected arcs will be re-simulated.
- Arcs containing errors from a previous characterization (`altos_error_flag` in the ldb) will be automatically re-characterized.
- Re-simulating only constraint arcs (for example, if `constraint_delay_degrade` was changed). This will be faster than re-characterizing the entire library:

```
read_ldb -remove_type {constraint}  
char_library
```

- Adding CCSN to a previous ldb that does not contain CCSN data. Use these commands:

```
read_ldb -incremental  
char_library -ccsn
```

CCS timing and/or power can be handled similarly.

Example

```
# Read an ldb to generate additional formats
read_ldb tt_all.ldb
# Write a HTML datasheet
write_datasheet -format html -dir tt_html tt
# Write .lib
write_library tt_all.lib
```

Note: When an ldb is read in, if the `scale_tran_by_template` and/or `scale_load_by_template` variables were set to 1 when the VDB was created, they will be reset to 0 to prevent double scaling of the indices.

read_library

Reads an existing Liberty format library into memory.

Options

{library_names} List of library file(s) to read in Liberty format.

This is a key command for a number of flows capable of outputting these types of files:

- **User data file** - Extracts non-characterized data such as area, function, etc.
See [write_userdata_library](#)
- **Template file** - Tcl script used as part of a characterization flow
See [write_template](#)
- **Datasheet** - Typical datasheet showing delays, power, pin capacitance, etc.
See [write_datasheet](#)
- **Verilog / Vital** - RTL model descriptions
See [write_verilog](#) and [write_vital](#)

Examples

Example 1

```
# Create a datasheet
read_library cdnTech1.lib
write_datasheet -format text myDatasheet
```

Example 2

`read_library` can also be used together with `write_library` to add margin or other attributes to a library:

```
# Add 20% relative margin to setup and hold
read_library cdnTech2.lib
add_margin -type {setup hold} -rel 0.20
write_library test.lib
```

Example 3

When a user wants to read a library with bus syntax “<>” and would like to use the output to generate another bus syntax (i.e. “[”]). The below commands in the same sequence will work. In this example the bus syntax is “<>” and the library is written with “[”].

```
set_var bus_syntax "<>"  
read_library input.lib  
write_library -bus_syntax "\[\]" -filename output.lib test1
```

read_spice

Reads in the SPICE netlists of the cells.

Options

```
-format {hspice | spectre}  
          SPICE netlist format. Default: hspice  
{<spice_netlist_file>}  
          List of file(s) with extracted circuit netlists in SPICE format.
```

The model files should be included with the circuit netlists. The SPICE netlist and model formats supported by Liberate are as follows:

- ❑ Hspice netlist, Level=49, Level=53 and Level=54 models,
- ❑ Spice3 netlist, BSIM3 and BSIM4 models. Well proximity effects are supported.
- ❑ PSP

Note: The `read_spice` command can read files that have been compressed using gzip. Specify the filename with or without the `.gz` suffix.

This command must be used before [char_library](#).

Examples

```
# Read in a group of SPICE cell netlists  
read_spice {nand2x2.spi nor2x2.spi inv2x4.spi 90nm_cmos.spi}  
# Read in a group of SPICE cell netlists  
set cells {nand2x2 nor2x2 inv2x4}  
set spice_netlists {90nm_cmos.spi}  
set csz [llength $cells]  
for {set c 0} {$c < $csz} {incr c 1} {  
    set cell [lindex $cells $c]  
    lappend spice_netlists subckts/$cell.spi  
}  
read_spice $spice_netlists
```

read_truth_table

Reads in and validates truth-table files.

Options

{<filename>} List of Truth Table files which contain one or more truth tables to load.

This command must be used before running the [write_template](#) command. Use the `write_template` command with the `-truth_table` and `-auto_index` options to output a template file built from the loaded truth table file data.

The `define_pin_load` template is added when there is a tri-state bi-directional pin in the cell and one or more of the following attributes: pull-up voltage, pull-up resistance, pull-down resistance or serial-resistance, is defined in the truth table.

The `-pin_load` and `-load_dir` options are not added to the `define_arc` command. If required, they need to be added manually.

Note: For pull-up and pull-down pins, only pin capacitance is characterized. The `-pin_load` and `-load_dir` options should be added to the `define_arc` commands in the template to specify these values.

Example input truth table:

```
# CMOS Tri-State Output Pad with Schmitt Trigger Input and Pull-Up,  
# High-V Tolerant  
* PULLUP_RES=1000  
* PULLDOWN_RES=1000  
* SERIES_RES=25  
* PULLUP_VOLT=3.3  
* CELL=PDIO12  
* TABLE= OEN I PAD @ PAD C  
        0 0 - @ 0 0  
        0 1 - @ 1 1  
        1 X 0 @ - 0  
        1 X 1 @ - 1  
        1 X - @ Z 1  
* TABLE_END  
* CELL_END
```

Virtuoso Liberate Reference Manual

Liberate Commands

Example output template:

```
Cell: PDIO12
define_cell \
    -input {OEN I} \
    -output {C} \
    -bidi {PAD} \
    -pinlist {OEN I PAD C} \
    -delay delay_template_7x7 \
    -power power_template_7x7 \
    PDIO12

define_pin_load \
    -pullup_voltage 3.3 \
    -pullup_resistance 1000 \
    -pulldown_resistance 1000 \
    -series_resistance 25 \
    pin_load_template_PDIO12

define_leakage -when "!OEN !I" {PDIO12}
define_leakage -when "!OEN I" {PDIO12}
define_leakage -when "OEN I !PAD" {PDIO12}
define_leakage -when "OEN !I !PAD" {PDIO12}
define_leakage -when "OEN I PAD" {PDIO12}
define_leakage -when "OEN !I PAD" {PDIO12}
define_leakage -when "OEN I" {PDIO12}
define_leakage -when "OEN !I" {PDIO12}

# OEN -> PAD
define_arc \
    -type enable \
    -vector "F0FX" \
    -related_pin OEN \
    -pin PAD \
    PDIO12

define_arc \
    -type enable \
    -vector "F1RX" \
    -related_pin OEN \
    -pin PAD \
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
PDIO12

define_arc \
  -type disable \
  -vector "RXRX" \
  -related_pin OEN \
  -pin PAD \
  PDIO12

define_arc \
  -type disable \
  -vector "RXFX" \
  -related_pin OEN \
  -pin PAD \
  PDIO12

# I -> C
# I -> PAD
define_arc \
  -vector "OFFX" \
  -related_pin I \
  -pin PAD \
  PDIO12

define_arc \
  -vector "ORRX" \
  -related_pin I \
  -pin PAD \
  PDIO12

# OEN -> C
define_arc \
  -vector "F0XF" \
  -related_pin OEN \
  -pin C \
  PDIO12

define_arc \
  -vector "F1XR" \
  -related_pin OEN \
  -pin C \
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
PDIO12

define_arc \
  -vector "RX0F" \
  -related_pin OEN \
  -pin C \
  PDIO12

define_arc \
  -vector "RX1R" \
  -related_pin OEN \
  -pin C \
  PDIO12

define_arc \
  -vector "RXXR" \
  -related_pin OEN \
  -pin C \
  PDIO12

# PAD -> C
define_arc \
  -vector "1XFF" \
  -related_pin PAD \
  -pin C \
  PDIO12

define_arc \
  -vector "1XRR" \
  -related_pin PAD \
  -pin C \
  PDIO12

set cells { \
  PDIO12 \
}
```

read_vdb

Reads in a VDB file that was previously created by the [write_vdb](#) command. The Tcl file must contain the same setup that was in place when the VDB file was created.

Options

{<*filename*>} The name of a VDB file to load.

Use the `read_vdb` command in a Tcl command file before the `char_library` command. When using a `read_vdb` command, do ***not*** reference a `template.tcl` file.

Use the `write_vdb/read_vdb` flow to speed up the analysis by storing the processed vectors and to enforce a specific structure across multiple PVT corners.

remove_pin_attribute

Removes the automatically-generated pin-level attributes from the output Liberty file.

Options

| | |
|-------------------------|--------------------|
| <i>{cells}</i> | List of cells |
| <i>{pins}</i> | List of pins |
| <i>{attribute list}</i> | List of attributes |

Pin-level attributes can be provided in the [write_library](#) -user_data file or automatically added by Liberate.

Note: To remove pin level attributes that are supplied in the user_data file, modify the user_data file.

This command supports wildcards.

Examples

```
# Remove input_signal_level from pins Q and CK on cell DFF
remove_pin_attribute {DFF} {Q CK} {input_signal_level}

# Same as above except all pins of all cells
remove_pin_attribute {*} {*} {input_signal_level}
```

reset_defaults

Restores the changed variable settings to the default values from a previous release.

Options

`-version <version>` Default: Reset all variables needed to restore the behavior in the previous release.

From release to release, defaults values of some Liberate variables might change. You can use this command to restore the default values from a previous release. The required default value changes are stored in the file located at the following location:

`${ALTOSHOME}/etc/backward_compatible.tcl`

The backward compatible settings echoes into the Tcl file.

Important

This command should be used at the start of your Tcl file and can only be used once per run.

Example

```
# Reset Liberate defaults to 12.1
reset_defaults -version 12.1
```

select_index

Specifies the template index values to be used for simulation.

Options

| | |
|--|--|
| <code>-cells {list}</code> | Specifies a list of cells. Default: all cells |
| <code>index_1 { list } index_2 { list }</code> | Specifies which index value positions specified by a <code>define_template</code> or <code>define_index</code> command are characterized. The position begins counting with 1. Default: all indexes (indexes are not reduced). The supported value is a list of specific positions within curly braces, such as <code>{ 2 3 5 }</code> . |
| <code>-style <value></code> | Specifies the method for selecting the template index values. The supported values are: <ul style="list-style-type: none">"index" Uses the settings of the <code>-index_1</code> and <code>index_2</code> options."1x1" Uses the first index value. This is equivalent to <code>"-index_1 {1} -index_2 {1}"</code>."2x2" Uses the first and last index values. For an index with seven values, this is equivalent to: <code>"-index_1 {1 7} -index_2 {1 7}"</code>."3x3" Uses the first, middle, and last index values. For an index with seven values, this is equivalent to: <code>"-index_1 {1 4 7} index_2 {1 4 7}"</code>."mid" Uses the middle index value. For an index with seven values, this is equivalent to: <code>"index_1 {4} -index_2 {4}"</code>. |
| <code>-type { list }</code> | Specifies a list of data types. Default: all supported data types The supported data types are: <code>delay</code> , <code>power</code> , <code>constraint</code> , and <code>mpw</code> . |

This command must be specified before the `char_library` command.

set_aging_criteria

Enables Liberate to use the Spectre circuit aging capabilities. Before using this command, ensure that you are using MMSIM12.1 ISR16 or a later release. Check the `$(ALTOSHOMEDIR)/README` file for the recommended version of MMSIM. You might get incorrect aging characterization data if you use an earlier version of the MMSIM release. In addition, use the `extsim_model_include` variable and the `define_leafcell` command with `set_aging_criteria`.

Note: The aging models are required for the aging flow to work.

Options

`-aging_supply list{}`

Specifies the voltage(s) at which aging is run. It accepts a single value (in volts) applied to all supplies or a list of `supply_name` voltage pairs. The default is the voltage specified by the [set_operating_condition](#) command.

`-aging_temp <temperature>`

Specifies the aging temperature in degrees celsius. The default is the temperature specified by the [set_operating_condition](#) command.

Note: The `-aging_supply` and `-aging_temp` options allow you to run the aging analysis at a different voltage and temperature than the and characterization.

`-attribute <value>` Settings about reliability analyses. (REQUIRED)

The `-attribute` option specifies the aging-specific settings for the Spectre reliability analysis. The following attributes should be included: age time, deltad value, and report_model_param value. For more information on the aging-specific attributes, refer to the *Virtuoso Spectre Circuit Simulator and Accelerated Parallel Simulator User Guide*. There are no default attributes. This option is required.

`-duty_cycle <value>`

Specifies the duty cycle that is applied to the period for stress simulation. The stress simulation duty cycle ranges from 0.1 (10% of period at a high logic level) to 0.9 (90% of the period at a high logic level). Default: 0.5 (50%)

If this option is not specified, the default value of 0.50 (50%) is applied to netlist aging.

`-period <value>` Specifies the period of stress simulation (in MKS units). (REQUIRED)

There is no default period. If you set a period value that is less than `max_slew` or `duty_cycle`, Liberate corrects it as necessary. An information message is also displayed.

`"INFO (LIB-1011): (set_aging_criteria):"`

This command must be used before char_library.

Example

```
# Set the aging criteria
set_aging_criteria \
    -period 300e-9 \
    -duty_cycle 0.5 \
    -attribute "    age time = \[10y\]\n    deltad value = 0.1\nreport_model_param value=yes\n"
```

set_attribute

Adds, replaces, deletes, or modifies the specified attribute in the output library.

Options

-cells {<cell_name>}

A list of cell names. This list can include wildcards. Note: When you specify -cells, the attribute is set at the specified cell level. However, if you do not specify any of the location options (that is -cells, -pins, -related, and -group), the attribute is set under the library level.

-pins {output_pin(s)}

Specifies a list of pin names on which the attribute is set.

Note: If this argument is specified, the -cells argument is required.

-type <type>

The type of attribute. Specify one of the following valid values: constraint, delay, em, leakage, min_pulse_width, power, or retain. Default: "" (Do not apply this option.)

-related <related_pin>

The related_pin of an arc. This is used to identify a timing or internal_power group under the specified pins in the specified cells where the specified attribute will be set. Default: "" (Do not apply this option.) Note: If -related is specified, the -cells and -pins options are required.

-group_name <group_name>

The group name under which the attribute will be set. Examples: "ff" and "latch". Default: "" (Do not look for a matching group). If -group_name is not specified, the attribute is set under the level specified by the other options.

-match_attributes {<attributes>}

A list of existing attributes and their values that exist in the library. This is used to identify a specific location to set the attribute. For example, {timing_type rising_edge timing_sense positive_unate}. Default: "" (Do not match any existing attributes.)

-method <method> The method to be used for setting the attributes. Specify one of the following valid values: append, augment, replace, or delete. Default: replace (Replaces the existing attribute and adds it if it does not already exist.)

-user_defined_type < string | integer | float | boolean > Specifies the type of this attribute as specified in the `define` command in the library header. Default: " " (no `user_defined_type`)

<attribute_name> The name of the attribute that needs to be edited.

<?attribute value?> (Optional) The new value of the specified attribute. The value is an arbitrary string and should contain all required punctuation such as quotes. Note: If the -method option is to delete, do not provide a value.

Caution

The use of this command can create a library that does not compile or is functionally incorrect. This is because there is minimal checking of any attribute changes applied by using the `set_attribute` command. In addition, the `set attribute` should have the legal Liberty syntax, otherwise the final library will not compile.

The `set_attribute` command allows you to add or modify attributes in the output library at the library, cell, pin, and group levels. In addition, you can add or modify attributes in groups of the following types: `delay`, `constraint`, `power`, `em` (electromigration), `leakage`, `min_pulse_width`, and `retain`.

How the attributes are applied depends on the value specified with the `-method` option. The `replace` method, which is the default, replaces an existing attribute value or adds it if the attribute does not exist. The `append` method appends additional copies of the attribute if it already exists; otherwise, it adds the attribute. The `augment` method adds an attribute only if the attribute does not already exist. The `delete` method removes the attribute and its value.

By default, the specified attribute is set at the library level. Use the `-cells` option to specify a list of cells (including wildcards) on which the attribute needs to be set. To set the attribute at a pin level, use both the `-cells` and `-pins` options.

Attributes that belong to a specific library, cell, or pin group can be set by specifying the `-group_name` option.

Attributes that appear under characterized arcs (such as `delay`, `leakage`, `constraint`, and `power`) can also be set using the appropriate `-type`, `-related`, and `-match_attributes` list.

For example, while editing attributes for a delay arc, the following options must be specified:

- `-cell <cell_name>`
- `-pin <output_pin>`
- `-related <related_pin>`
- `-type "delay"`

If a specific delay arc is to be edited, a list of attributes can be specified with the `-match_attributes` option. The specified attributes list uniquely identifies the arc to be edited, such as a specific `when` condition and `timing_type`.

Example

```
# add the "nom_process" attribute with value 3 to the library level if it does not
# already exist
set_attribute -method augment nom_process 3

# add the "default_hidden_power" attribute with value true to hidden power arc under
# pin B1 of cell AOI33 with the when condition "(A2 * !(A3) * B1 * B2 * !(B3) * ZN)"
set_attribute \
    -cells "AOI33" \
    -pins "B1" \
    -type power \
    -match_attributes {when "(A2 * !(A3) * B1 * B2 * !(B3) * ZN)"} \
    default_hidden_power \
    true

# add the "setup_mode" attribute with value "max" to a constraint arc under pin D of
# cell SDFF to all entries with timing_type setup_rising.
set_attribute \
    -cells SDFF \
    -pins D \
    -type constraint \
    -match_attributes {timing_type setup_rising} \
    setup_mode \
    \"max\"
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
# Change the value of the min_pulse_width_low attribute to 0.040 under pin CP of
# cell CKGATE.
set_attribute \
    -cells CKGATE \
    -pins CP \
    -type min_pulse_width \
    min_pulse_width_low \
    0.040

# Change the timing_type of the constraint arcs under pin D of cell SDFF from
# hold_rising to hold_falling
# Note: This example shows the flexibility of the set_attribute command. It
# lets you make arbitrary changes to the attributes in a library. But when used
# incorrectly, as it is in this example, it can break a library.
set_attribute \
    -cells SDFF \
    -pins D \
    -type constraint \
    -match_attributes {timing_type hold_rising} \
    timing_type \
    hold_falling

# Change the function attribute of pin Y for all cells prefixed with AND2 to "A*B"
set_attribute -cells AND2* -pins Y function "\"A*B\""

# Remove the direction attribute from pin Y for all cells prefixed with OR2
# Note the resulting library will not compile.
set_attribute -method delete -cells OR2* -pins Y direction
```

set_client

Specifies a client a machine to be used for distributed library characterization.

Options

-dir <directory_name>{ %N%U%P%S }

Defines a directory on the client machine to use as a temporary workspace for simulation jobs performed on that machine. Liberate creates the directory if it does not exist. (REQUIRED)

<machine_name> Name of client machine or queue name.

Liberate can perform distributed processing by explicitly defining the names of each of the client machines. To specify multiple machines, use multiple **set_client** commands.

The network port number to be used can also be set using the [set_network_port](#) command.

For more details on distributed parallel processing, see [Chapter 3, “Parallel Processing.”](#)

This command must be used before [char_library](#).

Examples

```
# set the machines to use (no queue)
set_client -dir /tmp/scratch/%U_%N_%S_%P linux1
set_client -dir /tmp/scratch/%U_%N_%S_%P linux2
```

The **-n** option supports a deprecated method of using a queuing system such as LSF. For more details on this option, see [Appendix D, “Deprecated and Legacy Commands and Variables.”](#)

set_conditional

Disables conditional arcs for delay and constraint groups for a list of cells. Currently other arc types such as power are not supported.

Options

| | |
|---|--|
| <code>-off</code> | Disable conditional states |
| <code>-type {<i>delay const</i>}</code> | Type of data for which conditional arcs are disabled. Default: <code>{delay const}</code>) |
| <code>-cells {<i>cell_names</i>}</code> | List of cell names. (REQUIRED) |

This command can be used after char_library, but before model generation.

Example

```
# Turn off conditional delay and constraint arcs on ao32      # a033 cells
set_conditional -off -type {delay const} -cells {ao32 ao33}
```

set_constraint

Adds the specified margin (in seconds) to a supported constraint type when a library or datasheet is output.

Options

-cells {cell_names}

Specifies a list of cell names on which the `set_constraint` command should be applied. Default: none

Note: This option currently applies only to the `-index1_factor` and `-index2_factor` options.

-index1_factor <value>

Specifies the multiplication factor. Default: 0 . 0

-index2_factor <value>

Specifies the multiplication factor. Default: 0 . 0

-margin <value> Specifies the margin to add to timing constraints. Default: 0

-max <value> Specifies the maximum constraint value. Default: 1e20

-max_margin <value>

Specifies the maximum margin allowed for the given constraint.
Default: 1e20

-max_risefall Specifies to use the maximum value from the `rise_constraint` and `fall_constraint`.

-min <value> Specifies the minimum constraint value. Default: -1e20

-min_margin <value>

Specifies the minimum margin allowed for the given constraint.
Default: -1e20

Virtuoso Liberate Reference Manual

Liberate Commands

`-min_recrem <value>`

Specifies a minimum value (in seconds) that the respective sum of recovery + removal must exceed. Default: no checking

If the minimum is not met, the removal values are adjusted as required to meet the minimum. When the `-min_recrem` value is not met, the warning level can be controlled by setting `-min_warning` as follows:

- 0 = no warnings
- 1 = 1 warning per table
- 2 = 1 warning per table value.

`-min_setuphold <value>`

Specifies a minimum value (in seconds) that the respective sum of setup + hold must exceed. If the minimum is not met, the hold values are adjusted as required to meet the minimum. When the `-min_setuphold` value is not met, the warning level can be controlled by setting `-min_warning` as follows:

- 0 = no warnings
- 1 = 1 warning per table
- 2 = 1 warning per table value

`-min_warning <0 | 1 | 2>`

Specifies the warning level when `-min_recrem` or `-min_setuphold` are exceeded. Default: 1

`-non_seq_sum_swap_direction`

Instructs the tool to sum `non_seq_setup`/`non_seq_hold` using the *opposite* direction to that used for summing `setup`/`hold`.

This only applies when the `-min_setuphold` option is set. The default is to follow the same direction as used for `setup/hold`, as specified by the `-sum_same_direction` option.

`-pin_dir <rise | fall | both>`

Specifies whether the margin is to be applied to just `rise` constraints, `fall` constraints, or `both` rising and falling. Default: `both`

Virtuoso Liberate Reference Manual

Liberate Commands

-pin_probe_factor Specifies the factors that are applied to the final margin based on the measurement of the pin to probe delay. Default: 0

This option is applied only when constraints are measured using path delay. The final margin for a given constraint is as follows:

```
Total_Margin = margin + (index1_factor * pin_slew) +  
(index2_factor * related_pin_slew) + (pin_probe_factor *  
pin_to_probe_path_delay) + (related_probe_factor *  
related_pin_to_probe_delay)
```

-pins {pin_names} Specifies a list of pin names on which the `set_constraint` command should be applied. Default: none

Note: This option currently applies only to the `-index1_factor` and `-index2_factor` options.

-related {list} List of related pins.

-rel_margin <value> Margin ratio to add to timing constraints. Default: 0

-related_probe_factor

Specifies the factors that are applied to the final margin based on the measurement of the related pin to probe delay. Default: 0

This option is applied only when constraints are measured using path delay. The final margin for a given constraint is as follows:

```
Total_Margin = margin + (index1_factor * pin_slew) +  
(index2_factor * related_pin_slew) + (pin_probe_factor *  
pin_to_probe_path_delay) + (related_probe_factor *  
related_pin_to_probe_delay)
```

-sum_both_directions

When set, all four constraint combinations (that is, `setup rise hold rise`, `setup fall hold rise`, `setup rise hold fall`, and `setup fall hold fall`) are checked for a negative sum. This is equivalent to using both settings of `-sum_same_direction` together. Default: off

-sum_same_direction Request summing using the same direction. Default: opposite direction.

The **-sum_same_direction** option can be used to check the **-min_setuphold** and **-min_recrem** options. Specify this option to use the same transition for checking the minimum sum of setup + recovery with hold + removal. The default is false, that is, sum rise + fall, fall + rise.

-type {setup | hold | recovery | removal | mpw}

Specifies the timing constraint type.

When the **-type** option is set to **setup** or **hold**, it enables a factor of the **index_1** and **index_2** transition values to be added to the characterized constraint values. The **index1_factor** is applied to the **index_1** values and the **index2_factor** is applied to the **index_2** values. The formula is:

```
new_constr_val = orig_constr_val
    + ( index_1 * index1_factor )
    + ( index_2 * index2_factor )
    + margin
```

-when <"function"> Logic state of side inputs.

The **set_constraint** command adds the specified **-margin** (in seconds) to the constraint type when a library or datasheet is output. Acceptable constraint types are: **setup**, **hold**, **recovery**, **removal**, and **mpw** (minimum pulse width). The margin is applied globally. If there are multiple **set_constraint** commands specified, the margin in the last command will override earlier ones. The margins are not cumulative. If no **-type** is given, the **-margin** is applied to all constraints. The **-margin** is applied before checking the value against the min/max limits. The default **-margin** is 0.0 seconds. The **-margin** is based on library thresholds and not on measurement thresholds. The **-rel_margin** option can be used to specify a relative margin to be added. The **-rel_margin** accepts a ratio number between 0 and 1.

To have an effect on the output library, this command must be executed before [write_library](#). If the **-define_arc**, **-pin_probe**, and **-related_probe** options are used and a margin is desired for the delay measurements that will be reported in the LDB, this command must be executed before [char_library](#).

Example

```
# Add 20ps margin to all constraints, 50ps to hold,  
# enable warnings if sum is less than 0ps  
set_constraint -margin 20e-12  
set_constraint -type hold -margin 50e-12  
set_constraint -min_recrem 0 -min_setuphold 0 -min_warning 2
```

set_constraint_criteria

Allows setting of constraint-related parameters with different values either globally or for different constraint arcs without having to specify the define_arc command for each constraint.

Options

-cells {cell_names}

Specifies a list of cells that the specified criteria will affect.
Default: *all cells*

If the **-cells** list is empty or not included, the specified criteria are applied globally.

-delay_degrade <value>

Specifies the maximum amount of delay degradation relative tolerance ratio permitted in the clock-to-constraint output pin (flip-flop) or the data-to-constraint output pin (latch) delay before an arriving signal is deemed to fail a timing constraint. Default: 0.1 (10%)

The *delay_degrade* is a percentage specified as a positive decimal (usually from 0.0 to 1.0). This option overrides the global variable constraint_delay_degrade when the **-cells** option is not used.

-delay_degrade_abstol <value>

Specifies the minimum delay degradation (minimum) absolute tolerance value permitted (in seconds) in the clock-to-constraint output pin (flip-flop) or the data-to-constraint_output_pin (latch) delay. Default: 5e-12 (5 ps)

This option overrides the global variable constraint_delay_degrade_abstol when the **-cells** option is not used. A constraint bisection search bound is deemed as failing if the delay degradation is greater than the maximum of the constraint_delay_degrade percentage of the clock-to-output-delay or data-to-output-delay and the constraint_delay_degrade_abstol exceeds the constraint_delay_degrade_abstol_max.

Virtuoso Liberate Reference Manual

Liberate Commands

`-delay_degrade_abstol_max <value>`

Specifies the maximum delay degradation absolute tolerance, in seconds. Default: -1 (off)

The `-delay_degrade_abstol` and `-delay_degrade_abstol_max` options when specified together set both an upper and a lower bound to the delay degradation. This option override the global variable `constraint_delay_degrade_abstol_max` when the `-cells` option is not used.

`-dependent`

The constraint is computed as a dependent constraint with respect to its complementary constraint (setup vs. hold, recovery vs. removal). This requires that `constraint_dependent_setuphold` should be equal to 0.

`-dependent_margin`

The absolute margin (in seconds) to be added to the independent constraint when computing the dependent constraint to which the command applies. See `constraint_dependent_setuphold_margin`.

`-dependent_margin_ratio`

The margin (as a multiple of the slew index) to be added to the independent constraint when computing the dependent constraint to which the command applies. See `constraint_dependent_setuphold_margin_ratio`.

`-glitch_peak <value>`

Specifies the maximum size of logic glitch peak relative tolerance ratio permitted on the probe node before an arriving signal is deemed to fail a timing constraint. Default: 0.1

This option overrides the global variable `constraint_glitch_peak` when the `-cells` option is not used.

`-independent`

The constraint is computed as an independent constraint. This option is used to override prior `-dependent` settings.

Virtuoso Liberate Reference Manual

Liberate Commands

-metric <list> Sets the timing constraint measurement criteria. The supported values are: delay, delay_both_edges, glitch, slew, width, or path_delta.

If the `-metric` option is specified both in `set_constraint_criteria` and `define_arc` commands, the `define_arc` setting takes precedence. If `-metric` is not specified for an arc, the usual inside view decision process applies. The `delay_both_edges` metric applies to constraints where the probe node transitions in response to both the related and the constrained pins. The constraint is tested with delays from both the constrained and related pins to the associated transition on the probed node. Degradation failure on either edge indicates a failure of the constraint.

-min_constraint {cell pin ...}

Instructs Liberate to put in the library the minimum constraint (instead of the maximum constraint) for the specified cell-pin pairs. This command option accepts a list of "cell pin" pairs where "cell" specifies the name of the cell (an asterisk "*" indicates all cells) and "pin" specifies the name of a pin in the specified cell.

-output_load <min | max | value | index_?>

Specifies the load to be applied to cell output pins during constraint characterization. The value must be in Farads. Default: min (from `constraint_output_load`)

This option can be used, for example, to remove glitch suppression for cells with an expected output glitch from the clock when a reset or set pin is active. This option overrides the global variable `constraint_output_load` when the cells option is not used.

-pin {pins}

Specifies a list of pins. Default: {} (no pins)

For a constraint, the pin is constrained so that it must arrive before (setup), holds after, or meets the minimum pulse width of the characterized value. That is, the "pin" is constrained.

-pin_dir <R | F>

Specifies the switching direction of the pin.

Virtuoso Liberate Reference Manual

Liberate Commands

- probe {probes}** Takes a list of nodes where one or more constraint criteria can be specified after any probe in the list. For related details, see [Using the -probe option](#).
- related_pin {pins}** Specifies a list of related_pin(s) for the constraint. The related pin is typically a clock pin.
- related_pin_dir <R | F>** Specifies the switching direction of the related_pin.
- slew_degrade <value>** Includes slew degradation relative tolerance ratio when determining timing constraints. Default: -1 (not enabled))
By setting the slew criteria, both the delay and slew degradations are checked and the first criteria to fail determines the setup and hold values. The slew degradation is a value (0.0 to 1.0) that represents the percentage of slew degradation and is measured using the `measure_slew_*` variables. This option overrides the global variable `constraint_slew_degrade` when the `cells` option is not used.
- type <hold | mpw | min_pulse_width | nochange | non_seq_hold | non_seq_setup | recovery | removal | setup>** Specifies that type of constraint to which the criteria applies.
Specify the type `nochange` to apply the criteria to any of the `nochange_*_*` arc types. Use the options `-pin_dir` and `-related_pin_dir` to apply the criteria to a specific `nochange` arc. The `define_arc` command must be specified to enable `nochange` arcs to be modeled.
Note: This option is ignored if the `-cells` option is not specified.
- when <"function">** Defines the logic conditions of the side and non-switching pins of the cell. It corresponds to the Liberty `-when` option.

`-width_degrade <value>`

Specifies the percentage of degradation in the width of a pulse when calculating setup/hold time at the output or internal node of the pulse generator in pulse latch cells. Default: follows the constraint_width_degrade variable.

This is used together with the `-metric width` option of the define_arc command. This option overrides the global variable constraint_width_degrade when the cells option is not used.

To set the global constraint-related criteria using the `set_constraint_criteria` command, do not include any of the arc-specific settings such as the options `-cells`, `-type`, `-pin`, `-pin_dir`, `-related_pin`, and `-related_pin_dir`. These options (`cells`, `-type`, `-pin`, `-pin_dir`, `-related_pin`, and `-related_pin_dir`) are used to identify specific constraint arcs to which the command will apply and can be used in any combination with any or all of the remaining options. Wildcard expressions (*) and (?) may be used with `-cells`, `-pin`, and `-related_pin`. Multiple `set_constraint_criteria` commands can be applied to a single constraint. If the same option (such as `-glitch_peak`) is supplied by more than one `set_constraint_criteria` command, the last value defined is used. To set a global parameter, the `cells`, `-type`, `-pin`, `-pin_dir`, `-related_pin`, and `-related_pin_dir` should not be used.

This command can be used to set the related global variables, but if both the global variable and the `set_constraint_criteria` (in global mode) commands are used, then the last one executed sets the global criteria.

If a constraint criteria is set globally, using `set_constraint_criteria` and using `define_arc -metric_thresh`, the order of precedence is as follows:

- 1st: `define_arc`
- 2nd: `set_constraint_criteria`
- 3rd: global variable

The `set_constraint_criteria` command can be used to override metric thresh values, but it does not control the metric used to measure a constraint. Use the `-metric` and `-metric_thresh` options of the `define_arc` command to control the metric to be used to measure a constraint. For example if `-glitch_peak` is supplied, it does not require that the matching constraint use a glitch measurement, but if the glitch metric is chosen by Liberate for the arc, the specified value is used in place of the global parameter value.

This command must be used before char_library.

Using the -probe option

The -probe option supports the following two use models:

- -probe { list of probe nodes }
- -probe { probeNode1 crit1 value1 < crit2 value2 > ... probeNode2 crit1 value1 < crit2 value2 > ... }

For example:

```
set_constraint_criteria -probe {a b}  
set_constraint_criteria -probe {a -metric glitch -glitch_peak 0.2 b -  
delay_degrade 0.1}
```

The set of supported criteria and their meaning is the same as the `set_constraint_criteria` command currently supports.

Specifying a criterion in the -probe option, such as `-delay_degrade`, modifies the criteria but does NOT enforces the metric. To enforce the metric, it must be specified explicitly using the `-metric` option; otherwise, Liberate chooses which metric should be applied.

The criteria specified for each probe overrides the criteria set by using `set_constraint_criteria` even if it is not specified in the same command. Also, if some criteria is specified with both `define_arc` and `set_constraint_criteria` commands, the criteria specified with the `define_arc` command takes precedence. One useful subtlety is that if a per -probe criterion is specified in a list of probe nodes and the -probe list is modified by a later `set_constraint_criteria` or `define_arc` command, the per -probe criteria still applies to probes in the new list, unless explicitly overridden. This allows the criterion to be specified once for all potential probe nodes and the probe node selection to be specified per-arc or per-cell.

Specifying probe nodes

To specify probe nodes to use on an arc basis, you may combine the `-related_pin`, `-related_pin_dir`, `-pin`, `-pin_dir`, `-probe`, `-cell`, and `-type` options. When specifying a probe, no other options can be used with the `set_constraint_criteria` command. That is, the probe(s) must be specified in a separate command from the other constraint criteria. For example, the criteria such as `delay_degrade` cannot be combined in a command with the `-probe` option. Separate commands may be used: one with the failure criteria, and one with the probe criteria.

Examples

Example 1

```
# Set the hold criteria to 10% glitch for clock_gater
set_constraint_criteria \
    -type hold \
    -glitch_peak 0.1 \
    -cells clock_gater

# Set the setup criteria to 10% delay degradation and
# 50% slew degradation for the clock_gater
set_constraint_criteria \
    -type setup \
    -delay_degrade 0.1 \
    -slew_degrade 0.5 \
    -cells clock_gater

# set the global criteria
set_constraint_criteria \
    -delay_degrade 0.15 \
    -delay_degrade_abstol 10e-12 \
    -glitch_peak 0.5 \
    -slew_degrade 0.5
```

Example 2

```
# Specify a probe node for setup of D to CK
set_constraint_criteria \
    -type setup \
    -pin D \
    -related_pin CK \
    -probe N1 \
    -cells { dff1 dff2 }

# Specify the delay pushout failure criteria for setup D to CK
set_constraint_criteria \
    -type setup \
    -delay_degrade 0.08 \
    -cells { dff1 dff2 }
```

set_default_group

Specifies the criteria for creating the default group. This command can be used to specify the global criteria for the default group for all cells or to specify the criteria for specific cells and arcs.

Options

-cells {cell_names}

Specifies a list of cell names to which this command should apply. Default: *all cells* (the command applies to all cells globally)

-criteria {<delay | power | leakage | cap> <off | force_off | min | avg | max>}

Specifies the method for selecting the values in the default group data tables. Supported values are different depending on the type of group. This option accepts a list of pairs of type and value. Default: {} (use max for all criteria except leakage that uses avg)

This option replaces the `default_timing` variable and, if used globally, overwrites the value of that variable. The available types, values and their default values are:

| Type | Values | Default |
|---------|-----------------------------|---------|
| delay | off force_off min max | max |
| power | off min avg max | max |
| leakage | min avg max | avg |
| cap | min avg max | max |

The value of `force_off` tells Liberate to remove one of the rise/fall delay groups that already have state-dependent groups from the default timing group. This value is supported in the global mode only. That is, the `force_off` option cannot be used together with the `-cells` option.

-method {<default | const> <bitwise | table>}

List of matched pairs. Default: `table`

-pin {list}

List of cell pins. Default: "*" (all cell bidirectional/output pins)

-pin_dir <string>

The direction of the pins. The supported values are: `r`, `R`, `f`, `F`, `b`, and `B`. Default: `B`

`-related_pin {list}`

List of related pins. Default: "*" (all cell input/bidi pins)

`-type <string>`

The type of the arc. The valid values are: delay, constraint, leakage, and power. Default: " " (do not apply a type)

`-unateness <merge | separate>`

Keeps positive_unate and negative_unate timing groups from being merged in the default group. Default: merge (if both positive_unate and negative_unate timing groups exist, they will be merged into a single non_unate default group)

This option only applies to default timing groups and has no effect on other data groups.

Note: If all timing_sense attributes are identical, the original timing sense remains unchanged during the merge operation.

This option replaces the default_unateness variable and, if used globally, will overwrite the value of that variable.

`-when <string>`

State of the arc. Default: " " (do not apply a state).

This command is used to choose which characterized state-dependent arc should be selected for the default group. To choose a specific arc, you must provide one or more of the options: `-when`, `-type`, `-pin`, `-pin_dir`, `-related_pin`, and `-cells` that corresponds to exactly one characterized arc for the list of cells. The `-when` and `-type` options are required. The other options are optional since they have reasonable defaults. The `-cell`, `-pin`, and `-related_pin` accept a wildcard. For example:

```
set_default_group -type leakage -when "!(A1)" -cells "XOR3D1BWP"  
set_default_group -type delay -pin "Z" -related_pin "A" -when "A2 & A3" \
```

This command must be used before char_library.

Important Points to Notes:

- ❑ Multiple occurrences of this command can be issued. If this command is issued more than once for the same cell, the last command issued for the cell overrides any previous settings for that cell.
- ❑ The parameters that are replaced by this command will still be used internally when no list of cells is provided and, if currently accessible, would still be accessible through Tcl. At some time in the future, these variables will not be supported. We strongly recommend this command be utilized instead of the global variables.

- ❑ This functionality does *not* support the ability to read in a library database (ldb), modify the default group settings and write out a modified library.
- ❑ The recommended settings for Liberate are to use default groups for NLDM models, but *not* CCS or ECSV models.

Using the -method option

The `-method` option specifies the algorithm for selecting the data when creating the default group. With this option, you can specify a unique selection criterion for `const` (constraints) and `default` (all other data groups). The default setting for 'default' is `table`; the default setting for `const` is `bitwise`.

When constructing a default group using the `table` method, Liberate will find the worst value (min/max) from all the relevant tables and select the table that contains that value, or if `avg` is requested, then the table with the greatest `avg` value is selected. When the method is `bitwise`, Liberate will construct a default group by selecting the worst value (min/max) from all the relevant tables, on a bitwise basis or for `avg`, the average of the values from each table on a bitwise basis.

This option allows the user to specify the method to be applied to the timing (delay/transition), power, and cap tables (see `default`), and to the constraint tables (see `const`). The transition table in the default group will follow the timing table selection. This option requires a list of paired values. Each pair consists of a group type and a method. Acceptable group types are `default` and `const`. Acceptable selection methods are `bitwise` and `table`. This option replaces the `default_group_method` variable and, if used globally, will overwrite the value of that variable. There are 2 algorithms available when requesting an `avg` default power group. For more information, see the `default_power_avg_mode` variable.

Example

```
set_default_group \
  -method { default table const table } \
  -unateness merge \
  -criteria { delay max power avg leakage avg cap avg } \
  -cells { inv nr2 }
```

set_dependent_load

Specifies a load to be added to the specified cell:pin when the specified cell:pin is a dependent output. A dependent output is an output port of the cell that is in the path, but is not the monitored output of the timing arc being characterized.

The dependent load is used for all characterizations such as *delay*, *power*, *constraint*, *min_pulse_width* and so on.

The default is to use the same load as the active output for input-to-output arcs. For hidden power arcs, since there is no 'observed' output port, all the output ports will be given the dependent load.

Options

-cells {cell_names} List of cell names. Default: all cells

-pinlist {pin_names}

List of pin names. Default: all pins

<min | max | load_value | index_?>

Default: Apply same load as arc output pin min is the minimum delay load index value.

max Maximum delay load index value.

load_value value of load to add to dependent pins, in Farads

index_? specifies the load index value to use from load index on the delay table. It requires an integer starting with 0. Therefore, index_0 would be the first index.

This command must be used before char_library.

Example

```
set_dependent_load -cells {fdrs15} -pin {X1} 5e-15
```

set_driver_cell

Defines an active pre-driver cell to be used instead of the linear waveform. By default, Liberate uses a linear ramp as the input waveform during characterization.

Options

| | |
|--|---|
| <code>-accuracy_mode < 0 1 ></code> | Enables an algorithm for generating normalized waveforms while observing the slew behavior. Default: 1 |
| <code>-char_pin <pin></code> | Primary output pin. |
| <code>-input_transition <value></code> | Input transition time, in seconds. Default: 5e-12 |
| <code>-input_use_index</code> | Forces the input transition of the driver to use the index from the characterized cells. This option overrides (ignores) the <code>-input_transition</code> option. |
| <code>-instantiate</code> | Instantiates the driver cell in the simulation decks. For related details, see Using the -instantiate option . |
| <code>-pin_map {<driver_pin>, <cell_pin>}</code> | Specifies a list of <code>driver_cell_pin</code> to <code>char_cell_pin</code> pairs. This option defines to which cell the pin pairs in the <code>-pinlist</code> are mapped. The <code>-pin_map</code> maps by position to the <code>-pinlist</code> pins with the first <code>-pin_map</code> cell corresponding to the first <code>-pinlist</code> pair, and so on. |
| <code>-pinlist {<cell> <pin>}</code> | Specifies a list of pin pairs between driver cell output pins and characterization cell input pins. If a <code>-pinlist</code> is given, the driver cell is used for only the specific cell and pin pairs in the <code>-pinlist</code> . The wildcard asterisk (*) can be used to specify the cell names. |
| <code><driver_cell></code> | Specifies the name of driver cell. |

The pre-driver cell that the `set_driver_cell` command defines is driven to its input by a linear ramp defined by the `-input_transition` option. Liberate determines the loading on the output of the pre-driver such that the output transition of the driver cell as measured on

the `char_pin` are equivalent to the input transitions specified in the [define template](#) or [define_index](#) commands when measured at the `measure_slew*` voltage levels.

Note: All driver cells must be scheduled for characterization by `char_library` or they will be ignored. To schedule a cell for characterization, add the cell to the `char_library -cells` list, or do not use the `-cells` option at all. If the `-cells` option is not used, then all cells with a `define_cell` command and a loaded netlist (see [read_spice](#)) will be scheduled for characterization. Also, if you do not want a driver cell to be modeled in an output library, then the `define_cell` command for the driver cell should *not* refer to any templates (see [define_template](#)).

When characterizing CCS data, Synopsys recommends using a CCS predriver waveform instead of an active driver cell. For more information about this, see the variable [predriver_waveform](#).

Note: The `predriver_waveform` will override the active driver for the `related_pin` in an arc. Also, the active driver can be used to drive the side pins (See the `-instantiate` option) while the `predriver_waveform` drives the `related_pin`.

Currently, there is no limit to the number of `set_driver_cell` commands you can specify.

This command must be used before [char_library](#).

Using the -instantiate option

The `-instantiate` option allows the specified driver cell to be used to drive the specified cell/pin in the SPICE deck when the pin is a side pin and is static. By default, the driver instantiation will only be applied to side input pins that connect to a transistor channel. To apply a driver cell to the gate input of a side pin, the variable `driver_cell_all_inputs` must be set to 1. The use of this functionality can result in a significant (typically > 20%) run time penalty due to the increase of the size of the simulation deck. (See variable [driver_cell_all_inputs](#).)

Liberate supports an active driver that can simultaneously drive multiple inputs to a characterization cell. This capability allows multiple inputs to include delay offsets between related signals such as CK and CKN. If the driver cell has more than 1 output pin, use the `-char_pin` option to specify the primary output pin where slew matching is performed. The transition will be measured on the `char_pin`. If the `-char_pin` option is specified, the `-pin_map` and `-pinlist` options are also required.

Examples

```
# Set the default pre-driver with a 10ps input ramp
set_driver_cell -input_transition 10e-12 bufx16
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
# Set the default pre-driver for all CLK pins, GATER:CLKIN
set_driver_cell \
    -input_transition 10e-12 \
    -pinlist {* CLK GATER CLKIN } \
    Clkbufx4

# connect driver cell output X to inputs CK and SE on cell DFF1, and
# connect driver cell output Y to inputs CKN and SEN on cell DFF1.
set_driver_cell \
    -input_transition 6e-12 \
    -char_pin X \
    -pinlist { X CK X SE Y CKN Y SEN } \
    -pin_map { DFF1 DFF1 DFF1 DFF1 } \
    active_driver_2

# Set the active driver mode for more accurate generation of
# driver waveforms
set_driver_cell -input_transition 3e-12 \
    -accuracy_mode 1 \
    GL_CKBUFX14
```

set_driver_waveforms_file

Specifies a filename that holds the driver waveforms computed on the server side (see [set_driver_cell](#)). This command is used with the packet mode (see [Chapter 3, “Parallel Processing.”](#)) where the clients can reuse the waveforms stored in the `driver_waveform_file` without having to regenerate them. This flow cannot be used with auto-index because the client determines the slews for the cell, and the server does not know all the indices before the client characterizes the cell.

Options

`-slew_select <all | hybrid | index>`

Specifies which slews to store for reuse by the clients. Default: `all`

The supported settings are:

- `all`: select the superset of all slews as specified in the `define_template` and `define_index` commands. (Default)
- `hybrid`: select all the slews that map to the user-specified `define_arc` command. This mode assumes that the inside view is not augmenting the characterized arcs and that all characterized arcs have associated `define_arc` commands (see `char_library -user_arcs_only` and `-io`).
- `index`: select the slews specified by all the `define_index` commands.

`<file_name>`

Specifies the name of the file where driver waveforms are stored by the server in packet flow. (Required)

This command must be used before [char_library](#).

set_em_skip_monitor

Specifies the nets to be ignored for electromigration (EM) analysis. The EM characteristics of the matching nets will not be monitored.

Options

-cells {<cell_names>}

Specifies a list of cells.

Note: This option supports the use of the wildcard character " * ".

-net {<list>} Specifies a list of pins.

Note: This option supports the use of the wildcard character " * ".

-layer {<list>} Specifies a list of layers.

Note: This option does not support use of any wildcard.

This command must be used before char_library.

Examples

```
set_em_skip_monitor \
  cell { DFF} \
  -net {NET1 NET2} \
  -layer {M0 M1} \
```

The above example instructs Liberate to ignore the EM results of the nets NET1 and NET2 on process layers M0 and M1 while calculating the max_toggle_rate for any of the pins on the cell DFF.

```
set_em_skip_monitor \
  -net {VDD} \
  -layer {M0} \
```

The above example instructs Liberate to ignore the EM result of net VDD on process layer M0 for all cells.

```
set_em_skip_monitor \
  -net {TVDD} \
  -layer {*} \
```

Virtuoso Liberate Reference Manual

Liberate Commands

The above example instructs Liberate to ignore the EM result of net TVDD on all process layers for all cells.

set_gnd / set_vdd

The `set_gnd` and `set_vdd` commands identify the name(s) of ground nets and power supply nets, respectively. Liberate must know the names of the supply nets and their respective voltages. Multiple `set_gnd` and `set_vdd` commands can be specified. It is strongly recommended that all supply pins are identified using `set_gnd` and `set_vdd`, as appropriate.

Note: A warning is generated if `set_gnd` sets a pin to large positive voltage or `set_vdd` sets a pin to 0 volts.

Options

`-attributes {name value ... }`

List of attributes specified as name-value pairs to include in the `pg_pin` group. Default: none

For example:

`-attributes {leakage_bin 1.1 power_negative allow}`

Note: The preferred method of adding these types of attributes is through `write_library` with the `-user_data` option. For more information, see the description of the [write_library](#) command and the [user_data_override](#) variable.

`-cells`

List of cells which will use this supply specification. When the `-cells` option is used, the `-name_map` option should also be used. If the `-cells` option is specified without using the `-name_map` option, the supply name will default to `"gnd_allos_<gnd_value>"`, where the `gnd_value` is the voltage value of the supply with the period symbol (".") replaced by the underscore ("_").

`-combine_rail`

Combines the power and leakage from the supplies specified in the `-include` option with the supply being declared by the `set_gnd` or `set_vdd` command. The power in the combined rail cannot be ignored. The command that specifies the combined power must be executed after all other `set_gnd` / `set_vdd` commands.

Virtuoso Liberate Reference Manual

Liberate Commands

-ignore_power Ignore the power contribution from this supply net. That is, the current in the specified supply net will not be summed into any power measurement.

Note: The `ignore_power` option will be skipped if the `-cells` option is specified.

-include { list } List of power rails to be merged into the power rail specified in the `set_gnd` or `set_vdd` command. Default: {}

Note: To merge the power, the `-combine_rail` option must also be specified.

-name_map <value> Specifies the name that this supply will be called in the `output.lib` file. Name mapping is only supported when the `pg_pin` (see the `pin_based_power` variable) syntax is enabled.

The `-cells` option is often used with the `-name_map` option to change the `pg_pin` name on an individual cell basis. This allows the mapping of a global supply to a local cell-specific supply possibly at a different voltage. For example:

```
set_vdd vdd1
set_vdd -cells {INV_X1} -name_map VDD_X1 VDD 0.9
set_vdd -cells {INV_X16} -name_map VDD_X16 VDD 0.8
```

-no_model Request to not include this supply in the output `.lib`.

-type <primary | backup | internal | nwell | pwell | deepnwell or deeppwell>

Specify the power supply type. (Default: `primary`)

<net_name> Name of ground/power supply net.

<voltage_value> Voltage value (in Volts).

Liberate automatically identifies 0, GND, and VSS (case-insensitive) as ground supplies and sets them to zero volts. Use the `set_gnd` command to set them to alternative values. It is not recommended to attempt to change the voltage of the ground net 0 since this is considered the reference ground.

Liberate automatically identifies VDD (case-insensitive) as power supplies and set them to the default voltage specified by the `set_operating_condition` command. Use the `set_vdd` command to set them to alternative values.

Note: Liberate will generate a warning if `set_vdd` is used to set a pin to 0 volts.

This command should be used before the read_spice command to have the desired effect.

This command must be used before char_library.

Example

```
set_vdd -combine_rail -cells {test1 test2} -include {VDD} VDD33 1.1
```

In the above example, the contributions for power and leakage from rail VDD will be merged to VDD33 for cell test1 and test2. The VDD will appear in the library as defined power rail only (that is, no power and leakage values is specified for VDD) if -no_model is not specified for VDD in test1 and test2 in other set_vdd command.

```
# Set VDD3 to 3 volts
set_vdd VDD3 3
set_gnd BULK_GND 0
```

set_input_voltage

Creates `input_voltage` groups at the library and pin level. For a default group the naming convention is: `(default_$vdd_$gnd_$direction)`. For example:

```
input_voltage(default_VDD1_VSS1_output)
```

For multiple groups, the naming convention is: `(user_voltage_$num)`.

Options

| | |
|-----------------------------------|----------------------------------|
| <code>-vil <value></code> | Specifies voltage input low. |
| <code>-vih <value></code> | Specifies voltage input high. |
| <code>-vimin <value></code> | Specifies minimum input voltage. |
| <code>-vimax <value></code> | Specifies max input voltage. |
| <code>-cells {list}</code> | Specifies a list of cells. |
| <code>-pins {list}</code> | Specifies a list of pins. |

Example

```
input_voltage(user_voltage_0)
input_voltage(user_voltage_1)
input_voltage(user_voltage_2)
```

set_logic_condition

Specifies a logic condition to insert into an arc to reduce/filter potential vectors and thereby reduce the simulation overhead.

Options

| | |
|---|---|
| <code>-cells {list}</code> | List of cell names. Default: "*" (all cells) |
| <code>-pin {list}</code> | List of pin names. Default: "*" (all pins) |
| <code>-pin_dir string ()</code> | The pin direction [RIF]. Default: "" (apply to all directions) |
| <code>-related_pin {list}</code> | List of related pin names. Default: "*" (all pins) |
| <code>-related_pin_dir string ()</code> | The related pin direction [RIF]. Default: "" (apply to all directions) |
| <code>-type {list}</code> | The type of the arc. Supported types are constraint, delay, hidden, leakage, mpw, setup, and hold. Default: "" (Apply to all types) |
| <code><logic_condition></code> | The logic condition to be applied to filter vectors. |

This command adds and applies the specified `logic_condition` to all matching arcs. Multiple `set_logic_condition` commands can be used. However, if more than one command maps to a particular arc, the one that most closely matches the arc or the last one specified is applied. This command works in the same way as the `-logic_condition` option of the `define_arc` command and can be used to augment any `define_arc` commands that overlap with the option settings of this command without having to edit the actual `define_arc` command.

This command must be used before `char_library`.

Example

```
set_logic_condition \
  -type delay \
  -pin "Z" \
  -related_pin "A1" \
  -cells { ao32 } \
  "B1 & C"
```

set_max_fanout

Liberate can compute the *max_fanout* attribute for all output and bidi pins. The value is computed as follows:

```
max_fanout = <max_capacitance> / <input cap for refcell/refcell_pin>
```

Where:

```
<max_capacitance> == max_capacitance attribute value for an output or bidi pin  
for the cell
```

```
<input cap for refcell/refcell_pin> == the input cap of chosen pin for the  
cell.
```

Options

-refcell <value> The reference cell.

-refcell_pin <value>
 The pin of the reference cell.

-refcell_dir <Rise | Fall>
 The direction of the reference pin.

-cells {list} List of cells.

set_network_port

Defines an explicit network port number to be used for distributed library creation.

Options

<port_number> Network port number.

By default, Liberate will search for an available port. To specify the machines to use for parallel processing use multiple `set_client` commands. For more details, see on distributed parallel processing see Chapter 6.

Note: The TCP/IP port is determined by the following equation, if it has not been specified with the `set_network_port` command:

`30003 + rand() % 1413`

and falls within 300003-30259, 30261-30998, 31000-31019, 31021-31415.

This command must be used before `char_library`.

Example

```
# Set the network port on the host machine
set_network_port 20000

# Set the client machines to use for parallel processing
set_client -dir /tmp/liberate linux1
set_client -dir /tmp/liberate linux2
```

set_operating_condition

Defines the process corner, temperature and default voltage to be used for library creation.

Note: The `set_operating_condition` command will not override any supply voltage that was previously specified using the `set_vdd` command.

Options

| | |
|--|--|
| <code>-name <value></code> | Specifies the operating condition name to be used in the output library. |
| <code>-supply_name <string></code> | Instructs Liberate to use the voltage set by the <code>set_vdd</code> command as the default operating condition voltage for the specified supply in the generated <code>.lib</code> file. If this option is not specified, Liberate, by default, uses <code>-voltage</code> and <code>-temp</code> as the default operating condition in the generated <code>.lib</code> file. |
| | Note: The <code>set_vdd</code> command should be specified before the <code>set_operating_condition</code> command. |
| <code>-temp <value></code> | Defines the temperature to characterize in °Celsius. (REQUIRED) |
| <code>-voltage <value></code> | Specifies the default positive supply voltage in volts. Default: 0V (REQUIRED) This voltage will be assigned to any VDD pin name. To specify additional power or ground supply nets and their appropriate values use the <code>set_gnd</code> / <code>set_vdd</code> commands. The default supply names are VDD for the positive supply, VSS, GND, and 0 for the negative supply. |

This command must be used before `char_library`.

Example

```
# Characterize using typical process, 25°C, 1.2 Volts
set_operating_condition \
    -temp 25 \
    -voltage 1.2
```

set_output_voltage

Creates output_voltage groups at the library and pin level.

Options

| | |
|-----------------------------------|-----------------------------------|
| <code>-vol <value></code> | Specifies voltage output low. |
| <code>-voh <value></code> | Specifies voltage output high. |
| <code>-vomin <value></code> | Specifies minimum output voltage. |
| <code>-vomax <value></code> | Specifies max output voltage. |
| <code>-cells {list}</code> | Specifies a list of cells. |
| <code>-pins {list}</code> | Specifies a list of pins. |

For a default group, the naming convention is: (default_\$vdd_\$gnd_\$direction). For example:

```
output_voltage(default_VDD1_VSS1_output)
```

For multiple groups, the naming convention is: (user_voltage_\$num). For example:

```
output_voltage(user_voltage_0)
output_voltage(user_voltage_1)
output_voltage(user_voltage_2)
```

set_pin_attribute

Modifies pin attributes in the library.

Options

| | |
|------------------------------|--|
| -abstol <value> | Absolute threshold (in seconds) is used to compare the result of <code>abs(cell_rise/cell_fall)</code> . (Required) |
| | The specified absolute threshold value must not be negative. For example, use <code>1e-12</code> for <code>1ps</code> . |
| -attributes {list} | List of attributes that will be updated. Default: " " |
| | When the <code>-type</code> option is set to <code>min_max_cap_tran</code> , the supported attributes are: <code>max_transition</code> , <code>min_transition</code> , <code>max_capacitance</code> , and <code>min_capacitance</code> . The <code>-attributes</code> option can be set to these and if it is not set, it will default to these attributes. |
| -cells {list} | List of cells. Currently, it accepts only one cell name and does not support use of wildcards. (Required) |
| -pin "string" | Arc pin, that is, the output or constrained pin. (Required) |
| -related_pin "string" | Arc related_pin. (Required) |
| -reltol <value> | Relative threshold that is used to compare the result of: $\text{abs}(\text{cell_rise} - \text{cell_fall}) / (\text{cell_rise} + \text{cell_fall})$. (Required) |
| | The specified relative threshold value must not be negative, but must be less than 1. For example, use <code>0.05</code> for 5%. |
| -type "string" | The type of data to be modified. Currently, it accepts only " <code>min_max_cap_tran</code> ". (Required) |
| | When <code>-type</code> is specified as <code>min_max_cap_tran</code> , a maximum delay sub-table is chosen from the characterized delay table for the arc specified by the pin, related_pin, and cell. This sub-table is chosen according to the absolute threshold and relative threshold values provided. The attributes <code>max_transition</code> , <code>min_transition</code> , <code>max_capacitance</code> , and <code>min_capacitance</code> can be set to match the sub-table. |

-when "string" The arc conditional state, that is, the when condition.

This command can be used after char_library, read_lDb, and read_library, but it must be used before model generation such as with write_library.

You can specify multiple `set_pin_attribute` commands.

Example

```
read_lDb test.ldb.gz
set_pin_attribute -type min_max_cap_tran -abstol 4e-10 -reltol 0.8 -pin Y
-related_pin A -cells {INVX1}
write_library -filename my.lib test
```

set_pin_capacitance

Specifies how the `pin_capacitance`, `rise_capacitance`, and `fall_capacitance` attributes and the default advanced table capacitance

(`ccs_receiver_capacitance_rise` and `ccs_receiver_capacitance_fall`) models for each pin are determined. By default, the NLDM `pin_capacitance` is selected from the maximum value of all the calculated capacitances without considering the direction (rise or fall), logic state, or slew/load in the characterized state-dependent capacitance tables. By default, the advanced table-based capacitance models are selected from one of the characterized capacitance tables.

Options

`-range_use_side_input`

Changes the `capacitance_range` calculation to observe the `-side_input` option setting.

This option can be used as a post-processing step, as long as the `set_pin_capacitance` command is invoked again after the `read_ldb` command. For example:

```
set_pin_capacitance -state avg -table avg \
    -direction min \
    -side_input noncontrolling -range_use_side_input
```

`-side_input <controlling | noncontrolling | all>`

Specifies the method to be used to select different groups of vectors. Default: `all`

When set to `controlling`, the pin capacitance will be measured only for vectors where there is a side input that is controlling the output. This is usually the case only for hidden vectors.

When set to `noncontrolling`, the pin capacitance will only be measured for vectors where no side input controls the output. That is, the `related_pin` controls the output. This will not include any hidden vectors.

The default is `all`, which means to measure pin capacitance for all vectors.

Virtuoso Liberate Reference Manual

Liberate Commands

`-state <min | avg | max>`

Specifies the method to be used to select between different states. Default: `max`

`-state_avg_method <merge | separate >`

Specifies the method to be used to compute the average capacitance. Default: `merge`

For the `-state_avg_method` option to have any effect, the `set_pin_capacitance` `-state` command must have a value of `avg` and the `.lib` must have timing groups with multiple states merged into a single timing group. This option supports the values of `merge` (Default) and `separate`. For example, if there are two sets of `when` conditions, and `W1, W2, W3` are all logic cubes encompassing all input pins `when`: `W1+W2, c1 and when: W3, c2`:

merge method: $c_{avg} = (c1 + c2) / 2$

separate method: $c_{avg} = (c1 + c1 + c2) / 3$

`-table <min | avg | max>`

Specifies the method to be used to select within a table. Default: `max`

`-direction <min | avg | max>`

Specifies the method to be used to select between rise and fall capacitance. Default: `max`

`-meas_supply_cap`

Instructs Liberate to take a capacitance measurement on the supply pin. The measurement is made using the leakage deck.

Note: This option does not affect leakage characterization.

The power supply capacitance is measured by ramping down the power supply voltage. The ramping is done after leakage is measured so as to not disturb the leakage acquisition. The `CCSP_parasitic_capacitance` data structures are used to store the result. The supply capacitance will be written to the output library as a commented pin group containing a capacitance attribute.

`-when <"function">` Logic conditions of side inputs.

-vector <"vector"> Logic conditions of side inputs that must match `define_cell -pinlist`.

The `-vector` option should be consistent with the `-pin_dir` option for the ports listed in the `-pins` option. If the `-vector` option has an "x" for the `-pins` option, Liberate enforces consistency. If the vector has an R or F, the `-pin_dir` option must have rise or fall. There is no character supported in the `-vector` option to indicate both directions. That is, there is no support for a "b" in the `-vector` option.

-pin {pins} List of pin names.

-pin_dir <rise | fall | both>
Specify the pin direction.

-cell <cell> Specifies the cell name.

Using this command enables numerous selection levels. For NLDM `pin_capacitance`, the first level (table) determines whether to use the minimum (min), average (avg), or maximum (max) values from each table to roll up. The next level (state) chooses the capacitance rolled up for each table in the previous step by logic state (when condition). From the capacitances selected from each table across all logic states, a single value for `rise_capacitance` and a single value for `fall_capacitance` will be selected. The final level of refinement determines how the `pin_capacitance` attribute is determined between the `rise_capacitance` and `fall_capacitance` from the previous step. The min, avg, and max values can be selected at each level (table, state, and direction). The default at all levels is max. This command can be used independently from characterization, only impacting the attributes output by `write_library` because the characterization database (LDB) contains all the rise and fall state-dependent capacitance tables for each pin. For the more advanced table-based models, such as the `ccs_receiver_capacitance_rise` and `ccs_receiver_capacitance_fall`, the capacitance data associated with the default `hidden_power` group will be written into the `.lib`.

This command can be used to specify a particular vector or vectors to use when measuring input pin capacitance. The following options are available to support this: `-when`, `-vector`, `-pin`, `-pin_dir`, and `-cell`. The `-pin_dir` option is required when specifying capacitance vectors. The 'vector' or the 'when' options should also be specified. If multiple vectors are desired, execute multiple `set_pin_capacitance` commands or use wildcards (x) in the vector. At least one of the `set_pin_capacitance` commands must specify satisfiable logic conditions (vectors) or the output library may not have valid pin capacitance values for the pin. In addition, the `set_pin_capacitance` command must reference an arc that is characterized; that is, the arc specified must have been characterized, matching the

when, vector, pin, pin_dir and cell. The vector option must specify the same number of pins as in the pinlist option of the corresponding define_cell command.

Example

```
# Set how the pin capacitance attributes are determined
set_pin_capacitance -state max -table avg -direction min
# specify a particular vector for the input cap of pin A on cell lag.
set_pin_capacitance -vector "x0x" -pin A -pin_dir rise -cell lag
```

set_pin_delay_threshold

Sets the `input_threshold_percent_fall` and `input_threshold_percent_rise` attributes at the pin level. This must be used together with `define_arc` to set the delay thresholds. This only adds the attributes into the library and does not control the characterization (which is done by `define_arc`).

Options

`-cells { list }` List of cells. (REQUIRED)
`-pins { list }` List of pins. (REQUIRED)
`{ values }` Specify two delay measurement thresholds:
 `input_threshold_percent_rise`,
 `input_threshold_percent_fall`

Example

```
set_pin_delay_threshold -cells {cellA cellB} -pins {pin1 pin2} {.3 .2}
```

set_pin_gnd

See [set_pin_vdd](#) for description of options.

Options

| | |
|--|--|
| <code>-add_supply</code> | Create and add supply_name to cell gnd list. |
| <code>-supply_name <name></code> | Name of the supply that drives this pin. (REQUIRED) |
| <code>{cell_names}</code> | List of cell names. (REQUIRED) |
| <code>{pin_names}</code> | List of pin names. For example, {a1 a2 a3 c din ckn}, or regexp like {a* c*}. (REQUIRED) |
| <code><gnd_value></code> | Ground supply value. (REQUIRED) |

set_pin_slew_threshold

Lets you set the measurement thresholds for specific cell pins that need different sets of threshold. When this command is set, the thresholds will overwrite the measure threshold set by the `measure_slew_*` variable and the `define arc -slew_thresh` command for the specific cell(s) and pin(s).

Options

- `-cells {list}` List of cells (REQUIRED).
- `-pins {list}` Pins that need special slew threshold (REQUIRED).
- `{slew_threshold}` Specify the four slew measurement thresholds: `lower_rise`, `upper_rise`, `lower_fall`, and `upper_fall`. (REQUIRED)

set_pin_vdd

The `set_pin_vdd` and `set_pin_gnd` commands associate a *pin* of a cell with a particular supply domain. This is particularly useful on cells that have multiple power and ground connections, such as level shifters where it may be difficult for Liberate to correctly identify the supplies for each pin. The cell and pin options accepts a list of names. The cell and pin names may be specified with wildcard characters * and ? and `regexp` expressions. In the event that multiple commands are given which map to the same cell and/or pin, then the last command given takes effect.

Options

`-add_supply` Instructs Liberate to create a new supply with the name specified by the `supply_name` option or an arbitrary internal supply name if the `supply_name` is not specified (if the supply does not already exist.)

`-leakage_add_to_supply <name>` Specifies the name of a supply to which all leakage for this supply should be added. This option should be used when gate leakage on an input pin is to be added to a power pin not controlling it. This is useful when characterizing level shifters where the input is driven by a supply domain that does not exist in the cell and you do not want to lose the input power.

`-supply_name <name>` Specifies the name of the supply associated with the specified pin. (REQUIRED)

Use this option to specify the name of the supply associated with the specified pin. This is often needed with level shifter cells to avoid Liberate matching an incorrect supply to the specified pin. In addition, it also fixes `ccsn` stage generation for level shifters so that Liberate not only considers the input/output voltages of a timing arc when deciding if an arc stage is to be used, It will also check to see if the input/output shares the same voltage supplies before using the arc based constructs.

When this option is used, and the `voltage_map` variable is set to 1, the specified `supply_name` will be output in `related_power_pin/related_ground_pin` format. If `voltage_map` is set to 2, the `input/output_signal_level` attributes are used instead.

| | |
|--------------|---|
| {cell_names} | List of cells. (REQUIRED) |
| {pin_names} | List of pin names. For example, {a1 a2 a3 c din ckn}, or regexp like {a* c*}. (REQUIRED) |
| <vdd_value> | <p>Power supply value (Optional)</p> <p>When you specify the <code>supply_name</code> but do not specify the <code>vdd_value</code> and the supply exists in the vdd list (see <code>set_vdd</code>), then the vdd voltage value from the <code>set_vdd</code> command is used. If you specify both <code>supply_name</code> and <code>vdd_value</code>, ensure that the vdd value is the same as the one supplied in the <code>set_vdd</code> command. Else, an error message is generated and the command is ignored.</p> |



- 1. If there is more than one `set_pin_vdd` commands for the same pin but with a different value or different supply name, then the second instance of the command overwrites the first.**
- 2. If there are two supplies with different names but with the same voltage value, and a `set_pin_vdd` command associates a net to this voltage but the `supply_name` is not given, Liberate reports a error and exits (after reporting all similar errors.)**

Example error cases:

```
# -supply_name not given:  
set_vdd VDD1 1.2  
set_vdd VDD2 1.2  
set_pin_vdd NAND2 Y 1.2  
  
# Attempt to redefine vdd2 voltage:  
set_pin_vdd -supply_name vdd2 busDriver PAD 3.3
```

This command must be used before `char_library`.

Examples

Example 1

```
# Set the voltage swing on the input pin of a level shifter  
set_pin_vdd -supply VDD3 level_shifter_3to1 A1 3.0
```

Example 2

```
set_vdd VDD $VOLT
set_vdd VDD1 $VOLT1
set_pin_vdd -supply_name VDD1 -leakage_add_to_supply VDD \
LVLHLD1HVTIN$VOLT
```

In the above example, gate leakage currents on pin IN will be multiplied by \$VOLT1 (controlling VDD1) to get the leakage power that will be added to the supply pin VDD.

set_receiver_cap_thresholds

Allows you to override the default behavior of Liberate with respect to `ecsm_capacitance` tables.

Options

| | |
|---------------------------|--|
| <code>-rise {list}</code> | List of receiver rise capacitance thresholds specified as a percentage in decimal. (that is, .5 = 50%) |
| <code>-fall {list}</code> | List of receiver fall capacitance thresholds specified as a percentage in decimal. (that is, .5 = 50%) |

For **1-piece tables**, Liberate will report `ecsm_capacitance` at `delay_inp_rise` and `delay_inp_fall`.

For **3-piece tables**, Liberate also reports `ecsm_capacitance` at `measure_slew_lower_rise`, `measure_slew_upper_rise`, `measure_slew_upper_fall`, and `measure_slew_lower_fall`. This command allows you to augment these tables with additional voltage thresholds.

Note: The receiver capacitance thresholds must be symmetric for rise and fall arcs. The first rise corresponds to last fall, the second rise corresponds to the next to last fall, and so on. Each pair should sum up to 1. Asymmetric thresholds are not currently supported and will introduce some unwanted changes in the capacitance models.

Examples

Example 1

```
# Create ecm_capacitance tables for 20%, 30%, 50%, 70%, 80%
# for both rising and falling arcs
#
set_receiver_cap_thresholds \
  -rise {0.2 0.3 0.5 0.7 0.8} \
  -fall {0.2 0.3 0.5 0.7 0.8}
```

Example 2

```
# Compact version of Example 1, where:
# delay_*=0.5
# measure_slew_lower_*=0.3
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
# measure_slew_upper_*=0.7
#
set_receiver_cap_thresholds -rise {0.2 0.8} -fall {0.2 0.8}
```

Example 3

```
# Create 8 piece ecssm_capacitance tables that are weighed towards the tail of the
arc
#
set_receiver_cap_thresholds \
  -rise { 0.01 0.3 0.5 0.6 0.7 0.8 0.9 0.9999} \
  -fall { 0.0001 0.1 0.2 0.3 0.4 0.5 0.7 0.9}
```

set_rsh_cmd

Defines an `rsh_cmd` string to be used with a specified list of cells when accessing a remote client through cell-based distributed parallel processing. This can be used to send a select group of cells to a specific queue (possibly one with different compute resources.)

Options

| | |
|-------------------------------|--|
| <code>{rsh_cmd_string}</code> | String to be used for the <code>rsh_cmd</code> . |
| <code>{cells}</code> | List of cells to which this <code>rsh_cmd</code> is applied. |

Important Points to Note

- ❑ If there are more than one cell in a packet and not all the cells in that packet use the same `rsh_cmd`, Liberate uses the `rsh_cmd` associated with the cell estimated to require the most effort.
- ❑ This command is supported only in cell packet mode (see [packet mode](#)).

This variable must be used before [char_library](#).

Example

```
# Setup an rsh_cmd for a "smallQue" machine
set_var packet_clients 10
set_var rsh_cmd "bsub -q smallQueue"

# Setup an rsh_cmd to send some big cells to a queue on a different machine
set_rsh_cmd "bsub -q bigQueue" {bigCellA bigCellB}
char_library
```

set_sim_init_condition

Instructs Liberate how to set initial conditions when presenting spice decks to external simulators, that is, it allows local/global control of `sim_init_condition`. See also [**char_library -extsim**](#).

Options

`-cells {list}` List of cell names to which this command should be applied.
Default: all cells

`-floating_node < init | ignore >`
Tells Liberate to initialize (default) floating nodes, or to ignore floating nodes. Default: `init`

When ignored, the external simulator can initialize the nodes or the user can provide the initialization using the `-extsim_deck_header` option with the [define_arc](#) command. When providing initial conditions with `-extsim_deck_header`, the node names must map into the deck. This requires the addition of an additional level of hierarchy of "x1 ." to the cell nodename in the `.ic` command.

`-method < hybrid | ic | ic_except_dc | ic_except_dc_plus_leakage>`
Specifies the method for setting initial conditions. Default: `hybrid`

This is the same as the `sim_init_condition` variable. Accepted values are:

- `hybrid`: Use `.ic` for any floating nodes and `.nodeset` everywhere else. (Default)
- `ic`: Always use `.ic` to initialize nodes in the external simulation.
- `ic_except_dc`: Use `.ic` to initialize nodes in all external simulations except when the measurement is a DC type measurement. This can occur for leakage measurements.
- `ic_except_dc_plus_leakage`: Same as `ic_except_dc` except that it forces to use `.ic` to do leakage characterization.

`-vector_skip_open_source_drain`

Tells Liberate to skip vectors where a floating low node has no pull-down or a floating high node has no pull-up. This option might be useful, for example, when there is an inverter surrounded by many decap cells. Here, the decap cell has bi-stable states - in one state both nodes are floating; in another state both nodes are connected. For example:

```
M1  VDD  N1  N2  VDD  PMOS
M2  N1  N2  VSS  VSS  NMOS
```

when $N1=0, N2=1$, both nodes are driven
when $N1=1, N2=0$, both nodes are floating, but this is an unstable state.

When the `-vector_skip_open_source_drain` option is used, the vector with $N1=1$ and $N2=0$ will be skipped.

This command must be used before char_library. It has multiple uses.

Examples

Example 1

```
# Request the use of .ic for all nodes needing initialization except for leakage decks (leakage uses the DC solution).
# Do not initialize floating nodes. Let the simulator handle the initialization.
set_sim_init_condition -method ic_except_dc -floating_node ignore
```

Example 2

```
# Request that the cell called myCell uses .ic to initialize all nodes
set_sim_init_condition -method ic -cells { myCell }
```

set_simultaneous_switch

Disables simultaneous switching input analysis on a cell-by-cell basis. See the [simultaneous_switch](#) variable for more information about simultaneous switching input analysis.

Options

| | |
|---------------------------|---|
| <code>-off</code> | Disable simultaneous_switch on the list of cells. |
| <code>{cell_names}</code> | Names of cells. (REQUIRED) |

Example

```
set_simultaneous_switch -off { mycell }
```

set_three_state

Disables three_state modeling for a list of cells. This can be used for cells such as one-hot muxes to disable the modeling of the outputs as three-state outputs.

Options

`-current_degradation_threshold <value>`

Specifies the current degradation threshold. Default: 0.1

The threshold specifies a ratio of the original current. When the current falls below this threshold, the output is considered off. When this option is used, all disable arcs will be characterized with a disable_method of 2 (current degradation).

`-off` Turn off three state modeling.

`{ }` List of cells.

This command can be used after **char_library** except in one case. If the

`-current_degradation_threshold` option is specified, this command must be used before `char_library`.

Example

```
set_three_state -off { my list of cells }
```

set_units

Specifies the timing, capacitance, leakage power, and current units to be used in the output library. This command *only* affects modeling, and has no effect on either the VDB or the template files.

Units can be uppercase or lowercase; internally Liberate is case-insensitive and understands that "1mw" and "1mW" are the same. The purpose of this command is to output a library containing the correct units for your down-stream tools.

Options

-capacitance < 1pf | 100ff | 10ff | 1ff >

Specify the capacitance units. Default: 1pf

-current < 1a | 1ma | 1ua >

Specify the current units. Default: 1ma

-leakage_power < 1mw | 1uw | 1nw | 1pw >

Specify the leakage power units. Default: 1nw

-pulling_resistance < 1ohm | 10ohm | 100ohm | 1kohm >

Specify the pulling resistance units. Default: 1kohm

-timing < 1ns | 100ps | 10ps | 1ps >

Specify the timing units. Default: 1ns

This command must be set before any command that creates a library, such as [write_library](#) or [write_verilog](#), and can be set after [char_library](#).

Example

```
# Set the timing units to 1 pico second.  
set_units -timing 1ps
```

set_var

Sets Liberate-specific parameters. The available Liberate parameters are defined in the [Liberate Parameters](#) chapter.

Options

| | |
|--|--|
| <code>-cells</code> | List of cells. Default: all cells |
| <code>-pin {pins}</code> | List of destination pins for the arc (typically, output pins for combinational arcs, input pins for timing constraint, or hidden power arcs). (REQUIRED) |
| <code>-pin_dir <R F></code> | Transition direction of pin(s). |
| <code>-related_pin {pins}</code> | List of related pin names (typically input pins for combinational arcs, clock pins for timing constraint arcs). |
| <code>-related_pin_dir <R F></code> | Transition direction of related pin(s). |
| <code>-type < constraint delay delay and power hold leakage mpw nochange power recovery removal setup ></code> | Type of arc (Default: <i>all types</i>) |
| <code><name></code> | Variable name (REQUIRED) |
| <code><value></code> | Variable value (REQUIRED) |

The options `-cells`, `-type`, `-pin`, `-pin_dir`, `-related_pin`, and `-related_pin_dir` are used to specify local cell and arc specific variables and their corresponding values. All options are not valid for all parameters. If an option is not allowed, an error is issued and the setting is ignored. If an option is omitted, any value for that option is allowed. The options `-cells`, `-pin`, and `-related_pin` support the usage of the wildcards `*` and `?`. Some variables can only be set at a global level. If you specify local cell and arc variable that can only be applied globally, a warning is issued in the log file and `set_var` is ignored.

Example

```
set_var -cells DFF* write_logic_function false
```

set_vdd

See [set_gnd / set_vdd](#) for details on this command.

write_datasheet

Writes a datasheet in the format specified by the `format` option. If the format is `text` the datasheet is written to a file named `filename` unless no `filename` option is specified in which case the datasheet is written to a file called `library_name.txt`.

Options

| | |
|---|---|
| <code>-cells</code> | Writes specified cells to the datasheet. |
| <code>-conditional</code> | Includes writing out all conditional arcs. Default: write out data in the default groups only. |
| <code>-dir <dir_name></code> | HTML directory name. |
| <code>-exclude</code> | Excludes specified cells from being written to the datasheet. |
| <code>-filename <file_name></code> | Output file name. |
| <code>-format <text html pdf ps></code> | Datasheet format. Default: "text" |
| <code>-groups</code> | Writes only specified cell groups to the datasheet. |
| <code>-include_indices</code> | Include indices in HTML reports. |
| <code>-logo <file_name></code> | Logo in GIF or JPG format. |
| <code>-map {list}</code> | List of name-map pairs that are used to map an internal name to an external name. Default: none (which means there is no name mapping.) |
| <code>-table_style</code> | Specifies where to create the datasheet tables from first-mid-last or min-avg-last. Default: first-mid-last |
| <code><library_name></code> | Library name for the datasheet and filename prefix. |

This command can be used after `char_library`, `read_ldb`, and `read_library`.

The datasheet includes library information for each cell group. A cell group can be specified using the `define_group` command or it can be inferred from the footprint attribute. The cell group information includes the name of each cell in the group, the logic function, the pin capacitance, the area and any relevant leakage, delay, power and timing constraint information. The delay, power and constraint information is written as a table that includes the minimum, middle and maximum entry from the respective characterized table values.

If the `html` format is requested, the `-dir` directory name must also be specified. A collection of `<cell_group>.html` files will be written to this directory. The datasheet can be viewed by opening the `<dir>/index.html` file with an internet browser. If the `-logo` option is specified, a reference to the logo (gif or jpg) will be included at the top of each groups `html` file. The logo file must exist in `<dir>/<logo>`. If a schematic symbol for a cell group is available in the file `<dir>/sym/<cell_group>.jpg` then it will also be included. The datasheet writer is written using Tcl API of Liberate and is available for customization in the file `$ALTOSHOME/etc/datasheet.tcl`.

Note: if the cell group include only one cell, the symbol picture should be named as `<cell_group>.jpg`. However, if the cell group includes more than one cell, the symbol picture should be named as `<cell_group>x.jpg`.

The `-format` option also supports the values `pdf` and `ps` for writing out PDF and PostScript (`.ps`) format files. This is done by converting the HTML datasheet output first to postscript and then to PDF using two tools: `html2ps` and `ps2pdf`. These tools need to be installed and in the command `PATH` for postscript and PDF generation to work. They are available for download from the web. The recommended setting is `html`.

The `-include_indices` option can be specified to request Liberate to include the indices for delay, power and constraint tables when the format `html` is generated.

By default, with the `-table_style` option, the datasheet tables are created from the first point in any table, the last point, and the mid point. If set to `min-avg-max`, the `min`, `avg`, and `max` values from the table are used.

The `-map` option provides for mapping internal signal names to an external name. Options are provided as a list of signal name pairs. Example:

```
write_datasheet -map {intA myA intB myB}
```

Signal "intA" will be output as "myA" in the datasheet. ("intB" and any other signals will be handled similarly.)

The `write_datasheet` command outputs the group description defined by the `define_group` command in the HTML datasheet format. If the `-conditional` option is used, all conditional arcs are also written out. The `write_datasheet` command will also create the directory specified by `-dir`, unless it already exists.

This command must be used after a database has been loaded (see `char_library`, `read_idb`, and `read_library`).

Sample output

```
Delay(ns) to Q rising (conditional):  
Cell NameTiming Arc(Dir)WhenDelay(ns) :MinMidMax
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
DFF3TCLK->Q (RR)-0.02610.04650.1452
SET->Q (FR)CLK&D0.04390.06600.2019
SET->Q (FR)CLK&!D0.04580.06590.2026
SET->Q (FR)!CLK&D0.03240.05330.1671
```

SET->Q (FF)



Direction of related pin.

Examples

```
# Output the new datasheet in text format to tt.txt
write_datasheet tt

# Write a HTML datasheet
write_datasheet -format html -dir tt_html \
    -logo liberate.gif tt
```

write_ibis_file

Writes an IBIS formatted model. See [Creating an IBIS file with multiple \[Model\] sections for a single device](#) (the section on IBIS file creation in this manual for more information).

Options

-append_file Tells Liberate to append to an existing file instead of writing a new file. Default: `off` (don't append)
Note: This option must be used with `-no_header`.

-gndclamp_max <value> Power Clamp max cutoff. Default: $1e-6$

-gndclamp_min <value> Power Clamp min cutoff. Default: $-1e-6$

-no_header Tells Liberate to create only the [Model] section, and to omit the file header, the [Component] section, and the [Model Selector] section. Default: `off` (print all sections.)
This option together with `-append_file` are used to create an IBIS file with multiple [Model] sections for a single device. (See [Creating an IBIS file with multiple \[Model\] sections for a single device](#).)

-pulldn_max <value> Pull-up max cutoff. Default: $1e-6$

-pulldn_min <value> Pull-up min cutoff. Default: $-1e-6$

-pullup_max <value> Pull-up max cutoff. Default: $1e-6$

-pullup_min <value> Pull-up min cutoff. Default: $-1e-6$

-pwrclamp_max <value> Power Clamp max cutoff. Default: $1e-6$

-pwrclamp_min <value> Power Clamp min cutoff. Default: $-1e-6$

Virtuoso Liberate Reference Manual

Liberate Commands

```
-rev <value>           IBIS file revision. Default: "v1.0"
-versions { }           IBIS versions. Default: "4.2"
files {filenames}      List of IBIS files to create.
ldbs {ldb_filenames}   List of LDB files for typ/min/max corners.
cells {cell_names}     List of cell names
```

Note: The `-pwrclamp*`, `-gndclamp*`, `-pullup*`, and `-pulldn*` options can be used to filter from the output `.ibs` file the data that might not be desired.

write_lDb

Creates a library database (LDB). The LDB can then be used in a later Liberate session for formatting the library data, for example, creating a datasheet or generating a Verilog file.

Options

| | |
|-------------------------------|--|
| <code>-overwrite</code> | Disables the automatic version control, and if the output library already exists, it is overwritten. |
| <code>-rename</code> | Renames an existing LDB file. The default is to not rename the previous LDB and create a unique file name for the new LDB instead. |
| <code><filename></code> | A library database file in ldb format. |

Liberate will automatically save each cell as it is characterized to an LDB in the current directory named *altos.ldb.<#>*, where # is the process ID. The `write_lDb` command renames this file to the name given in the `write_lDb` command.

Important

It is recommended that the `write_lDb` command should be executed immediately after the `char_library` command and before any model creation commands such as `write_library`. This is highly recommended to ensure that a clean, unmodified copy of the LDB is saved for future use. This is important because, for example, when user data is loaded with `write_library -user_data`, the user data will modify the internal database and any LDB subsequently saved will contain those modifications.

If the `gzip` utility from GNU exists in the search path, the LDB will be gzipped automatically. This library database is named `<filename>.gz`.

Example

```
# characterize the library
char_library
# save the library database to tt.ldb
write_lDb tt.ldb
```

write_library

Outputs the library in Liberty format using Tcl API routines. (See [ALAPI Reference Manual](#)). The Tcl script for formatting the library is provided in the file \$ALTOSHOME/etc/format_library.tcl. The write_library command should be specified in this file.

Options

`-bus_syntax "<>" | "[]" | "()"`

Specifies the bus syntax characters used when outputting the library and the bus_naming_style attribute. (See also the `-expand_buses` option.)

`-capacitance_only`

Disables the output of `rise_capacitance` and `fall_capacitance` attributes. The output library will only have a single capacitance attribute.

Note: This option is useful for backward compatibility. Care should be taken when using this option.

`-capacitance_range < 0 | 1 | 2 >`

Requests the output of `rise/fall_capacitance_range` attributes into the library. The supported values are:

- 0: Omit.
- 1: Include the rise and fall range spanning from the min of the `min_capacitance` values to the max of the `max_capacitance` values. This method has been reported to cause timing issues in PrimeTime. (Default)
- 2: Include the rise and fall capacitance ranges where both range limits are both set to the rise/fall capacitance attribute values:

```
rise_capacitance_range = "<rise_capacitance>,  
<rise_capacitance>"
```

```
fall_capacitance_range = "<fall_capacitance>,  
<fall_capacitance>"
```

`-ccs`

Include CCS data.

`-ccs_compact`

Outputs a library in the compact CCS format.

`-ccs_compact_lc`

Generates a compact CCS output by feeding the Liberate output library into Library Compiler.

Virtuoso Liberate Reference Manual

Liberate Commands

`-ccs_precision` Controls the precision used when writing CCS waveform data into the generated library. The value for this option must conform to the standard Tcl formatting. Default: "%g"

`-ccs_voltage_waveform`
Includes CCS driver voltage waveforms in the output library. The construct is known as `normalized_output_voltage_waveform`. The `-ccs` option must also be used. Default: Do not include.

`-ccsn` Include CCSN (noise) data.

`-ccsp` Include CCSP (power) data.

`-ccsp_compact` Outputs a library in the compact CCSP format.

`-cells {cell_names}`
Specifies which cells get written to the output library. If `-exclude` is also set, then only cells not listed in the `cells` list will be output. Default: all cells get written

Note: This option supports the use of a wildcard.

`-dcnoise_abstol <tolerance>`

Specifies tolerance used while merging similar DC noise templates. Default: 1e-6 (volts)

If the noise values are less than this tolerance, the templates will be merged into a single group.

`-dcnoise_prefix <prefix>`

Controls the prefix used for writing DC noise templates. Default: "DC_"

Virtuoso Liberate Reference Manual

Liberate Commands

`-derive_max_capacitance`

Computes the pin-based `max_capacitance` simple attribute from the characterized values in the `rise_transition`/`fall_transition` tables. In order for this option to work, the `rise_transition`/`fall_transition` tables must be characterized with `index_1` values covering the `max_transition` which is specified by the control variable `max_transition` or the `define_max_transition` command. If the `index_1` (transition) values do not cover the `max_transition`, then Liberate will issue a warning during the `write_library` phase. In addition, this option will honor all of the following option/commands as long as they are used in the same Tcl script:

```
define_max_transition
define_max_capacitance_attr_limit
define_cell -ignore_input_for_auto_cap
```

`-driver_waveform`

Outputs normalized driver waveforms into the library. For the output to include the driver waveform, the LDB/VDB must contain the driver waveform data. If the Tcl contains multiple `write_library` commands, the first command using this option will enable the waveform output for all subsequent `write_library` commands. Normalized driver waveforms will not be output for user defined PWLs which are incompletely specified or use wildcards.

`-driver_waveform_size <value>`

Sets the number of voltage points in the normalized driver waveform `index_2`. The normalized waveform currently uses an arbitrary number of voltage points uniformly distributed from gnd to vdd. The number of points can be controlled using this option. Default: 500

`-ecsm`

Include ECSM data.

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|----------------------|---|
| -ecsm_ccs | Requests a library that contains both CCS and ECSV timing constructs. This option can be combined with "-ccs -ccsn -ccsp" or "-ecsm -ecsmn -ecsmpl". The CCS and ECSV noise and power formats cannot be merged in the output library. The correct usage is "-ecsm_ccs -ccsn -ccsp" (<i>recommended</i>) or "-ecsm_ccs -ecsmn -ecsmpl". Merging these formats produces a very large library. Due to the large size of a combined library, it is preferable to build a library with only CCS data for PrimeTime and only ECSV data for Tempus. Use the -ecsm_ccs option when a single library is desired for use with both timers. |
| -ecsmn | Include ECSV noise data. |
| -ecsmpl | Include ECSV power data. |
| -em | Include Electromigration data. |
| -exclude | Exclude cells given by -cells. |
| -expand_buses | Specifies the library should be output with individual pins and no buses. |
| -filename <filename> | Specifies the name for Liberate to write out the library. If <i>filename</i> is not specified the library is written to < <i>library_name</i> >.lib. |
| -gzip | Compresses the output file using gzip. If the output library file already exists, a warning will be given and a unique file name will be generated using the given name suffixed with a unique number. (See <i>overwrite</i> option.) |
| -indent <number> | Specifies the number of spaces to indent by. Default: 2 |
| -map {list} | List of name-map pairs, used to map internal names to an external name. Default: none (No name mapping.) |
| -internal_name | The option modifies the final name used for the internal node in the library. It is usually the case that the internal pin node name contains character that can not be used in a .lib format. Therefore, it is required to use a simpler name for such internal pins. |
| -overwrite | Disables the automatic version control, and if the output library already exists, it is overwritten. |

Virtuoso Liberate Reference Manual

Liberate Commands

`-precision <precision>`

Controls the precision used when writing out the library. The value for this option must conform to standard Tcl formatting. Default: "%g"

`-preserve_user_data_precision { }`

Tells Liberate to preserve the precision of attributes in the `user_data` file and not to apply the `precision` to them.

Note: If user data contains quotation marks, such as `myData ("1.001")`, this will be preserved in the output.

`-remove_failed <none | data | cell>`

Instructs Liberate on how to handle failed characterization data when writing to the library. Default: "" (data for Liberate MX, Liberate AMS, Variety and none for other products.)

The following are the supported values:

- `none`: Does not remove any failed data. See the variables `mark_failed_data`, `mark_failed_data_replacement`, and `constraint_failed_value` for available user controls.
- `data`: Removes the arcs with failed data from the cell in the output library.
- `cell`: Removes the cell that includes any arc with failed data from the output library.

`-rename`

Instructs Liberate to test for the existence of the output library. If the output library exists, rename the existing file before writing the library. The existing file will be renamed using the next available unused numerical index. By default, the `write_library` command tests for the existence of the output library and if it exists, a warning is printed and the output is written to the next available unused numerical index.

`-scan_ccsn_remove` Remove CCSN from scan dummy cell.

`-scan_dummy_scale_power`

Enables power data scaling for power tables while removing scan pins. The `define_cell` command must include scan-related information such as the scan pins for this option to work.

Virtuoso Liberate Reference Manual

Liberate Commands

-scan_output_dummy Converts sequential cells (latches, flops) to scan dummy cells by removing the scan pins. This option removes all scan pins from a cell and writes out the reduced cell.

-sdf_cond_equals { "==" | "===" | "==" logical" | " "

Specifies how `sdf_cond` attributes are written. Default: " "
(none)

The following table explains supported values:

| Value | Output |
|---------------|--------------------------------|
| "==" | "a == 1'b1 && b == 1'b0 ..." |
| "===" | "a === 1'b1 && b === 1'b0 ..." |
| "==" logical" | "a == 1 && b == 0 ..." |
| default | "a && ~b ..." |

-sdf_edges Enables the output of the `sdf_edges` attribute.

-sensitivity_file <filename>

When this option is specified, Liberate reads the specified sensitivity file. The sensitivity file is created by Variety using the `write_variation` command with the `-format "sensitivity"` option. When this option is used, the OCV delay groups in the sensitivity file will be merged into the nominal timing library. Both the `add_margin -sensitivity_file` and `write_library -sensitivity_file` options can be used in the same run.

-si Include SI data.

-skip {leakage | power | hidden_power | conditional_hidden_power | nochange}

Disables the output of the specified data type into the output `.lib` file. Default: none (do not skip any data)

The characterization of power (see `char_library -skip {power}`) cannot be skipped when CCS data is desired since Liberate needs the hidden power simulations to generate the receiver pin caps. Specifying `hidden_power` skips the output of hidden power arcs. Specifying `conditional_hidden_power` skips the output of conditional hidden power arcs. This capability is useful when characterizing a library using different SPICE models for timing and power.

Virtuoso Liberate Reference Manual

Liberate Commands

-swap_index_order Swaps the index order for 2D tables specified by the `swap_index_order_templates` list. Default: Use the order specified by the `read_library`, `define_template` or `read_ldb` commands.

-swap_index_order_templates {list}

Specifies LUT template types whose indexes will be swapped.
Default: all

It is a list of 2D templates processed by `swap_index_order`.
Valid values are one or more specific template names used in
the library or all. Default is all 2D templates.

An example that swaps constraint indices for all constraints that
use a template named `constraint_template_3x3` is given
below:

```
write_library -swap_index_order -  
swap_index_order_templates { constraint_template_3x3 } -  
filename test.lib test
```

-sync_ldb

Forces Liberate to read the ldb on disk prior to writing. This
option is used to address possible precision issues between the
`char_library` and `read_ldb/write_library` flows.

-thread <number>

Number of different CPU threads to use. This option defines the
maximum number of threads to use on the current machine. By
default, Liberate uses single thread when formatting and writing
out library files. Running on two or more threads will reduce the
wall time needed to output library files. The number of threads
should be chosen such that the machine will not be overloaded.

-unique_pin_data

Outputs unique data, such as timing and power, for each bus or
bundle member. It specifies that original pin names are to be
used inside the when condition string without going through the
post-processing of changing pin names to bundle names.

Virtuoso Liberate Reference Manual

Liberate Commands

`-user_data <filename>`

Specifies user-provided data in Liberty format to be merged with the current library. This is useful for including non-characterized data such as wire-load models in the output library. Once this user-data is merged into the current library, all subsequent `write_library` commands will output the merged constructs as part of the output library. If this is not desired, then separate runs of Liberate consisting of `read_lDb` and `write_library` must be executed. Any valid construct that is present in the user-provided but not present in the current library database will be copied to the output library, with the following exceptions:

- Attribute `slew_derate_from_library` is not copied.
- Attributes `function`, `state_function` and `area` will override values in the current library.
- Groups `state_table`, `ff` and `latch` will override the equivalent groups in the current library.

`-vector <"stimulus">`

Vector stimulus used to simulate the leakage, each bit can be one of `X` `1` `1` `0`.

`-vector <vectors>`

Outputs the user-defined sensitization vectors into the output library. The vectors specified using the `-vector` and `-prevector` options of the `define_arc` command are known as user-defined vectors.

`<libname>`

The output library name.

The `write_library` supports buses. Buses can be defined by `define_cell`, in the `ldb`, or by the `define_bus` command. For each defined bus, a bus template is created in the library header (group name "type"). A `bus_naming_style` attribute is also created. For each bus, all the timing, power, ccsn data, etc. for that bus pin is represented once under the bus group with only capacitance, min/max_transition attributes given for each pin. Note that this can result in a loss in accuracy, as all the data is taken from the first bus bit (the from index).

The `-ccs`, `-ccsn`, `-ccsp`, `-ecsm`, `-ecsmn`, `-ecsmpp`, `-em`, and `-si` options enable inclusion of Liberty CCS timing, CCS noise, CCS power, ECSV, ECSV noise, ECSV power, electromigration (EM), and SI data in the output library if it exists in the characterized database (`ldb`). By default, only leakage values and NLDM timing and NLPM power table data are written. ECSV and CCS data are not written into the same library.



Due to differences in the power data liberty format for CCSP and ECSMP, `write_library` can output *either* power format in a Liberate run, *but not both*. If both power formats are required, then separate Liberate runs must be used with a `read_ldb/write_library` flow.

Libraries that contain CCSP data or ECSMP data cannot be created using `write_library` in the same run with other libraries. That is, do not use `write_library -ccsp` or `set_var voltage_map <1|2>` in the same Liberate run with other `write_library` commands. Currently, the following libraries can be written in a single Liberate run: NLDM only, CCS, SI, ECSM and/or CCSN. The CCSP library must be in its own separate Liberate run. Example:

```
read_ldb <ldb>
write_library -ccsp <ccsp.lib>
```

Examples

Example 1

```
read_ldb complete.ldb

# Output a library without SI and merge my.lib
write_library -ccs -ecsm -user_data my.lib no_si

# Output a library with SI but no ECSM or CCS
write_library -si -user_data my.lib si_only

# Output a library for cells "AND2" and "OR2"
# Output cells AND2 and OR2 only, no SI, CCS or ECSM
write_library -cells {AND2 OR2} and2_or2_only

# Omit cells AND2 and OR2, no SI, CCS or ECSM
write_library -cells{AND2 OR2} -exclude no_and2_or2

# changed the sdf_cond style
write_library -sdf_cond_equals "==" sdf_equals.lib
```

Example 2

By default, that is, without `-unique_pin_data`:

```
bus (DOUT) {
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
bus_type : bus_DDR3_DOUT_2_0;
pin (DOUT[2]) {
    direction : output;
}
pin (DOUT[1]) {
    direction : output;
}
pin (DOUT[0]) {
    direction : output;
}
timing () {
    related_pin : "CONTROL";
    timing_sense : non_unate;
    timing_type : rising_edge;
    cell_rise (delay_template_7x7) {
        ....
    }
    rise_transition (delay_template_7x7) {
        ....
    }
    cell_fall (delay_template_7x7) {
        ....
    }
    fall_transition (delay_template_7x7) {
        ....
    }
}
internal_power () {
    related_pin : "CONTROL";
    rise_power (power_template_7x7) {
        ....
    }
    fall_power (power_template_7x7) {
        ....
    }
}
```

Example 3

With `-unique_pin_data`:

```
bus (DOUT) {
    bus_type : bus_DDR3_DOUT_2_0;
    pin (DOUT[2]) {
        direction : output;
        timing () {
            related_pin : "CONTROL";
            timing_sense : non_unate;
            timing_type : rising_edge;
            cell_rise (delay_template_7x7) {
                .....
            }
            rise_transition (delay_template_7x7) {
                .....
            }
            cell_fall (delay_template_7x7) {
                .....
            }
            fall_transition (delay_template_7x7) {
                .....
            }
        }
        internal_power () {
            rise_power (power_template_7x7) {
                .....
            }
            fall_power (power_template_7x7) {
                .....
            }
        }
    }
    pin (DOUT[1]) {
        direction : output;
        timing () {
            related_pin : "CONTROL";
            timing_sense : non_unate;
            timing_type : rising_edge;
            cell_rise (delay_template_7x7) {
                .....
            }
        }
    }
}
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
}

rise_transition (delay_template_7x7) {
    .....
}

cell_fall (delay_template_7x7) {
    .....
}

fall_transition (delay_template_7x7) {
    .....
}

internal_power () {
    rise_power (power_template_7x7) {
        .....
    }

    fall_power (power_template_7x7) {
        .....
    }
}

pin (DOUT[0]) {
    direction : output;
    timing () {
        related_pin : "CONTROL";
        timing_sense : non_unate;
        timing_type : rising_edge;
        cell_rise (delay_template_7x7) {
            .....
        }

        rise_transition (delay_template_7x7) {
            .....
        }

        cell_fall (delay_template_7x7) {
            .....
        }

        fall_transition (delay_template_7x7) {
            .....
        }
    }

    internal_power () {
        rise_power (power_template_7x7) {
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
      .....
}
fall_power (power_template_7x7) {
      .....
}
}
}
}
```

write_template

Creates a Liberate Tcl command file template by reading an existing library (.lib) or library database (LDB). The Tcl command file is named based on the user-input provided for the `<filename>` option (.tcl is appended to the name if it does not end with .tcl). The Tcl file includes all the necessary `define_template` and `define_cell` commands needed to run Liberate. This function provides a convenient way to use an existing library's templates to create the Tcl file to characterize a new library.

Options

| | |
|---------------------------------------|---|
| <code>-abs_tol <value></code> | Tolerance for comparing templates. Default: 0.0 |
| <code>-auto_index</code> | Generates templates suitable for use with <code>-auto_index</code> option of <code>char_library</code> . When <code>-auto_index</code> is used, all cells refer to a single delay and power template whose size is denoted by the <code>-index_delay</code> string (default "7x7") and a single constraint template whose size is denoted by the <code>-index_const</code> string (default "3x3"). If the <code>-si</code> and <code>-index_si</code> options are set, the cells will refer to a single SI immunity template of size <code>index_si</code> (default <code>index_delay</code>). The values of the indices in the templates are used as scaling factors for the indices automatically determined from the minimum/maximum transition and minimum load where these are extracted from the input library by <code>write_template</code> . |
| <code>-ccsn</code> | Set this to request <code>define_arc</code> commands for CCSN arcs in the output template. |
| <code>-cells {cell_names}</code> | Controls which cells should be written to the template. Default: <i>all cells</i> This option supports the use of a wildcard. Note: If <code>-exclude</code> is also set, then the cells not listed in the <code>-cells</code> list will be excluded. |
| <code>-combine_rise_fall_index</code> | Create <code>index_1</code> and <code>index_2</code> ranges spanning the maximum/minimum values from both rise and fall arcs. |

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|--|--|
| <code>-define_index</code> | Generates a <u>define_index</u> command for each arc of a cell whose indices differ from the default indices for the cell. By default, <u>write_template</u> assumes all the data of the same type (delay, power etc.) within a cell uses the same template. For certain types of cells there may be distinct indices for different paths within that cell, where the slew and loading conditions used for characterization are different. |
| | Note: The <code>-define_index</code> option cannot be used with the <code>-auto_index</code> option. |
| <code>-dir <directory_name></code> | Specifies a directory that stores the templates spilt into sub-templates for cells or cell groups. Default: do not split |
| | The files will be named in the following format: <code><dir>/<cell>_template.tcl</code> |
| | Note: To create unique templates per cell group, specify the <code>-dir</code> , <code>-group_cells</code> , and <code>-unique</code> options. |
| <code>-driver_waveforms</code> | Converts normalized driver waveforms into <u>define_input_waveform</u> in template. |
| <code>-exclude</code> | Excludes the cells given by the <code>-cells</code> option. |
| <code>-expand_buses</code> | Generate <u>define_cell</u> commands using <u>bus_syntax</u> if buses exist in the input library. |
| <code>-group <number></code> | Specifies the Tcl file will only contain cell definitions and templates for a <code><number></code> of cells per group. A group is created by either using the <u>define_group</u> command or by sharing the same footprint name. This option can be useful to generate a list of cells for a trial characterization run with a representative subset of the cells in the library. Default: all |
| <code>-group_cells</code> | Generate unique template per cell group; use with the <code>-dir</code> option. |
| | Combines cells belonging to the same group into a single file as <code><group_name>_template.tcl</code> . (See <u>group_attribute</u> and <u>define_group</u>) |
| <code>-ibis</code> | IBIS template mode. |
| <code>-ibis_char_prefix "string"</code> | IBIS char script file prefix. Default: <code>ibis</code> |

-ibis_gen_prefix "string"
IBIS generation script file prefix. Default: igen

-ibis_slew "value" IBIS input slew. Default: 1n (ns)

-ibis_tmpl_name "file_name"
IBIS template file name. Default: "template_ibis.tcl"

-index_const "value"
Generates the number of constraint indices for "DATAxREFERENCE". Default: 3x3
For related details, see also [Using the index * options](#).

-index_delay "value"
Generates the number of delay/power indices for "SLEWSxLOADS". Default: 7x7

-index_mpw
Generate this number of mpw indices for "SLEWSxLOADS". Default: 3x2

-index_si
Generates the number of si_immunity indices for "WIDTHSxLOADS". Default: the value specified using the -index_delay option

-input_supply_pin
Outputs `set_pin_gnd` and `set_pin_vdd` commands into the template file. For this feature to work properly, the original library must have `pg_pin` syntax with related power nodes in the pin group.

-io
Create templates with `define_arc` commands for I/O's.
The -io option additionally generates `define_arc` (including vectors), `define_leakage`, and `define_index` commands for use with IO cell characterization. For pad pins, a default `define_pin_load` template will also be generated that includes a pull-up voltage equal to twice (Tcl variable `template_pullup_voltage_scale`) the voltage on that pin and a pullup and pulldown resistance of 4000 ohms (Tcl variable `template_resistance`). The `-verbose` option is equivalent to the `-io` option with the exception that the `pin_load` will not be output.

Virtuoso Liberate Reference Manual

Liberate Commands

`-map {list}`

List of name-map pairs, used to map internal names to an external name. Default: `none` (No name mapping.)

The `-map` option modifies the name used for the internal node in the library when writing the template file. It is usually the case that the internal pin node name contains character that can not be used in a `.lib` format. Therefore, it is required to use a simpler name for such internal pins. This option allows the user to specify the actual name for the internal pin that needs to be used in the various definitions contained in the template file (`define_cell`, `define_arc`, and so on).

`-merge <"skip_delay">`

Generates a "merged verbose" template with merged *when* conditions. The option accepts the value `"skip_delay"`, which merges only `constraint`, `mpw`, and `hidden power` arcs. This option must be used together with the `-verbose` option. For example:

```
write_template -merge skip_delay -verbose
```

Note: When using a "merged verbose" template, you must also set the `define_arc_preserve_when_string` variable to 1.

`-mpw`

Generates MPW templates for each cell if two-dimensional MPW tables exist in the loaded library. If two-dimensional MPW tables do not exist, no MPW templates will be output.

`-no_internal_supply`

Disables the default behavior of recognizing `internal_ground` and `internal_power` attribute in the `pg_pin` library group and writing out `define_cell` with the `internal_supply` constructs. Default: `off`

`-sdf_cond`

Writes `define arcs` with `sdf_cond` conditions for timing arcs.

Note: The `-sdf_cond` and `-when_as_vector` options of the `write_template` command cannot be used together. Using them together generates an error message.

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|-----------------|--|
| -si | <p>Generates <code>si_iv_curve</code> and <code>si_immunity</code> templates for each cell. If the current library does include any signal integrity data, a default <code>si_iv_curve</code> template with 35 points (Tcl variable <code>template_siv_points</code>) is generated along with <code>si_immunity</code> templates for each unique delay template. The noise width (<code>index_1</code>) for the <code>si_immunity</code> template is created by multiplying the slew index (<code>index_1</code>) of the delay template by a factor of 10 (Tcl variable <code>template_slew_to_width</code>).</p> <p>If the switching power indices are swapped and <code>index_1</code> actually points to load, the tool recognizes the actual slew index used by the switching power template and pads the hidden power template based on that.</p> <p>If the input library does not have a two-dimensional noise immunity template, a two-dimensional template will be created from the delay indices.</p> |
| -skip {leakage} | <p>Disables the output of <code>define_leakage</code> commands into the template file. This option should only be used when <i>Inside View</i> is enabled. It cannot be used with the <code>char_library -io</code> option that disables the <i>Inside View</i>. If there are no <code>define_leakage</code> commands loaded and the <i>Inside View</i> is not enabled, the resulting library will <i>not</i> have any leakage states characterized. Currently, out of all supported values, you can skip only "leakage".</p> |
| -sort_pinlist | <p>Accepts a value of <code>in_bi_ou</code>. When this option is enabled, the <code>define_cell -pinlist</code> option in the template will have the pins sorted as follows: "input bidi output".</p> |
| -truth_table | <p>Specifies that a template file will be written out from the truth table data that has been loaded using the <code>read_truth_table</code> command. When this option is used, the <code>-auto_index</code> option setting will be used. The output template might then be edited to define index values. The following options will be ignored when using this option: <code>-cell</code>, <code>-exclude</code>, <code>-io</code>, <code>-unique</code>, <code>-unique_power</code>, <code>-group</code>, and <code>-define_index</code>. For more information about the truth table syntax, see Truth Table Format.</p> |

-unique Specifies that each cell will have its own unique set of template definitions, otherwise cells will share templates where the templates are identical. Two templates are deemed identical if they have the same type, the same number of indices and each index is within `abstol` (default 0.0) to each other.

The `-unique` option cannot be used with the `-auto_index` option.

Note: The `-unique` option always writes unqualified templates for each cell by adding a sequential numerical suffix. This option should not be combined with `-use_lu_table_name` option because the resulting library will not match the original library.

-unique_power Generate unique power templates. (Default: *use same template indices for delay and power*)

If `-unique_power` is used, the power templates can have indices that are different from the timing templates. If the arc conditions for delay and power are identical, then only `define_arc` commands are written for "delay" as these will force both power and delay characterization. If the conditions are different then a full set of delay arcs and a full set of power arcs are written into the template. By default, the power templates will use the same indices as the delay templates.



Using this option can have a significant impact on run time because additional simulations to characterize the power will be required.

-use_lu_table_name Reuse the lu_table names from the original library in the Liberate `define_template` commands.

-verbose Create verbose templates with `define_arc` commands.

-when_as_vector Converts the `when` conditions into a vector sequence rather than having a `-when` for each `define_arc`.

<filename> File name for the Tcl template

This command should be used after `char_library`, `read_ldb`, `read_library`, or `read_truth_table`.

When using the `-verbose` option, it is common to see invalid `define_arc` commands in the output template. This can occur if the `timing_sense` and/or unateness of an arc is not specified in the input library. Liberate cannot determine the real arcs; therefore, it outputs all combinations of `define_arc` commands for all possible cases from the `related_pin` to the output pin. For example, `define_arc` commands will cover all four cases of RR, RF, FF, FR for an arc where the `timing_sense` and unateness are absent. Two of the `define_arc` commands will be invalid and may cause an error condition during `char_library` run depending on the value of `def_arc_msg_level`. To avoid having invalid `define_arc` commands in the template, the input library must contain complete `timing_sense` and unateness attributes. The template can be modified manually to remove the invalid arcs.

When generating `define_index` commands (see the `verbose`, `io` and `define_index` options) `write_template` will increase the number of indices to the same size as the default template if it is smaller. For example, a pad pin might have a default 7x7 template, but for some arcs, it might use 4x4. As `define_index` requires 7x7, `write_template` will pad the 4x4 indices by inserting mid-points starting between the 1st and 2nd index, 2nd and 3rd, and so on until there are enough indices.

If buses exist in the input library, the pins will be output in bus syntax in the `define_cell` `-input`, `-output`, `-bidi`, `-clock`, `-async`, and `-pinlist` options. The `-expand_buses` option is used to make `write_template` generate the `define_cell` commands without using `bus_syntax`. For example, a bus DOUT with 3-bits will be represented as:

```
define_cell {-output { DOUT [2-0] }}
```

instead of

```
define_cell -output { DOUT [2] DOUT [1] DOUT [0] }
```

The bus syntax used for the template can be changed by setting the `bus_syntax` variable. For example, if the library uses "[]" and the SPICE netlist uses "<>", then using `set_var bus_syntax "<>"` before `write_template` will cause the bus definition to appear in the template using the `<>` syntax. For example, `-output { DOUT[2-0] }` will be output as `-output { DOUT<2-0> }`

Using the `index_*` options

The `-index_constraint`, `-index_delay`, `-index_mpw`, and `-index_si` can be used with the `auto_index` option to specify the required data size. These can also be used to change the data table size from the original `.lib` to a modified table. Liberate does not support 3D data tables. When the golden `.lib` file has 3D data tables (possibly due to a dependent load) for constraints (3D setup/hold table) and load dependent mpw (2D mpw table), the `write_template` command ignores complete the 3d tables. The `-index_*` options can be used to adjust the data template size. If the user-specified dimension is

different from the library then appropriate warnings will be printed and the data size will be adjusted. For example:

```
#User wants a one dimensional mpw table: 4x1
write_template -index_mpw 4x1 -mpw -define_index new_data/templates.tcl
```

Using the -driver_waveforms option

The `-driver_waveforms` option writes out a template including `define_input_waveform` commands that can be used to re-characterize cells using the same input waveforms as the existing library. The normalized driver waveform data *must* be present in the source library.

This template can only be used with version **LIBERATE 12.1 ISR1** or later.

The `driver_waveforms` written by `write_template` can be re-used for a different voltage level by changing the `-vdd_val` parameter to `define_input_waveform`.

Following is an example showing the use of `-driver_waveforms` in a `read_library`, `write_template` flow:

```
read_library my.lib
write_template -driver_waveforms template.tcl
```

Its output will appear as follows:

```
set PreDriver10_dot_5_colon_rise_0_dot_004_pwl {0.0 0 6.5e-13 0.065 1.175e-12
0.215835 1.675e-12 0.34251 2.25e-12 0.470272 2.85e-12 0.587447 3.85e-12
0.755671 5e-12 0.920547 5.775e-12 0.942784 6.65e-12 0.960506 7.65e-12 0.974145
8.825e-12 0.984284 1.175e-11 0.995448 1.6975e-11 1}

set PreDriver10_dot_5_colon_rise_0_dot_072_pwl {0.0 0 1.17e-11 0.065 2.115e-11
0.215835 3.015e-11 0.34251 4.05e-11 0.470272 5.13e-11 0.587447 6.93e-11
0.755671 9e-11 0.920547 1.0395e-10 0.942784 1.197e-10 0.960506 1.377e-10
0.974145 1.5885e-10 0.984284 2.115e-10 0.995448 3.0555e-10 1}

set PreDriver10_dot_5_colon_fall_0_dot_004_pwl {0.0 0 6.5e-13 0.065 1.175e-12
0.215835 1.675e-12 0.34251 2.25e-12 0.470272 2.85e-12 0.587447 3.85e-12
0.755671 5e-12 0.920547 5.775e-12 0.942784 6.65e-12 0.960506 7.65e-12 0.974145
8.825e-12 0.984284 1.175e-11 0.995448 1.6975e-11 1}

set PreDriver10_dot_5_colon_fall_0_dot_072_pwl {0.0 0 1.17e-11 0.065 2.115e-11
0.215835 3.015e-11 0.34251 4.05e-11 0.470272 5.13e-11 0.587447 6.93e-11
0.755671 9e-11 0.920547 1.0395e-10 0.942784 1.197e-10 0.960506 1.377e-10
0.974145 1.5885e-10 0.984284 2.115e-10 0.995448 3.0555e-10 1}

define_template -type delay \
    -index_1 {0.004 0.072} \
    -index_2 {0.0013 0.0142} \
```

Virtuoso Liberate Reference Manual

Liberate Commands

```
delay_template_2x2

define_template -type power \
    -index_1 {0.004 0.072} \
    -index_2 {0.0013 0.0142} \
    power_template_2x2

define_cell \
    -input { A1 A2 A3 B1 B2 B3 } \
    -output { ZN } \
    -pinlist { A1 A2 A3 B1 B2 B3 ZN } \
    -delay delay_template_2x2 \
    -power power_template_2x2 \
AOI33D4

define_input_waveform -slew_index 0.004 -dir fall -pwl
$PreDriver10_dot_5_colon_fall_0_dot_004_pwl -vdd_val 1.05 -gnd_val 0 -scale
-pinlist { AOI33D4 A1 AOI33D4 A2 AOI33D4 A3 AOI33D4 B1 AOI33D4 B2 AOI33D4 B3 }

define_input_waveform -slew_index 0.072 -dir fall -pwl
$PreDriver10_dot_5_colon_fall_0_dot_072_pwl -vdd_val 1.05 -gnd_val 0 -scale
-pinlist { AOI33D4 A1 AOI33D4 A2 AOI33D4 A3 AOI33D4 B1 AOI33D4 B2 AOI33D4 B3 }

define_input_waveform -slew_index 0.004 -dir rise -pwl
$PreDriver10_dot_5_colon_rise_0_dot_004_pwl -vdd_val 1.05 -gnd_val 0 -scale
-pinlist { AOI33D4 A1 AOI33D4 A2 AOI33D4 A3 AOI33D4 B1 AOI33D4 B2 AOI33D4 B3 }

define_input_waveform -slew_index 0.072 -dir rise -pwl
$PreDriver10_dot_5_colon_rise_0_dot_072_pwl -vdd_val 1.05 -gnd_val 0 -scale
-pinlist { AOI33D4 A1 AOI33D4 A2 AOI33D4 A3 AOI33D4 B1 AOI33D4 B2 AOI33D4 B3 }
```

Examples

```
read_library my.lib
# Output a Liberate Tcl command file with templates
set template_slew_to_width 15
write_template -si my_template

# Output a Liberate Tcl command file for auto_index
write_template -auto_index -index_delay 8x8 ai_template

# Output a Liberate Tcl command file IO cell char
write_template -io io_template
```

Virtuoso Liberate Reference Manual

Liberate Commands

Note: For `ibis_slew`, the user should consult with their design manual to determine an appropriate input slew number to drive the IBIS characterization. A number too large might result in waveform lag when annotating the generated IBIS file into simulation. For example, by comparing IBIS (the B component) annotated waveform with original (direct simulation) waveform, the following waveform lag might occur. If you see behavior like this, please adjust the input slew number to fix the problem.

write_top_netlist

Creates a Verilog file that contains an instantiation of each cell within the library. The output is written to the specified output Verilog *<filename>*. If the given *<filename>* does not end with a *.v*, the suffix *.v* is added automatically. Each instance and pin is named as follows: *inst_<cell>, pin_<cell>_<pin>*

Options

-cells {list_of_cells}

List of cells to include in the output. Default: all cells

Note: If the **-exclude** option is specified, the list of cells specified by the **-cells** option will be excluded from the output file.

-exclude Exclude the list of cells. Default: treat cells list as include list

-pin_prefix <string>

Prefix for each pin. Default: *pin_*

-instance_prefix <string>

Prefix for each instance. Default: *inst_*

-module <name> Top module name. Default: *top*

<filename> Name of the output Verilog file.

The top-level Verilog can be used to verify SDF timing back-annotation as follows. Read the top-level Verilog generated by **write_top_netlist** into a timing tool along with the *.lib* from **write_library**. Generate an SDF file from the timer. Read the SDF file, the top-level Verilog plus the Verilog (from **write_verilog**) or Vital (from **write_vital**) into a gate-level simulator. The gate-level simulator will report any SDF annotation errors.

This command must be used after a database has been loaded.

write_userdata_library

Takes a Liberty file as input, and writes out a Liberty-formatted file containing only the cells, attributes, and groups specified by the user. The purpose of this command is to create a userdata file for use with the [write_library](#) command.

Options

| | |
|---|---|
| <code>-cells {list}</code> | List of cells to write |
| <code>-exclude {list}</code> | List of cells to exclude from -cells list |
| <code>-include_attributes {list}</code> | List of attributes to include. |
| <code>-exclude_attributes {list}</code> | List of attributes to exclude. |
| <code>-include_groups {list}</code> | List of groups to include. |
| <code>-exclude_groups {list}</code> | List of groups to exclude. |
| <code><filename></code> | User data filename for output |

This command maintains a list of default groups and attributes to include, as well as a default list of groups and attributes to exclude. (*NOTE: the default lists to "include" and "exclude" do not contain the same items. See list below.*)

If you do not explicitly specify items to include or exclude, the output file contains only the defaults. To add or subtract from this list, you must specify the groups/attributes you wish to include (or exclude). The command also supports the keyword "default" as a convenient way of referring to the default lists. (*See examples.*)

The list of attributes to exclude does not apply to the attributes under groups that are included. If a group is included then all the attributes and sub-groups of that group are also included.

■ **Default include attributes:**

| |
|--------------------------------|
| <code>area</code> |
| <code>cell_footprint</code> |
| <code>clear</code> |
| <code>clear_preset_var1</code> |

Virtuoso Liberate Reference Manual

Liberate Commands

```
clear_preset_var2
clock
clocked_on
clock_gate_clock_pin
clock_gate_enable_pin
clock_gate_out_pin
clock_gate_test_pin
clock_gating_integrated_cell
data_in
direction
dont_touch
dont_use
enable
function
input_map
input_voltage_range
internal_node
is_level_shifter
level_shifter_data_pin
level_shifter_enable_pin
level_shifter_type
next_state
nextstate_type
output_voltage_range
power_down_function
preset
signal_type
state_function
table
three_state
```

■ Default exclude attributes:

```
capacitance
cell_leakage_power
input_voltage
max_capacitance
max_transition
min_pulse_width_low
min_pulse_width_high
output_voltage
```

■ Default include groups:

```
ff
latch
statetable
test_cell
```

■ **Default exclude groups:**

```
hyperbolic_noise_above_high
hyperbolic_noise_below_low
hyperbolic_noise_high
hyperbolic_noise_low
input_voltage
output_voltage
pin
propagation_lut_template
```

Examples

```
# Include all the default groups plus the "input_voltage" group
read_library myLibrary.lib
write_userdata_library -include_groups {default input_voltage} userData.lib

# Exclude all the default groups plus the "statetable" group
read_library myLibrary.lib
write_userdata_library -exclude_groups {default statetable} userData.lib
```

write_vdb

Creates a vector library database (VDB) file for the current library.



The `write_vdb` command is currently incompatible with a non-zero value for `packet_clients`. If your flow uses both of these, make sure that `packet_clients` is set to 0 immediately prior to the `write_vdb` command.

Options

| | |
|-------------------------------------|---|
| <code>-auto_index</code> | Instructs Liberate to automatically generate table indices for all constructs (except <code>si_immunity</code>) overriding the values specified in the given templates. The number of entries for each index is taken for the appropriate pre-defined template. This feature uses the <code>-max_transition</code> parameter to determine the range of output loads for each cell. To automatically generate <code>si_immunity</code> indices set the <u>max_noise_width</u> variable. |
| <code>-ccsn</code> | Include CCSN (noise) data. |
| <code>-cells {list_of_cells}</code> | List of cells to include in the output. Default: all cells |
| <code>-extsim <name></code> | Name of external simulator to use. For related details, see <u>Using the -extsim option</u> . |
| <code>-io</code> | Enable IO mode. |
| <code><vdb_filename></code> | Name of the output VDB file. |

The VDB file created by the `write_vdb` command includes vector data that is created during the preprocessing stage in Liberate. This file can be used to speed up preprocessing by storing the processed vector data and library structure in the VDB file.

Typically, the VDB that is created is used to drive separate characterization runs, each designed to process the same library with different corners (process, voltage, temperature.). The characterization script for a given run would first load the database with a read_vdb command. Once loaded, the `char_library` command will use the vector and structure information stored in the VDB file, and will not rerun the vector processing. This is true for both the server and the client processes.

For generating a VDB, the `write_vdb` command takes the place of `char_library` in a Tcl command file that has been fully set up to characterize a library. Note that `char_library` should not be executed in the same run as `write_vdb`.

The `-ccsn` and `-io` options are used to generate and store the CCSN templates, structures, and vectors into the VDB in the normal and in the IO mode. This VDB can be read back in using `read_vdb` and used in the subsequent `char_library` flow for CCSN generation.

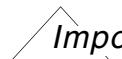
Note: CCSN in libraries across corners is based on internal node consistency. Using extractions and netlists with varying wire names in the `write_vdb` and `char_library` stages of the flow will produce unreliable results and is not supported.

This command must be used after a database has been loaded.

Using the -extsim option

The `-extsim` option instructs Liberate to use an external SPICE simulator instead of Alspice. Alspice is the default built-in SPICE simulator. The license for the external simulator must be available. Currently, the following external simulators are supported:

- HSPICE
- Spectre



The setting for `-extsim` must correspond to the setting for the `extsim_cmd` variable.

When the `-extsim` option is specified, temporary run directories named `altos.<unique_id>.0`, `altos.<unique_id>.1`, and so on will be created to store the external simulation run-time files based on the thread number (0, 1, and so on). The `<unique_id>` is a unique ID based on the date, time, and the Liberate process ID.

If distributed processing is requested using the `set_client` command, these temporary run directories will be created in the directory specified by the `-dir` option of the `set_client` command.

If distributed processing is requested using the `packet_log_filename` parameter, the `TMPDIR` environment variable and the `tmpdir` parameter determine where the temporary simulation files will be stored. If the specified directory does not exist, Liberate tries to create it using the `mkdir -p` system command.

write_verilog

Creates a Verilog file for the current library. The file is saved with the name given with the `<verilog_filename>` option. If this name does not have a `.v` file extension at the end, a `.v` suffix will be added automatically to the filenames.

Options

| | |
|---|--|
| <code>-add_default_udp</code> | Allows you to provide user data where all the primitives are not included; default primitives will be used for all other cells. This option works only with <code>-user_data</code> . |
| <code>-cells {cell_names}</code> | Controls which cells should get written to the output. Default: <i>all cells</i> This option supports the use of a wildcard. Note: If <code>-exclude</code> is also set, only the cells not listed in the <code>-cells</code> list will be output. |
| <code>-compare</code> | Enables function comparison using the <code>-conformal</code> option of the <code>compare_function</code> command of Liberate LV. To use this flag, you must have license to run Liberate_LV and Conformal (Liberate_LV_Server, Liberate_LV_Client, and LEC licenses). In addition, the environment must be setup to enable these tools. |
| <code>-delayed</code> | Controls naming convention for creating <i>delayed</i> signals. Default: "delayed_%P" (where %P is the pin name.) |
| <code>-exclude</code> | Excludes the cells specified by using the <code>-cells</code> option. |
| <code>-fwire_prefix <prefix></code> | Specifies the prefix used for internal wires created when generating logic functions. Default: <code>int_fwire_</code> |
| <code>-indent <number></code> | Specifies the number of spaces to use for indentation. Default: <code>tab</code> |
| <code>-merge</code> | Merges cell modules from <code>user_data</code> . The merge operation includes the cells that are not specified in the library, but are present in the <code>user_data</code> file. |

`-mpw_include_output_state`

Includes the logic state of the output in the condition for MPW checks on "clear" and "preset" signals in the generated Verilog. Default: *do not use*

This option requires the `sdf_cond_style` variable is set to 1. If this variable is not set, the `-mpw_include_output_state` option will force this setting. Note that this variable should be set prior to creating an equivalent library (.lib) with `write_library`, to ensure consistency between the library and the Verilog. Otherwise, there might be warnings during SDF back-annotation.

The `-mpw_include_output_state` option should be used *before* `read_library`, `char_library`, or `read_Idb`. When this variable is used, the MPW check (`$width`) will be checked by Verilog only when the output pin of the cell is high for clear inputs and low for preset inputs. The Verilog will contain extra "timing" gates to add the logic necessary to "and" the logic state of the output pin to logic representing the "when" condition given in the library for each `min_pulse_width` timing arc.

`-mux`

Instructs to use MUX primitives instead of basic logic primitives to represent MUX functions. Default: *basic logic primitives*

This option converts pins whose functions are a 2x1 or 4x1 MUX into a pre-defined, user-defined primitive (UDP) named `altos_mux2` and `altos_mux4` respectively.

`-notifier`

Sets the name of the Verilog register that is used to flag timing violations. Default: "notifier"

Note: This option is needed only if `-user_data` is used.

Virtuoso Liberate Reference Manual

Liberate Commands

-no_edge Excludes 'posedge' or 'negedge' on edge-triggered arcs.
Default: *include edges*

This option will change the following:

```
specify
  if ((CP & SN))
    (negedge RN => (QN-:1'b0)) = 0;
```

...to this:

```
specify
  if ((CP & SN))
    (RN => QN) = 0;
```

-no_err_primitives Causes *_err primitive cells to be replaced with buf cells.

This option is available because some tools such as ATPG do not accept *_err primitives.

-path <path> Controls the delimiter used for delay assignments. It must be either "=>" or "*>". Default: "=>"

-power_pin Adds the power-down function to the Verilog model. This option is used when the power-down function is needed in `liberty.lib` and the power pins should be included in the cell-port list.

-sdf_version <version>

Controls the format of the output Verilog for use with SDF annotation. SDF version must be 2.1 or 3.0. Default: 2.1

Setting this option to 3.0 generates a Verilog file that is compatible with SDF version 3.0. SDF version 3.0 permits *recrem* constructs in the Verilog to represent recovery and removal of timing constraints.

-specparams Causes delay assignments in the Verilog to be assigned to *specparam* variables rather than directly to values.

| | |
|--|--|
| <code>-split_nonunate</code> | Instructs Liberate to convert a non_unate timing group in the .lib into two arcs in the Verilog output file. |
| | This option is useful when using the Cadence ETS static timer to generate the SDF for annotation into the Verilog simulation. ETS will convert the single non_unate timing group into 2 SDF entries. This can cause an error condition in Verilog if the Verilog has only one non_unate arc and the SDF has two-delay entries. |
| <code>-split_notifier</code> | When writing Verilog modules for multi-bit cells, it is required to output separate notifier commands for each DFF. The <code>-split_notifier</code> option is used to output separate notifier commands for each DFF. |
| <code>-table_style <min-avg-max first-mid-last></code> | Instructs Liberate to add real timing data into the Verilog file using the specified specparams syntax. Default: " " (do not use real timing). |
| | This option specifies the syntax to be used in the specparams. When this is not used, Liberate defaults to writing the Verilog with zero (0) for all timing and will expect that an Standard Delay Format (SDF) file is supplied from the STA tool. |
| <code>-timescale "timescale_value"</code> | Change the timescale when writing out Verilog. Default: "1ns / 10ps" |
| <code>-twire_prefix <prefix></code> | Specifies the prefix used for internal wires created when generating additional functions for state dependent timing constraints. Default: <code>int_twire_</code> |
| <code>-udp_prefix <prefix></code> | Specifies the prefix used for built-in user-defined primitives (UDPs) that are created for latches and/or flip-flops. Set <code>-udp_prefix</code> to a null string (" ") to exclude generating UDPs. Default: <code>altos_</code> |

`-use_liberate_function`

Generates Liberate functions when user data does not contain any function. If user data has functions and this option is specified, Liberate generates the functions again.

Note: This option works with the `-user_data` option.

`-user_data <filename>`

Specifies a user-provided Verilog file to merge with the generated Verilog data. If a `user_data` file is provided, timing information (paths and any additional wires required to specify the conditions for those paths) are merged with the user file and written to the output file, replacing any existing user-provided timing information. Without a `user_data` file, a complete Verilog file is written including function descriptions.

`<verilog_filename>`

Name of the output Verilog file.

The `write_verilog` command should *not* be used in the same run as the `char_library`, `read_libraries`, or `write_library` command. Instead, it should be used in a separate Liberate run after a `read_library` command. This is because the Liberty file might have formatting that is required for the Verilog output to be properly formatted. For example:

```
read_library my.lib
write_verilog my.v
```



This command must be used after a database has been loaded.

Using the `-delayed` option

The `-delayed` option controls the naming convention for creating "delayed" signals. When using user-data with `write_verilog`, it is necessary to match these delayed signals with the equivalent signals used in the user-provided functional description. By default, delayed output signals are created for the signals passed to timing checks such as setup-hold and/or recrem in Verilog.

The delayed option uses a special variable "%P" to return the pin name and combine it with a user-defined string. For example:

```
write_verilog -delayed "delayed_%P"
```

If there is no "%P" and the -delayed string begins with "_", it is used as a suffix. In the following example, the wire for the pin "CK" is "CK_mySuffix".

```
write_verilog -delayed "_mySuffix" ...
```

if there is no "%P" and the -delayed string does not begin with "_", the string is treated as a prefix. In the following example, the wire for the pin "CK" is "myPreFixCK".

```
write_verilog -delayed "myPreFix" ...
```

For a pin named "myPin", this will produce a delayed signal name "delayed_myPin". Some examples are below:

Example 1:

```
module DFFSRN (QN, D, CP, RN, SN);
    output QN;
    input D, CP, RN, SN;
    reg notifier;
    wire delayed_D, delayed_CP, delayed_RN, delayed_SN;

    // Function
    ...
    // Timing
    specify
        ...
        $setuphold (posedge CP, posedge D, 0, 0, notifier,,, delayed_CP, delayed_D);
        ...
        $recrem (posedge RN, posedge CP, 0, 0, notifier,,, delayed_RN, delayed_CP);
        ...
    endspecify
endmodule
```

Example 2:

```
write_verilog -delayed "dly_%P"
```

... will produce:

```
wire dly_D, dly_CP, dly_RN, dly_SN;
...
$setuphold (posedge CP, posedge D, 0, 0, notifier,,, dly_CP, dly_D);
```

Example 3:

```
write_verilog -delayed "%P_d"
```

... will produce:

```
  wire D_d, CP_d, RN_d, SN_d;  
  ...  
  $setuphold (posedge CP, posedge D, 0, 0, notifier,,, CP_d, D_d);
```

The default name for these delayed signals is "delayed_<pin_name>" (for example, delayed_%P) where <pin_name> is a pin that is involved in a timing check.

To disable generation of delayed signals, use " " (empty double quotes), as illustrated below:

```
module DFFSRN (QN, D, CP, RN, SN);  
  output QN;  
  input D, CP, RN, SN;  
  reg notifier;  
  
  // Function  
  ....  
  // Timing  
  specify  
  ...  
  $setuphold (posedge CP, posedge D, 0, 0, notifier);  
  ...  
  $recrem (posedge RN, posedge CP, 0, 0, notifier);  
  ...  
  endspecify  
endmodule
```

Tcl Variables to Control the Verilog Output Format

The following Tcl variables can also be used to control the format of the Verilog output:

verilog_delay_value The delay value. Default: 0

verilog_delay_Zvalue

The delay value for tristates. Default: 0

verilog_delay_clk2q_value

The delay value for clock to Q arcs on sequential cells.
Default: 0

verilog_IQ

The name map for the internal state of flip-flops (for example, IQ) to the Verilog state function.

verilog_IQN

The name map for the internal state of flip-flops (for example, IQN) to the Verilog state function.

Virtuoso Liberate Reference Manual

Liberate Commands

| | |
|--------------------|--|
| verilog_start_skip | Line to mark the start of the timing section in the <code>user_data</code> file which is to be replaced. Must match exactly apart for leading or trailing white space. Default: "specify" |
| verilog_stop_skip | Line to mark the end of the timing section in the <code>user_data</code> file which is to be replaced. Must match exactly apart for leading or trailing white space. Default: "endspecify" |

Example

```
read_library my.lib
# Output a Verilog file
write_verilog -user_data my_verilog my.v
```

write_vital

Creates a Vital VHDL file for the current library. The file is saved with the name given with the `<vital_filename>` option. If this name does not have a `.vhd` file extension at the end, a `.vhd` suffix will be added automatically to the filenames. The output will include sequential gates and tristates.

Options

`-cells {cell_names}`

Controls which cells should get written to the output. Default: *all cells*

This option supports the use of a wildcard.

Note: If `-exclude` is also set, only the cells not listed in the `-cells` list will be output.

`-compare`

Enables function comparison using the `-conformal` option of the `compare_function` command of Liberate LV. To use this flag, you must have license to run Liberate_LV and Conformal (Liberate_LV_Server, Liberate_LV_Client, and LEC licenses). In addition, the environment must be setup to enable these tools.

`-component <filename>`

Instructs Liberate to print a list of components into the specified output file. By default, no component file is output.

`-exclude`

Excludes the cells given by using the `-cells` option.

`-indent <number>`

Specifies the number of spaces to use for indentation. Default: tab

`-merge`

Merges cell modules from `user_data`.

The merge operation includes the cells that are not specified in the library, but are present in the `user_data` file.

Virtuoso Liberate Reference Manual

Liberate Commands

-no_edge Excludes 'posedge' or 'negedge' on edge-triggered arcs.
Default: *include edges*

This option will change the following:

```
specify
  if ((CP & SN))
    (negedge RN => (QN-:1'b0)) = 0;
```

...to this:

```
specify
  if ((CP & SN))
    (RN => QN) = 0;
```

-sdf_version <2.1 | 3.0>

Specifies the desired SDF version in the output file. SDF version must be 2.1 or 3.0. Default: 2.1

Setting this option to 3.0 generates a Vital file that is compatible with SDF version 3.0.

-user_data <filename>

Specifies a user-provided Vital file to merge with the generated Vital data. If a `user_data` file is provided the port names and functional behavior information is extracted from the user-specified file. Without a `user_data` file, a complete Vital file is written. The sections in the Vital file that are extracted from the user data are marked with comments indicating that they were user-provided and not automatically generated, for example:

```
-- FUNCTIONALITY SECTION (USER PROVIDED) --
-- END FUNCTIONALITY SECTION (USER PROVIDED) --
```

<vital_filename> Name of the output Vital (vhd) file.

The `write_vital` command should *not* be used in the same run as the `char_library`, `read_libraries`, or `write_library` command. Instead, it should be used in a separate Liberate run after a `read_library` command. This is because the Liberty file might have information that is required for the Vital output to be properly formatted. For example:

```
read_library my.lib
write_vital my.vhd
```



This command must be used after a database has been loaded.

Tcl Variables to Control the Vital Output Format

The following Tcl variables can be used to control the format of the Vital output:

| | |
|-----------------------------|---|
| vital_config | Include config section. Default: 0 |
| vital_port_type | Port type. Default: STD_LOGIC |
| vital_delay_value | The delay value. Default: 0ns |
| vital_delay_Zvalue | The delay value for tristates. Default: 0ns |
| vital_delay_variables | Flag to enable assignment to Delay variables instead of values. Default: 1 |
| vital_timingViolationFormat | The prefix for Violation variables used by setup/hold timing checks. Default: "Tviol_\\\$count", where \$count is incremented for each VitalSetupHoldCheck entry within a cell |
| vital_recremViolationFormat | The prefix for Violation variables used by recovery/removal timing checks. Default: "Rviol_\\\$count" where \$count is incremented for each VitalRecoveryRemovalCheck entry within a cell |
| vital_pulseViolationFormat | The prefix for Violation variables used by pulse/period timing checks. Default: "Pviol_\\\$count" where \$count is incremented for each VitalPeriodPulseCheck entry within a cell |
| vital_timingInfoFormat | The prefix for TimingData variables used by setup/hold timing checks. Default: "SetupHoldInfo_\\\$count" where \$count is incremented for each VitalSetupHoldCheck entry within a cell |

Virtuoso Liberate Reference Manual

Liberate Commands

`vital_recrem_info_format`

The prefix for `TimingData` variables used by recovery/removal timing checks. Default: "RecoRemoInfo_\\$count" where \$count is incremented for each `VitalRecoveryRemovalCheck` entry within a cell

`vital_pulse_info_format`

The prefix for `PeriodData` variables used by period/pulse timing checks. Default: "PeriodDataInfo_\\$count" where \$count is incremented for each `VitalPeriodPulseCheck` entry within a cell

`vital_removal_check` The prefix to use for `Entity` variables used in removal checks. For SDF 3.0 annotation change this variable to "removal". Default: "hold"

`vital_start_architecture`

The keyword that indicates the start of the architecture section in the `user_data` file. Default: "architecture"

`vital_stop_architecture`

The line that indicates the end of the architecture section in the `user_data` file. Default: "end \\$cell_arch" where \\$cell is substituted with the current cell name

`vital_start_cell` The line to indicate the start of a vital cell description in the `user_data` file. Default: "-- %BEGIN \\$cell" where \$cell is substituted with the current cell name

`vital_stop_cell` The line to indicate the stop of a Vital cell description in the `user_data` file. Default: "-- %END \\$cell" where \$cell is substituted with the current cell name

`vital_start_function`

The line to indicate the start of a Vital function description in the `user_data` file. Default: "function"

Note: The spaces and dashes (-) and case will be ignored.

`vital_stop_function` The line to indicate the stop of a vital function description in the `user_data` file. Default: "\\$vital_path_delay" where \$vital_path_delay is substituted with the value of the `vital_path_delay` variable, (Default: "VitalPathDelay01Z")

Tcl Variables to Customize Formatting of Generated Timing Check Variables

The following Tcl variables can be used to customize the formatting of the generated timing check variables (note that these variables must be preceded with a "\\" and suffixed with "\\\" when used in the middle of a string to avoid early evaluation by the Tcl interpreter):

| | |
|------------|--|
| \$count | The current count of VitalSetupHoldCheck, VitalRecoveryRemovalCheck, or VitalPeriodPulseCheck entries. |
| \$ip | The input pin name. |
| \$rp | The reference pin name. |
| \$edge | The edge of the input pin transition. |
| \$ref_edge | The edge of the reference pin transition. |

Example

```
read_library my.lib
# set timing check variables
set vital_timingViolationFormat "Tviol_\\$ip\\_\\$rp\\_\\$refEdge"
set vital_recremViolationFormat "Rviol_\\$ip\\_\\$rp\\_\\$refEdge"
set vital_timingInfoFormat      "Tinfo_\\$ip\\_\\$rp\\_\\$refEdge"
set vital_recremInfoFormat     "Rinfo_\\$ip\\_\\$rp\\_\\$refEdge"
# Output a Vital file
write_vital -user_data my_vital my.vhd
```

Virtuoso Liberate Reference Manual
Liberate Commands

Liberate Parameters

This chapter describes the Liberate specific parameters that impact library creation.

Note: Liberate specific variables are set using the `set_var` command.

| a... | |
|--|---|
| <code>adjust_tristate_load</code> | <code>alspice_option</code> |
| <code>adjust_tristate_load_ccsp</code> | <code>auto_index_distinct_risefall</code> |
| <code>alspice_diode</code> | <code>auto_index_input_slew</code> |
| <code>alspice_leakage_option</code> | <code>auto_index_weak_driver_mode</code> |
| b... | |
| <code>binning_detail</code> | <code>bundle_arc_mode</code> |
| <code>bisection_info</code> | <code>bus_syntax</code> |
| <code>bundle_when</code> | |
| c... | |
| <code>capacitance_attr_mode</code> | <code>ccsp_prune_second_tol</code> |
| <code>capacitance_pin_rollup_k</code> | <code>ccsp_prune_start_tol</code> |
| <code>capacitance_pin_rollup_mode</code> | <code>ccsp_quantization_num_steps</code> |
| <code>capacitance_range_mode</code> | <code>ccsp_rel_tol</code> |
| <code>capacitance_save_mode</code> | <code>ccsp_related_pin_mode</code> |
| <code>ccs_abs_tol</code> | <code>ccsp_segmentation_effort</code> |
| <code>ccs_base_curve_points</code> | <code>ccsp_table_reduction</code> |
| <code>ccs_base_curve_share_mode</code> | <code>ccsp_tail_tol</code> |
| <code>ccs_cap_duplicate_risefall</code> | <code>cell_port_case</code> |
| <code>ccs_cap_hidden_pin</code> | <code>char_mos_term_cap</code> |

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|--|---|
| <u>ccs cap hidden pin mode</u> | <u>cleanup tmpdir</u> |
| <u>ccs cap is propagating</u> | <u>combinational out to out arc</u> |
| <u>ccs cap mode add missing</u> | <u>combinational risefall</u> |
| <u>ccs cap use input transition</u> | <u>conditional arc</u> |
| <u>ccs cap use input transition tristate</u> | <u>conditional cap hidden pin</u> |
| <u>ccs correct current by area</u> | <u>conditional cap hidden pin mode</u> |
| <u>ccs current model pin load</u> | <u>conditional cap hidden pin thresh</u> |
| <u>ccs enable sawtooth out</u> | <u>conditional constraint</u> |
| <u>ccsn floating init mode</u> | <u>conditional expression</u> |
| <u>ccs force grid delay</u> | <u>conditional hidden power</u> |
| <u>ccs infer output dir</u> | <u>conditional immunity</u> |
| <u>ccs init voltage comp thresh</u> | <u>conditional include constant</u> |
| <u>ccs max current thresh</u> | <u>conditional include output</u> |
| <u>ccs max pts</u> | <u>conditional leakage</u> |
| <u>ccs multiple switching output mode</u> | <u>conditional min period</u> |
| <u>ccs rel tol</u> | <u>conditional mpw</u> |
| <u>ccs segmentation effort</u> | <u>conditional rcvr cap select criteria</u> |
| <u>ccs smooth lower rise</u> | <u>constraint async probe internal</u> |
| <u>ccs smooth upper fall</u> | <u>constraint bisection mode</u> |
| <u>ccs voltage smooth thresh</u> | <u>constraint check final state</u> |
| <u>ccs voltage tail tol</u> | <u>constraint check final state threshold</u> |
| <u>ccs voltage tail tol mode</u> | <u>constraint check rebound</u> |
| <u>ccs voltage tail trim tol</u> | <u>constraint clock gater</u> |
| <u>ccs warn negative rcvr caps</u> | <u>constraint combinational</u> |
| <u>ccs waveform min time step</u> | <u>constraint combinational step limit</u> |
| <u>ccs waveform smooth mode</u> | <u>constraint combinational step size</u> |
| <u>ccsn active ccr recognition mode</u> | <u>constraint delay degrade</u> |
| <u>ccsn allow duplicate condition</u> | <u>constraint delay degrade abstol</u> |
| <u>ccsn allow multiple input switching</u> | <u>constraint delay degrade abstol max</u> |

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|---|---|
| <u>ccsn allow overlap when</u> | <u>constraint delay degrade minimize dtoq</u> |
| <u>ccsn allow partial voltage swing</u> | <u>constraint delay degrade minimize dtoq c lock only</u> |
| <u>ccsn arc channel check</u> | <u>constraint delay degrade minimize dtoq mode</u> |
| <u>ccsn arc consistent cut</u> | <u>constraint delay degrade minimize dtoq tol</u> |
| <u>ccsn arc high effort</u> | <u>constraint delay min check</u> |
| <u>ccsn bus holder mode</u> | <u>constraint dependent nominal</u> |
| <u>ccsn channel inputs high effort</u> | <u>constraint dependent recrem</u> |
| <u>ccsn compatibility mode</u> | <u>constraint dependent setuphold</u> |
| <u>ccsn consistent side inputs</u> | <u>constraint dependent setuphold input threshold</u> |
| <u>ccsn controlling path check</u> | <u>constraint dependent setuphold margin</u> |
| <u>ccsn dc static check</u> | <u>constraint dependent setuphold margin ratio</u> |
| <u>ccsn dc static check mode</u> | <u>constraint dependent setuphold pessimism</u> |
| <u>ccsn dc static check thresh</u> | <u>constraint failed value</u> |
| <u>ccsn dc template size</u> | <u>constraint glitch hold</u> |
| <u>ccsn default group add when</u> | <u>constraint glitch peak</u> |
| <u>ccsn default group criteria mode</u> | <u>constraint glitch peak internal</u> |
| <u>ccsn dual tie enable</u> | <u>constraint glitch peak max</u> |
| <u>ccsn extra default stages</u> | <u>constraint glitch peak mode</u> |
| <u>ccsn fanout select mode</u> | <u>constraint glitch peak report inherent</u> |
| <u>ccsn include passgate attr</u> | <u>constraint hold probe</u> |
| <u>ccsn input xfr probe mode</u> | <u>constraint info</u> |
| <u>ccsn io allow multiples</u> | <u>constraint info pass fail</u> |
| <u>ccsn io mode</u> | <u>constraint linear waveform</u> |
| <u>ccsn io mode enable</u> | <u>constraint margin</u> |
| <u>ccsn miller init mode</u> | <u>constraint merge state</u> |

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|---|--|
| <u>ccsn miller init vin thresh</u> | <u>constraint output load</u> |
| <u>ccsn miller vout delta variation</u> | <u>constraint output pin</u> |
| <u>ccsn model unbuffered output</u> | <u>constraint output pin mode</u> |
| <u>ccsn no input dc current mode</u> | <u>constraint probe internal</u> |
| <u>ccsn one sided tristate</u> | <u>constraint probe lower fall</u> |
| <u>ccsn pin criteria mode</u> | <u>constraint probe lower rise</u> |
| <u>ccsn pin high effort</u> | <u>constraint probe mode</u> |
| <u>ccsn pin stage lshift</u> | <u>constraint probe multiple</u> |
| <u>ccsn pin stage merge mode</u> | <u>constraint probe upper fall</u> |
| <u>ccsn pin unconditional</u> | <u>constraint probe upper rise</u> |
| <u>ccsn pin voltage level attrib</u> | <u>constraint search bound</u> |
| <u>ccsn prefer min vt probe</u> | <u>constraint search bound bisection mode</u> |
| <u>ccsn prefer two sided stages</u> | <u>constraint search bound estimation mode</u> |
| <u>ccsn print is needed if false attr value</u> | <u>constraint search bound probe mode</u> |
| <u>ccsn prop noise peak mode</u> | <u>constraint search iteration limit</u> |
| <u>ccsn prop retry duration incr</u> | <u>constraint search time abstol</u> |
| <u>ccsn prop retry peak incr</u> | <u>constraint slew degrade</u> |
| <u>ccsn probe mode</u> | <u>constraint snap to bound</u> |
| <u>ccsn prune last stage</u> | <u>constraint sweep pulse detection mode</u> |
| <u>ccsn simultaneous switch save vecdata</u> | <u>constraint sweep pulse width max</u> |
| <u>ccsn sort merge hidden mode</u> | <u>constraint tran end extend</u> |
| <u>ccsn switch cell partition mode</u> | <u>constraint tran end extend retry</u> |
| <u>ccsn tempus promote mode</u> | <u>constraint tran end mode</u> |
| <u>ccsn use io ccb format</u> | <u>constraint user defined probe mode</u> |
| <u>ccsn xfr ccc probe mode</u> | <u>constraint vector equivalence mode</u> |
| <u>ccsp base curve points</u> | <u>constraint vector mode</u> |
| <u>ccsp default group</u> | <u>constraint width degrade</u> |
| <u>ccsp leakage current abstol</u> | <u>constraint width degrade abstol</u> |
| <u>ccsp leakage current compensation mode</u> | <u>constraint width degrade abstol max</u> |

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|--|--|
| <u>ccsp min pts</u> | <u>constraint worst vector abstol</u> |
| <u>ccsp pin direction post default</u> | |
| <u>ccsp prune factor</u> | |
| d... | |
| <u>debug flow</u> | <u>discard timing sense after merge</u> |
| <u>def arc drive side bidi</u> | <u>disk wait time</u> |
| <u>def arc msg level</u> | <u>driver cell acc mode</u> |
| <u>def arc vector consistency check</u> | <u>driver cell all inputs</u> |
| <u>default power avg mode</u> | <u>driver cell info</u> |
| <u>define arc ignore mode</u> | <u>driver cell load all outputs</u> |
| <u>define arc preserve when string</u> | <u>driver cell load ldb cmd</u> |
| <u>define duplicate cap mode</u> | <u>driver waveform arcs only</u> |
| <u>delay constrained by setup recovery</u> | <u>driver waveform output precision</u> |
| <u>delay inp fall</u> | <u>driver waveform pulse mode</u> |
| <u>delay inp rise</u> | <u>driver waveform wildcard mode</u> |
| <u>delay out fall</u> | <u>duplicate pin attr mode</u> |
| <u>delay out rise</u> | <u>duplicate risefall power</u> |
| <u>disable method</u> | <u>duplicate risefall power ccsp</u> |
| e... | |
| <u>ecsm arctype enable</u> | <u>em window estimate mode</u> |
| <u>ecsm cap hidden pin</u> | <u>enable command history</u> |
| <u>ecsm cap input slew mode</u> | <u>extsim ccs option</u> |
| <u>ecsm cap load effect tol</u> | <u>extsim cells use nodeset for io pad</u> |
| <u>ecsm cap mode</u> | <u>extsim cmd</u> |
| <u>ecsm cap style</u> | <u>extsim cmd option</u> |
| <u>ecsm cap use input transition</u> | <u>extsim constraint option</u> |
| <u>ecsm capacitance factor</u> | <u>extsim deck dir</u> |
| <u>ecsm capacitance precision</u> | <u>extsim deck header</u> |
| <u>ecsm factor mode</u> | <u>extsim deck style</u> |

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|--|-------------------------------------|
| <u>ecsm invert gnd current</u> | <u>extsim exclusive</u> |
| <u>ecsm measure output range</u> | <u>extsim flatten netlist</u> |
| <u>ecsm version</u> | <u>extsim immunity option</u> |
| <u>ecsm waveform for bidi pin</u> | <u>extsim leakage option</u> |
| <u>ecsm waveform style</u> | <u>extsim lic keep</u> |
| <u>ecsm waveform time factor</u> | <u>extsim line length limit</u> |
| <u>ecsm waveform time precision</u> | <u>extsim model include</u> |
| <u>ecsmn loadcap mode</u> | <u>extsim model include leakage</u> |
| <u>ecsmn mode</u> | <u>extsim model include mode</u> |
| <u>ecsmn vtol mode</u> | <u>extsim monitor deck dir</u> |
| <u>em calculation include input</u> | <u>extsim monitor enable</u> |
| <u>em calculation monitor rails</u> | <u>extsim monitor timeout</u> |
| <u>em calculation monitor rails skip layer</u> | <u>extsim mpw option</u> |
| <u>em char arcs mode</u> | <u>extsim node name prefix</u> |
| <u>em clock freq</u> | <u>extsim option</u> |
| <u>em current type</u> | <u>extsim option presim</u> |
| <u>em data file</u> | <u>extsim reuse ic</u> |
| <u>em freq mode</u> | <u>extsim sanitize param name</u> |
| <u>em iacpeak mode</u> | <u>extsim save failed</u> |
| <u>em include string</u> | <u>extsim save passed</u> |
| <u>em maxcap</u> | <u>extsim save verify</u> |
| <u>em maxcap type</u> | <u>extsim tar cmd</u> |
| <u>em maxcap frequency</u> | <u>extsim timestep</u> |
| <u>em report data usage mode</u> | <u>extsim tran append</u> |
| <u>em tech file</u> | <u>extsim use node name</u> |
| <u>em user string</u> | |
| f... | |
| <u>floating channel bias</u> | <u>force default group</u> |
| <u>floating channel mode</u> | <u>force edge timing type</u> |

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|---------------------------------------|--|
| <u>floating node initialize mode</u> | <u>force leakage if no pg pin</u> |
| <u>force avg default select order</u> | <u>force related power pin</u> |
| <u>force condition</u> | <u>force unconnected pg pin</u> |
| g... | |
| <u>group attribute</u> | |
| h... | |
| <u>heartbeat initial timeout</u> | <u>hidden power</u> |
| <u>heartbeat timeout</u> | |
| i... | |
| <u>ibis compensate odt</u> | <u>init clock period mode</u> |
| <u>ibis has weak hold</u> | <u>init comb num cycles</u> |
| <u>ibis iv max step factor</u> | <u>init comb related pin period</u> |
| <u>ibis iv mode</u> | <u>init constraint period</u> |
| <u>ibis iv step</u> | <u>init constraint period binning mode</u> |
| <u>ibis odt min current</u> | <u>init constraint period check mode</u> |
| <u>ibis sim duration</u> | <u>init delay period</u> |
| <u>ibis t2b cmd</u> | <u>init pin hidden period</u> |
| <u>ibis tend factor</u> | <u>init pin hidden num cycles</u> |
| <u>ibis vt max num pts</u> | <u>init pin hidden period mode</u> |
| <u>ibis vt min num pts</u> | <u>input noise</u> |
| <u>immunity glitch peak</u> | <u>input output voltage</u> |
| <u>immunity noise skew ratio</u> | |
| k... | |
| <u>keep dcap leakage</u> | <u>keep empty cells</u> |
| <u>keep default leakage group</u> | <u>keep user defined arc failed data</u> |
| l... | |
| <u>ldb_checkpoint_dir</u> | <u>leakage precision</u> |
| <u>ldb_precision</u> | <u>leakage ramp vsrc</u> |
| <u>ldb_save_all_cells</u> | <u>leakage sim duration</u> |

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|---|-------------------------------------|
| <u>leakage add input pin</u> | <u>library copyright</u> |
| <u>leakage add missing group</u> | <u>library revision</u> |
| <u>leakage cell attribute</u> | <u>library revision mode</u> |
| <u>leakage expand state</u> | <u>lic max timeout</u> |
| <u>leakage float internal supply</u> | <u>lic queue timeout</u> |
| <u>leakage force tristate pin</u> | <u>logic and</u> |
| <u>leakage merge state</u> | <u>logic not</u> |
| <u>leakage mode</u> | <u>logic or</u> |
| <u>leakage model internal pin</u> | |
| m.... | |
| <u>mac address query timeout</u> | <u>measure target occurrence</u> |
| <u>mark failed data</u> | <u>mega bundle mode</u> |
| <u>mark failed data replacement</u> | <u>mega enable</u> |
| <u>max capacitance attr limit</u> | <u>mega mode constraint</u> |
| <u>max capacitance attr mode</u> | <u>mega mode delay</u> |
| <u>max capacitance derive limit maxload</u> | <u>mega mode hidden</u> |
| <u>max capacitance factor</u> | <u>merge related preset clear</u> |
| <u>max capacitance limit</u> | <u>min capacitance for outputs</u> |
| <u>max hidden vector</u> | <u>min output cap</u> |
| <u>max leakage vector</u> | <u>min period</u> |
| <u>max noise width</u> | <u>min period when</u> |
| <u>max transition</u> | <u>min transition</u> |
| <u>max transition attr limit</u> | <u>min transition attr limit</u> |
| <u>max transition factor</u> | <u>min transition factor</u> |
| <u>max transition for outputs</u> | <u>min transition for outputs</u> |
| <u>max transition include power</u> | <u>min transition include power</u> |
| <u>measure cap active driver mode</u> | <u>mpw criteria</u> |
| <u>measure cap lower fall</u> | <u>mpw glitch peak</u> |
| <u>measure cap lower rise</u> | <u>mpw input threshold</u> |

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|---|---|
| <u>measure cap upper fall</u> | <u>mpw linear waveform</u> |
| <u>measure cap upper rise</u> | <u>mpw search bound</u> |
| <u>measure ccs cap lower rise</u> | <u>mpw search mode</u> |
| <u>measure ccs cap upper fall</u> | <u>mpw skew factor</u> |
| <u>measure em target occurrence</u> | <u>mpw slew</u> |
| <u>measure output range</u> | <u>mpw slew clock factor</u> |
| <u>measure output range abstol</u> | <u>mpw table</u> |
| <u>measure slew lower fall</u> | <u>mpw vector bin mode</u> |
| <u>measure slew lower rise</u> | <u>msg level</u> |
| <u>measure slew upper fall</u> | <u>msg level user data override</u> |
| <u>measure slew upper rise</u> | <u>msg limit per type per cell</u> |
| n... | |
| <u>net batch mode</u> | <u>non seq copy dst pin</u> |
| <u>nochange mode</u> | <u>non seq copy src pin</u> |
| <u>nochange value</u> | <u>non seq pin swap</u> |
| <u>normalized driver waveform</u> | <u>nonseq as recrem</u> |
| o... | |
| <u>output internal pin</u> | |
| p... | |
| <u>packet arc notification interval</u> | <u>power adjust for pin load</u> |
| <u>packet arc notification limit</u> | <u>power binate arc</u> |
| <u>packet arc notification list</u> | <u>power combinational include output</u> |
| <u>packet arc optimize idle clients</u> | <u>power divide num switching mode</u> |
| <u>packet arcs per thread</u> | <u>power info</u> |
| <u>packet arcs per thread auto adjust</u> | <u>power info filename</u> |
| <u>packet require spectre char opt</u> | <u>power minimize switching</u> |
| <u>packet cell max fets</u> | <u>power model gnd waveform data mode</u> |
| <u>packet clients</u> | <u>power multi output binning mode</u> |

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|--------------------------------------|--|
| <u>packet client idle count</u> | <u>power sequential include complementary output</u> |
| <u>packet client resubmit count</u> | <u>power sim estimate duration</u> |
| <u>packet client timeout</u> | <u>power subtract leakage</u> |
| <u>packet client timeout action</u> | <u>power subtract leakage msg level</u> |
| <u>packet log filename</u> | <u>power subtract leakage mode</u> |
| <u>packet mode</u> | <u>power subtract output load</u> |
| <u>packet rdb mode</u> | <u>power subtract output load mode</u> |
| <u>packet rsh mode</u> | <u>power tend match tran</u> |
| <u>parenthesize not</u> | <u>predriver waveform</u> |
| <u>parenthesize sdf cond</u> | <u>predriver waveform mode</u> |
| <u>parse auto define leafcell</u> | <u>predriver waveform npts</u> |
| <u>parse space bang is comment</u> | <u>predriver waveform ratio</u> |
| <u>pin based leakage</u> | <u>preserve user function</u> |
| <u>pin based power</u> | <u>prevector period</u> |
| <u>pin based signal level mode</u> | <u>prevector slew</u> |
| <u>pin capacitance matching mode</u> | <u>prevector voltage waveform mode</u> |
| <u>pin type order</u> | <u>process match pins to ports</u> |
| <u>pin vdd supply style</u> | <u>process node</u> |
| <u>power add input pin</u> | |
| r... | |
| <u>ramp vsrc</u> | <u>reset leakage current mode</u> |
| <u>rc floating cap mode</u> | <u>reset negative leakage power value</u> |
| <u>rcp cmd</u> | <u>reset negative constraint</u> |
| <u>rdb checkpoint dir</u> | <u>reset negative delay</u> |
| <u>rdb exit if source differ</u> | <u>reset negative leakage power</u> |
| <u>rechar checksum</u> | <u>reset negative power</u> |
| <u>removal glitch peak</u> | <u>resolve collision</u> |
| <u>res merge</u> | <u>retry count</u> |

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|---|---|
| <u>res_open_tol</u> | <u>retry_count_file_operation</u> |
| <u>res_tol</u> | <u>rsh_cmd</u> |
| s... | |
| <u>scale_load_by_template</u> | <u>ski_mdlthreshold_exact</u> |
| <u>scale_tran_by_template</u> | <u>ski_power_subtract_output_load_match_ext_sim</u> |
| <u>scan_dummy_include_leakage_power</u> | <u>ski_reset_cnt</u> |
| <u>sdf_cond_equals</u> | <u>ski_use_large_memory</u> |
| <u>sdf_cond_postfix</u> | <u>skip_nfs_sync</u> |
| <u>sdf_cond_prefix</u> | <u>slew_lower_fall</u> |
| <u>sdf_cond_style</u> | <u>slew_lower_rise</u> |
| <u>sdf_cond_variable_map</u> | <u>slew_normalize</u> |
| <u>sdf_logic_and</u> | <u>slew_upper_fall</u> |
| <u>sdf_logic_not</u> | <u>slew_upper_rise</u> |
| <u>sdf_logic_or</u> | <u>sort_cells</u> |
| <u>server_timeout</u> | <u>sort_groups_under_pin</u> |
| <u>set_var_failure_action</u> | <u>sort_pins</u> |
| <u>sim_default_engine</u> | <u>sort_pins_under_when</u> |
| <u>sim_duration</u> | <u>spectre_dash_log</u> |
| <u>sim_estimate_duration</u> | <u>spectre_use_char_opt_license</u> |
| <u>sim_init_condition</u> | <u>spice_character_map / spectre_character_map</u> |
| <u>sim_init_condition_estimation_mode</u> | <u>spice_delimiter</u> |
| <u>sim_init_duration</u> | <u>spice_delimiter_replacement</u> |
| <u>sim_power_duration_extend</u> | <u>spice_instance_name_require_x_prefix</u> |
| <u>sim_use_init_duration</u> | <u>spice_logical_netname_mode</u> |
| <u>simultaneous_switch</u> | <u>subtract_hidden_power</u> |
| <u>simultaneous_switch_from_cell_when</u> | <u>subtract_hidden_power_use_default</u> |
| <u>simultaneous_switch_offset</u> | <u>supply_define_mode</u> |

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|--|--|
| <u>simultaneous switch use arc when</u> | <u>switch cell bounded dc current</u> |
| <u>simultaneous switch worst vector</u> | <u>switch cell dc current</u> |
| <u>ski alter mode</u> | <u>switch cell dc current output offset</u> |
| <u>ski clean mode</u> | <u>switch cell internal node timing arc</u> |
| <u>ski compatibility mode</u> | <u>switch cell powerdown function</u> |
| <u>ski enable</u> | <u>switch function attr mode</u> |
| t... | |
| <u>template unique power mode</u> | <u>toggle leakage state</u> |
| <u>test cell at end</u> | <u>tran tend estimation mode</u> |
| <u>timing group unateness</u> | <u>tristate disable transition</u> |
| <u>tmpdir</u> | <u>tristate pin cap always on res mode</u> |
| u... | |
| <u>unique pin data</u> | <u>user data override</u> |
| <u>use pid tmpdir</u> | <u>user data quote attributes</u> |
| <u>user data attr order</u> | <u>user data quote simple attr</u> |
| v... | |
| <u>variation mean nominal model mode</u> | <u>verilog cg filter edge</u> |
| <u>vector check mode</u> | <u>voltage map</u> |
| <u>vector estimate dump</u> | <u>vsrc slope mode</u> |
| <u>vector side input</u> | |
| w... | |
| <u>waveform report</u> | <u>write logic function group at end</u> |
| <u>wnflag</u> | <u>write logic function mode</u> |
| <u>write library is unbuffered</u> | <u>write logic function statetable limit</u> |
| <u>write library mode</u> | <u>write logic function statetable mode</u> |
| <u>write library sync ldb</u> | <u>write min transition attr</u> |
| <u>write logic function</u> | <u>write template force power</u> |

adjust_tristate_load

<0 | 1 | 2 | 21 | 22>

Controls whether pin capacitance is added to the load indices on tri-state pins.

Default: 1

- | | |
|----|---|
| 0 | Turns off these adjustments, that is, the library and template do not add or subtract the tri-state pin capacitance. |
| 1 | Adds the pin capacitance of the tri-state pin to each of the load indices when outputting the library. (Default) |
| 2 | Enables functionality similar to 1 with the addition that instead of adding the rise_capacitance or fall_capacitance, the pin attribute capacitance is added to the load indices for the index_2 values. When using the write_template command, the pin capacitance is subtracted from the load indices specified in the input library to create the appropriate define_template commands for tri-state pins. The value of the capacitance attribute can be modified using the set_pin_capacitance command. |
| 21 | Provides an effect that is the same as a setting of 1, but power arc loads are not adjusted. |
| 22 | Provides an effect that is the same as 2, but power arcs are not adjusted. |

By default, Liberate adds the pin capacitance of the tri-state pin to each of the load indices when outputting the library. The rise index_2 adds the rise_capacitance and the fall index_2 adds the fall_capacitance. In addition, when using the write_template command to create a Liberate Tcl command file, the tri-state pin rise_capacitance and fall_capacitance is subtracted from the load indices specified in the input library to create the appropriate define_template commands for tri-state pins.

This variable must be used before the char_library command.

Note: This variable can be used after `char_library`, but must be used before any models are generated.

Example:

```
# Disable adjusting tristate pin load indices
set_var adjust_tristate_load 0
```

adjust_tristate_load_ccsp

| | |
|---------|---|
| <0 1> | Controls adjusting the <code>index_2</code> ccsp dynamic waveforms for tri-state load. This variable must be used in conjunction with <code>adjust_tristate_load</code> . Default: 0 |
| 0 | Load indices of the ccsp current waveforms to be modified even for tri-state caps. |
| 1 | Modification of load indices in the ccsp current groups is avoided when <code>adjust_tristate_load</code> is set to 1 or 2 |

This variable must be used before the `write_library` command.

alspice_diode

| | |
|---------|---|
| <0 1> | Enables an improved algorithm for handling diodes in Alspice. Default: 1 |
| 0 | Enables you to achieve the behavior of releases prior to 3.1 |
| 1 | Default and Recommended. |

This variable must be used before the `read_spice` command.

alspice_leakage_option

{options} Functions the same as `alspice_option`, except that it is targeted for leakage simulations and override values set by `alspice_option`. The available options are: `sim_method`, `sim_gmin`, and `sim_step`.
The recommended setting is `sim_method` to override the default homotopy.

This variable must be used before the `char_library` command.

Example:

```
# Set the leakage options for Alspice
set_var alspice_leakage_option "sim_method=trap sim_gmin=1e-14 sim_step=1e-14"
```

alspice_option

{options} Accepts a list of simulation control options to be used by Alspice for delay, power, and timing constraint characterization. The options are specified in the format "name=value".
Default: There are no fixed default value. Alspice determines appropriate values on a case to case basis.

Note: This variable has no effect on simulations performed by an external simulator.

`sim_method` Specifies the integration method. The valid values are: "trap" and "gear".

Note: The recommend setting for this is either trap or gear. This is in order to override the default homotopy, which is a proprietary algorithm.

`sim_gmin` Specifies gmin for Alspice.

`sim_step` Specifies time step for Alspice.

If the values for `gmin` and `tstep` is not specified, Alspice examines the data and formulate its own determination about appropriate values. However, Alspice always uses the explicitly specified

This variable must be used before the `char_library` command.

Example

```
# Set the simulation options for Alspice
set_var alspice_option "sim_method=trap sim_gmin=1e-12 sim_step=1e-12"
```

auto_index_distinct_risefall

<0 | 1> Controls the computation of both the rising and falling maximum capacitance and load index values when the `-auto_index` option of the `char_library` command is used.
Default: 0

This variable must be used before the `char_library` command.

auto_index_input_slew

<value> Specifies the value (in seconds) of the input slew to be used to calculate `max_capacitance` during `auto_index` characterization. The input slew is measured using the slew threshold values defined by the following parameters: `slew_lower_rise`, `slew_lower_fall`, `slew_upper_rise`, and `slew_upper_fall`. The default is to use `max_transition`.

This variable must be used before the `char_library` command.

auto_index_weak_driver_mode

<0 | 1> Controls if the `-auto_index` option of the `char_library` command adjusts `min_capacitance` of cells that cannot satisfy the `min_transition` at 0 load.
Default: 0 (Do not compensate).
Recommended: 1

| | |
|---|--|
| 0 | Prints a warning if <code>min_slew</code> does not meet the cell output load. Choose the last attempted valid load. |
| 1 | Same as 0. However, ensure that the methodology is consistent between Alspice and <code>extsim_exclusive</code> modes. |

Some cells do not have sufficient drive strength to satisfy the `min_slew` condition when not loaded. Setting this variable allows for better correlation between the `min_load` values generated during `-auto_index` between Alspice and `extsim_exclusive` characterizations.

This variable must be used before the `char_library` command.

binning_detail

`<low | medium | high>`

Sets the level of detail for state dependency.
Default: `high`

| | |
|---------------------|--|
| <code>low</code> | Results in a progressive reduction of the size of the resulting library. |
| <code>medium</code> | Uses detailed state dependency. |
| <code>high</code> | Results in a progressive increase of the size of the resulting library. |

This variable is used to set the criteria for determining how individual state-dependent groups are merged. This variable affects the characterization and should be specified before the `char_library` command.

This variable must be used before the `char_library` command.

Example:

```
# Enable low binning detail
set_var binning_detail low
```

bisection_info

`<0 | 1 | 2 | 3 | 4>` Prints additional constraint bisection information.
Default and recommended: `0`

| | |
|----------------|---|
| <code>0</code> | No information is provided. This is the default and recommended production setting because it minimizes file input/output and provides better runtime and disk usage. |
|----------------|---|

Virtuoso Liberate Reference Manual

Liberate Parameters

- | | |
|---|--|
| 1 | External circuit simulation decks are annotated with bisection information. This mode forces <code>constraint_bisection_mode</code> to 0 (pure bisection). Note: Because many spice decks are utilized in a bisection search, it is recommended to use a setting of 2 or 3, which provides additional information in the log file. |
| 2 | External circuit simulation decks are annotated with search information. The search method is not modified. |
| 3 | The search iteration information is added to the log file as a <code>LIB-405</code> message. For debugging a search, this is the recommended setting. |
| 4 | Similar to mode 3, but adds the pin direction and path to decks for each iteration when the decks are written. |

For constraint and MPW searches, circuit simulation decks can be annotated with comments that give the slew, alignment, and iteration number for each search iteration in that deck. In addition, the log file can contain, along with the measurement result for each search iteration, the same information as the deck. This variable controls the information that is included.

Use this variable for information about debugging constraint bisection searches. To obtain the circuit simulation decks, see the `extsim_deck_dir`, `extsim_save_passed`, `extsim_save_failed` variables, and the `-extsim` argument of the `char_library` command.

Note: It is often easier to follow the debug information while characterizing a single slew or load.

This variable must be used before the `char_library` command.

bundle_when

`<0 | 1>`

Controls the mapping of bundles in a `when` condition.
Default: 1

Virtuoso Liberate Reference Manual

Liberate Parameters

- 0 Use a reference bit from the bundle in the `when` condition. The bit is controlled by the `-use_pin` option of the `define_bundle_pins` command . For example:

```
when : "D1"
```

- 1 Use the bundle name as specified in the `define_bundle_pins` command in the `when` condition.

This setting assumes that all members of the bundles have the same state value:

Supported:

```
when "D" == when " D2 & D1"  
when "!D" == when "!D1 & !D2"
```

Not possible if using bundle name in the `when`:

```
" D1 * !D2"  
" !D1 * D2"
```

Bundle pins that occur in `when` or `sdf_cond` can either use the bundle name, or a reference bit from the bundle. For information on defining bundles, see the [define_bundle_pins](#) command. In the example given below, assume a bundle being defined as follows:

```
define_bundle_pins -use_pin { D1 } MyCell D { D2 D1 }
```

This variable must be used before the `char_library` command.

bundle_arc_mode

`<0 | 1>` Specifies the method to choose the data to output in the bundle.
Default: 0

- 0 If the `-use_pin` option is specified, it acquires the arc data from the specified use pin. If it is not specified, the data is acquired from the first bit of the bundle.

- 1 Liberate automatically selects the timing and power arcs from the characterized pins based on the user-defined criteria.

This variable affects how the `define_bundle_pins` command chooses the pin data to output in the bundle.

This variable must be used before the `write_library` command.

bus_syntax

`<string>`

Specifies the characters that are used to delimit bus indices from the bus name. The valid characters are: "<>", "()" , or "[]" .
Default: "[]"

This variable should only need to be set in conjunction with the `define_bus` command, if the input spice netlist or ldb do not use the default bus syntax. If the bus is defined using the `define_cell` command, then this variable is set automatically. If buses are defined during library characterization, the `bus_syntax` is stored in the ldb and does not need to be reset when reading the ldb.

Note: This variable does not control the `bus_syntax` used when writing the output library. That is controlled by the `-bus_syntax.` option of the `write_library` command.

Any pin in the ldb that ends with a number surrounded by the `bus_syntax` and belongs to a defined bus is outputted under a bus group in the Liberty file. Buses are defined either with the `define_bus` command or by the `-input`, `-output`, or `-inout` options of the `define_cell` command, using the following naming convention:

`<bus_name><bus_syntax_open><from_index>:<to_index><bus_syntax_close>"`

This variable must be used before the `write_library` command.

Exmaples

Example1:

```
"dout[3:0]"  
or  
"din<0:4>"
```

Example 2:

If you want to change the bus syntax in a library, then the following commands in same sequence will work. In this example the bus syntax is "<>" and the library is written with "[]".

```
set_var bus_syntax "<>"  
read_library input.lib  
write_library -bus_syntax "\[\]" -filename output.lib test1
```

capacitance_attr_mode

| | |
|---------|--|
| <0 1> | Controls when the <code>rise_capacitance</code> and <code>fall_capacitance</code> attributes should be written separately. Default: 0 |
| 0 | Separate <code>rise_capacitance</code> and <code>fall_capacitance</code> attributes are written only when both capacitances are measured and are greater than 0. |
| 1 | Separate <code>rise_capacitance</code> and <code>fall_capacitance</code> attributes are always written. |

This variable must be used before the `char_library` command.

capacitance_pin_rollup_k

| | |
|--|--|
| <value> | User-defined multiplier is used in formula for calculating pin capacitance range. The value can be set between 0 to 1.0. Default: 0.5 |
| You can also refer to <u>capacitance_pin_rollup_mode</u> . | |

This variable must be used before the `char_library` command.

capacitance_pin_rollup_mode

| | |
|---------|---|
| <0 1> | Enables an alternate algorithm for calculating pin capacitance and pin capacitance range. Default: 0 |
| 0 | Pin capacitance (<code>rise_capacitance</code> and <code>fall_capacitance</code>) is determined by the command <code>set_pin_capacitance</code> . Pin capacitance range is determined according to the setting of <code>capacitance_range_mode</code> . |

1

Pin capacitance is calculated as an average across all states, timing arcs. and output loads. See [Calculating Pin Capacitance Range](#).

Calculating Pin Capacitance Range

Pin capacitance range [C_{nldm_min} , C_{nldm_max}] is computed as follows:

$$C_{nldm_min} = K * C_{avg} + (1-K) * C_{min}$$

where: $C_{min} = \min [C_{pin}(inpSlew, load, when)]$

$$C_{avg} = \text{avg}[C_{pin}(inpSlew, load, when)]$$

- and -

$$C_{nldm_max} = K * C_{avg} + (1-K) * C_{max}$$

where: $C_{max} = \max [C_{pin}(inpSlew, load, when)]$

$$C_{avg} = \text{avg}[C_{pin}(inpSlew, load, when)]$$

Here,

"K" is a user-defined multiplier set by the variable `capacitance_pin_rollup_k`. The K-factor only changes the values in capacitance rise/fall range and has no effect on "capacitance" attribute itself.

Recommendations:

When using `capacitance_pin_rollup_mode`, set the pin capacitance "state" and "table" to average:

```
set_pin_capacitance -state avg -table avg
```

Also set the capacitance measurement thresholds to 0-100%

```
set_var measure_cap_lower_rise 0.0
set_var measure_cap_upper_rise 1.0
set_var measure_cap_lower_fall 0.0
set_var measure_cap_upper_fall 1.0
```

This variable must be used before the `char_library` command.

capacitance_range_mode

| | |
|---------|--|
| <0 1> | Controls how the capacitance range is calculated. Use the absolute min or max value or avg of min and avg of max. Default: 0 (Use absolute min or max values.) |
| 0 | The lower/upper boundary is the min/max value of all the capacitances. |
| 1 | The lower boundary value is calculated as the average across all states, timing arcs, and output loads for the smallest input slew. The upper boundary value is calculated as the average across all states, timing arcs, and output loads for the largest input slew |

This variable must be used before the `write_library` command.

capacitance_save_mode

| | |
|---------|---|
| <0 1> | Saves all pin capacitances. Default: 0 Recommended: 1 |
| 0 | Remove negative capacitances if <code>ccs_warn_negative_rcvr_caps</code> = 1. Keep negative capacitances if <code>ccs_warn_negative_rcvr_caps</code> = 0. |
| 1 | Saves all capacitances. Negative capacitances are saved as positive capacitances. If the tool-chain libraries are used, the recommendation is to keep negative receiver capacitances as this improves accuracy. To do this, leave the <code>capacitance_save_mode</code> variable at the default value of 0 and set <code>ccs_warn_negative_rcvr_caps</code> to 0. |

Liberate derives capacitance values by integrating current. In cases where current direction is reversed from expectations, it is possible that capacitance values are negative. This condition is rare and usually is only seen on some pass-gate or tristate designs.

This variable must be set before the `char_library` command.

ccs_abs_tol

`<value>` Specifies the CCS absolute tolerance.
Default: `1e-13`

Use this control variable to set the CCS absolute tolerance (in seconds). When determining how many points are needed to reproduce the original SPICE waveform, Liberate stops adding points to the CCS data when the absolute error between the reduced CCS waveform and the original SPICE waveform is less than the specified tolerance.

This variable must be used before the `char_library` command.

ccs_base_curve_points

`<value>` Specifies the number of base curve points.
Default: 15

Use this variable to specify the number of base curve points used when generating compact CCS natively in Liberate. To output compact CCS format data, use the `write_library -ccs_compact` option.

This variable can be used after the `char_library` command.

ccs_base_curve_share_mode

`<0 | 1 | 2>` Determines which algorithm is selected for reusing CCS base curves.
Default and recommended: 2

| | |
|---|---|
| 0 | Set this variable to 0 to revert to release 2.3p2 and prior release behavior. |
| 1 | Selects an algorithm that uses a more aggressive base curve re-use rate without impacting accuracy. There is no significant impact on accuracy when using either mode 1 or 2. |

| | |
|---|--|
| 2 | Selects an algorithm that uses a more aggressive CCS compaction algorithm. There is no significant impact on accuracy when using either mode 1 or 2. |
|---|--|

This variable can be used after the `char_library` command.

ccs_cap_duplicate_risefall

`<-1 | 0 | 1>`

Populates a missing rise or fall CCS receiver capacitance by duplicating the appropriate rise or fall CCS receiver capacitance from existing arc-based or pin-based data.
Default: 0 (copy from arc-based data)

Note: If a timing arc has a missing rise or fall CCS receiver capacitance, this variable controls how Liberate handles various scenarios.

`-1`

Leaves the missing group unpopulated. What is represented in the Liberty model is exactly what was characterized. It is recommended to use this value if all meaningful switching and hidden arcs are characterized.

`0`

Copies the appropriate CCS receiver capacitance from the input pin involved in the timing arc.

Note: This default setting is a more accurate representation of the pin capacitance but might cause problems with downstream tools.

`1`

Creates the missing rise or fall CCS receiver capacitance by duplicating the existing rise or fall CCS receiver capacitance.

Note: Two-sided arcs are functionally impossible in certain designs. Therefore, setting `-1` is the most accurate representation of such designs. Settings `0` and `1` were implemented to prevent Library Compiler and certain downstream tools from rejecting libraries that only have one-sided arcs, but are not as accurate as `-1` setting. If the tool-chain requires one-sided arcs to be fully populated, setting `0` is recommended.

This variable must be used before the `write_library` command.

`ccs_cap_hidden_pin`

| | |
|--------------------------------|--|
| <code><0 1 2></code> | Controls the output of the CCS receiver pin capacitance on inputs with hidden power arcs. Default: 2 (Output pin capacitance for hidden arcs) |
| 0 | Uses the behavior of release 2.3p1 and prior where Liberate only saved the CCS receiver capacitance on "hidden" pins such as the D pin of a flip-flop. |
| 1 | Outputs CCS receiver capacitance on input pins that have "hidden" transitions such as clock, clear, preset, <code>combinational_rise</code> , <code>combinational_fall</code> , <code>tristate_enable</code> , and <code>tristate_disable</code> pins. |
| 2 | Output CCS receiver capacitance on all input pins that have potential "hidden" conditions; <i>any</i> pin that has a hidden power arc will also have CCS receiver capacitance. |

This variable must be specified before the `char_library` command.

`ccs_cap_hidden_pin_mode`

| | |
|-----------------------------|---|
| <code><0 1 ></code> | Enables Liberate to store the CCS receiver capacitance data into the LDB for all <code>hidden_power</code> arcs. Default: 0 (Output pin capacitance for hidden arcs) |
| | Recommended: 1 |

Note: The `-state` option of the `set_pin_capacitance` command specifies the method used to determine the capacitance in the default receiver capacitance under the pin.

| | |
|---|---|
| 0 | Only stores the CCS receiver capacitance data for one of the characterized <code>hidden_power</code> arcs. |
| 1 | Stores the CCS receiver capacitance data for all characterized <code>hidden_power</code> arcs. (Recommended) |

This variable must be specified before the `char_library` command.

ccs_cap_is_propagating

| | |
|----------|--|
| <0 1 > | Outputs <code>is_propagating</code> attributes. Default: 1 |
| 0 | Does not output the <code>is_propagating</code> attribute. |
| 1 | Adds the <code>is_propagating</code> attribute to pin-based <code>ccs_receiver</code> cap tables if there is also a timing arc from the same pin with the same when condition. |

This variable must be specified before writing out a library (see [write_library](#)).

ccs_cap_mode_add_missing

| | |
|----------|---|
| <0 1 > | Specifies whether to get 2.4-style libraries with missing receiver capacitance for one-sided timing arcs. Default: 1 |
| 0 | Gets 2.4-style libraries with missing receiver capacitance for one-sided timing arcs |
| 1 | Gets 2.4-style libraries without missing receiver capacitance for one-sided timing arcs. |

This variable must be used before the `write_library` command.

ccs_cap_use_input_transition

| | |
|-------------|---|
| <0 1 2> | Determines the pin transition direction followed to determine the CCS receiver capacitance. Default and recommended: 1 |
|-------------|---|

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|---|--|
| 0 | Reverts to the old behavior where the CCS receiver capacitance follows the <i>output</i> pin direction. Set this variable to 0 after <code>read_ldb</code> or <code>char_library</code> and before <code>write_library</code> . |
| 1 | Instructs Liberate to use the <i>input</i> pin transition direction for CCS receiver capacitance. |
| 2 | Enables an algorithm that improves the accuracy of the <code>index_1</code> in the CCS receiver capacitance when the input waveform has an unusual shape and the input slew thresholds are non-standard. The input waveform may have an unusual shape, for example, when an active driver is used (see <code>set_driver_cell</code>). The standard input slew thresholds are 10/90, 20/80, and 30/70 with a delay threshold at 50% of supply. |

The `read_library` command resets this variable to 0. The `char_library` and `read_ldb` commands sets this variable to 1.

If `ccs_cap_use_input_transition` and `ccs_cap_hidden_pin` is set to 1, then for one sided arcs such as "clock rising_edge to Q", the inactive edge (falling) uses the CCS receiver capacitance from the clock pin. The load index for this entry (`index_2`) is set to "0" since this data has no load dependency.

This variable must be set before the `char_library` command.

`ccs_cap_use_input_transition_tristate`

| | |
|----------|--|
| <0 1 > | Specifies that the CCS receiver capacitance for tristate arcs follows the <u>input</u> pin direction. Default: 1 |
| 0 | No check is performed to ensure that the CCS receiver capacitance for tristate arcs follows the input pin direction. |
| 1 | Checks that the CCS receiver capacitance for tristate arcs follows the <u>input</u> pin direction. |

This variable must be used before the `char_library` command.

`ccs_correct_current_by_area`

| | |
|----------|---|
| <0 1 > | Applies an alternate smoothing method that tracks the total area of current to be conserved during the smoothing process, if <code>ccs_waveform_smooth_mode</code> is set to 1. Do not use with <code>ccs_smooth_lower_rise</code> or <code>ccs_smooth_upper_fall</code> . Default: 0 (disabled) |
| 0 | Disables correct method. |
| 1 | Enables correct current by area method for all simplified waveforms. |

This variable must be used before the `char_library` command.

`ccs_current_model_pin_load`

| | |
|----------|---|
| <0 1 > | Controls whether to characterize and model the CCS current (<code>output_current_rise</code> and <code>output_current_fall</code>) tables even when the output has a <code>pin_load</code> applied. See also the <code>define_pin_load</code> command and the <code>-pin_load</code> and <code>-load_dir</code> options of the <code>define_arc</code> command. Default: 0 |
| 0 | Instructs Liberate to not to characterize and model CCS current if the output has a parasitic load applied. |
| 1 | Instructs Liberate to measure and report the CCS current even if a <code>define_pin_load</code> has been run. |

This variable must be set prior to the `char_library` command.

`ccs_enable_sawtooth_out`

| | |
|---------|--|
| <0 1> | Enables the detection and correction of CCS with double-peaks current when smoothing is enabled. Default: 0 |
|---------|--|

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|---|--|
| 0 | Does not detect double-peak (sawtooth) CCS current waveform data. |
| 1 | Detects double-peak CCS current waveform data and applies smoothing to the CCS current waveform. |

This variable must be used before the `char_library` command.

ccsn_floating_init_mode

`<0 | 1 | 2 | 3>` Controls how Liberate initializes floating internal cell simulation wires in the Channel Connected Block (CCB) for CCSN simulation. These are nodes that are internal to the cell (not primary ports).

Default: 1 (use `.nodeset`)

Recommended: 3

0 Uses `.ic` in all CCSN simulations. This setting is used for backward compatibility for 12.1 ISR3 and prior releases.

1 Uses `.nodeset` in all CCSN simulations.

2 Uses `.nodeset` in CCSN dc simulations and `.ic` in CCSN transient simulations.

3 Uses `.nodeset` in all CCSN simulations except `dc_current` simulations which have no `.nodeset` or `.ic`. The simulator will need to initialize all nodes for the `dc_current` simulations.

This variable must be set prior to the `char_library` command.

ccs_force_grid_delay

`<0 | 1 | 2>` Controls accuracy algorithms for checking CCS waveforms. Default and recommended: 1

0 Does not force CCS waveforms to use the existing grid, so the size of the grid might change if the waveforms require it.

- | | |
|---|---|
| 1 | Forces CCS waveforms to use the existing grid and checks delay accuracy and transition accuracy when segmenting the waveform. |
| 2 | Forces CCS waveforms to use the existing grid but does not check delay accuracy when segmenting the waveform. |

This variable must be used before the `char_library` command.

`ccs_infer_output_dir`

- | | |
|----------|--|
| <0 1 > | Detects whether all outputs have both rising and falling waveforms. Default: 1 |
| 0 | Address CCST issues with pulse generators having rising and falling waveforms in every output. |
| 1 | Detect whether all outputs have both rising and falling waveforms. |

This variable must be used before the `char_library` command.

`ccs_init_voltage_comp_thresh`

- | | |
|----------------------------|--|
| <code><value></code> | Sets a voltage threshold to enable a proprietary initial voltage offset. Default: 1.1 (fully-compensated to improve correlation for PrimeTime 2010.06 and prior releases) Set this variable to -1 to disable the compensation for improving correlation for PrimeTime 2010.12 and later releases. |
|----------------------------|--|

The Synopsys CCS standard requires that the CCS waveform start from an initial rail voltage. If the SPICE output waveform starts from a non-rail value (usually due to a relatively large leakage), the resulting CCS waveform may have an additional time delay and might not reach the final voltage rail. Liberate compensates for the non-rail initial voltage to ensure the CCS waveform matches the NLDM values and reaches the final rail voltage.

This variable represents a percentage of the full-rail swing over which compensation takes place. If set to a value larger than 1, the maximum threshold for the compensation is controlled by the `measure_slew_lower_rise` and `measure_slew_upper_fall` variables. If set to a value between 0 and 1, Liberate compensates to the value given. If set to a value less than 0, compensation is disabled.

The NLDM and CCS models are currently limited in how to handle non-rail starting or ending voltage levels, since `output_signal_level` attributes must be applied at the pin level. In cases where non-rail behavior is only observed for specific `WHEN` conditions, the accuracy of those models is degraded. See `ecsm_measure_output_range` for accurate modeling of these effects.

This variable must be set before the `char_library` command.

`ccs_max_current_thresh`

`<value>` Checks for CCS currents greater than the specified threshold.
Default: 0.2A (200mA)

Use MKS units for `<value>`. If the absolute CCS current is larger than the threshold, Liberate issues a warning.

This variable must be used before the `char_library` command.

`ccs_max_pts`

`<value>` Sets the maximum number of CCS points that are allowed in the CCS waveform data.
Default: 50

When determining how many points are needed to reproduce the original SPICE waveform, Liberate stops adding points to the CCS data when the number of points reaches the limit specified by this variable.

This variable must be used before the `char_library` command.

`ccs_multiple_switching_output_mode`

`<0 | 1>` Enables handling of multiple switching output voltage.
Default: 0

| | |
|---|--|
| 0 | The output waveform does not have special processing. |
| 1 | Filters a multiple switching output voltage waveform. Keeps the first or last rise or fall transition waveform according to the setting of the variable <code>measure_target_occurrence</code> . |

When the variable `simultaneous_switch` is enabled and is greater than 0, the output voltage behavior can be a multiple switching waveform (that is a waveform with multiple rise or fall edges). This can produce undesirable negative and/or positive current values in the rise or fall waveform. Liberate uses a fixed value to make the CCS library data LC friendly. This could lead to NLDM versus CCS comparison (`compare_ccs_nldm`) failures.

This variable must be used before the `char_library` command.

`ccs_rel_tol`

`<value>` Sets the CCS relative tolerance.
Default: 0.001

When determining how many points are needed to reproduce the original SPICE waveform, Liberate stops adding points to the composite current source (CCS) data when the relative error between the reduced CCS waveform and the original SPICE waveform is less than this relative tolerance.

This variable must be used before the `char_library` command.

`ccs_segmentation_effort`

`<0 | 1 | 2 | 3 | 4>` Enables selection of the method used to acquire the CCS current waveform from simulated $I(t)$ data.
Default and recommended: 1

| | |
|---|--|
| 0 | Disables CCS waveform processing. Use values measured from the current waveform. |
| 1 | Smoothed algorithm. |
| 2 | dv/dt -based algorithm |

- | | |
|---|---|
| 3 | Use both 1 and 2, but choose the waveform that best correlates to NLDM. |
| 3 | Same as 1, but also apply CCS retry criteria (ccs_retry_mode) to check the reconstructed waveform instead of the original current waveform. |

This variable must be used before the `char_library` command.

`ccs_smooth_lower_rise`

`<value>` Specifies a lower rise threshold to enable current waveform smoothing for rising outputs.
Default: -1 (Disabled)

Sometimes, the simulator reports a severely non-monotonic CCS current waveform. If this *distorted* waveform is provided to the timer, it might cause significant timing differences. To avoid this error in the timer, Liberate can smooth the current waveform. To enable this smoothing, set the rise and fall thresholds by using the `ccs_smooth_lower_rise` and `ccs_smooth_upper_fall` variables, respectively. A higher `ccs_smooth_lower_rise` value corresponds to more aggressive smoothing. To minimize the introduction of errors, the specified value should be equal to or less than the `measure_slew_lower_rise` threshold.

When a value is set for this variable, Liberate checks if the CCS current waveform generated by the simulator has multiple peaks, where the initial peak is greater than the second peak. When detected, Liberate applies a proprietary smoothing algorithm to smooth the initial current peak to ensure the waveform meets the Liberty syntax rules while maintaining the original voltage/time on the output waveform at the specified threshold.

Ensure this variable is used along with the corresponding fall threshold set by using the `ccs_smooth_upper_fall` variable.

Note: The smoothing algorithm can have an impact on downstream timing or noise tools that are sensitive to the specified threshold.

This variable must be used before the `char_library` command.

Example

```
set_var ccs_smooth_lower_rise 0.2
```

ccs_smooth_upper_fall

`<value>` Specifies a upper fall threshold to enable current waveform smoothing for falling outputs.
Default: -1 (Disabled)

Sometimes, the simulator reports a severely non-monotonic CCS current waveform. If this *distorted* waveform is provided to the timer, it might cause significant timing differences. To avoid this error in the timer, Liberate can smooth the current waveform. To enable this smoothing, set the rise and fall thresholds by using the `ccs_smooth_lower_rise` and `ccs_smooth_upper_fall` variables, respectively. A lower `ccs_smooth_upper_fall` value corresponds to more aggressive smoothing. To minimize the introduction of error, the specified value should be equal to or greater than the `measure_slew_upper_fall` threshold.

When a value is set for this variable, Liberate checks if the CCS current waveform generated by the simulator has multiple peaks, where the initial peak is greater than the second peak. When detected, Liberate applies a proprietary smoothing algorithm to smooth the initial current peak to ensure the waveform meets the Liberty syntax rules while maintaining the original voltage/time on the output waveform at the specified threshold.

Ensure this variable is used along with the corresponding rise threshold set by using the `ccs_smooth_lower_rise` variable.

Note: The smoothing algorithm can have an impact on downstream timing or noise tools that are sensitive to the specified threshold.

This variable must be used before the `char_library` command.

Example

```
set_var ccs_smooth_upper_fall 0.8
```

ccs_voltage_smooth_thresh

`<voltage | -1>` Sets a voltage threshold (for example, 0.2) to enable a proprietary smoothing algorithm.
Default and recommended: -1 (Disabled)

Note: The smoothing algorithm may have an impact on downstream timing or noise tools that are sensitive to that threshold.

| | |
|---------|--|
| voltage | Enables the smoothing algorithm. A higher number corresponds to a more aggressive smoothing. |
| -1 | Disables the smoothing |

To reduce CCS interpolation error for some corner cases, Liberate can artificially smoothen the waveform.

This variable must be used before the `write_library` command.

ccs_voltage_tail_tol

| | |
|----------------------------|--|
| <code><value></code> | The stopping point as a ratio of supply of the CCS waveform for modeling purposes. Default: 0.955 (Output must swing to within 4.5% of the supply.) |
|----------------------------|--|

For CCS timing waveform data, if the tail of the integrated $v(t)$ obtained from the sampled $i(t)$ does not reach $\$ \{ccs_voltage_tail_tol\} \times \text{supply_swing}$, then $i(t)$ is padded to ensure that the integrated voltage reaches the supply rail.

As this parameter can cause Liberate to pad the current waveform, it is recommended to set this value lower than `ccs_voltage_tail_trim_tol`, but higher than the requirement of downstream tools (for example, 5% in Synopsys' Library Compiler).

This variable must be used before the `char_library` command.

Example

```
set_var ccs_voltage_tail_tol 0.951
```

ccs_voltage_tail_tol_mode

| | |
|--------------------------------|---|
| <code><0 1 2></code> | Controls the padding for the CCS "tail". Default: 2 |
| 0 | Extends the tail as long as possible to reach the <code>ccs_voltage_tail_tol</code> value with the last current (I) close to 0. |

- | | |
|---|---|
| 1 | Extends the tail with limited step. The last current might have a small spike to reach the <code>ccs_voltage_tail_tol</code> value. |
| 2 | Pads the current to ensure the integrated voltage reaches the supply rail tolerance. If the selected pad time is small, this results in a large spike in the current. |

This variable determines how to pad the CCS tail if the tail of the integrated $v(t)$ obtained from the sampled $i(t)$ does not reach $\$ \{ccs_voltage_tail_tol\} * supply_swing$.

This variable must be used before the `char_library` command.

`ccs_voltage_tail_trim_tol`

`<value>` The stopping point as a ratio of supply of the CCS waveform during simulation waveform capture.
Default: 0.999 (within 0.1% of rail supply voltage)

For CCS timing waveform data, the tail of the integrated $v(t)$ obtained from the sampled $i(t)$ is captured until $\$ \{ccs_voltage_tail_trim_tol\} * supply_swing$, is reached.

This parameter controls the current waveform capture and works with `ccs_voltage_tail_tol` to control the modeling of the CCS current waveforms. For 28nm processes and below, it is strongly recommended to set this value higher than `ccs_voltage_tail_tol`, but not to 100%. This helps to remove "N-curve" phenomena in the CCS current waveforms.

The default setting should only be used for large geometries where capturing to within a tighter range creates such a large waveform that in time duration causes problems in legacy downstream tools.

This variable must be used before the `char_library` command.

Example

```
set_var ccs_voltage_tail_trim_tol 0.999
```

ccs_warn_negative_rcvr_caps

| | |
|-------------|--|
| <0 1 2> | Controls the type of data that are checked for handling of negative receiver capacitances. Default: 0 |
| 0 | Disables the check. |
| 1 | Check both nominal and sensitivity. |
| 2 | Only check nominal or Liberate capacitance. |

Liberate derives capacitance values by integrating current. In cases where current direction is reversed from expectations, it is possible that capacitance values are negative. This condition is rare and is usually only seen on some pass-gate or tristate designs.

If the tool-chain libraries are used, the recommendation is to keep negative receiver capacitances because this will improve accuracy. In order to do this, leave capacitance_save_mode at its default value of 0 and set ccs_warn_negative_rcvr_caps to 0.

This variable must be used before the `write_library` command.

ccs_waveform_min_time_step

| | |
|---------|---|
| <value> | Specifies the minimum time step supported in the <code>ccs_waveform(index_3)</code> . This is used when adjusting time for monotonicity. Default: 1e-16 (in seconds) |
|---------|---|

This variable must be set prior to the `write_library` command.

ccs_waveform_smooth_mode

| | |
|---------|---|
| <0 1> | Enables an algorithm to smooth non-monotonic voltages resulting from the <code>output_current_*</code> waveforms. Do not use this variable with <code>ccs_smooth_lower_rise</code> or <code>ccs_smooth_upper_fall</code> . Default: 0 (disabled) |
| 0 | Disables ccs smooth method. |
| 1 | Enables new ccs smooth method. |

This variable must be used before the `char_library` command.

`ccsn_active_ccr_recognition_mode`

Deprecated. See [Backward Compatibility Variables](#).

`ccsn_allow_duplicate_condition`

| | |
|-----------------------------|---|
| <code><0 1 ></code> | Allows multiple CCSN groups to be written to the library. Default: 0 |
| 0 | Keep only the worst-case group, and do not include the rest. |
| 1 | Allow multiple CCSN groups to be written to library. |

Sometimes CCSN characterization results in having multiple `ccsn` groups with the same WHEN condition (e.g. internal nodes might be initialized differently and are not captured in the WHEN condition). By default these groups are written into the library.

Note: We recommend setting this variable consistent with the requirements of the noise analysis tool that will be using the library. If the noise analysis tool does not support duplicate conditions, then set this variable to 0.

This variable must be used before the `char_library` command.

`ccsn_allow_multiple_input_switching`

| | |
|--------------------------------|---|
| <code><0 1 2></code> | Allows the vectors on the various input pins on a Channel Connected Region (CCR) to switch independently. Default and recommended: 2 |
| 0 | Use this setting when the input pins to the CCR cannot switch independently. |
| 1 | Allow the input pins to the CCR to switch independently. Use this when characterizing CCRs that only propagate noise when the inputs can be toggled in opposite directions independently. |

| | |
|---|---|
| 2 | This is an enhancement over the setting of 1 to improve the recognition of multiple switching events in CCSN decks. |
|---|---|

This variable must be used before the `char_library` command.

ccsn_allow_overlap_when

| | |
|-----------------------------|--|
| <code><0 1 ></code> | Control output of arc-based ccsn data. Default: 1 |
| 0 | Revert back to the release 2.3 and earlier behavior. |
| 1 | Allow checking for boolean overlap |

Liberate checks for boolean overlap instead of only an explicit *when* string comparisons. This improves on an issue in Liberate where vector data is not properly matched up to the corresponding timing groups for arc based CCSN generation. This occurs only in the `write_template` verbose flow because the timing group can have a non-empty *when* string, but the Vecdata would have a null string for the *when* condition.

This variable must be used before the `char_library` command.

ccsn_allow_partial_voltage_swing

| | |
|-----------------------------|---|
| <code><0 1 ></code> | Allows CCSN <code>output_voltage</code> groups to use partial voltage swings instead of the values recommended in the CCSN Characterization Guideline. Default: 1 |
| 1 | Liberate overrides the recommendations with the voltage levels measured in the transition. Where possible, the voltage levels from the recommendations are used. If necessary, indexes and values are padded to insure monotonicity |

By default, Liberate uses the voltages recommended in the CCSN Characterization Guidelines for `output_voltage_*` groups. These are 10%, 30%, 50%, 70%, and 90% of the rail voltage. In certain cases, cells are unable to meet these voltage levels - usually due to large

leakage current. In these cases, Liberate issues a warning during characterization and a warning during modeling. These CCSN groups are not written out in the library.

This variable must be used before the `char_library` command.

`ccsn_arc_channel_check`

| | |
|-----------------------------|---|
| <code><0 1 ></code> | Enables an aggressive pruning of non channel connected CCSN stages that might end up as arc-level CCSN constructs by enabling a more thorough vector dependent check. Default and recommended: 1 |
| 0 | Enables pruning. |
| 1 | Disables pruning. (Achieves 3.0p3 and prior behavior.) |

This variable must be used before the `char_library` command.

`ccsn_arc_consistent_cut`

| | |
|-----------------------------|--|
| <code><0 1 ></code> | Uses an enhanced algorithm to ensure that a consistent cut node is chosen for arc based CCSN. CCSN generation in Liberate chooses a single stage CCSN arc over a two stage CCSN arc, whenever possible.. Default: 1 |
| 0 | Revert back to release 2.4p2 and prior behavior |
| 1 | Enable an enhanced CCS cut algorithm. |

This variable must be used before the `char_library` command.

`ccsn_arc_high_effort`

| | |
|--------------------------------|--|
| <code><0 1 2></code> | Control output of arc-based ccsn data. Default: 1 |
| 0 | Revert to the pre 2.2p2 behavior, which does not output state-dependent ccsn arcs. |

- | | |
|---|---|
| 1 | Enable the state dependent CCSN arcs. |
| 2 | Same as 1, also outputs a message into the log file for each cell that triggers state-dependent arc code. |

Liberate outputs more arc-based ccsn stages by applying "looser" side input requirements. The result is that some types of cells, such as muxes, have an increase in the number of arc-based ccsn data. This control variable can be used to revert back to the pre 2.2p2 behavior.

This variable must be used before the `char_library` command.

Example:

```
set_var ccsn_arc_high_effort 0
```

ccsn_bus_holder_mode

- | | |
|-------------|---|
| <0 1 2> | Specifies methodology for modeling unbuffered output or inout pins. Default and recommended: 2 |
| 0 | Do not model unbuffered outputs. (Behavior of release 3.2p2 or earlier.) |
| 1 | Cleanup first stage only; do not add a last stage. |
| 2 | Cleanup feedback in first stage (on pins having <code>bidi</code> ports) and add a last stage. |

This variable must be used before the `char_library` command..

ccsn_channel_inputs_high_effort

- | | |
|----------|--|
| <0 1 > | Controls merging of CCSN data. Default: 1 |
| 0 | Revert to release 2.3 and earlier behavior. |
| 1 | Enable CCSN constructs for un-buffered input pins. |

This variable must be used before the `char_library` command.

ccsn_compatibility_mode

Deprecated. See [Backward Compatibility Variables](#).

ccsn_consistent_side_inputs

| | |
|----------|--|
| <0 1 > | Enables to avoid a crash that can occur when there are CP->Q arcs with no when conditions but different internal side-input values. Default: 1 |
| 0 | Revert to release 2.3 and earlier behavior. |
| 1 | Avoids a crash. |

This variable must be used before the `char_library` command.

ccsn_controlling_path_check

| | |
|----------|---|
| <0 1 > | Specifies how to handle CCSN output nodes that are driven by multiple paths including pass gates. Default: 0 |
| 0 | Apply default algorithm. |
| 1 | Check the CCSN CCCs for the presence of multiple active paths that could affect the output pin. When found, Liberate preserves the primary active path from the related pin to the CCSN CCC output while disabling secondary paths. Liberate identifies and modifies the controlling signals at the pass gates along the secondary paths. If such a CCC is identified at the arc level, the CCSN group is skipped and a CCSN group is identified at the pin-level instead that can be modified as previously described. |

This may occur when the CCC output probe wire is controlled by pass gates as well as normal driver stages. In addition, some of the pass gates are controlled by the related pin directly (via

the same CCC) or indirectly by other signals. Turning them OFF or ON completely as a part of the same CCSN stage is problematic.

This variable must be used before the `char_library` command.

ccsn_dc_static_check

| | |
|--------------------------------|---|
| <code><0 2 3></code> | Performs a check to determine if the CCSN partition is valid. Default: 0 |
| 0 | Do not perform any data checks on CCSN DC tables. |
| 2 | Apply basic data checks on <code>ccsn_first_stage</code> DC tables. |
| 3 | Apply a data check by walking across the DC table and comparing the values at 0 and VDD. A potential error is flagged if the values indicate that the CCSN partition has not been "turned on", that is, the values must show a sign change, or magnitude difference of 10 or greater to be considered a valid CCSN partition. For more information, see <u>ccsn_dc_static_check_thresh</u> . |

This variable must be used before the `write_library` command.

ccsn_dc_static_check_mode

| | |
|-----------------------------|--|
| <code><0 1 ></code> | Controls severity of messaging when a problem in the CCSN data is found. Default: 1 |
| 0 | Report a warning when a <code>ccsn_dc_static_check</code> is enabled and a problem in the data is found. |

| | |
|---|--|
| 1 | Report an error when a <code>ccsn_dc_static_check</code> is enabled and a problem in the data is found. In addition, remove the problem CCSN data table from the output library. |
|---|--|

This variable must be used before the `write_library` command.

`ccsn_dc_static_check_thresh`

| | |
|----------------------------|--|
| <code><value></code> | Specifies the threshold applied when <code>ccsn_dc_static_check=3</code> . Set this to the desired threshold that identifies a turn on or a non-static behavior in a CCSN DC table. Default: 10 |
|----------------------------|--|

This variable must be used before the `write_library` command.

`ccsn_dc_template_size`

| | |
|----------------------------|--|
| <code><value></code> | Specifies the size of the CCSN DC current table to be used. Default and recommended: 29 |
|----------------------------|--|

For larger geometries, this variable can be set to 17 to allow for better characterization runtime. For advanced nodes or if using a version of PrimeTime that is 2012.x or later.

This variable must be used before the `char_library` command.

`ccsn_default_group_add_when`

| | |
|-----------------------------|--|
| <code><0 1 ></code> | Adds WHEN condition inside default CCSN groups. Default: 0 (Do not add WHEN.) |
| 0 | <u>Does not add</u> WHEN condition inside the default CCSN groups. |
| 1 | <u>Add</u> WHEN condition inside the default CCSN groups. |

This variable must be used before the `char_library` command.

`ccsn_default_group_criteria_mode`

| | |
|----------|--|
| <0 1 > | Enables generation of worst case CCSN default groups at the pin-level and at the default timing group. Default and recommended: 0 |
| 0 | Generate libraries that are structurally independent of per-corner data. |
| 1 | Generate the most conservative per-corner libraries. |

 *Important*

For libraries to be used in voltage or temperature scaling flows, we highly recommended setting this parameter to 0

This variable has effect only if the library has timing arcs without CCSN groups.

Note: This may result in multiple default CCSN groups, depending on the data generated during characterization.

This variable must be used before the `char_library` command.

`ccsn_dual_tie_enable`

| | |
|----------|--|
| <0 1 > | Enables CCSN modeling for <code>tieHi</code> and <code>tieLow</code> cells. Default: 1 |
| 0 | Revert back to 2.5 and prior release behavior, where CCSN data was not output for <code>tieHi</code> or <code>tieLow</code> cells. |
| 1 | Enables CCSN modeling for <code>tieHi</code> and <code>tieLow</code> cells. |

This variable must be used before the `char_library` command.

ccsn_extra_default_stages

`<value>` Control merging of ccsn data.
Default: 1

When this parameter is set to 2, Liberate will try to add redundant default CCSN stages on all cell pins.

This variable must be used after the `char_library` command.

ccsn_fanout_select_mode

Deprecated. See [Backward Compatibility Variables](#).

ccsn_include_passgate_attr

| | |
|-----------------------------|---|
| <code><0 1 ></code> | Enables modeling of <code>pass_gate</code> attributes. Default: 0 |
| 0 | Does not automatically model these <code>pass_gate</code> related attributes. |
| 1 | Automatically model these <code>pass_gate</code> related attributes. |

The Liberty syntax supports the following pin-level attributes for unbuffered output latches:

- `is_unbuffered`: Indicates that a pin is unbuffered.
- `has_pass_gate`: Indicates whether the pin is internally connected to at least one pass gate.
- `is_pass_gate`: Indicates whether the `ccsn_*_stage` data is modeled for a pass gate and is specified in a `ccsn_*_stage` group (such as `ccsn_first_stage`).

This variable must be used before the `char_library` command.

ccsn_input_xfr_probe_mode

Deprecated. See [Backward Compatibility Variables](#).

ccsn_io_allow_multiples

| | |
|----------|---|
| <0 1 > | Enables exhaustive CCSN partition analysis when the IO (port-based) partitioning algorithm is used. Default: 1 |
| 0 | Liberate constructs a CCSN partition CCC using a heuristic based algorithm. |
| 1 | Liberate constructs and analyzes all possible CCSN partitions when the IO partitioning algorithm is used. This setting increases the runtime. You can use this setting when a more detailed IO partition based CCSN analysis is desired |

The generation of CCSN data requires special circuit partitions. When inside view is unable to sensitize a Channel Connected Component/Region (CCC) or is disabled, the CCSN circuit partitions are constructed by tracing the CCC connected to the cell port. This variable can be used to enable a faster or a more detailed CCSN port-based partitioning algorithm.

This variable must be used before the `char_library` command.

ccsn_io_mode

| | |
|----------|--|
| <0 1 > | Enables CCSN model generation when <code>define_cell -type</code> is set to UDA or when <code>char_library -io</code> is used or the <code>inside_view</code> algorithm fails. Default: 4 |
| 0 | Disable the IO mode ccsn generation algorithm. |
| 1-3 | These settings are deprecated and are for backward compatibility only. |

4

Enable enhanced recognition of Channel Connected Blocks (CCBs). The sensitization includes:

- Recognition of complementary signals, either external or internally generated.
- Recognition of a larger cone of logic so that all of the CCB side pins are initialized properly for a clean switching CCB output.
- Handles situations where there is no proper inversion in the CCB and inputs to complementary PMOS and NMOS devices are controlled by interdependent signals that must switch simultaneously for a clean event at the CCB output.
- Handles multiple topologies where transmission gates are involved.
- This setting must be used with `ccsn_allow_multiple_input_swit`ching set to 2 (or a compatible setting).

This variable must be used before the `char_library` command.

`ccsn_io_mode_enable`

`<0 | 1 >`

Enables IO CCSN circuit partitioning algorithm.

Default and recommended: 1

0

Do not use the IO CCSN circuit partitioning algorithm when Inside View is enabled.

1

Allow Liberate to use the IO CCSN circuit partitioning algorithm if the Inside View-based partitioning fails to successfully characterize the CCSN.

The generation of CCSN data requires special circuit partitions. When the Liberate Inside View algorithm is enabled, the circuit partitions are constructed from the characterized timing

arcs. When the `char_library -io` mode is used, Inside View is disabled. In this case, the CCSN circuit partitions are constructed from the circuit by tracing the Channel Connected Component/Region (CCC) connected to the cell port.

This variable must be used before the `char_library` command.

`ccsn_miller_init_mode`

| | |
|-----------------------------|---|
| <code><0 1 ></code> | Controls whether to use <code>.nodeset</code> or <code>.ic</code> to initialize the primary input to the Channel Connected Block (CCB) used for measuring the input pin miller capacitance. Default: 1 (use <code>.ic</code>) |
| 0 | Uses <code>.nodeset</code> to initialize the input probe while writing miller decks. |
| 1 | Uses <code>.ic</code> to initialize the input probe while writing miller decks. |

This variable must be set before the `char_library` command.

`ccsn_miller_init_vin_thresh`

| | |
|----------------------------|---|
| <code><value></code> | Controls how close the inputs are initialized to the starting rail voltage for CCSN miller capacitance characterization. The default value 0.2 initializes input rise to 0.8 Vdd and input fall to 0.2 Vdd. Default: 0.2 |
|----------------------------|---|

This variable must be used before the `char_library` command.

`ccsn_miller_vout_delta_variation`

| | |
|--|--|
| <code><value></code> | Controls the output swing for CCSN miller capacitance characterization. For FinFET technologies, the output swing needs to be large enough to model the complete impact of backward miller effect. Default: 0.5 |
| Note: The default value 0.5 implies $0.5 * (Vdd - gnd)$ output swing. | |

This variable must be used before the `char_library` command.

`ccsn_model_unbuffered_output`

Deprecated. See [Backward Compatibility Variables](#).

`ccsn_no_input_dc_current_mode`

| | |
|--------------------------------|--|
| <code><0 1 2></code> | Controls the DC current characterization of CCSN for the cells that has no inputs. Default: 2 |
| 0 | Do not perform any simulation reduction and post process for no input cell like tie cell. |
| 1 | Post process for tie cell to duplicate the first line across the rest of the DC table. |
| 2 | Only simulate first row of <code>dc_current</code> table and do post process similar to 1 |

This variable must be used before the `char_library` command.

`ccsn_one_sided_tristate`

| | |
|----------------------------|---|
| <code><0 1></code> | Enables checking for one sided tristates. Default: 0 |
|----------------------------|---|

In CCSN modeling, the output is needed to be able to swing rail-rail (from a 1 to a 0) in order to propagate noise pulses. This may not always be possible with tri-state cells. If the output of the CCSN stage can swing to a "Z" state, based on its input switching, then additional side inputs might need simultaneous toggling to pull it to a 1 or a 0. Setting `ccsn_one_sided_tristate` allows Liberate to evaluate combinations of the primary inputs and tie them together until the Z state is resolved to a 1 or a 0. This is a pessimistic solution that physically translates to noise affecting multiple inputs at the same time.

This variable must be used before the `char_library` command.

ccsn_pin_criteria_mode

| | |
|---------|--|
| <0 1> | Adds additional criteria for modeling CCSN groups as pin-based instead of arc-based. Default and recommended: 1 (pin-based) |
| 0 | Favor arc-based modeling. Should only be used for backward-compatibility purposes to match 12.1.5 or prior releases. |
| 1 | Favor pin-based modeling. |

Multiple criteria may be considered when deciding whether to model a stage as pin-based or arc-based. Setting this variable adds additional criteria that pushes a stage to the pin.

This variable must be used before the `char_library` command.

ccsn_pin_high_effort

| | |
|---------|---|
| <0 1> | Selects a CCS effort algorithm. Default and recommended: 1 |
| 0 | Revert to release 2.4p2 and prior behavior. |
| 1 | Create the CCSN stages input pins that are not modeled by any CCSN stages (either arc based or pin based) The default behavior of Liberate was changed in release 2.5 to create these CCSN stages because this behavior results in a more complete CCSN .lib model. |

This variable must be used before the `char_library` command.

ccsn_pin_stage_lshift

| | |
|---------|---|
| <0 1> | Specifies the methodology for modeling CCSN for level-shifters. Default: 1 Recommended: 0 |
| 0 | Model level-shifters as arc-based or pin-based as per the CCSN specification for that CCB. |

| | |
|---|---|
| 1 | Model level-shifters as pin-based to work around LBDB-716 issues. |
|---|---|

The CCSN Characterization guidelines have rules to outline when CCSN stages should be modeled as pin-based and when as arc-based. However, if level-shifters included arc-based CCSN models, older versions of Library Compiler generates "LBDB-716" errors and fail to compile the library. The default setting works around this problem by forcing pin-based models. Library Compiler versions post 12.1 ISR6 does not have this issue,. Therefore, it is recommended to model these as arc-based.

This variable must be used before the `char_library` command.

`ccsn_pin_stage_merge_mode`

`<0 | 1 | 2 | 3 | 4>` Controls merging of ccsn data.

Default and recommended: 4

| | |
|---|--|
| 0 | Enables behavior prior to release 2.3. (Use only for backward compatibility.) |
| 1 | Enables behavior of release 2.3. (Use only for backward compatibility.) |
| 2 | Allows Liberate to also choose inverter outputs whenever possible. |
| 3 | Enables mode 1 and 2, and for output CCSN stages, chooses inverter input whenever possible. |
| 4 | Gives preference to probes directly on inverter outputs. This setting provides the expected functionality when the output is also channel connected to an input. |

It is often desirable to merge CCSN `stage_type` `pull_up` and `pull_down` data to `stage_type` "both". Use this variable to instruct Liberate to prefer `ccsn_stage_both` for input/output probe nodes that use inverter input/outputs.

Set this variable to enable enhanced conditional checks to understand "don't care" conditions. Previously, Liberate considered two groups to be equivalent (which allowed Liberate to merge them together into one group.) Liberate required all CCR (Channel Connected Regions of transistors) inputs to have the same state even when a particular input is a "don't care" for an output. This caused more CCSN stages.

This variable must be used before the `char_library` command.

Example

```
set_var ccsn_pin_stage_merge_mode 0
```

ccsn_pin_unconditional

Deprecated. See [Backward Compatibility Variables](#).

ccsn_pin_voltage_level_attrib

| | |
|----------------------------|--|
| <code><0 1></code> | Writes the Liberty attributes <code>input_signal_level</code> and <code>output_signal_level</code> into CCSN groups. Default and recommended: 1 |
| 0 | Do not output <code>input_signal_level</code> and <code>output_signal_level</code> attributes into the library. |
| 1 | Output <code>input_signal_level</code> and <code>output_signal_level</code> pin attributes into the CCSN (<code>ccsn_first_stage</code> and <code>ccsn_last_stage</code>) groups. This is compatible with the 2013.03 version of LC. |

This variable must be used before the `char_library` command.

ccsn_prefer_min_vt_probe

| | |
|----------------------------|---|
| <code><0 1></code> | Methodology for determining CCB output probe point selection. Default: 0 Recommended: 1 |
| | This variable is used when a CCSN stage (Channel Connected Region) has more than one output that can be selected as the output probe. |

| | |
|---|---|
| 0 | (Worst-case noise model) Select the output probe node that results in the largest propagated noise, even if a full voltage swing is not possible such as when the output is in the middle of an <code>nfet</code> or <code>pfet</code> stack. |
| 1 | (Noise-on-delay model) Liberate selects the output probe for the CCSN stage that connects to both <code>NMOS</code> and <code>PMOS</code> devices whenever possible. This node may not produce the largest propagated noise but it maximizes the observed output voltage swing. This might mean that the probe point is not the one that is most susceptible to noise but may have better results in noise-on-delay analysis. |

This variable must be used before the `char_library` command.

`ccsn_prefer_two_sided_stages`

Deprecated. See [Backward Compatibility Variables](#).

`ccsn_print_is_needed_if_false_attr_value`

| | |
|---------|--|
| <0 1> | Controls the output of the <code>is_needed</code> attribute in CCSN groups in the output library when this attribute has a value of <code>false</code> . This indicates that the CCSN group is not needed. Default: 0 |
| 0 | Do not model CCSN groups when the Liberty attribute <code>is_needed</code> has a value of <code>false</code> . |
| 1 | Write <code>ccsn_first_stage/ccsn_last_stage</code> groups into the output library when the Liberty attribute <code>is_needed</code> has a value of <code>false</code> . This setting may clear up library compilation issues. |

This variable affects cells with pins that do not require CCSN data. Print the following CCSN group with `is_needed` pin attribute.

```
pin (a) {
```

```
ccsn_first_stage () {  
    is_needed :  false;  
}  
}
```

Note: The `ccsn_*` groups with the attribute `is_needed` with value `false` cannot contain miller cap or any such attribute ("when" may be allowed).

This variable must be set before the `char_library` command.

`ccsn_prop_noise_peak_mode`

| | |
|---------|--|
| <0 1> | Used with CCSN characterization of level shifters if Library Compiler issues warnings about incorrect <code>index_1</code> voltage ranges. Default: 0 |
| 0 | Used with CCSN characterization of level shifters. |
| 1 | Control the propagated noise input values |

This variable must be used before the `char_library` command.

`ccsn_prop_retry_duration_incr`

Deprecated. See [Backward Compatibility Variables](#).

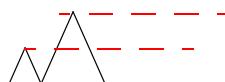
`ccsn_prop_retry_peak_incr`

| | |
|----------------------------|--|
| <code><value></code> | Controls the input PWL generation for propagated noise. This is the percentage of vdd-vss voltage swing (for that partition) used to increment the peak during the next retry attempt. A larger value results in larger peaks being generated for the input, when the initial set of inputs fails to produce the desired output signals. |
|----------------------------|--|

Example:

Given vdd=1, gnd=0, and initial peak=0.1

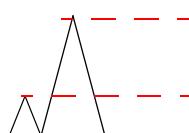
If `ccsn_prop_retry_peak_incr` = 0.2



... and the second would be $0.1 + (1 * 0.2) = 0.3v$

The first peak would be at 0.1

If `ccsn_prop_retry_peak_incr` = 0.8



... and the second would be $0.1 + (1 * 0.8) = 0.9v$

The first peak would be at 0.1

This variable must be used before the `char_library` command.

`ccsn_probe_mode`

Deprecated. See [Backward Compatibility Variables](#).

`ccsn_prune_last_stage`

| | |
|---------|---|
| <0 1> | Set to prefer two-sided stages. Default: 1 |
| 0 | Compatibility to release 2.3 and prior behavior where the last stage is not skipped |
| 1 | Liberate skips <code>ccsn_last_stage</code> generation on some output pins. When enabled, causes Liberate to skip generation of all pin based CCSN stages for output pins that are not destinations of timing() arcs and outputs two sided (stage_type: both) CCSN stages, whenever possible. |

This variable must be used before the `char_library` command.

ccsn_simultaneous_switch_save_vecdata

| | |
|---------|--|
| <0 1> | Enables the generation and saving of <code>vecdata</code> for select pins. The <code>vecdata</code> is required when generating CCSN data for an arc in a library. Default and recommended: 1 |
| 0 | CSN might not be generated for the <code>select</code> pin of a mux when <code>simultaneous_switch_from_cell_when=1</code> . This setting is provided for backward compatibility to Release 12.1 ISR4 and prior. |
| 1 | Generate and save all <code>vecdata</code> for select pin even when <code>simultaneous_switch_from_cell_when=1</code> . |

This variable must be used before the `char_library` command.

ccsn_sort_merge_hidden_mode

| | |
|-------------|---|
| <0 1 2> | Controls pin-based CCSN structural compatibility of CCSN models that are generated. Default and recommended: 0 |
| 0 | Maintains compatibility with the 12.1.2 release. |
| 1 | Maintains compatibility with the 3.2p4_lcs release. |
| 2 | General behavior of setting 1, behavior of setting 0 only for tristate buffers. |

This variable must be used before the `char_library` command.

ccsn_switch_cell_partition_mode

| | |
|---------|---|
| <0 1> | Enables better handling of power gates for DC current characterization. Default: 0 |
|---------|---|

| | |
|---|---|
| 0 | Does not distinguish between power gating MOS and switched rail powered devices for cell level <code>dc_current</code> generation purposes. Iterates across all possible gate connections to obtain the <code>dc_current</code> . |
| 1 | Chooses the gate terminal of the power gating MOS device for the <code>dc_current</code> characterization. It handles the inverter chain appropriately for first and last stage CCSN characterization on the respective input and output pins. You can use enhanced identification of first and last stage Channel Connected Blocks (CCBs) on switch rail powered gates, and thus avoid treating them as a single simple CCB. |

This variable must be set prior to the `char_library` command.

ccsn_tempus_promote_mode

<0 | 1> Enables additional internal node information in CCSN groups as user-defined library attributes.
Default: 0

Note: This variable controls the output of the two user defined attributes in `ccsn_first_stage`. The additional data is used by Tempus.

| | |
|---|--|
| 0 | Does not output the additional attributes. |
| 1 | Enables output of the additional attributes: <code>related_spice_node</code> , <code>load_cap_rise</code> , and <code>load_cap_fall</code> . |

This variable must be set before model creation (see `write_library`).

Example

```
define ( related_spice_node, ccsn_first_stage, string );
define ( load_cap_rise, ccsn_first_stage, float );
define ( load_cap_fall, ccsn_first_stage, float );
...
```

```
ccsn_first_stage () {  
    related_spice_node : "CKN";  
    load_cap_fall : 0.012899;  
    load_cap_rise : 0.012899;
```

ccsn_use_io_ccb_format

<0 | 1> Specifies CCS Noise characterization and modeling methodology.
Default: 0 (Use stage-based format)

Recently, a new format for modeling CCS Noise was introduced to replace the `ccsn_first_stage` and `ccsn_last_stage` formats with the `input_ccb` and `output_ccb` formats. A number of methodology changes for how CCS-N should be characterized were also implemented. This variable specifies which format to use for characterization and modeling. As per the specification, the formats are incompatible, and so the decision must be made prior to characterization.

- | | |
|---|---|
| 0 | Characterize and model as per the stage-based format. |
| 1 | Characterize and model as per the CCB-based format. |

The CCB-based format offers a number of improvements over the stage-based format, some of which were only previously available to customers using CDB/ECSM-N. However, since adopting this new format means updating most of the tool-chain, Cadence recommends that customers allocate sufficient time and resources for the necessary qualifications prior to making any decision on production use.

This variable must be used before the `char_library` command.

ccsn_xfr_ccc_probe_mode

<0 | 1> Ensures that, when modeling `ccsn` first-stage groups, Liberate does not probe transmission gate outputs when the transmission gate input was the arc pin.
Default: 1

| | |
|---|---|
| 0 | Revert to release 3.0 and prior behavior. |
|---|---|

| | |
|---|---|
| 1 | Disables probing of transmission gate outputs when modeling ccsn first-stage groups |
|---|---|

This variable must be used before the `char_library` command.

ccsp_base_curve_points

`<value>` The maximum number of points in a CCSP base curve.
Default: 15

When generating CCSP curve data, Liberate can search for common curve shapes. When found, the common shape can be stored into a lookup table and reused multiple times. This is done to reduce the size of the Liberty file. Liberate uses up to the specified number of points in a shared base curve.

This variable must be used before the `write_library` command.

ccsp_default_group

`<min | max | off>` Enable writing the output of a default CCSP power group.
Default: off

| | |
|-----|---|
| min | Select the <code>dynamic_current</code> which has the minimum <code>internal_power</code> . |
| max | Select the <code>dynamic_current</code> which has the maximum <code>internal_power</code> . |
| max | No default CCSP group is outputted. |

This variable may be used after the `char_library` command.

Example:

```
# Enable a max default ccsp power group
set_var ccsp_default_group max
```

ccsp_leakage_current_abstol

`<value>` Sets an absolute tolerance of when Liberate adjusts CCSP leakage current for Library Compiler compliance. The value `1.0e-10` conforms to the requirements in Library Compiler 2008-09 SP1. The value `0` conforms to the older releases of Library Compiler such as 2007-12 SP1.
Default: `0`

This variable must be used before the `char_library` command.

ccsp_leakage_current_compensation_mode

`<0 | 1 | 2>` Controls the methodology used for compensating `leakage_current` values for Library Compiler compliance.
Default and recommended: `1`

- | | |
|---|--|
| 0 | Adjust the rail with smallest <code>leakage_current</code> . |
| 1 | Adjust a 0v rail. If multiple 0v rails are attached to a cell, then adjust the <code>primary_ground</code> with the <i>most</i> <code>leakage_current</code> . |

Note: Some adjustments may be the result of truncation errors due to unit scale. For 28nm libraries and smaller geometries, we recommend using the following units prior to changing this variable:
`set_units -timing 1ps -capacitance 1ff -current 1ua -leakage_power 1pw`
This `set_units` change usually has the added benefit of decreased file size with the same (or slightly improved) accuracy.

- | | |
|---|--|
| 2 | Reduces the largest magnitude leakage current values enough to produce a zero sum, but not farther than 0. For header and footer cells, it removes the virtual rail current from the largest supply, and also from the gate leakage, if necessary. In all cases, retains the correct sign on all currents. |
|---|--|

This variable must be used before the `write_library` command.

ccsp_min_pts

`<value>` Sets the minimum number of desired CCSP samples. This variable only has an effect when the variable `ccsp_segmentation_effort` is enabled.
Default: 6

This variable must be used before the `char_library` command..

ccsp_pin_direction_post_default

`<0 | 1>` Controls finding unateness of CCSP dynamic groups.
Default: 0

1 When determining the pin direction (risefall) of the CCSP power group, Liberate examines the default timing group when no state dependent timing group matches the "when" condition in the ccsp power group.

This variable may be used before the `char_library` command.

ccsp_prune_factor

`<value>` Set the CCSP tail decay factor.
Default: 2

Use this variable to control the decay to be applied after the `ccsp_prune_second_tol`. If this variable is set to 2, the ccsp waveform is pruned at `ccsp_prune_factor * (ccsp_prune_start_tol - ccsp_prune_second_tol)` after `ccsp_prune_second_tol`. See [ccsp_prune_start_tol](#).

This variable must be used before the `char_library` command.

ccsp_prune_second_tol

<ratio> Set the second CCSP tail prune tolerance.
Default: 0.03 (3%)

This variable must be used before the `char_library` command.

ccsp_prune_start_tol

<ratio> Set the first CCSP tail prune tolerance.
Default: 0.0 (0%. Pruning disabled)

If set to a value greater than 0, Liberate terminates the CCSP waveform early. Liberate follows the raw current waveform backwards from the end to find the point where the waveform is the `ccsp_prune_start_tol` of the waveform peak. Liberate then follows the current forwards to the point where it reaches `ccsp_prune_second_tol` (default 3%) of the peak. The difference between these two points is the elapsed time ("x"). Assuming exponential decay, Liberate terminates the waveform after a time $2x$ (see [ccsp_prune_factor](#)) from the `ccsp_prune_second_tol` point.

When this variable is set to 0 (default), no waveform trimming is done. Note that `ccsp_prune_second_tol` must be less than `ccsp_prune_start_tol`. If this check fails, Liberate automatically adjusts `ccsp_prune_second_tol` to $0.5 * ccsp_prune_start_tol$.

Setting this variable to the desired value (that is 0.10, for 10% of peak) would act as described above to prune the tail of the CCSP waveform. The pruning is performed on the raw current waveforms and is done before any other CCSP waveform shaping methods. This waveform pruning has been useful to trim long waveform tails that have caused problems in some power tools.

Liberate also checks for the starting leakage value in calculating the percentages. For example, 10% prune time is reached at time t when the $i(t) - i(0) < 0.1 * (peak - i(0))$.

This variable must be used before the `char_library` command.

ccsp_quantization_num_steps

<value> Set the number of CCSP quantization steps used to preprocess the supply current waveform to smooth out kinks.
Default: 100

This variable must be used before the `char_library` command.

ccsp_rel_tol

<ratio> Set the CCSP relative tolerance of the area segmentation current as compared against the original area of the supply current.
Default: 0.02

This variable must be used before the `char_library` command.

ccsp_related_pin_mode

<0 | 1 | 2 | 3> Controls modeling of dynamic current groups for CCSP. In CCSP, `dynamic_current` groups contain the power data for switching events.
Default and Recommended: 2

| | |
|---|--|
| 0 | Writes out separate <code>dynamic_current</code> groups for each output pin. Backward compatible to 14.1 ISR2 and prior releases. (Deprecated. Should be used only for regression purposes.) |
| 1 | Enables combining the dynamic current groups under each output pin into one common group whenever possible. (Deprecated. Should be used only for regression purposes.) |
| 2 | Combines <code>dynamic_current</code> groups into a common group where possible. If multiple input pins switch as in the case of CLK/CLKB or a one-hot mux, then only the group on the first alphabetical <code>related_pin</code> is modeled. |

| | |
|---|--|
| 3 | Similar to 2, but the related_inputs and input_switching_condition attributes of the dynamic_current group accurately reflects all switching activity. This is the most accurate CCSP format, but is currently not supported in all downstream tools. Note: Verify with the tool vendor before using. |
|---|--|

Liberate writes out separate dynamic current groups for each output pin. This can give wrong power reports when using CCSP format data.

This variable can be used before the `write_library` command.

ccsp_segmentation_effort

| | |
|------------------------------------|--|
| <code><0 1 2 3></code> | Enables a more accurate CCSP segmentation algorithm. Default and recommended: 3 |
| 0 | Disables segmentation for supply current waveforms. |
| 1 | Liberate segments the supply current waveform and minimizes the number of points |
| 2 | This is currently just for the engineering use. The recommendation is not to use this setting. |
| 3 | Uses more points when segmenting supply current waveforms. Increasing the number of points provides the best balance of improved accuracy and file size, with no discernible impact on runtime |

This variable must be used before the `char_library` command.

ccsp_table_reduction

| | |
|--------------------------------|--|
| <code><0 1 2></code> | Enables reduced CCSP table data. Default: 1 |
|--------------------------------|--|

| | |
|---|--------------------------------|
| 0 | Disables table size reduction. |
| 1 | Use a 3x3 table for CCSP data. |
| 2 | Enables a 2x2 table |

The Liberty specification for CCSP allows for the table size to be reduced. Power results are usually not significantly impacted by using a smaller data table size. A smaller table size will result in a smaller file size.

This variable must be used before the `char_library` command.

ccsp_tail_tol

`<value>` Set the percentage of area considered as the waveform "tail".
The tails are discarded during the preprocessing of CCSP waveforms.
Default: 0.05

This variable must be used before the `char_library` command.

cell_port_case

`<upper | lower | preserve>`

Provides control for how all cell pins are output.
Default: `preserve`

| | |
|-----------------------|--|
| <code>upper</code> | Convert cell pin names to all upper case in the output ldb and library. |
| <code>lower</code> | Convert cell pin names to all lower case in the output ldb and library. |
| <code>preserve</code> | Keeps the current behavior of maintaining the case from the input netlist. |

Important

Do not use `cell_port_case` with Spectre. Spectre is case-sensitive and could have multiple port names that differ only in case.

This variable must be used before the `read_spice` command.

char_mos_term_cap

| | |
|-------------|---|
| <0 1 2> | Enables different algorithms for estimating the transistor device pin capacitance. The capacitance is used at various places in the Liberate algorithm to provide equivalent capacitance when the effective loading of the capacitance is required. Default and recommended: 2 |
| 0 | Enables the biased method. |
| 1 | Use this value to revert to the prior behavior before Liberate 13.1 ISR3. |
| 2 | Enables the integration method, which is used automatically if extsim_exclusive=1. |

This variable must be specified before the `char_library` command.

cleanup_tmpdir

| | |
|---------|--|
| <0 1> | Allows Liberate to delete temporary data directories. When using external simulators, Liberate creates temporary directories and store data files in them (example: altos.<pid>.<index>) Default: 1 |
| 0 | The temporary directories are not removed. Note: The temporary directories are not removed when Liberate exits due to a simulation error or when terminated by the host system |
| 1 | Liberate attempts to remove temporary directories when it exits |

This variable must be used before the `char_library` command.

combinational_out_to_out_arc

| | |
|---------------------------------|---|
| <all all_timing buf none> | Controls the stage-types that Inside View considers when determining output-to-output arcs. Default: all_timing |
| all | All feasible output to output arcs are generated, regardless of circuit topology. |
| all_timing | Similar to all, but only timing arcs are generated. This avoids double-counting power arcs. |
| buf | Limits checking for output-to-output arc to single input gates between the two outputs, such as a buffer or inverter. |
| none | Output-to-output arc is disabled, except for user-defined ones using <code>define_arc</code> . |

This variable must be used before the `char_library` command.

combinational_risefall

| | |
|----------------------------|--|
| <code><0 1></code> | Enables characterization of combinational_rise and combinational_fall timing type arcs for preset and clear pins. These arcs occur when both the preset and clear pins are active on a sequential cell and one of them turns off, that is, changes from its active state to its inactive state. See the <code>merge_related_preset_clear</code> variable to specify how to model these combinational arcs. Default: 1 (enabled) |
| 0 | Instructs the <i>Inside View</i> to characterize only the active arcs on asynchronous pins in sequential cells. |
| 1 | Characterizes the asynchronous arcs for the active edge of the asynchronous pins and the combinational arcs on sequential cells. |

Note: If combinational arcs for asynchronous pins are specified using the `define_arc` command, these arcs are characterized without considering the value of this variable.

This variable must be specified before the `char_library` command.

Example

```
#Disable combinational_rise or combinational_fall arcs for preset and clear pins
set_var combinational_risefall 0
```

conditional_arc

| | |
|----------------------------|--|
| <code><0 1></code> | Control conditional arcs. All states are characterized using a proprietary algorithm which results in a faster runtime but only one state being stored in the ldb. This reduces the size of the ldb and also results in all output libraries being generated with no state dependency. Default: 1 (output conditional arcs) |
| 0 | Turn off the storing of conditional arcs in the library database. When set to 0, this variable overrides <code>conditional_constraint</code> . |
| 1 | Outputs conditional arcs with full state dependency. |

This variable must be used before the `char_library` command.

conditional_cap_hidden_pin

`<0 | 1>`

Enables to characterize and model conditional (state-dependent) CCS/ECSM pin capacitance. This type of pin capacitance is measured during hidden power arc characterization. To get state-dependent pin capacitance, state-dependent hidden_power arcs must be characterized.

Default: 1

0

Outputs unconditional (state-independent) CCS/ECSM `receiver_cap` for each input pin if the capacitance data was characterized and stored. The variable `ccs_cap_hidden_pin` or `ecsm_cap_hidden_pin` determines if the capacitance is characterized and saved for the hidden arcs.

1

Characterizes and saves all conditional (state-dependent) pin-based receiver capacitance associated with each characterized hidden power arc.

This variable must be specified before the `char_library` command.

conditional_cap_hidden_pin_mode

`<0 | 1 | 2 | 3>`

Keeps or removes the output pin(s) from the `when` of the receiver capacitance groups in the output library.

Default: 0

0

Removes the output pin from the `when` of the CCS and ECSM receiver capacitance.

1

Removes the output pin from the `when` of the CCS and ECSM receiver capacitance with better structural matching across PVT.

Note: This is an improvement over the setting of 0.

- | | |
|---|--|
| 2 | Allows the output pins in the when of the CCS and ECSM receiver capacitance. |
| 3 | Allows the output pins in when condition with better structural matching across PVT. |
- Note:** This is an improvement over the setting of 2.

When a setting of 0 or 1 is used to remove the output pins, the variable `conditional_rcvr_cap_select_criteria` can be used to select one of the overlapping capacitance groups to model in the output library.

Note: The settings of 0 and 1 can create libraries with receiver capacitance groups that have the same when condition. For example, if the hidden power states `D&Q` and `D&!Q` are characterized and the output pin is removed, there are two receiver capacitance groups both with a when of `D`. These two groups may have different pin capacitance values. It is up to the timer to choose which one to use.

This variable must be specified after the `char_library` command.

conditional_cap_hidden_pin_thresh

`<value>` Controls the modeling of conditional (state-dependent) receiver capacitance under the input pin. When the number of input pin capacitance with "when" conditions (states) exceeds the specified value, Liberate writes into the `.lib` file two CCS/ECSM `receiver_capacitance` groups, where one is the minimum capacitance and the other is the maximum capacitance. Also, each group will have a `char_when` attribute (but no when condition). This is done to minimize the size of the output library.
Default: 1024

The minimum or maximum capacitance table is determined by finding the capacitance value from all conditional receiver capacitance tables and selecting the table that contains the capacitance. If the minimum and maximum capacitance groups are equal, then only one capacitance group is output.

This variable must be set before the `write_library` command.

conditional_constraint

| | |
|-------------|--|
| <0 1 2> | Controls conditional constraint characterization for timing constraints. Default: 1 (on) |
| 0 | Timing constraints are characterized under worst-case conditions. |
| 1 | Characterize and model each unique when condition. Note: Using this value increases the run time. |
| 2 | The worst-case constraint condition is characterized similar to 0, but a combined when condition is generated in the library. If conditional_arc is set to 0, conditional_constraint will affect characterization, but only one state-independent timing group will be stored. |

This variable is used to enable characterization of conditional (state-dependent) timing constraints including setup, hold, recovery, removal, and minimum pulse width.

This variable must be specified before the `char_library` command.

Example

```
# Enable conditional arcs for timing constraints
set_var conditional_constraint 1
```

conditional_expression

| | |
|--------------------|--|
| <merge separate> | Controls whether conditional groups are merged when they contain sub-expressions that are OR'd together. Default: merge |
| merge | Merges all conditions that result in identical values by ORing each of the sub-expressions. |

`separate` Splits conditions containing OR sub-expressions into separate groups. The default timing and default power groups are also created if enabled by the `default_timing` and `default_power` control variables.

This variable must be used before the `char_library` command.

Example

```
# Split groups with OR'd conditions
set_var conditional_expression separate
```

conditional_hidden_power

| | |
|---------|--|
| <0 1> | Controls state dependence of hidden power. Default: 1 (output state-dependent hidden power) |
| 0 | Outputs only unconditional hidden power. |
| 1 | Outputs only state-dependent hidden power. |

This variable must be used before the `char_library` command.

Example

```
# Turn off conditional states for hidden power
set_var conditional_hidden_power 0
```

conditional_immunity

| | |
|---------|---|
| <0 1> | Controls characterization of conditional Noise Immunity Curve (NIC) data. Default: 0 |
| 0 | Outputs the worst case NIC. |

1 Request the output of unique NICs for each when condition

Note: Using this value increases the run time.

This variable must be used before the `char_library` command.

Example

```
# Turn on conditional states for noise immunity
set_var conditional_immunity 1
```

conditional_include_constant

| | |
|---------|--|
| <0 1> | Controls whether constant nets are included in conditional when statements. Default: 1 (on) |
| 0 | Constant nets are not included in conditional when statements. |
| 1 | Constant nets are included in conditional when statements. |

This variable must be used before the `char_library` command.

Example:

```
# Disable constants in 'when' conditions
set_var conditional_include_constant 0
```

conditional_include_output

| | |
|---------|--|
| <0 1> | Controls whether output pins are included in the when condition of hidden power and leakage constructs. Default: 1 (on) |
| 0 | Output pins are not included in conditional when statements of hidden power and leakage groups. |

| | |
|---|---|
| 1 | Output pins are included in conditional when statements of hidden power and leakage groups. |
|---|---|

This variable must be used before the `char_library` command.

Example

```
# no output pins in hidden power & leakage 'when' conditions
set_var conditional_include_output 0
```

conditional_leakage

| | |
|---------|---|
| <0 1> | Controls state dependence of leakage. Default: 1 |
| 0 | Disable state dependent leakage. |
| 1 | Output state dependent leakage. |

This variable must be used before the `char_library` command.

conditional_min_period

| | |
|---------|---|
| <0 1> | Enables modeling of conditional (state dependent) <code>min_period</code> . Default: 0 |
| 0 | A single state independent <code>min_period</code> group is written to the output library. If using <code>min_period_when</code> , then a single <code>min_period</code> group is modeled with the <code>when</code> set to the <code>min_period_when</code> . |
| 1 | Write <code>min_period</code> groups to the output library corresponding to all characterized mpw states (see <code>conditional_mpw</code>). If using <code>min_period_when</code> then a <code>min_period</code> group is modeled for each mpw group where the <code>when</code> functionally overlaps with the <code>min_period_when</code> value. |

This variable must be set before the `write_library` command.

Example

```
# Disable conditional states for leakage
set_var conditional_leakage 0
```

conditional_mpw

| | |
|--------------|---|
| <-1 0 1> | Controls the output of conditional mpw timing constraints. Default: -1 |
| -1 | Follows the <code>conditional_constraint</code> setting. |
| 0 | Turns off conditional mpw, regardless of <code>conditional_constraint</code> . The mpw is modeled without any state and the MPW low/high is the worst case of all of the MPW values characterized as follows: <code>min_pulse_width_low = max (all mpwLo)</code> <code>min_pulse_width_high = max (all mpwHi)</code> |
| 1 | Turns on conditional (state dependent) mpw, regardless of <code>conditional_constraint</code> . The mpw is characterized and modeled with state dependence. For a given state and the MPW low/high is the worst case of all of the MPW values characterized for the specific state and is determined as follows: <code>min_pulse_width_low = max (all mpwLo)</code> <code>min_pulse_width_high = max (all mpwHi)</code> |

This variable must be set to 1 prior to `char_library` to enable the characterization of state dependent mpw data. It may be reset to 0 prior to `write_library` to disable the modeling of state dependent mpw.

conditional_rcvr_cap_select_criteria

`<none | first | mid | last>`

Specifies the position in the receiver capacitance table to be used to decide which table to select to write to the output library when there are multiple tables with the same identical "when" condition.

Default: none

Sometimes Liberate outputs multiple receiver capacitance tables for the same "when" condition. This usually occurs because the output state is removed from the when resulting in identical states. For example, the states $D \& Q$ and $D \& !Q$ are characterized for clock falling hidden power. If the output pin Q is removed from the when, then there are two capacitance groups with the same when of "D".

Note: When the duplication occurs because conditional_cap_hidden_pin_thresh is exceeded, then duplicate receiver capacitance tables are expected since they correspond to the minimum and maximum input receiver capacitance. The timer expects to see both of these capacitance groups and the variable should be set to "none".

| | |
|-------|---|
| none | No selection is performed. All receiver capacitance tables are written to the output library. |
| first | Selects the table with the largest value in the first index position. |
| mid | Selects the table with the largest value in the middle of the table. For example, if there are eight points in index_1, compare 4th positional value. |
| last | Selects the table with the largest value in the last index position. For example, if there are eight points in index_1, compare 8th positional value. |

Note: This selection is disabled when conditional_cap_hidden_pin_mode is set to 1 and the number of receiver capacitance tables exceeds conditional_cap_hidden_pin_thresh. This is because in this case, the timer is expecting to see both the minimum and the maximum capacitance tables.

This variable must be used after the `char_library` command.

constraint_async_probe_internal

`<0 | 1>`

Specifies whether to probe internal nodes on non-sequential constraints. This variable only applies to `non_seq_setup` and `non_seq_hold` constraints. (See [constraint_probe_internal](#)).
Default: 1 (permits probing internal nodes)

Liberate automatically finds probe nodes for measuring constraints.

0 Prevents from choosing an internal node as the probe node for non-sequential constraint acquisition.

1 Permits to probe internal nodes.

This variable must be specified before the `char_library` command.

constraint_bisection_mode

`<0 | 1 | 2>`

Controls the methodology behind bisection searches.
Default and recommended: 2 (Updated Brent's Method)

Note: Prior to version 3.2p2, only setting 1 was available.

0 Use pure bisection. Result will be correct.

1 Use Brent's Method. Could be artificially pessimistic when compared to pure bisection in some corner cases.

Note: This value should only be used for backward compatibility.

2 Use improved Brent's Method. Result should be within 1 constraint tolerance of pure bisection, but use fewer iterations.

In extremely rare cases, setting 1 may result in additional iterations and/or pessimism in the constraint when delay degradation is the success criteria due to numerical noise in the delay measurements. This effect is usually seen if one version of Liberate generates one or two

constraint values out of an entire library that are more pessimistic than the constraint_search_time_abstol window should allow.

The results for settings 0 and 2 should always be within the constraint_search_time_abstol window. The benefit to using 2 is slightly better performance.

To verify a constraint, use extsim_save_verify=1 to generate a sweep deck, then run in a standalone simulation. Note the nominal delay value and calculate the delay+degrade value. Look for the constraint that results in a measured delay that is less than the delay+degrade value.

If there are multiple places in the sweep results that cross the delay+degrade threshold, than this cell exhibits "multi-bump" or multiple solutions to the delay+degrade success criteria. The criteria should be adjusted to result in only one solution for consistency.

This variable must be used before the char_library command.

constraint_check_final_state

<0 | 1 | 2 | 3>

Enforces additional criteria on constraint measurements. It enables a check on the final state of transitioning nodes in the cell. The final state of internal probe nodes and feedback wires is checked to have fully transitioned. The final voltage on each of the internal probe and feedback wires must agree with their intended final logic level. Checking the final state of these additional nodes provides an assurance that the probe node is stable and will not revert to its original state. The additional node checks normally only affect constraints that use delay degradation as their measurement criteria.

Default and recommended: 3

0

Does not perform extra checks to determine the final state of internal nodes.

1

The circuit simulation continues until each of the internal probe and feedback wires transition to within 5% of their expected final voltage. This setting ensures that these wires reach 95% of their voltage swing.

- | | |
|---|--|
| 2 | The simulation does not stop early. This setting ensures that the probe node transition has fully settled before measuring the degradation by providing extra settling time. |
| 3 | Same as option 2, but also check for stability of internal feedback wire at the end of the simulation. If any internal feedback node voltage is below <code>constraint_check_final_state_threshold</code> , the iteration is considered a failing bound. Also, affects constraints with glitch-peak criteria, as glitches larger than the <code>constraint_check_final_state_threshold</code> are considered as failing measurements. |

Note: A setting of 2 or 3 may increase a small number of constraint values and will slightly increase characterization time.

The additional node checks also affect constraints that use glitch-peak criteria where glitches larger than the `constraint_check_final_state_threshold` are considered as failing measurements.

This variable must be used before the `char_library` command.

`constraint_check_final_state_threshold`

`<value>` Sets the final state threshold to be used when `constraint_check_final_state` is enabled. The threshold value is a ratio between 0 and 1.
Default: 0.9 (use `delay_out_fall` / `rise`)

When allowed to default, the threshold uses the `delay_out_rise` threshold for a rising transition and the `delay_out_fall` threshold for a falling transition.

This variable must be used before the `char_library` command.

constraint_check_rebound

| | |
|---------|---|
| <0 1> | Checks if the output rebounds. Default and recommended: 0 |
| 0 | Does not check for probe node reversals that lead to multiple measurements. |
| 1 | Reversals on the probe node transition greater than 1pS mark the constraint simulation as failing. If a probe node transition reversal crosses the delay measurement threshold (see <u>delay_out_fall</u> and <u>delay_out_rise</u>) multiple times, then measure the time delta between the first and last threshold crossing. If the delay delta between the two crossings exceeds 1pS, then the bisection bound is treated as a failing bound. |

When measuring constraints using delay criteria as the metric, Liberate expects that the probe node transitions from `gnd` to `vdd` monotonically. It does not check if the probe node reverses direction. Sometimes the probe node transition reverses course momentarily.

Note: This variable is not recommended because jitter in the output can exceed the 1pS criteria, leading to unpredictable constraint results. If checks on the output waveform are desired, it is recommended to use constraint_check_final_state and constraint_slew_degrade instead.

This variable must be used before the `char_library` command.

constraint_clock_gater

| | |
|---------|--|
| <0 1> | Controls the use of special clock gater circuit constraints. Default: 1 (enable special checks) |
| 0 | Disables special handling for clock gater constraints. |
| 1 | Enables special handling for clock gater constraints. |

Clock-gating circuits require special techniques when measuring constraints such as setup and hold. These techniques are documented in the Timing Constraints section of Chapter 7.

This variable must be used before the `char_library` command.

constraint_combinational

| | |
|-------------|--|
| <0 1 2> | Disables combinational measurement of constraints. . Default: 0 |
| 0 | Use a bisection search method when characterizing constraints. |
| 1 | Attempt to use a bisection search, but when that fails, it uses a simple delay-degradation based sweep in an attempt to characterize the constraint. |
| 2 | Note: This value should be used to characterize a constraint for a combinational cell such as a <code>nand</code> gate. Enables a path-difference, formula-based constraint estimation if the bisection search fails to characterize the constraint. |

Note: Setting this variable to 1 or 2 can significantly slow down the Liberate run.

This variable must be used before the `char_library` command.

constraint_combinational_step_limit

| | |
|---------|--|
| <value> | Specify the number of sweep steps. Default: 1000 |
| | When <code>constraint_combinational</code> is enabled (set to 1 or 2), this variable determines how many sweep steps the constraint characterization can take. |

This variable must be used before the `char_library` command.

constraint_combinational_step_size

<value> Specifies the desired combinational constraint step size (in MKS units) to be used for linear constraint searches (non-bisection). Also, see [constraint_combinational](#).
Default: 10e-12 seconds

This variable must be used before the `char_library` command.

constraint_delay_degrade

<value> Specifies the percentage of delay degradation permitted in the clock-to-constraint-output-pin delay (flip-flop) or the data-to-constraint-output-pin delay (latch) before an arriving signal is deemed to fail a timing constraint (setup, hold, recovery, removal).
Default: 0.1 (10%)

The `set_constraint_criteria` command can also be used along with this variable. If both this variable and the `set_constraint_criteria` are used, the last one executed sets the value to be used by Liberate.

This variable must be used before the `char_library` command.

constraint_delay_degrade_abstol

<delay> Specifies the minimum delay degradation value (in seconds) permitted in the clock-to-constraint-output-pin delay (flip-flop) or the data-to-constraint-output-pin delay (latch). The maximum of the `constraint_delay_degrade` percentage of the clock-to-output-delay or data-to-output-delay and the `constraint_delay_degrade_abstol` is used as the delay degradation criteria.
Default: 2e-12 (2ps)

Note: For 28nm and below, or for high performance designs the recommended value is 2e-12. However, for 16nm FinFet and below, the recommended value is 1e-12.

The `set_constraint_criteria` command can also be used to set this variable. If both this variable and the `set_constraint_criteria` are used, the last one executed sets the value to be used by Liberate.

This variable must be used before the `char_library` command.

constraint_delay_degrade_abstol_max

<delay> Specifies the maximum delay degradation value (in seconds) permitted in the clock-to-constraint-output-pin delay (flip-flop) or the data-to-constraint-output-pin delay (latch). The minimum of the `constraint_delay_degrade` percentage of the clock-to-output-delay or data-to-output-delay and the `constraint_delay_degrade_abstol_max` is used as the delay degradation criteria. Both `constraint_delay_degrade_abstol` and `constraint_delay_degrade_abstol_max` can be used simultaneously to set both an upper and a lower bound to the delay degradation.
Default: -1 (not used)

This variable must be used before the `char_library` command.

Example

```
# use a 10% delay degradation with min=10ps and max=60ps
set_var constraint_delay_degrade          0.10
set_var constraint_delay_degrade_abstol    10e-12
set_var constraint_delay_degrade_abstol_max 60e-12
```

constraint_delay_degrade_minimize_dtoq

`<0 | 1 | 2 | 3>` Sets to use the minimize minimum `<setup | recovery> + clk-to-Q` delay as the setup and recovery criteria instead of the `constraint_delay_degrade` criteria. Note that `D` does not propagate to `Q`, but rather, `minimize D-to-Q` is shorthand for minimizing the sum of `D->clk` and `clk->Q`.
Default: 0 (Do not minimize `D` to `Q` delay)

Note: When this variable is set to a value other than 0 or 1, the `clk` to `Q` delay measurement are simulated with both `D` and `clk` switching with the time offset equal to the setup time derived previously.

- | | |
|---|--|
| 0 | Use the <code>constraint_delay_degrade</code> criteria when measuring setup and recovery. |
| 1 | The setup measurement threshold for degradation cases is to minimize the <code>D->clk</code> (setup time) + <code>clk->Q</code> delay. Instead of searching for a fixed delay degradation percentage (see <code>constraint_delay_degrade</code>), This method searches for the minimum of <code>D</code> to <code>clk</code> plus <code>clk</code> to <code>Q</code> delays; in other words, minimize the <code>D</code> to <code>Q</code> delay. |
| 2 | In addition to the setup measurement method used when set to a 1, the clock to output delay, transition, ECSM, and CCS data is all simulated based on the constrained data switching. After the setup simulations are done, delay simulations are performed and the data is stored in the ldb under <code>rise_constrained_timing</code> and <code>fall_constrained_timing</code> groups. |
| 3 | Same as 2, but employs additional calculations to minimize the maximum of <code>setup0</code> or <code>setup1</code> (setup to latch a 0 or a 1), plus the maximum for a <code>launch0</code> or <code>launch1</code> (output transitioning to a 0 or to a 1). |

This variable must be used before the `char_library` command.

constraint_delay_degrade_minimize_dtoq_clock_only

| | |
|---------|--|
| <0 1> | Controls whether the constraint_delay_degrade_minimize_dtoq algorithm is applied only to clocked constraints (where related_pin is a clock) or to all constraints. Default: 0 (apply to all constraint arcs). |
| 0 | When the constraint_delay_degrade_minimize_dtoq algorithm is enabled, apply the minimize dtoq algorithm to all constraints. |
| 1 | Apply the minimize dtoq algorithm only to clocked constraints (the related pin is a clock). |

This variable must be specified before the `char_library` command.

constraint_delay_degrade_minimize_dtoq_mode

| | |
|---------------------------------|---|
| <0 1 2 3 4 5 6 7> | Avoids clock to output delay simulations where data switches too early with respect to clock. Default: 0 |
| 0 | Use whatever setup time is available. |
| 1 | Constraint delay to probe only. |
| 2 | Constraint delay for worst setup data only. |
| 3 | Constraint delay for probe and worst setup data only. |
| 4 | Constraint delay for non-scan setup only. |
| 5 | Constraint delay for probe and non-scan setup only. |
| 6 | Constraint delay for worst setup and non-scan setup only. |
| 7 | Constraint delay for probe, worst setup, and non-scan setup data. |

This variable has an effect only when constraint_delay_degrade_minimize_dtoq is enabled. For example, a DFF (Flip-Flop) has a `ck - q` arc and assume there are 2 inputs, `D` and `TE` that have a setup constraint with respect to `CK`. When **constraint_delay_degrade_minimize_dtoq** is enabled, the `dtoq` delay must be evaluated for setup times of both `D` and `TE`.

Incase there are four DFF flops (4 bits with `D0-D3` and `Q0-Q3` and `TE`). Here the problem is that there really are four `TE` setup times, one for each probe node of `Q0-Q3`. The worst case `TE - CK` setup time can come from any probe. In a verbose template for multi-bit cells, it is possible that the setup time only gets measured using one probe node such as `Q0`. There can be four `CK-Qn` delays. A problem can occur when the setup time that matches to the specific output pin for each `Qn` is not measured. A special algorithm is needed to be able to link up specific setup times to specific `CK - Qn` arcs. For the minimize `dtoq` algorithm to work correctly, Liberate needs to be certain that the input `-CK` setup is measured for all four probes or that the worst case probe is used.

This variable must be specified before the `char_library` command.

constraint_delay_degrade_minimize_dtoq_tol

`<value>` Specifies a relative tolerance (floating point number) to apply to the minimum `dtoq` delay so that setup and recovery times can be reduced, while increasing clock to `q` delay.
Default: 0 (Do not apply the tolerance)

The total `dtoq` delay is:

```
(1+constraint_delay_degrade_minimize_dtoq_tol) *  
min_dtoq.
```

This variable must be used before the `char_library` command.

constraint_delay_min_check

`<min_time>` Minimum valid reference delay for constraint acquisition.
Default: -1 seconds (disables the check)

The total `dtoq` delay is:

```
(1+constraint_delay_degrade_minimize_dtoq_tol) *  
min_dtoq.
```

During constraint acquisition, Liberate finds the nominal CK → Q delay. This reference delay is used as a base for all delay push-out calculations. Due to some circuit design techniques or artifacts of measuring delays at non-threshold voltages (for example, 50%), the reference delay may be close to 0 or even negative.

Liberate checks the reference delay against the value of `constraint_delay_min_check`, and report an informational message if the reference delay is smaller.

This variable must be used before the `char_library` command.

constraint_dependent_nominal

| | |
|----------------------------|---|
| <code><0 1></code> | Determines how to compute the reference input-to-probe delay for a dependent constraint. Default: 0 |
| 0 | Compute the dependent nominal delay using the margined independent constraint value and maximum dependent constraint value. |
| 1 | Compute the dependent nominal delay using maximum independent and dependent constraint values. This matches the nominal delay used for the independent constraint |

This variable must be used before the `char_library` command.

constraint_dependent_recrem

| | |
|--------------------------------|--|
| <code><0 1 2></code> | Provides separate control over dependent characterization for recovery and removal timing constraints. Default: 0 (standard characterization) |
| 0 | For detailed information, see <u>constraint_dependent_setuphold</u> . Standard characterization of timing constraints. |
| 1 | Enable re-characterization of recovery timing constraints. |
| 2 | Enable re-characterization of removal timing constraints. |

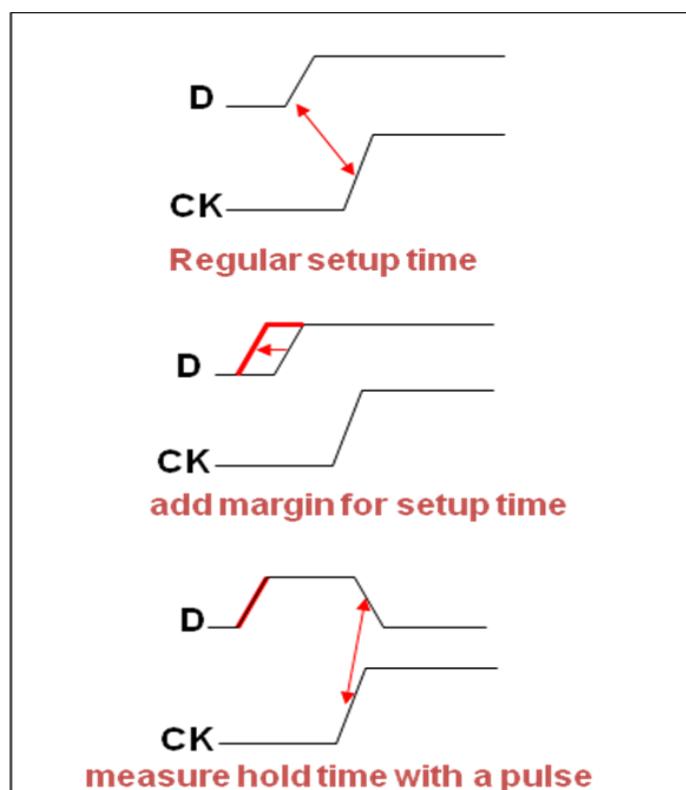
This variable must be used before the `char_library` command.

constraint_dependent_setuphold

| | |
|-------------|--|
| <0 1 2> | Enables dependent setup and hold characterization. Default: 0 (standard characterization) |
| | For detailed information, see constraint_dependent_setuphold . |
| 0 | Standard standard setup and hold characterization. |
| 1 | Enable re-characterization of setup. |
| 2 | Enable re-characterization of hold. |

To re-characterize for dependent setup and hold, an ldb must be loaded using `read_1db` that includes standard setup and hold data.

An example of value 2 (dependent hold time) is shown in the figure below.



This variable must be used before the `char_library` command.

constraint_dependent_setuphold_input_threshold

<value> Controls the minimum peak threshold of the input triangular waveform. The acceptable value is a ratio of the supply voltage and is between 0 and 1.

Default: -1 (use the value of the control variables `delay_inp_rise` and `delay_inp_fall` depending on the input transition direction)

When dependent setup/hold is enabled the data input is a pulse. The input pulse can narrow until it is reduced to a triangular waveform. The peak of this input triangular waveform can drop away from the supply voltage.

This variable must be used before the `char_library` command.

Example

```
# Set the dependent setuphold input threshold
set_var constraint_dependent_setuphold_input_threshold 0.7
```

constraint_dependent_setuphold_margin

<value> Adds a setup or hold margin to the hold (setup) time applied when re-characterizing for dependent setuphold constraints. Default: 0 (seconds)

This variable must be used before the `char_library` command.

constraint_dependent_setuphold_margin_ratio

<ratio> Adds a setup or hold margin to the hold (setup) time applied when re-characterizing for dependent setuphold constraints. The total margin is determined using the formula specified below.

$$<\text{dependent_margin}> + <\text{dependent_ratio}> * (\text{index_1} + \text{index_2})$$

Default: 0 (standard characterization)

This variable must be used before the `char_library` command.

constraint_dependent_setuphold_pessimism

<0 | 1> Ensures pessimism over independent setup or hold values during dependent setuphold characterization
Default: 0 (seconds)

| | |
|---|---|
| 0 | Always use dependent characterized constraint value. |
| 1 | Report a more pessimistic value between the dependent and independent characterized constraint values |

This variable must be used before the `char_library` command.

Example

```
# Set the output pin name for constraints
set_var constraint_dependent_setuphold_pessimism 1
```

constraint_failed_value

<string> Inserts the value of this variable into the LDB and resulting library for failed constraint data when `mark_failed_data` is not enabled.
Default: 1 (1e9 in Liberate LDB units of nS)

| | |
|-----------------------|---|
| -1 | Reverts to the 2.5 (and prior) behavior where, for example, if the constraint arc search simulation failed with a “too close to search bounds” error, the resulting LDB and library contained the failing bound of the constraint search. In rare cases, this could be a large negative value, which resulted in an optimistic constraint value in the library. |
| <i><string></i> | Specifies a value to be inserted into the LDB and resulting library when a constraint characterization fails (as when the result is too close to search bounds). The units of this value is in LDB time units of 1e-9S. |

If the string set by `constraint_failed_value` is like a number (e.g. "5.5e+09"), then the replacement for the failed constraint value is the result of dividing the number by the `time_unit` (for example, 1ns), that is, a scaled number (example, 5.5e+18) overwrites the failed values in the output library.

Examples

Here are some examples for failed constraint replacement.

Example1

```
set_var constraint_failed_value "5.5e+09" with a library time unit of 1ns:  
fall_constraint (constraint_template_3x3) {  
    index_1 ("0.0125, 0.5, 1");  
    index_2 ("0.0125, 0.5, 1");  
    values ( \  
        "5.5e+18, 5.5e+18, 5.5e+18", \  
        "5.5e+18, 5.5e+18, 5.5e+18", \  
        "5.5e+18, 5.5e+18, 5.5e+18" \  
    );
```

Example2

```
set_var constraint_failed_value "inf"  
fall_constraint (constraint_template_3x3) {  
    index_1 ("0.0125, 0.5, 1");  
    index_2 ("0.0125, 0.5, 1");
```

```
values ( \
    "inf, inf, inf", \
    "inf, inf, inf", \
    "inf, inf, inf" \
);
```

Example3

```
set_var constraint_failed_value    "NaN"
fall_constraint (constraint_template_3x3) {
    index_1 ("0.0125, 0.5, 1");
    index_2 ("0.0125, 0.5, 1");
    values ( \
        "NaN, NaN, NaN", \
        "NaN, NaN, NaN", \
        "NaN, NaN, NaN" \
);
```

This variable must be used before the `char_library` command.

constraint_glitch_hold

| | |
|---------|--|
| <0 1> | Enables glitch height as the failure criteria for characterizing hold time constraints rather than delay degradation. Default: 0 |
| 0 | Use delay degradation as the failure criteria. This results in a more pessimistic hold constraint time than probing for a glitch in the output waveform to signify failure. |
| 1 | Use glitch height as the failure criteria. |

This variable must be used before the `char_library` command.

constraint_glitch_peak

| | |
|---------|---|
| <value> | Specifies the maximum size of logic glitch permitted on the constraint output pin before an arriving signal is deemed to fail a timing constraint (setup, hold, recovery, removal). Default: 0.1 (10%) |
|---------|---|

The `set_constraint_criteria` command can also be used to set this variable. If both this variable and the `set_constraint_criteria` are used, the last one executed sets the value to be used by Liberate.

This variable must be used before the `char_library` command.

constraint_glitch_peak_internal

`<value>` Specifies the maximum size of logic glitch permitted on an internal node before an arriving signal is deemed to fail a timing constraint (setup, hold, recovery, removal).
Default and recommended: 0.2

Note: See `constraint_glitch_peak_mode=2` for recommendations for clearing most "Too close to search bounds" errors.

It is frequently seen that internal nodes have larger glitches than external nodes. Most of these cases are inherent glitches resulting from the momentary contention when a pass-gate is half-open or the circuit is switching between drive paths. In general, these cases represent correct circuit functionality and should be ignored for characterization purposes. The most conservative solution is to slightly increase this parameter to clear these errors. For example, setting `constraint_glitch_peak_internal=0.2` may remove all of the search bound errors. However, this threshold may vary on a case-by-case basis, and keeping track of all node settings may become tedious.

This variable must be used before the `char_library` command.

constraint_glitch_peak_max

`<value>` Specifies the maximum threshold for `constraint_glitch_peak_mode`.
Default: 0.5 (Range: 0.0 - 2.0)

If the new threshold from using `constraint_glitch_peak_mode` exceeds the ratio of `constraint_glitch_peak_max` times VDD, then the threshold is limited to the voltage represented by the ratio of VDD specified by `constraint_glitch_peak_max`. This can result in a search bound error.

If there is a rail-to-rail intrinsic glitch (i.e. always present) then this can be disabled by setting it to a value greater than 1.0 but less than 2.0.

This variable must be used before the `char_library` command.

constraint_glitch_peak_mode

| | |
|-------------|--|
| <0 1 2> | Applies <code>constraint_glitch_peak</code> on top of inherent glitch. Default: 0 Recommended: 1 |
| 0 | Do not apply <code>constraint_glitch_peak</code> on top of inherent glitch. Note: This value is also used if a constraint arc probes an external pin. |
| 1 | Measures the inherent glitch magnitude (noise on the net) and then add that to the <code>constraint_glitch_peak</code> to use as a new threshold. If the new threshold exceeds <code>constraint_glitch_peak_max</code> , the threshold is limited to <code>constraint_glitch_peak_max</code> . This helps prevent warnings about "Too close to search bound" for glitch-based constraint measurements in the Liberate log file. Note: This value is also used if a constraint arc probes an internal node. |
| 2 | Same as option 1, but only pertains to constraint arcs that probe internal nodes (such as clock gaters). This option is useful when the inherent glitch on internal nodes are larger than the inherent glitch on external pins. |

Many cells exhibit inherent glitches immediately upon clock transition. This is a glitch that occurs on a node as a direct result of the clock switching and is not related to any race condition between data and clock. If this inherent glitch occurs at a node that Liberate identifies as the probe node, then the logfile contains the warning message "Too close to search bound". If the constraint measurement criteria is glitch, then it is possible that an inherent glitch occurs on the probe node. Setting `constraint_glitch_peak_mode` can work around this by accounting for the inherent glitch.

This variable must be used before the `char_library` command.

constraint_glitch_peak_report_inherent

`<0 | 1>` Enables the reporting (in the log file) of the measured inherent glitch on the probe node.
Default: 0

Note: The inherent glitch is a glitch that occurs as a result of the `related_pin` switching and is not caused by race conditions between the pin and the related pin. It is measured only when `constraint_glitch_peak_mode` is set to 1 or 2.

- | | |
|---|---|
| 0 | Does not report the measured inherent glitch. |
| 1 | The measured inherent glitch is reported in the log file. |

This variable must be specified before the `char_library` command.

constraint_hold_probe

`<value>` Specifies the name of the cell probe pin to use for hold timing constraint characterization.

Default: '' (Use the name specified by the `constraint_output_pin` variable.)

The `<value>` can also be an internal node specified using the form `<transistor_name>: [S|D|G]` where S is the transistor source, D is the transistor drain, and G is the transistor gate terminal.

This variable must be used before the `char_library` command.

Example

```
# Set the output pin name for constraints
set_var constraint_hold_probe ProbePin
```

constraint_info

| | |
|-------------|--|
| <0 1 2> | Enables printing of constraint measurement details. Default: 1 Recommended: 2 |
| 0 | Does not print any constraint characterization details |
| 1 | Turns on printing of constraint characterization details (including the probe point) into the log file. The information printed includes the cell name, timing type, constraint type, pin, related pin, probe node, and criteria. The information is printed prior to the actual characterization and is therefore based on the characterization plan. |
| 2 | Turns on printing of constraint mapping information after a cell is finished with characterization. Constraint criteria can be updated during simulation. This is the recommended setting. |

This variable must be used before the `char_library` command.

Example

The following is an example of the log file output:

```
constraint_map: Cell: CLKGater; Type: setup_rising; Constraint: rise_constraint;
Pin: EN; Related: CLK; probe:EN->I2:26; criteria: degradation
```

constraint_info_pass_fail

| | |
|---------|--|
| <0 1> | Controls the printing of glitch metric as "glitch" or "pass/fail". Default: 1 |
| 0 | Print glitch metric as "glitch". |
| 1 | Print glitch metric as "pass/fail". |

This variable must be specified before the `char_library` command.

constraint_linear_waveform

| | |
|---------|--|
| <0 1> | Controls whether a linear waveform is used to drive the inputs during constraint characterization. Default: 0 |
| 0 | Uses the same driver that delay and transition use. |
| 1 | Requests a linear waveform for constraint characterization. |

This variable must be used before the `char_library` command.

constraint_margin

| | |
|---------|---|
| <value> | Specifies to add a margin (in MKS units) to apply when the combinational constraint method is used. Default: 2e-12 (seconds) |
|---------|---|

This variable can be used after the `char_library` command.

constraint_merge_state

| | |
|---------|--|
| <0 1> | Controls whether timing groups are merged when the <code>when</code> states are the same. This can occur when there are multiple user-defined constraint arcs with the same <code>when</code> to be characterized using different metrics or probes. Default: 1 |
| 0 | Uses the release 2.4 and prior behavior. |
| 1 | Merges these arcs using the bitwise worst case among the values. |

This variable must be used before the `char_library` command.

constraint_output_load

<min | max | value | index_?>

Sets the type of load on the output pin used for constraint characterization.

Default: min

| | |
|---------|---|
| min | Uses the minimum load index value. |
| max | Uses the maximum load index value. |
| value | Specifies an exact load to use. |
| index_? | Specifies the load index value to use from load index on the delay table where index_0 is the same as min and index_6 is the same as max. The load index contains 7 points. |

This variable must be used before the `char_library` command.

Example

Here are some of the examples:

```
# Set the output pin load to 10ff for constraints
set_var constraint_output_load 10e-15
```

```
# Set the output pin load to use the fourth load index from the delay table
set_var constraint_output_load index_3
```

constraint output pin

<pin>

Specifies the name of the output pin used for determining timing constraint characterization (setup, hold, recovery, and removal values).

Default: " " (alphabetic first output pin name)

The `<pin>` can also be an internal node specified using the form `<transistor_name>: [S|D|G]` where S is the transistor source, D the transistor drain, and G the transistor gate terminal

When calculating constraints, a search is performed by switching the relevant data signal with respect to the clock signal and determining when there is significant delay or voltage impact at a particular pin or node of the cell. This variable defines which of the pins of the sequential cell to monitor.

This variable may be set to " * ". In this case, the constraint searches are repeated on all output pins. The worst-case measurement is reported. Using this functionality can result in approximately a 12% increase in run-time, depending on the number of constraints to be characterized.

This variable must be used before the `char_library` command.

Examples

```
# Set the output pin name for constraints
set_var constraint_output_pin Q
define_cell \
  -input {D} \
  -output {Q QN} \
  -clock {CK} \
  -delay delay_5x5 \
  -constraint constraint_3x3 \
  DFFX1

# Set a pin name for constraints
set_var constraint_output_pin M1:D
```

constraint_output_pin_mode

| | |
|---------|--|
| <0 1> | Controls the selection method of constraint and MPW probe nodes when using the <code>-io</code> option of the <code>char_library</code> command, if the <code>define_arc</code> command does not specify the desired probe node. In the <code>-io</code> option , it is recommended to specify the desired probe node. Default: 0 Recommended: 1 |
| 0 | Issues an error. |
| 1 | Determines the probe node by checking the setting of the <code>constraint_output_pin</code> variable. If <code>constraint_output_pin</code> is a valid pin or node name, then, that is used. However, if <code>constraint_output_pin</code> is ' * ', then uses the first output pin in the alphabetical order. Else, issues an error. |

This variable must be specified before the `char_library` command.

constraint_probe_internal

| | |
|---------|---|
| <0 1> | Specifies whether to probe internal nodes for constraints. Default: 1 |
| | Liberate's Inside View automatically finds probe nodes for measuring constraints. These probe nodes can be primary cell ports or can be internal nodes. This variable applies to all constraints. |
| 0 | Prevents Liberate from choosing an internal node as the probe node for constraint acquisition. |
| 1 | Allows probing internal nodes. |

This variable must be specified before the `char_library` command.

constraint_probe_lower_fall

| | |
|---------|---|
| <value> | Probe lower fall threshold. Default: 0.3 |
|---------|---|

constraint_probe_lower_rise

| | |
|---------|---|
| <value> | Probe lower rise threshold. Default: 0.3 |
|---------|---|

constraint_probe_mode

| | |
|-----------------|---|
| <0 1 2 3> | Specifies the Inside View methodology for selecting internal probe nodes for constraints. This variable applies to all constraints. Default and recommended: 2 |
| 0 | Selects an output node if possible, otherwise select probes from back-to-back CCCs. |
| 1 | Selects an output node if possible, otherwise select probes from any internal simulation wire that connects to a transistor gate. |

- | | |
|---|---|
| 2 | Similar to mode 0. However, choose internal node with the preferred metric over an output requiring a different metric. |
| 3 | Similar to mode 1. However, choose internal node with the preferred metric over an output requiring a different metric. |

Note: The recommendation is to use setting 2. However, if Liberate fails to find an appropriate probe node with this setting, it results in all constraints of a given type being skipped. In this case, use setting 3.

The Liberate Inside View (patented) algorithm automatically finds probe nodes for measuring constraints. Output ports (see `-output` and `-bidi` options of the `define_cell` command) nodes are selected first. When no output has an observable change then an internal probe node must be used. Depending on which node is selected to be the probe, Liberate may change the characterization criteria (for example, from delay to glitch peak) if appropriate for the new node. Inside View has several criteria for judging whether or not a node is suitable to be a probe point.

The probe can be set manually using the `-probe` option of the `set_constraint_criteria` or the `define_arc` commands. If a verbose template is available, then manually adding a `-probe <node>` to the `define_arc` command for a particular constraint forces Liberate to use that probe node, regardless of the setting of this variable. The `-probe node` option of the `define_arc` command overrides the `-probe node` option of the `set_constraint_criteria` command, which in turn overrides the nodes selected by the Inside View algorithm.

This variable must be specified before the `char_library` command.

Example

```
set_var constraint_probe_mode 2
```

constraint_probe_multiple

| | |
|---|---|
| <separate merge> | Allows for multiple probe nodes in a single constraint search. Default: separate Recommended: merge |
| <p>Note: For some designs, it may be desirable to test multiple probe nodes in order to find the worst-case constraint. These could be internal nodes or output pins. Combining probe nodes can give a performance increase over separate simulations.</p> | |
| separate | Uses separate search decks for each probe node. |
| merge | Combines all probe nodes in a single search. |
| <p>Note: In order to enable this feature, the <code>define_arc</code> statements need to have multiple probe nodes specified or multiple probes must be specified with <code>set_constraint_criteria</code>.</p> | |

Examples

Example 1: Scan Flip-Flop with two outputs

```
define_arc -type setup -pin D -related_pin CK -probe {Q QN} SDFQNX1
```

Example 2: 4-bit Flip-Flop

```
define_arc -type setup -pin SE -related_pin CK -probe {Q1 Q2 Q3 Q4} MB4X1
```

Example 3: simple Flip-Flop

```
set_constraint_criteria -probe {ILAT Q} -cell DFF
```

This variable must be used before the `char_library` command.

constraint_probe_upper_fall

| | |
|---------|---|
| <value> | Probe upper fall threshold. Default: 0.7 |
|---------|---|

This variable must be used before the `char_library` command.

constraint_probe_upper_rise

<value> Probe upper rise threshold.

Default: 0.7

This variable is used to set the voltage thresholds to be used when measuring clock and data delays. For more information, see the `-pin_probe_threshold` and `-related_probe_threshold` options of the `define_arc` command.

This variable must be used before the `char_library` command.

constraint_search_bound

<min_time> Specifies the minimum search bound.

Default: -1

Use this variable to set the initial constraint search bounds. The constraint bisection search will start at a maximum of +/- the search bound value. By default, Liberate automatically determines the search bound for optimal run time. When this variable is set to a time (in seconds), Liberate starts the bisection search using +/- the `constraint_search_bound` value that will override the automatically-determined bound value.

When constraint estimation is not done (see `constraint_search_bound_estimation_mode`). This variable determines the range of constraint values to search. The default is 10ns. The search range is +/- the minimum of `constraint_search_bound` (or 10ns) and 0.5*`sim_duration`, adjusted to allow for the constrained and related pin slews.

This variable must be used before the `char_library` command.

constraint_search_bound_bisection_mode

<0 | 1> Determines the definition of a pass/fail for bisection initial bound algorithm.

Default: 1

The bisection search requires an initial passing search bound and an initial failing search bound.

- | | |
|---|--|
| 0 | The initial pass search bound indicates a transition on the probe node and the initial fail search bound indicates the probe node failed to transition. For example, a flip-flop whose output switches (passing search bound) when data arrives well in front of clock, and fails to switch (failing search bound) when data arrives long after clock. |
| 1 | The initial pass search bound indicates a transition on the output. The initial fail search bound also indicates a transition on the probe node, but the delay pushout is greater than the constraint_delay_degrade. This option is useful when characterizing combinational gates for constraints (see combinational_constraint). For example, a nand gate is used to gate a clock where the user wants the setup/hold measured between the data and the clock. The output always switches and the delay pushout determines the pass/fail for the bisection search. |

Note: This value is applicable if delay pushout is the criteria. This is because the glitch criteria should already work with the bisection algorithm.

This variable must be used before the `char_library` command.

constraint_search_bound_estimation_mode

<0 | 1 | 2 | 3> Controls the method used to determine the constraint search bound.

Default and recommended: 2

Note: It is recommended not to change this variable except for backwards-compatibility purposes.

| | |
|---|---|
| 0 | Set this variable to 0 to revert to release 3.1 and prior release behavior. |
|---|---|

- | | |
|---|--|
| 1 | Set this variable to 1 to revert to release 12.1.4 and prior release behavior |
| 2 | Uses a method that is consistent for all settings of <code>extsim_model_include</code> and <code>extsim_exclusive</code> . |
| 3 | Disable search bound estimation (see <u>constraint_search_bound</u>) |

This variable must be specified before the `char_library` command.

constraint_search_bound_probe_mode

- | | |
|----------------------------|--|
| <code><0 1></code> | Set this to expand probe selection. Liberate supports any node that is an output of a channel or the input to a gate to be used as a probe node. Default and recommended: 1 |
| 0 | Uses the default available nodes for expanding probe selection. |
| 1 | Allows additional nodes such as input nodes connected to pass transistors. |

This variable must be used before the `char_library` command.

constraint_search_iteration_limit

- | | |
|------------------------------|---|
| <code><integer></code> | Specifies the maximum number of constraint-related bisection search iterations. When this limit is reached, the bisection-based constraint search terminates. The constraint search fails and an error is flagged if the search has not converged within the specified number of iterations. The minimum supported value is 3. Default: 50 |
|------------------------------|---|

Note: A value of 25 provides backward compatibility to the Liberate 15.1 ISR3 release.

This variable must be set prior to the `char_library` command.

constraint_search_time_abstol

`<min_time>` Set this variable to the minimum constraint binary search window.
Default: 2e-12 (seconds)

This variable affects characterization and should be set prior to the `char_library` command. If `<min_time>` is set too small, this variable can have a significant impact on the run time.

This variable must be used before the `char_library` command.

constraint_slew_degrade

`<value>` Includes slew degradation percentage along with delay degradation as a criteria for determining timing constraints.
Default: -1 (only use delay degradation)

By setting the slew criteria both delay degradation and slew degradation are checked and the first criteria to fail determines the setup and hold values. The slew degradation is a value (0.0 to 1.0) that represents the percentage of slew degradation and is measured using the `measure_slew_*` variables.

The `set_constraint_criteria` command can also be used to set this variable. If both this variable and the `set_constraint_criteria` are used, the last one executed sets the value to be used by Liberate.

This variable must be used before the `char_library` command.

Examples

```
# Set the setup time delay degradation criteria to 12%
set_var constraint_delay_degrade 0.12
# Set the minimum delay degradation criteria to 10ps
set_var constraint_delay_degrade_abstol 1e-11
# Set the slew degradation criteria to 50%
set_var constraint_slew_degrade 0.5
# Set the hold time glitch peak criteria to 15% of Vdd
set_var constraint_glitch_peak 0.15
set_var constraint_glitch_hold 1
```

constraint_snap_to_bound

| | |
|------------------------------|---|
| <code>< 0 1 ></code> | Determines which constraint characterization algorithm to use to snap to the passing bound. Default: 1 (Report the last passing bound) |
| 0 | Uses a proprietary algorithm to determine the final constraint value. |
| 1 | Enables Liberate bisection-based constraint characterization to snap to the last passing bound while characterizing constraints when processing a binary search. This can result in a bit more pessimism and run time. By reporting the last passing bound, you can be assured that the reported constraint value was simulated by Variety. |

This variable must be used before the `char_library` command.

constraint_sweep_pulse_detection_mode

| | |
|------------------------------|---|
| <code>< 0 1 ></code> | Enables a sweep based algorithm instead of the bisection algorithm. The sweep algorithm can significantly increase the characterization runtime. Default and recommended: 0 |
| 0 | Use the bisection algorithm to characterize constraints. |
| 1 | Assumes the constraints exhibit a bump or pulse profile of the delay degradation at the output instead of a single valid constraint value. When set to 1 the constraint is acquired using a sweep algorithm. You can provide the expected minimum width of the passing pulse (bump) using the variable <code>constraint_sweep_pulse_width_max</code> . The minimum step (accuracy resolution) is controlled by the variable <code>constraint_combinatorial_step_size</code> . |

Some cell constraints exhibit what is commonly referred to as a double bump. This occurs when a constraint has 2 legal constraint values. For example, in a DFF (D-Flip-Flop), as the data sweeps setup time toward the clock, the failure criteria is exceeded and a setup time is found. But if the data continues forward, the circuit starts to pass the criteria and eventually again fails the criteria.

This double bump is sometimes related to reconvergence or coupling issues in the circuit design. The default constraint characterization algorithm in Liberate is based on a bisection search. If the constraint exhibits a double bump (aka a constraint pulse), then the bisection search can report either of the setup times.

This variable must be specified before the `char_library` command.

constraint_sweep_pulse_width_max

`<value>` Specifies the maximum width of a constraint pulse (double bump) in seconds.

Liberate uses a step size 2.5 times smaller than this value to ensure hitting the bump. The variable `constraint_sweep_pulse_detection_mode` must be set to enable the sweep pulse algorithm.

This variable must be specified before the `char_library` command.

Example

```
set_var constraint_sweep_pulse_width_max 20e-12 # (20pS)
```

constraint_tran_end_extend

`<value>` Specifies an absolute incremental increase (in seconds) in the transient simulation end time for constraints. This allows you to increase the `.tran` end time by an arbitrary time if the automated `.tran` end time is not sufficient for the simulation measurements to return proper values.
Default: 500e-12 seconds

This variable specifies a duration in seconds to add to the transient simulation end time (see `.tran` in a SPICE format simulation deck). Liberate checks if a transition on the constraint probe occurs within `constraint_tran_end_extend/2` of the transient simulation end

time. This check is performed to detect the possibility that a transition degrading beyond the `.tran end` could be mistaken for a failed constraint bisection iteration. If such a case is detected, a warning is generated that suggests increasing the value of this variable. It is important that the transient simulation does not terminate early because for glitch-based constraints, the reported constraint value might be optimistic while for delay degradation, the reported constraint might result in a pessimistic value.

This variable must be specified before the `char_library` command.

constraint_tran_end_extend_retry

| | |
|----------------------------|---|
| <code><value></code> | When set to greater than 0, provides an alternate value for <code>constraint_tran_end_extend</code> when Liberate detects a condition where the simulation time may be too short to properly detect a glitch. |
| | When the value is specified as 0, this feature is disabled and no retry is done. Default: 0.001seconds |

constraint_tran_end_mode

| | |
|----------------------------------|---|
| <code>< 0 1 2 ></code> | Enables a methodology for setting transient simulation end time in constraint bisection searches when using various simulators. Default and recommended: 1 (Optimized per table entry) |
| | Note: SPICE simulators have different capabilities and limitations. |
| 0 | This value is used for backward compatibility only. You can use mode 2 for SPECTRE, SKI, and some proprietary simulators; otherwise use mode 1. |
| 1 | Uses an optimized <code>tran_tend</code> (transient simulation end time) for each entry of each iteration except for some proprietary simulators, which uses mode 2. |

| | |
|---|--|
| 1 | Uses the maximum <code>tran_tend</code> for each entry in the constraint data table for all simulations. Use this value only when needed and usually only for testing purposes. This setting may increase runtime. |
|---|--|

This variable must be used before the `char_library` command.

constraint_user_defined_probe_mode

| | |
|------------------------------|--|
| <code>< 0 1 ></code> | Enables to use only user-defined probes for setup/hold, recovery/removal. Default: 1 |
| 0 | Behavior of release 12.1.1 and earlier where Liberate treats the user-defined probes as a suggestion. |
| 1 | Liberate uses only user-defined probes for setup, hold, recovery and removal (this is already the behavior for MPW). User-defined probes are specified with the <code>-probe</code> option of either <code>define_arc</code> or <code>set_constraint_criteria</code> . |

This variable must be specified before the `char_library` command.

constraint_vector_equivalence_mode

| | |
|------------------------------|--|
| <code>< 0 1 ></code> | Enables an algorithm that determines which vectors need to be simulated for constraints and which can be skipped as equivalent to the simulated vectors. Default: 0 Recommended: 1 |
| 0 | Uses base algorithm for reducing vectors based on matching all switching modes |
| 1 | Enables a vector equivalency check that helps reduce the number of vectors simulated when not using mega-mode. |

In the case of complex multi-register cells, there can be many delay paths active besides the ones relevant to the particular measurement. For instance, when measuring a constraint at one element of a latch bank, the behavior of the other latches is usually not significant. Previously, Liberate required that the switching activity of all nodes be the same before two vectors could be considered equivalent. With this algorithm, nodes that do not contribute to the measured delay paths are not considered in the vector comparison. This can result in a substantial reduction in the number of simulations required.

This feature is not required when using mega-mode (`mega_enable=2` or the `-type` option of `define_cell` is `mega`) since that mode performs the same function more efficiently. Therefore, `constraint_vector_equivalence_mode` is disabled.

This variable must be used before the `char_library` command.

constraint_vector_mode

| | |
|---------------------------------|---|
| <code>< 0 1 2></code> | Specifies the mode for controlling recovery, removal, and MPW vectors when using the <code>define_arc</code> command. Default and recommended: 2 |
| 0 | Liberate uses the pre-3.0p3 algorithm for vector generation. |
| 1 | If a <code>define_arc</code> is supplied for recovery, removal, or <code>min_pulse_width</code> types, Liberate uses an enhanced algorithm that increases the vector generation effort. |
| 2 | Enables the probing of internal nodes for all types of constraints/mpw when default probing fails to find a vector. This provides more automation in a recharacterization flow (especially for PMK) where certain arcs cannot be acquired without specifying the internal probe node. Examples include MPW on SAVE pin, and <code>non_seq</code> constraints between RDN and RETN pins. |

This variable must be specified before the `char_library` command.

constraint_width_degrade

<value> Specifies the percentage of degradation in the width of a pulse when calculating setup/hold time at the output or internal node of the pulse generator in pulse latch cells. This variable can be used with the `-metric width` option of `define_arc` command.
Default: 0.1 (10%)

constraint_width_degrade_abstol

<value> Specifies the minimum pulse width degradation value permitted (in seconds) at the output or internal node of the pulse generator in pulse latch cells when measuring timing constraints (setup, hold, recovery, removal) with `define_arc -metric width` and `constraint_width_degrade`. The maximum of the `constraint_width_degrade` percentage of the nominal pulse width and the `constraint_width_degrade_abstol` is used as the width degradation criteria.
Default: 5e-12 (5ps)

constraint_width_degrade_abstol_max

<value> Specifies the maximum width degradation value (in seconds) permitted at the output or internal node of the pulse generator in pulse latch cells when measuring timing constraints (setup, hold, recovery, removal) with `define_arc -metric width` and `constraint_width_degrade`. The minimum of the `constraint_width_degrade` percentage of the nominal pulse width and the `constraint_width_degrade_abstol_max` is used as the width degradation criteria. Both `constraint_width_degrade_abstol` and `constraint_width_degrade_abstol_max` can be used simultaneously to set an upper and a lower bound to the delay degradation.
Default: -1 (not used)

constraint_worst_vector_abstol

`<min_time>` Enables this variable to control minimum constraint binning. Liberate performs simulation-based vector binning when an arc contains multiple vectors. The worst vectors are selected based on results falling within `constraint_worst_vector_abstol` of the worst vector. It is recommended to set this variable to the same value as `constraint_search_time_abstol`. Default: 0 (do not apply an absolute tolerance)

This variable must be used before the `char_library` command.

debug_flow

`<1x1 | 2x2>` Speeds up the characterization by reducing the template as defined by the `define_template` command to either a `1x1` or a `2x2` data matrix. This can significantly decrease the run time for a quick debug flow. You can also check the selected points in a slew or load matrix by shrinking the template to a `1x1` or a `2x2` data matrix.

| | |
|------------------|---|
| <code>1x1</code> | Uses the first value in each index. |
| <code>2x2</code> | Uses the first and last values in each index. |

This variable must be specified before the first `define_template` command.

Example

```
set_var debug_flow 2x2
```

def_arc_drive_side_bidi

`< 0 | 1 >` Enables to drive bidi pins that are not the `pin` or `related_pin` of an arc.
Default: 1

| | |
|----------------|---|
| <code>0</code> | Does not to drive side pins that are bidirectional. |
|----------------|---|

| | |
|---|---|
| 1 | Drives any bidirectional pin that is not the pin or the <code>related_pin</code> of an arc. |
|---|---|

This variable must be specified before the `char_library` command.

def_arc_msg_level

| | |
|-----------|---|
| < 0 1 > | Instructs Liberate to report <code>define_arc</code> commands that are not characterized. Default: 1 |
| 0 | Reports an error when no valid vectors for the arc are found. |
| 1 | Reports a warning for every <code>define_arc</code> command that is <i>not</i> characterized. |

Note: The `write_template` command is enhanced to add "`set_var def_arc_msg_level 0`". This change ensures that all new template files created from `write_template` follows the behavior prior to release 3.2.

This variable must be specified before the `char_library` command.

def_arc_vector_consistency_check

| | |
|-----------|--|
| < 0 1 > | Switches to specify the "when" state must agree with the pre-switching state of the -vector. Default: 1 |
| 0 | Sets the behavior of release 3.1p4 and earlier. |
| 1 | The "when" state must agree with the pre-switching state of the -vector. |

This variable must be specified before the `char_library` command.

default_power_avg_mode

| | |
|-----------|---|
| < 0 1 > | Enables a weighted average of all <code>internal_power</code> states under a pin. Default: 1 (weighted average) |
| 0 | Uses simple average. |
| 1 | Uses weighted average. When weighted average is used, the <code>default_power</code> is computed as though <code>conditional_expression</code> was set to separate. |

When **conditional expression** is set to `merge`, the number of `internal_power` groups under a pin can be reduced due to merging of states. When this happens and an average default power group was requested, an average of the actual groups under the pin is used. The default `internal_power` is computed as a simple average or a weighted average of all of the available states.

This variable must be used before the `write_library` command.

define_arc_ignore_mode

| | |
|---------------|---|
| < 0 1 2 > | Ignores all arcs that satisfy the <code>when</code> condition for the <code>-ignore</code> option of the <code>define_arc</code> command and when <code>define_arc_ignore_mode</code> is set to 1. This variable allows similar fine-grained control for the <code>define_arc</code> command. Default: 1 |
| 0 | Ignores all arcs for the <code>pin</code> and <code>related_pin</code> if the <code>-ignore</code> option of the <code>define_arc</code> command is used. All other <code>define_arc</code> options are also ignored. |
| 1 | Considers <code>when</code> condition in the <code>define_arc</code> command in addition to 0. |
| 2 | Allows to ignore arcs of type <code>power</code> for the specified <code>pin</code> or <code>related_pin</code> . |

Inside View uses the information given in the `define_cell` command as a starting point for determining a cell's functionality. Liberate's default behavior is to find all vectors possible from this general description.

In cases where the design intent is different from the actual circuit behavior, there are additional controls to restrict the vectors that Inside View generates. The `define_cell` command accepts a `-when` option which, when used with "set_var simultaneous_switch_from_cell_when 1" restricts the full vector set to only vectors that satisfy the when condition.

Example

```
define_arc \
-ignore \
-type power \
-related_pin CDN \
-pin Q \
DFF
define_arc_ignore_mode
```

This variable must be used before the `char_library` command.

define_arc_preserve_when_string

| | |
|-----------|--|
| < 0 1 > | Enables to preserve a <code>when</code> string in its original form and output to the library. Default: 0 (off) |
| 0 | Allows Liberate to simplify the <code>when</code> string. |
| 1 | Preserve the <code>when</code> string in its original form and output to library. |

Normally Liberate attempts to simplify a `when` condition according to Boolean logic. (For example, in the following command: `define_arc -when "A&B | A&!B"`, Liberate could simplify the "when" condition to `-when "A"`.)

define_duplicate_cap_mode

| | |
|-----------|--|
| < 0 1 > | Specifies the aliasing of pins for the determination of pin capacitance. Default: 0 (off) |
| 0 | No pin aliasing for pin cap characterizations; each pin cap will call for a separate simulation to characterize its value. |

1

Use aliasing to determine pin capacitance; a given pin will be characterized, and its value is applied to all the other pins specified with the `define_duplicate_pins` command.

This works together with the `define_duplicate_pins` command to specify which pin is used to characterize the pin capacitance, and then applied to the "duplicate" (aliased) pins.

This variable must be used before the `char_library` command.

delay_constrained_by_setup_recovery

< 0 | 1 >

Controls how timing constraints are characterized.
Default and recommended: 0

0

1

Characterizes the timing constraints with the normal delay degradation method, but clock to output delays are measured with the data pin also switching – at a time offset as determined by the measured setup/recovery time.

Also, the `constraint_delay_degrade_minimize_dtoq` variable is forced to 0.

This variable must be used before the `char_library` command.

delay_inp_fall

`<value>`

Specifies the % point on the cell input falling waveform from which to measure delays.
Default: 0.5 (50% of supply)

delay_inp_rise

<value> Specifies the % point on the cell input rising waveform from which to measure delays.
Default: 0.5 (50% of supply)

delay_out_fall

<value> Specifies the % point on the cell output falling waveform from which to measure delays.
Default: 0.5 (50% of supply)

delay_out_rise

<value> Specifies the % point on the cell output rising waveform from which to measure delays.
Default: 0.5 (50% of supply)

These variables set the input and output rising and falling transition crossing-points for measuring delays. It is possible to modify the delay measurement thresholds after characterization. To modify these thresholds, use the `read_1db` command, then modify the required delay-measurement thresholds (see `delay_inp_fall`, `delay_inp_rise`, `delay_out_fall`, and `delay_out_rise`) and write the new library.

Note: If the library is characterized with a predriver cell, there may be a small accuracy impact when changing the measurement thresholds without re-characterizing. Only combinational rise/fall arc-related delays can be modified in this way. Timing constraints cannot be modified without rerunning the characterization

These variable must be used before the `char_library` command.

Examples

```
# Set the delay measurement from 45% to 55%
set_var delay_inp_fall 0.45
set_var delay_inp_rise 0.45
set_var delay_out_fall 0.55
set_var delay_out_rise 0.55
```

disable_method

<0 | 1 | 2>

Controls how the timing arcs for tri-state devices are calculated when the device is transitioning to the OFF state (three_state_disable). See the figure below.

Default: 0

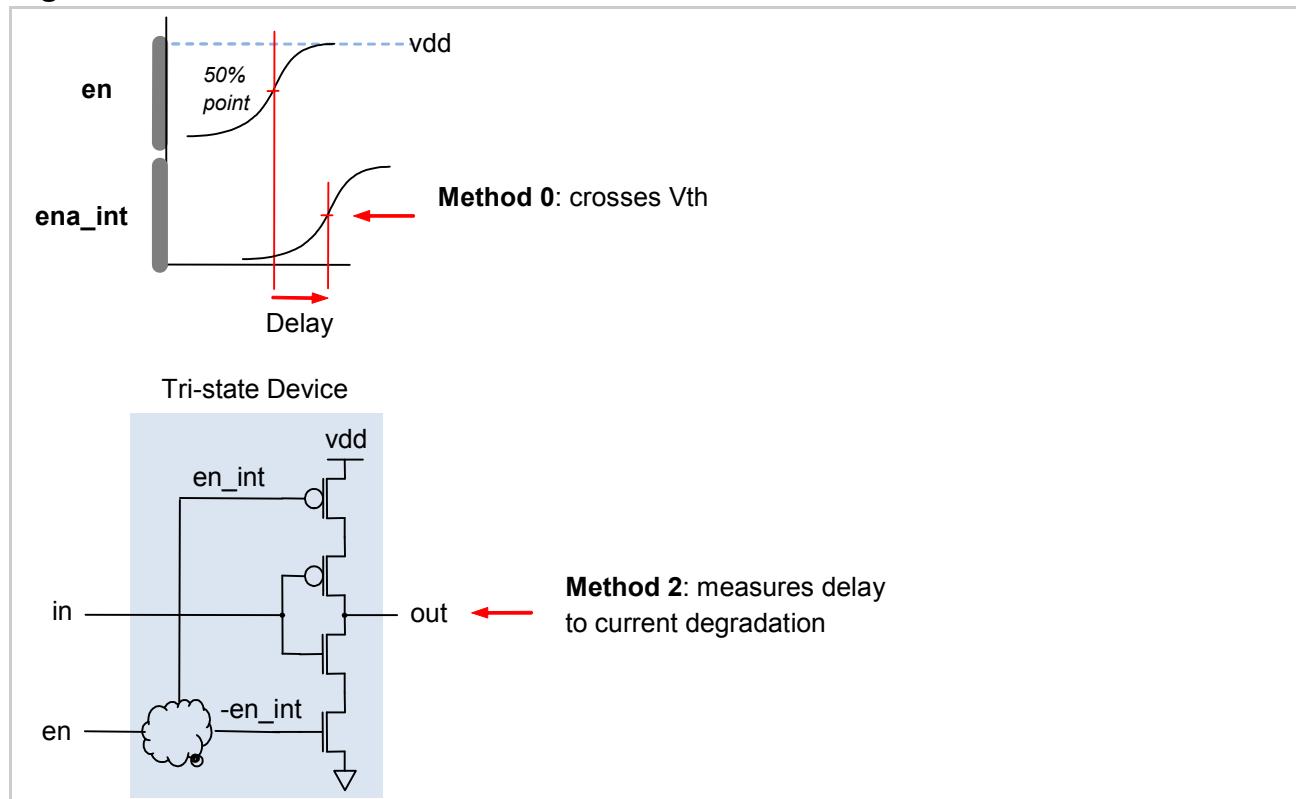
0 Calculates the disable timing arcs (three_state_disable) by measuring the delay time from the input disable pin to the time the input signal to the last transistor stage (channel connected block) crosses the lowest threshold voltage (V_{th}) of the impacted transistors.

1 Measures the delay from the enable pin (en) crossing the 50% threshold, to the input pin of the channel-connected region (en_int or -en_int, whichever is worse) reaching half the supply voltage ($V_{dd}/2$)

2 Measures the delay from the enable pin (en) crossing the 50% threshold, to the time when the output drive current degrades to 10% of the short-circuit current.

Note: This method is automatically enabled if disable_method is set to something other than 2 but Liberate fails to find any last stage CCR input to probe. However, if the user specifies an internal probe, Liberate respects that regardless of the disable_method.

Figure 5-1



Note: The `timing_sense` for the `three_state_enable` or `three_state_disable` arc may have a different meaning from the default one. According to the Synopsys Liberty specification, if a value of 1 on the control pin of a three-state cell causes a `z` value on the output pin, the `timing_sense` is `positive_unate` for the `three_state_disable` timing arc and `negative_unate` for the `three_state_enable` timing arc. If a value of 0 on the control pin of a three-state cell causes a `z` value on the output pin, the `timing_sense` is `negative_unate` for the `three_state_disable` timing arc and `positive_unate` for the `three_state_enable` timing arc.

This variable must be used before the `char_library` command.

Example

```
# Measure disable delay to Vdd/2
set_var disable_method 1
```

discard_timing_sense_after_merge

| | |
|-----------|--|
| < 0 1 > | Enables removal of <code>timing_sense</code> after the merging of two timing groups. Default: 0 (Do not remove <code>timing_sense</code> .) |
| 0 | Does not remove <code>timing_sense</code> . |
| 1 | Removes the <code>timing_sense</code> after the merging of two timing groups. |

See [timing_group_unateness](#).

This variable must be set before the `char_library` command.

disk_wait_time

| | |
|---------|--|
| <value> | Specifies the number of seconds you want Liberate to wait before it attempts to read the external simulator output file (<code>sim.lis</code>). When Liberate is using an external simulator (see <code>char_library -extsim</code>), Liberate attempts to read the simulator output file (<code>sim.lis</code> for Hspice) as soon as the simulator terminates. Due to network latencies, the <code>sim.lis</code> file may not yet be ready to read. Default: 0 (seconds) |
|---------|--|

This variable must be used before the `char_library` command.

driver_cell_acc_mode

| | |
|-----------|---|
| < 0 1 > | Generates a more accurate driver waveform for minimum slews. Default: 1 (on) |
| 0 | Reverts to 3.1p2 (or earlier) behavior in sampling active driver waveforms. |
| 1 | Generates a more accurate driver waveform for the minimum slews. |

This variable must be used before the `char_library` command.

driver_cell_all_inputs

| | |
|------------------------------|--|
| <code>< 0 1 ></code> | Holds side pin constant with driver cell. Default: 0 |
| 0 | When a driver cell is specified for the cell/pin, and the pin is a constant side input connected to a transistor channel (source/drain), then the side pin is held to a constant value by the specified driver cell. |
| 1 | Same as setting '0' except all constant side pins will be driven by the specified driver cell. |

This variable must be used before the `char_library` command.

driver_cell_info

| | |
|------------------------------|---|
| <code>< 0 1 ></code> | Enables reporting for each cell input. The report includes the driver cell that is the characterized driver for the input. This report is useful to determine the actual driver cell assigned to each input when many <code>set_driver_cell</code> commands are used. Default: 1 |
| 0 | Does not report for each cell input. |
| 1 | Requests a report for each cell input. |

This variable must be used before the `char_library` command.

driver_cell_load_all_outputs

| | |
|------------------------------|--|
| <code>< 0 1 ></code> | Enables reporting if the active driver load is applied to secondary (side) pins while adjusting loads to match slews on primary pins. For this functionality to work, the <code>set_driver_cell</code> <code>char_pin</code> and <code>pin_map</code> options must be used to map active driver pins to cell pins. Default: 1 (apply the load to the secondary side pins) |
| 1 | Applies the load to the secondary side pins. |

This variable must be used before the `char_library` command.

driver_cell_load_lDb_cmd

< 0 | 1 > Instructs Liberate to honor all `set_driver_cell` commands stored in an ldb that is loaded using the `read_ldb` command. The presence of the `set_driver_cell` commands is required to enable the output of Normalized Driver Waveforms (NDW) into the library when starting from an ldb.
Default: 0

This variable must be used before the `read_ldb` command.

driver_waveform_arcs_only

< 0 | 1 > Enables to select the method to be used to filter the input driver waveforms that are stored and available for output into the `.lib` file.
Default: 0

0 Writes out all of the waveforms that are used during characterization into the output library without any regard to the type of data the waveform was used to characterize.

1 Enables to store only those input waveforms used to characterize arcs for delay and power. If there are no delay or power arcs, then no input driver waveform data is stored. When the library (or pin) is characterized only for input capacitance (see `char_library -skip`), the library database does not have any characterization data from delay or power tables to save.

Note: Use this value to revert to the release 2.4p2 and prior behavior.

This variable must be specified before the `write_library` command.

driver_waveform_output_precision

<value> Specifies the number of digits of precision to maintain when generating driver waveform data.
Default: 0

The default setting of zero allows Liberate to select the number of digits to be used.

This variable must be specified before the `read_library`, `read_ldb`, or any other command that generates driver waveform data. This includes `write_template` and `write_library`.

Example

```
set_var driver_waveform_output_precision 10
read_library my.lib
write_template template.tcl
```

driver_waveform_pulse_mode

<0 | 1> Enables to choose the pulse algorithm. When combining leading and trailing driver waveforms to form a pulse, sometimes a sharp spike could occur depending on the time delta between the last point chosen on the leading waveform and the first point chosen on the trailing waveform.
Default: 1

| | |
|---|---|
| 0 | Reverts to release 2.4 and prior behavior, |
| 1 | Creates a new point to approximate where the two waveforms cross. |

This variable must be used before the `char_library` command.

driver_waveform_wildcard_mode

| | |
|---------|--|
| <0 1> | Enables to control formatting of the Normalized Driver Waveform (NDW) data in the output library when the <code>define_input_waveform</code> command is used to specify the input waveforms and the pin names specified with it contain wildcards, Default and recommended: 0 |
| 0 | Wildcards in the <code>define_input_waveform</code> command are not supported when writing NDW data in the output library. If wildcards exist in a <code>define_input_waveform</code> command, NDW is not written out. In addition, it is recommended that wildcards are not used in the <code>define_input_waveform</code> command. |
| 1 | Wildcards in the <code>define_input_waveform</code> command are expanded when writing NDW data into the output library. This can lead to many large NDW tables in the output library. |

This variable must be used before the `char_library` command.

duplicate_pin_attr_mode

| | |
|-------------------------------|---|
| <use_master augment_master> | Controls how master pin attributes are populated to duplicated pins. Default: <code>use_master</code> |
| <code>use_master</code> | All attributes from the master pin are copied to the duplicate pin <u>except</u> those generated from characterization, that is, pin capacitance. |
| <code>augment_master</code> | Copy all the attributes from the master pin to duplicated pin, but do not overwrite attributes that have already been set for that pin. |

duplicate_risefall_power

< 0 | 1 > Enables power duplication.

Default: 0

Note: In the Liberate MX and AMS flows, `duplicate_risefall_power` is set to 1 by default for backward compatibility in the regression.

0 The missing power direction is represented by a scalar group with value of 0.

1 Duplicates the power values when only rise or fall power (but not both) exist, For example, if only `rise_power` exists but not `fall_power`, then copy the `rise_power` values into `fall_power`. This duplication only applies to `internal_power` groups where the output pin is switching. It does not apply to hidden power.

This variable can be used after the `char_library` command.

Example

```
# Duplicate existing power to missing power
set_var duplicate_risefall_power 1
```

duplicate_risefall_power_ccsp

<0 | 1 | 2> Controls copying rise and fall CCSP dynamic currents into a missing rise and fall group.

Default: 0

0 Does not copy the rise and fall CCSP dynamic currents.

1 Copies rise and fall CCSP dynamic currents into the missing rise and fall group (similar to what the `duplicate_risefall_power` variable does).

Virtuoso Liberate Reference Manual

Liberate Parameters

| | |
|---|---|
| 2 | Same as setting it to 1, except Liberate includes the <code>input_switching_condition</code> in the copied group. |
|---|---|

This variable must be used before the `char_library` command.

ecsm_arctype_enable

<0 | 1> Enable modeling of the `ecsm_arctype` attribute in ECSM format libraries. Default: 0

Note: In the Liberate MX and AMS flows, `duplicate_risefall_power` is set to 1 by default for backward compatibility in the regression.

| | |
|---|--|
| 0 | Does not output the <code>ecsm_arctype</code> attribute in ECSM format libraries. |
| 1 | Enable modeling of the attribute <code>ecsm_arctype</code> for <code>ecsm_vivo_current_waveform</code> groups in ECSM noise model. The optional <code>ecsm_arctype</code> attribute is a string with four valid values: <code>max_rise</code> , <code>max_fall</code> , <code>min_rise</code> , and <code>min_fall</code> . The <code>max_rise</code> and <code>max_fall</code> values represent the Vivo arcs corresponding to the slow transition of the signal on the output. While <code>min_rise</code> and <code>min_fall</code> are for the fast transition. Including this attribute may help to clear up messages when using the <code>ecsm_checker</code> utility. |

This variable must be set before the `char_library` command.

ecsm_cap_hidden_pin

<0 | 1 | 2> Specifies how ECSM capacitance is written for input pins.
Default: 2

Prior to version 3.1 the variable `ccs_cap_hidden_pin` (default = 1) controlled how the ECSM capacitance was written. The variable `ecsm_cap_hidden_pin` provides a separate control for this.

| | |
|---|--|
| 0 | Hidden pins only. |
| 1 | Hidden and half-hidden pins. (<i>Set this for pre-3.1 behavior.</i>) |
| 2 | All input pins. |

This variable must be used before the `write_library` command.

ecsm_cap_input_slew_mode

<0 | 1> Enables ECSM capacitance input slew adjustment mode.
Default: 0

| | |
|---|---|
| 0 | Writes the original <code>index_1</code> for <code>ecsm_capacitance</code> . In this mode, the <code>index_1</code> slew values in ECSM data tables are not adjusted. |
| 1 | Use the adjusted <code>index_1</code> slew values from release 3.1p2 and earlier. |

Note: This adjustment is no longer recommended.

This variable must be used before the `write_library` command.

ecsm_cap_load_effect_tol

<value> Specifies the `ecsm_capacitance_set` effective threshold.
Default: 0.001 (0.1%)

If an arc, for example, `C1k -> Q`, has load dependent eCsm capacitance values that are within the tolerance limits, it means that the pin is shielded from the output load. The verification is done for each arc. When the load effect is less than this tolerance, then output the average cap over all the loads.

This variable specifies the tolerance used when the variable `eCsm_cap_style` is set to `capacitance_set`.

This variable must be set before the `write_library` command.

eCsm_cap_mode

`<0 | 1 | 2>`

Sets the ECSV capacitance modelling mode.

Default: 2 (Output a 1 or 8-piece cap table.)

This variable enables the output of a capacitance table with multiple steps. Liberate characterizes and saves the capacitance data into the ldb. It can then be written into a library. The three capacitance steps are fixed at the levels specified by the control variables `measure_slew_lower_fall`, `measure_slew_upper_fall`, and `delay_inp_fall` for a falling input transition, and by `measure_slew_lower_rise`, `measure_slew_upper_rise`, and `delay_inp_rise` for a rising input transition.

0

Outputs 1 (when `eCsm_version = 1.0`) or 2 piece cap table (when `eCsm_version >= 2.1`). This setting is provided for backward compatibility.

1

Outputs a 3-piece cap table when `eCsm_version >= 2.1`.

Note: To output a 3-piece cap table for input pins, we also recommend setting the variable `ccs_cap_hidden_pin` to 2

2

Outputs an 8-piece cap table when `ecsm_version >= 2.1`. The 8 pieces default to the following and can be overridden by the `set_receiver_cap_thresholds` command:

```
-rise {0.1 0.3 0.5 0.6 0.7 0.8 0.9  
0.9999}  
-fall {0.9 0.7 0.5 0.4 0.3 0.2 0.1  
0.0001}
```

This variable must be used before the `char_library` command.

ecsm_cap_style

`<capacitance | capacitance_set>`

Enables modeling of ecm pin capacitance in a compact data structure known as: `ecsm_capacitance_set`. This data structure is used when the ECSM pin capacitance does not depend on output load.

Default: `capacitance`

`capacitance` Outputs load dependent
 `ecsm_capacitance` value tables.

`capacitance_set`

Outputs compact
`ecsm_capacitance_set` value tables. The load dependent capacitance data is verified. If the effect of the load is less than a specified tolerance (see `ecsm_cap_load_effect_tol`), then ignore the load effect; if not, keep all load dependent data.

This variable must be specified before the `write_library` command.

ecsm_cap_use_input_transition

`<0 | 1>`

Sets `ecsm_capacitance` to follow the input transition.
Default: 1 (Follow the input.)

- | | |
|---|---|
| 0 | ecsm_capacitance follows the <u>output</u> transition. |
| 1 | ecsm_capacitance tables follow the <u>input</u> transition. |

This variable must be used before the `write_library` command.

ecsm_capacitance_factor

- | | |
|----------------------------|--|
| <code><value></code> | Specifies a multiplication factor to apply to the ECSM capacitance data. Use this factor in combination with the <code>ecsm_capacitance_precision</code> to reduce the number of digits of ECSM capacitance output into the library. This factoring reduces the number of leading zeros without compromising accuracy. Default: 10000 |
|----------------------------|--|

This variable must be specified before the `write_library` command.

ecsm_capacitance_precision

- | | |
|----------------------------|---|
| <code><value></code> | Specifies the number of digits of precision to output into the library for ECSM capacitance data. Default: 3 |
|----------------------------|---|
- The default value of 3 is consistent with the default tolerance for capacitance (0.001) in the `compare_library` utility.

Ensure that the precision is not set too small so that accuracy is not compromised.

This variable must be specified before the `write_library` command.

ecsm_factor_mode

| | |
|---------|---|
| <0 1> | Enables reduction in the physical size of the output library by factoring the ECSM data and controlling the precision. For more information, see the <code>ecsm_capacitance_factor</code> , <code>ecsm_waveform_time_factor</code> , <code>ecsm_capacitance_precision</code> , and <code>ecsm_waveform_time_precision</code> variables. Default: 0 |
| 0 | Does not apply any factoring to the ECSM data. |
| 1 | Apply factoring and precision control to the ECSM data in order to reduce the physical size of the output library. |

Example

The characterized ECSM capacitance data is 0.123456 ff in the `.lib`. The `ecsm_capacitance_factor` is set to 1000. The `ecsm_capacitance_precision` is set to 5. Therefore, $0.123456 * 1000 = 123.456$; keeping five digits this data becomes 123.45 in the final library.

Note: The variable `ldb_precision` controls the number of digits stored in the LDB.

This variable must be specified before the `write_library` command.

ecsm_invert_gnd_current

| | |
|---------|---|
| <0 1> | Inverts current values for <code>ecsm_current_waveform</code> groups. Default: 1 (Invert current.) |
| 0 | Does not invert currents. |
| 1 | Inverts the current values (<code>index_1</code>) for <code>ecsm_current_waveform</code> groups associated with ground supplies. (Currents are normally positive instead of normally negative.) This is for compatibility with EPS. |

This variable must be specified before the `char_library` command.

ecsm_measure_output_range

<0 | 1>

Enable scaling of the voltage span for the ECSM format. The ECSM format requires that the data for the output voltage swings from Vss to Vdd. When there is an output that does not swing from Vss to Vdd, the data must be shifted such that it does swing from rail to rail (for example, Vss to Vdd).

Default: 1

0

Provides backward compatibility to the 3.2 and prior releases.

1

Modifies the measurement algorithm to scale the voltage span according to the simulation outcome. For example, if the output swings between $0.03*Vdd$ and $0.94*Vdd$ then Liberate will sample the output crossing times at the ECSM voltage grid adjusted for the span:

$$(0.03*Vdd) + (span * [0.02, 0.05, 0.1, 0.2, \dots 0.9, 0.95, 0.98])$$

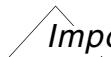
where:

$$\begin{aligned} span &= (0.94*Vdd) - (0.03*Vdd) \\ &= 0.91*Vdd \end{aligned}$$

In high-leakage / low voltage threshold designs, the voltage may not start nor end at the rail. If we add a stricter voltage requirement (within 2% of the rail instead of 5%), then there is more characterization failures on those designs, unless this parameter is set to 1.

In addition, the following ECSM attributes are added to the timing group:

ecsm_base_rail_vdd_rise
ecsm_base_rail_vdd_fall
ecsm_base_rail_gnd_rise
ecsm_base_rail_gnd_fall



Check if your version of ETS supports these attributes before setting this variable to 1. If this is supported, we recommend a setting of 1.

This variable must be specified before the `char_library` command.

ecsm_version

| | |
|-------------------|---|
| <1.1 2.0 2.1> | Specifies the ECSM format version. The supported versions are: 1.1, 2.0, and 2.1. Default: 2.1 |
| 1.1 | Refers to output transition. |
| 2.0 | Includes information on each cell's average N and P threshold voltage, in addition to ecsm_capacitance information on hidden pins such as the D pin of a flip-flop. In ECSM 2.0 the ecsm_capacitance group within a rise/fall_transition group refers to the <u>input pin transition</u> not the output transition as was the case for ECSM 1.1. If no advanced multi-piece receiver capacitance data is stored in the ldb , then a <u>single</u> capacitance value is outputted. |
| 2.1 | Supports an ecsm_capacitance with 2-piece or 3-piece capacitance tables, dependent on the setting of <u>ecsm_cap_mode</u> . |

This variable can be used after the `char_library` command.

Example

```
# Set the ECSM version to 2.1
set_var ects_version 2.1
```

ecsm_waveform_for_bidi_pin

| | |
|---------|---|
| <0 1> | Enables generation of ecsm_waveform groups under bidir pins. Default: 0 |
| 0 | Used for backward compatibility. |
| 1 | Generates ecsm_waveform groups under bidir pins. |

This variable must be used before the `char_library` command.

ecsm_waveform_style

`<waveform | waveform_set>`

Requests output of `ecsm_waveform_set` compatible ECSM format libraries. The *ECSM Waveform Set* stores time points without including voltage because the voltages are shared. This allows for a compact representation of the ECSM model.

Default: `waveform_set`

| | |
|-----------------------|---|
| <code>waveform</code> | Uses this value for compatibility with Liberate 14.1 ISR1 and prior releases. |
|-----------------------|---|

| | |
|---------------------------|--|
| <code>waveform_set</code> | Uses this value to enable the more compact <i>ECSM Waveform Set</i> data format. |
|---------------------------|--|

The variable must be specified before the `write_library` command.

ecsm_waveform_time_factor

`<value>` Specifies a multiplication factor to apply to the ECSM time waveform data. Use this factor in combination with the `ecsm_waveform_time_precision` to reduce the number of digits of ECSM time waveform data output into the library. This factoring reduces the number of leading zeros without compromising accuracy.

Default: 1000

This variable must be specified before the `write_library` command.

ecsm_waveform_time_precision

`<value>` Specifies the number of digits of precision to output into the library for ECSM waveform based time data.

Default: 3

The default value of 3 is consistent with the default tolerance for capacitance (0.001) in the `compare_library` utility

Ensure that the precision is not set too small as this could impact accuracy.

This variable must be specified before the `write_library` command.

ecsmn_loadcap_mode

| | |
|---------|---|
| <0 1> | Specify whether the miller or device caps will be included in the <code>ecsm_loadcap</code> attribute. Default and recommended: 1 |
| 0 | Liberate includes wire caps and miller caps in the <code>ecsm_loadcap</code> attribute. This setting is provided for backward compatibility with LIBERATE14.1 ISR2. |
| 1 | Liberate includes wire caps and device caps in the <code>ecsm_loadcap</code> attribute. |

This variable must be used before the `char_library` command.

ecsmn_mode

| | |
|---------|---|
| <0 1> | Controls ECSMN noise modeling behavior for pin-level or arc-level constructs. Default: 1 |
| 0 | Use pin-based ECSM-N models for all groups. |
| 1 | Generates arc-based ECSM Noise models for single-stage Channel Connected Circuits (CCC) and pin-based models for 2+ stages. This can improve accuracy at smaller geometries, and is recommended by Cadence. |

This variable must be used before the `char_library` command.

ecsmn_vtol_mode

| | |
|---------|--|
| <0 1> | Requests <code>vltolerance</code> and <code>vhtolerance</code> characterization. Default: 0 Recommended: 1 |
|---------|--|

| | |
|---|--|
| 0 | Uses a fixed value for vltolerance and vhtolerance. |
| 1 | Computes the vltolerance and vhtolerance based on the DC transfer characteristics of the cell. |

This variable must be set prior to the `char_library` command.

`em_calculation_include_input`

| | |
|---------|---|
| <0 1> | Specifies whether EM calculation should be performed on the input pins. Default: 1 |
| 0 | Do not consider input pins for EM calculation. |
| 1 | Consider input pins for EM calculation. |

This variable must be used before the `char_library` command.

`em_calculation_monitor_rails`

| | |
|---------|---|
| <0 1> | Specifies if the EM calculation should be performed on the power supply rails. Runtime can be improved by skipping the rails. Default: 1 |
| 0 | Limits the EM calculation to just the signal nets eliminating both supply and ground rails from being considered. |
| 1 | EM calculation is performed on all nets including the rails. |

For more information, see the Liberate Details section on Electromigration.

This variable must be used before the `char_library` command.

em_calculation_monitor_rails_skip_layer

<layer> Specifies the names of specific process layers to be skipped when performing Electromigration calculation on power supply rails.
Default: all layers are considered

Example

```
# Will consider all layers except M0 and M1 layer for monitoring em on rail nets.
```

```
# This setting affects power nets only.
```

```
# All layers including M0 and M1 are considered for signal nets.
```

```
set_var em_calculation_monitor_rails 1
set_var em_calculation_monitor_rails_skip_layer {M0 M1}
```

This variable must be set prior to the `char_library` command.

em_char_arcs_mode

<0 | 1 | 2 | 3> Controls the arcs simulated during EM characterization. For EM characterization, Liberate can gather data from switching and hidden arcs in either rising or falling directions.
Default: 3

0 Use all switching and hidden arcs on both rising and falling directions. This option has the longest runtime and provides the most comprehensive set of EM data.

1 Same as setting 0, but use pulses with an initial rising edge (RF) and skip pulses starting with initial falling edge (FR). This setting takes approximately 50% of the runtime of option 0, and provides similar accuracy.

2 Same as setting 0, but skip hidden vectors for all related pins on switching vectors. Use hidden vectors for any input pin that is not a related pin on switching arcs. This provides the same EM data for output pins, but removes redundant or unnecessary data and runtime on input pins.

| | |
|---|--|
| 3 | Combination of 1 and 2. This setting provides all necessary EM data with the best performance. |
|---|--|

This variable must be used before the `char_library` command.

em_clock_freq

`<value>` Specifies the clock frequency at which to characterize electromigration. The value specified must be a number in Hertz. The scientific notation is supported.
Default: `1e-9`

For more information, see the [Electromigration Models](#) section in the Liberate Details chapter.

This variable must be used before the `char_library` command.

Example

```
# Set the EM Clock Frequency to 20MHz
set_var em_clock_freq 20e6
```

em_current_type

`<0 | 1>` Specifies the EM current type used for modeling. Liberate can model EM using only average currents, or the three current types: peak, avg, and rms.
Default: 1

The variable affects the electromigration characterization flow, which is enabled by using the `-em` option of the `char_library` command.

- | | |
|---|--|
| 0 | Model EM groups using only average currents. |
| 1 | Model EM groups using all 3 current types: peak, avg, and rms. |

This variable must be used before the `char_library` command.

em_data_file

<string> Specifies the Spectre APS EMIR data file needed to characterize electromigration. If the `em_tech_file` is also provided, the `em_data_file` is ignored
Default: “ ”

For more information, see the [Electromigration Models](#) section in the Liberate Details chapter.

This variable must be used before the `char_library` command.

Example

```
set_var em_data_file "/home/work/MyTech.dat"
```

em_freq_mode

<0 | 1> Controls the selection of time window used for electromigration characterization.
Default: 1

| | |
|---|---|
| 0 | Uses the electromigration time window calculated based on frequency specified by <code>em_clock_freq</code> . |
| 1 | Estimates time window based on the time needed for output toggling. |

This variable must be used before the `char_library` command.

em_iacpeak_mode

<0 | 1> Enables to save iacpeak type from `aps`.
Default: 1

| | |
|---|---|
| 0 | Do not save iacpeak type information |
| 1 | Saves the iacpeak type from Spectre APS during EM characterization under max data. This uses the iacpeak results from APS instead of the max results. |

This variable must be specified the `char_library` command.

em_include_string

`<string>` Specifies the string to be used when loading the subckt netlists for EM analysis into SPECTRE.
Default and recommended: ".dspf_include"

In MMSIM 14.1 and later versions, support for the `.inc` statement in the SPICE deck changed for EM analysis. When a `.inc` statement is used, only the top-level nodes of a testbench are analyzed for EMIR. The analysis nodes defined in the EM config file are completely ignored. Therefore, the signal nets are no longer included in the EM analysis. In order to access these nodes, the `.inc` statement needs to be changed to `.dspf_include`. Once this is done, the lower level nets appear in the EM report.

Note: ".dspf_include" is default and recommended for advanced FinFET nodes.
".inc" is recommended for 28nm nodes and older nodes.

This variable must be specified before the `char_library` command.

Example

```
set_var em_include_string ".dspf_include"
```

em_maxcap

`<0 | 1 | 2>` Specifies what type of EM data to output into the library.
Default: 0

| | |
|---|---|
| 0 | Outputs <code>em_maxtoggle_rate</code> format data. |
| 1 | Converts <code>em_maxtoggle_rate</code> data into maxcap table based on frequency and slew. Output EM maxcap table data. |
| 2 | Converts <code>em_maxtoggle_rate</code> data as in 1. Output both <code>em_maxtoggle_rate</code> and maxcap data. |

The maxcap table values must be zero or greater and limited by the `max_capacitance` pin attribute. There is no pin-based EM maxcap data because this is only based on slew and not on load. For more information, see the [Electromigration Models](#) section in the chapter [Liberate Details](#).

This variable must be used before the `char_library` command.

`em_maxcap_type`

`<avg | rms | peak>` Specifies the type of EM data to use when creating maxcap tables
Default: avg

This variable must be used before the `char_library` command.

`em_maxcap_frequency`

`<value>` Specifies a list of frequency values to be used as `index_1` for all EM maxcap tables in Hertz. If not specified, the `index_1` frequency is chosen from the middle slew in the `em_maxtoggle_rate` table.
Default: ""

This variable must be specified before the `char_library` command.

`em_report_data_usage_mode`

`<0 | 1>` Controls how EM violations are calculated.
Default: 0
Recommended: 1

| | |
|---|---|
| 0 | Use pass or fail rate in the EM report to calculate EM violation. |
| 1 | Use "I limit" and "Current" in the EM report to calculate EM violation. |

EM violations can be calculated by Pass/Fail rate or "I limit" and "Current" in the EM report. Pass/fail rate has only one significant digit. When this variable is set to 1, Liberate can use "I limit" and "Current" that have more significant digits in the EM report to calculate EM violation.

This variable must be used before the `char_library` command.

em_tech_file

<string> Specifies the QRC technology file to be used to characterize electromigration. It is recommended to use a QRC technology file specified using this variable instead of the Spectre APS EMIR data file specified using `em_data_file`.
Default: " "

For more information, see the [Electromigration Models](#) section in the Liberate Details chapter.

This variable must be used before the `char_library` command.

Example

```
set_var em_tech_file "/home/work/MyTech.qrc"
```

em_user_string

<string> Specifies a string containing Spectre-APS EMIR configuration commands that are included in the emir configuration file.
Default: " "

For more information, see the [Electromigration Models](#) section in the Liberate Details chapter.

This variable must be used before the `char_library` command.

Example

```
set_var em_data_file "emirutil blech_length=longest_path"
```

em_window_estimate_mode

<0 | 1 | 2> Enables EM window estimation on both the rising and falling edges.
Default: 2

0 Enables backward compatibility to LIBERATE 14.1 ISR4.

| | |
|---|--|
| 1 | Enables EM window estimation based on both the rising and falling edges. Setting up EM windows based on <code>clq->q</code> delays may still have constraint violations on some of the fast slew or small load data points. Liberate buffers all such windows to avoid constraint violations. To buffer only a specified set of cells that are known to have failed before, use the <u>em_buffer_cell_pin_bounds</u> command. |
| 2 | Estimates the em windows needed for the pulse generation based on the rise and fall transitions for any specific slew load. This has a visible impact if the rise and fall delays are different for a cell. |

This variable must be specified before the `char_library` command.

enable_command_history

| | |
|---------|--|
| <0 1> | Enables to log commands. Default: 0 |
| 0 | Do not log commands |
| 1 | Enables command logging. |

extsim_ccs_option

| | |
|----------------------|--|
| <" <i>options</i> "> | Specifies the list of options to be used by the external SPICE simulator when characterizing CCS timing. Default: set to <code>extsim_option</code> |
|----------------------|--|

This variable only applies when the `-ccs` option is used with `char_library`. For accurate CCS current measurements it is advisable to use an accurate simulator setting (e.g. `"runlvl=6"` for Hspice). As delay and power characterization share the same simulation as CCS timing, this option also impacts NLDM delay, transition, power, and ECSM waveform values. The `options` string is passed as an `.option` line in the SPICE decks Liberate creates for characterization.

This variable must be used before the `char_library` command.

extsim_cells_use_nodeset_for_io_pad

<"list">

Specifies one or more cell(s) for which Liberate uses .nodeset instead of .ic in the simulation deck when the char_library -io option is specified. The .ic/.nodeset spice command is added on output pad nodes to help the external simulator find a DC solution. In some cases, the external simulator may find a circuit has difficulty reaching DC convergence with .ic but a .nodeset may help. In other cases, the opposite may be true. By default this parameter is empty, meaning there is no change in the Liberate default behavior.

Example

To use .nodeset instead of .ic for cell_A and cell_B only:

```
set_var extsim_cells_use_nodeset_for_io_pad "cell_A cell_B"
```

To use .nodeset instead of .ic for all cells:

This variable must be used before the **char_library** command.

extsim_cmd

<string>

Specifies a command string to be used to call the external SPICE simulator. This variable can be used to override the default command used by Liberate to call the external simulator. Default: see the information given below

This variable can be used to override the default command used by Liberate to call the external simulator. The default settings are:

Important

The simulator specified with **extsim_cmd** must agree with the **-extsim** option specified with char_library.

The default settings are:

- For Hspice (Synopsys): extsim_cmd="hspice"

```
# call HSPICE
```

```
$extsim_cmd $extsim_cmd_option \
-i $spicedir/sim.sp -o $spicedir/sim \
>& /dev/null
```

- For Spectre (Cadence): extsim_cmd="spectre"

```
# call Spectre
$extsim_cmd $extsim_cmd_option \
    sim.sp >& sim.lis
```

This variable must be used before the `char_library` command.

Example

```
set_var extsim_cmd "spectre2"
set_var extsim_cmd_option "+log mylogfile"
```

File spectre2 contains:

```
#!/bin/sh
exec spectre $*
```

extsim_cmd_option

`"string"` Specifies options to be passed to the external simulator.

This variable can be used to override the default command options used by Liberate to call the external simulator. It is recommended that Spectre models are provided when using the Spectre simulator. If Hspice models are used with Spectre, then this variable may need to include "+spice".

The default settings are:

- For HSPICE:
extsim_cmd.c
 - For Spectre:

To run Spectre in APS mode, set the `-run -` option:

This variable must be used before the `char_library` command

extsim_constraint_option

"options" Specifies the list of options to be used by the external_simulator for constraint characterization (setup, hold, mpw). The options string is passed as a `.option` line in the SPICE decks that Liberate creates for constraint characterization.

- If `extsim_constraint_option` is not defined, the default is `extsim_option`.
- If `extsim_mpw_option` is not defined, the default is `extsim_constraint_option`.
- If neither `extsim_constraint_option` nor `extsim_mpw_option` is defined, the default for both the parameters is `extsim_option`.

This variable must be used before the `char_library` command.

extsim_deck_dir

<directory_name> Specifies a directory in which to save all of the SPICE decks created when using an external simulator for characterization.
Default: follows `$TMPDIR/decks.$PID`

A tar'ed and gzipped file is written for each cell named `<cellname>.tgz` in the named directory. The directory must be visible to all of the server and client machines being used. If the directory does not exist, it is created.

All of the decks are written to a sub-directory based on cell names within the named directory. The files in the directory use the following naming convention:

`<cell>_<pin>_<dir>_<relPin>_<relPinDir>_<type>_<iter#>.sp`

SPICE decks are only saved when the `char_library -extsim` option is enabled.

A file called `map.1st` is included for each cell which gives a mapping of the simulation for each saved spice deck. Every line in `map.1st` represents one simulation setup. SPICE decks used only to validate vectors are indicated by a keyword `CHK` at the end of the entry. These show failure when the vector is false. In this file, you can see:

1. Simulation status (PASS/FAIL)
2. Sim deck directory

3. Pin, related-pin and pin toggle directions
4. Simulation type (combinational, three_state_enable, three_state_disable,...)
5. Transition type (rise/fall_transition, steady_state_current_low/high,...)
6. WHEN condition and pin state vector (e.g. from -vector of define_arc)
7. An optional CHK indicates this is an arc checking simulation

This variable must be used before the `char_library` command.

Example

```
# Set the directory for storing SPICE decks
set_var extsim_deck_dir "/home/char/decks"
char_library -extsim hspice -cells {INVX1 DFFX1}
```

extsim_deck_header

| | |
|------------------|--|
| <i>"options"</i> | Specifies a string of SPICE commands that are used to overwrite the headers of the external SPICE simulation decks. Default: for HSPICE, ".protect" |
| | These options might be superceded during simulation by duplicate options specified in <code>extsim_option</code> , or <code>extsim_leakage_option</code> , or similar options. |

The `extsim_deck_header` variable is available so the external simulator commands can be provided directly to the external simulator without having Liberate process or review them. This variable is intended to be used in a `define_leafcell` (see the option `-extsim`) flow where some models are loaded into `read_spice` and some are not. That is, it is intended to be used to load models directly into the external simulator engine bypassing the Liberate circuit reading code. When you set the scale using `extsim_deck_header`, Liberate does not know the scale.

The preferred method to load the scale is in the model file. If `read_spice` is not reading the model file then use the command "`define_leafcell -scale <value> ...`" to tell Liberate what the scale is to apply to the netlist.

Note: With some simulators, this variable defaults to "\n.n.protect". If you encounter any simulation issues caused by the presence of this default value, use the following command to override the default:

```
set_var extsim_deck_header ""
```

This variable must be used before the `char_library` command.

extsim_deck_style

| | |
|--------------------|---|
| <merge separate> | Controls whether the netlist and models from the external simulator SPICE decks are saved into a separate file. Default: merge |
| merge | Keeps the netlist and models in-line. |
| separate | Separates the netlist and the models from the external simulator SPICE decks. The netlist and models are saved into a separate file and loaded using the <code>.include</code> SPICE command. |

This variable must be used before the `char_library` command.

extsim_exclusive

| | |
|---------|--|
| <0 1> | Controls the SPICE engine used for pre-simulation. Default and recommended: 1 |
| 0 | Uses Alspice for pre-simulation measurements. |
| 1 | Uses the external simulation engine for all simulations when <code>char_library -extsim</code> is specified. |

This variable is intended for use in cases where Liberate Alspice cannot support the process model, netlists, or both (such as when encrypted netlists or models, or Verilog-A models are used). In these cases, the `extsim_model_include` variable and the `define_leafcell` command should first be set properly. Note that using the `define_leafcell` command sets `extsim_exclusive` to 1. In cases where the netlist cannot be processed, the `extsim_flatten_netlist` variable might also need to be reset to 0.

This variable must be specified before the `char_library` command.

extsim_flatten_netlist

| | |
|--------------|--|
| <-1 0 1> | Only applies when extsim_model_include is used Default: -1 (disabled) |
| -1 | Flattens the cell netlist file but loads the model file directly with a .inc SPICE command when extsim_model_include is used and define_leafcell command exists. |
| 0 | Forces Liberate to use .inc SPICE command to load cell netlist when define_leafcell command exists. |

Example

Following examples shows how the combination of **extsim_model_include**/**extsim_flatten_netlist**/**define_leafcell** changes the way that liberate composes SPICE simulation deck:

■ Case 1:

```
char.tcl
set_var extsim_model_include "/path/model.sp"
sim.sp
.inc '/path/cell.sp'
...
.inc '/path/model.sp'
```

■ Case 2:

```
char.tcl
set_var extsim_model_include "/path/model.sp"
define_leafcell -type nmos -pin_position {0 1 2 3} nch
define_leafcell -type pmos -pin_position {0 1 2 3} pch
sim.sp
XMMPO_0 MP0:DRN MP0:GATE MP1:SRC VDD pch L=4.2e-08 W=6.4e-07
XMMN0_0 MN0:DRN MN0:GATE MN1:SRC VSS nch L=4.0e-08 W=5.9e-07
...
.inc '/path/model.sp'
```

■ Case 3:

```
char.tcl
set_var extsim_model_include "/path/model.sp"
define_leafcell -type nmos -pin_position {0 1 2 3} nch
define_leafcell -type pmos -pin_position {0 1 2 3} pch
set_var extsim_flatten_netlist 0

sim.sp
.inc '/path/cell.sp'
...
.inc '/path/model.sp'
```

Note: In Case 3 above, the simulation decks for advanced noise models, such as CCSN, does not include the original netlist. Instead, a flattened netlist is always used. This is because these noise models require simulation of individual Channel Connected Regions (CCRs) on the boundary of the cells and not the complete cell.

This variable must be specified before the `char_library` command.

extsim_immunity_option

`"options"` Specifies the list of options to be used by external SPICE characterization for Noise Immunity Curves (NIC).
Default: `""` (use `"runlvl=3 rmax=25"` for Hspice, `""` for Spectre.)

The `options` string is passed as a `.option` line in the SPICE deck. This option only works with Hspice.

This variable must be used before the `char_library` command.

extsim_leakage_option

`"options"` Specifies the list of options to be used by external SPICE simulator during leakage characterization .
Default: `<see below>`

The `options` string is passed as a `.option` line in the SPICE decks that Liberate creates for leakage characterization.

If `extsim_leakage_option` is not set and `extsim_option` is set, then the value of the `extsim_option` is applied to the leakage decks.

If `extsim_leakage_option` and `extsim_option` are not set, then the leakage simulations uses a Liberate default setting which can change at any time. For Spectre, these default options are:

```
"accurate nomod numdgt=6 measdgt=6 ingold=2 measout=0 gmindc=1e-14 gmin=1e-14  
pivtol=1e-15"
```

It is recommended to set `extsim_leakage_option` explicitly to the desired value to ensure that the leakage simulations uses the desired simulator options.

This variable must be used before the `char_library` command.

`extsim_lic_keep`

`<0 | 1>`

Enables the HSPICE `-cc` mode of operation.

Default: 0

Recommended: Set this variable is 1 if you are using a 2008.09 or newer version of HSPICE. If you are using a version of HSPICE that does not support keeping a license checked out, do not use this variable.

0

Disables the HSPICE `-cc` mode of operation.

1

Enables the HSPICE `-cc` mode of operation. This setting can significantly improve the run time when using HSPICE as the external simulator.

The external simulator must be enabled with the `char_library -extsim` argument.

This variable must be specified before the `char_library` command.

Example

```
set_var extsim_lic_keep 1
```

`extsim_line_length_limit`

`<value>`

Limits the number of characters on a line in a spice deck.

Default: 0 (No limit.)

This variable limits the number of characters on a line in spice deck in order to accommodate the difference in line-length between various simulators. If a line is too long, it will be broken into multiple lines, with the maximum character count as specified. (For example, Hspice has a limit of 1024 chars.)

Setting this to 0 means no limit.

This variable must be specified before the `char_library` command.

extsim_model_include

`<value>` Specifies a full path to a file that loads the SPICE models when using an external SPICE simulator. An error will result if a full path is *not* provided. (See error messages below.)
Default: Uses flattened models in the external simulation input decks

When this variable is used, Liberate uses the file specified instead of the flattened models in the external simulation input deck. Liberate places a statement in the extsim SPICE decks such as:

```
.include <extsim_model_include_file>
```

This variable must be used before the `char_library` command.

Example

```
set_var extsim_model_include "/home/user1/models/include_ff"
...
char_library -extsim hspice
```

Where `include_ff` contains:

```
.include '/home/user1/models/models.l' ff
```

Error Messages

■ **File does not exist**

Error : The `extsim_model_include` file `<filename>` cannot be found! Check the variable for errors and rerun. Liberate will now exit.

■ **File is not readable**

Error : The `extsim_model_include` file `<filename>` could not be read! Check the variable for errors and rerun. Liberate will now exit.

■ **extsim_model_include** file is not a full path

Warning (set_var) : The extsim_model_include value should include a full path. If the external simulator cannot locate the model file, it will terminate and will cause Liberate to issue errors about failed simulations. Add the full path and rerun.

extsim_model_include_leakage

`<"model_file">` Specifies full path to model file. Used with HSPICE.
Default "" (none)

Set this to characterize leakage with a separate process model file. This will **.include** the leakage model into the leakage decks. (This will not affect the normal read_spice flow on the regular process models.) The leakage decks written out will not flatten the netlist.

This variable must be used before the `char_library` command.

Example

```
set_var extsim_model_include_leakage "/home/myDir/model.inc"
```

extsim_model_include_mode

`<0 | 1 | 2>` Instructs the tool to add a $1e-5$ Ohm resistor to all internal cell nodes.
Default and Recommended: 2

| | |
|---|--|
| 0 | Adds a $1e-5$ Ohm resistor to all internal nodes of a cell. Uses the behavior of 2.4p1 and earlier releases of the tool. |
| 1 | Does not add a resistor for leakage. The <code>.nodeset</code> and <code>.ic</code> statements in the extsim SPICE decks for <i>leakage</i> reference internal nodes directly instead of through a $1e-5$ Ohm resistor. |
| 2 | The <code>.nodeset</code> and <code>.ic</code> statements in <i>all</i> extsim SPICE decks reference internal nodes directly. |

To assist the extsim spice engine with DC convergence, Liberate adds a 1e-5 Ohm (10 micro-ohm) resistor to internal nodes of a cell. With some versions of spice, this can cause DC convergence issues with leakage simulations.

Note: When using Spectre Kernel Interface (SKI) with `extsim_model_include` and `extsim_flatten_netlist=0`, if `extsim_model_include_mode` is not 2, Liberate issues a warning and automatically set this variable to 2.

This variable must be used before the `char_library` command.

`extsim_monitor_deck_dir`

`<"directory_name">` Create a directory to store the SPICE decks affected by the `extsim_monitor` script.
Default: follows `${extsim_deck_dir} _monitor`

This variable specifies a directory to write all the SPICE decks affected by the `extsim_monitor` script when using an external simulator. The directory must be visible to all server and client machines. A `sim README` file is created in each simulation directory to help trouble-shoot why this simulation was affected. See also: `extsim_monitor_enable` and `extsim_monitor_timeout`

This variable must be used before the `char_library` command.

Example

```
# Set the directory for storing extsim monitor SPICE decks
set_var extsim_monitor_deck_dir "/home/char/decks_monitor"
```

extsim monitor enable

<0 | 1> Enables an external simulator monitoring script that can detect if the SPICE simulation is hanging, and will take corrective action. If necessary, Liberate will re-submit a simulation upto "retry_count" times before skipping the job. The only external simulator currently supported for monitoring is HSPICE.

Default and recommended: 0 (disabled)

Note: Only certain versions of HSPICE have been observed to hang indefinitely. Many of these hanging conditions have been resolved by updating the version of HSPICE or the process models.

0 Disables the external simulator monitoring script.

- 1 Enables the external simulator monitoring script. If it is necessary to enable this functionality to work around hanging conditions with an external simulator, we recommend you update the simulator version.

Important

The actual script is "extsim_monitor.sh" in the \$ALTOSHOMEDIR/bin directory, but we do not recommend that users modify this script.

See also: `extsim monitor deck dir` and `extsim monitor timeout`

This variable must be used before the `char` library command.

extsim monitor timeout

<time> Specifies the time in seconds that an extsim process is inactive before action is taken by the `extsim_monitor.sh` script. Default: 1800 (30 minutes)

This variable is used to enable recovery from hanging external SPICE processes during characterization. It controls how long the `extsim_monitor.sh` script should wait for an update from the simulator prior to taking action. If the external simulator process is

terminated, then Liberate will retry this simulation "retry_count" times before skipping this simulation. At the end of the characterization run any cells that are not fully characterized will be reported.

Any affected simulation decks will be copied to `extsim_monitor_deck_dir`. The variable `extsim_monitor_enable` must be set in order for this variable to have any effect.

The default value is 1800 (30 minutes). For most standard-cell library applications where single simulations complete in less than 5 minutes, the recommended value is 300 (5 minutes). For complicated or large cells with long simulation times, it may be necessary to increase this value to 3600 or larger.

See also: [extsim_monitor_enable](#) and [extsim_monitor_deck_dir](#)

This variable must be used before the `char_library` command.

Example

```
# Set the extsim monitor timeout to 5 minutes
set_var extsim_monitor_timeout 300
```

`extsim_mpw_option`

| | |
|--------------------------------|--|
| <code><"options"></code> | Specifies the options to be used for MPW characterization with external SPICE simulator. |
|--------------------------------|--|

This variable specifies the list of options to be used by the `external_simulator` characterization for `mpw`. The options string is passed as a `.option` line in the SPICE decks that Liberate creates for constraint characterization.

- If `extsim_constraint_option` is not defined, the default is `extsim_option`.
- If `extsim_mpw_option` is not defined, the default is `extsim_constraint_option`.
- If neither `extsim_constraint_option` nor `extsim_mpw_option` is defined, the default for both the parameters is `extsim_option`.

This variable must be used before the `char_library` command.

`extsim_node_name_prefix`

| | |
|-----------------------------|--|
| <code><string></code> | Specifies the generated node prefix string. Default: "altos_" |
|-----------------------------|--|

Use this variable to specify a string that is used as the prefix of all Liberate generated node names when simulation decks are written out using the original node names (see `extsim_use_node_name=1`).

The prefix is used to avoid conflicts between node names generated by Liberate and those in the cell netlist, which could occur if alpha-numeric node names are used.

This variable must be used before the `char_library` command.

extsim_option

"options" Specifies the list of options to be used by external SPICE characterization for delay, power, or timing constraint characterization.
Default: for HSPICE, "`runlvl=5`"; and for Spectre, "`save=none`"

The `options` string is passed as a `.option` line in the external SPICE decks Liberate creates for characterization.

Note: The last `extsim_option` overwrites the previous setting.

This variable must be used before the `char_library` command.

Example

```
# Set the .options for external SPICE, leakage and CCS
set_var extsim_option "runlvl=5"
set_var extsim_leakage_option "gmindc=1e-14 pivtol=1e-15"
set_var extsim_immunity_option "runlvl=4 rmax=24"
```

extsim_option_presim

"options" Options to be used for characterization with external SPICE.
Default: " " (none)

When checking a UDA (user-defined arc) with an external simulator (`char_library - extsim`), liberate creates a deck used for checking if the arc can be simulated. This deck is called a "CHK" deck and uses faster simulator option settings. Sometimes, different options should be used to help DC convergence. Use this variable to specify simulator options to be used during the presim stage.

This variable must be used before the `char_library` command.

Example

```
set_var extsim_option_presim "ITL1=300"
```

extsim_reuse_ic

| | |
|-----------------|--|
| <0 1 2 3> | Reuses DC solution for a group of transient simulations when Spectre is the external simulator. Default: 3 |
| 0 | Do not reuse DC solutions during <code>.alter</code> simulations. |
| 1 | Reuses DC solution by adding <code>write/readns</code> to the <code>.tran</code> statement. |
| 2 | Reuses DC solution by adding <code>restart=no</code> option to the <code>.tran</code> statement. This allows the previous solution to be used as the IC for <code>.alter</code> simulations. However, you may not want this because if you run two subsequent transients, then the last timepoint of the first transient would be the IC for the second transient. Therefore, this setting is not recommended. |
| 3 | Reuses DC solution by adding <code>skipdc=useprevic</code> option to the <code>.tran</code> statement. (Recommended, and the only supported option for Spectre-SKI.) (Default) |



This variable is supported only in Spectre.

The `extsim_reuse_ic` variable allows for the reuse of the first DC solution in a group of `.alters`, which can significantly speed up simulations for large cells. (See the Spectre manual for a full description of all Spectre options.)

This variable must be used before the `char_library` command.

extsim_sanitize_param_name

| | |
|---------|---|
| <0 1> | Enables sanitizing of node names in the output SPICE deck. Default: 1 |
| 0 | No sanitization (cleaning) is done. This may lead to an early termination of the simulation because of inconsistency of node names. |
| 1 | Liberate will sanitize the name before using it in .param, .data, etc. sections of the output netlist for simulation. |

This variable enables Liberate to sanitize a net/port name that uses characters that could be interpreted incorrectly as an equation when using an external simulator. (Characters such as: \, <, >, etc.)

This variable must be used before the `char_library` command.

extsim_save_failed

| | |
|---|---|
| <none deck all log all_plus_em_reports> | Saves the SPICE decks for failing simulations. Default: all |
| none | |
| deck | Saves only the input SPICE deck. |
| all | Saves both the input deck and the output listings. |
| log | Saves only the input deck and output log file from the simulator. The data is saved in the directory defined by the <code>extsim_deck_dir</code> variable. |
| all_plus_em_reports | Enables saving of simulation related files for ElectroMigration (EM) related characterization. The SPICE decks, <code>emir.conf</code> , <code>log</code> , <code>sim.measure</code> , <code>sim.print</code> , and EM report files for the EM simulations will be saved. |

This parameter is used to save SPICE decks and listings for failed simulations.

Note: Output SPICE decks are only saved when the `char_library -extsim` option is enabled.

This variable must be used before the `char_library` command.

extsim_save_passed

`<none | deck | all | log | all_plus_em_reports>`

Saves the SPICE decks for passing simulations.

Default: `none`

`none`

`deck` Saves only the input SPICE deck.

`all` Saves both the input deck and the output listings.

`log` Saves only the input deck and output log file from the simulator.

`all_plus_em_reports`

Enables saving of simulation-related files for electromigration (EM) related characterization. The SPICE decks, `emir.conf`, `log`, `sim.measure`, `sim.print`, and EM report files for the EM simulations will be saved.

This parameter is used to save SPICE decks and listings for successful simulations.

As the number of simulation decks is very large, it is advisable to only use this setting when characterizing a small number of cells. The data is saved in the directory defined by the `extsim_deck_dir` variable.

Note: Output SPICE decks are only saved when the `char_library -extsim` option is enabled.

This variable must be used before the `char_library` command.

extsim_save_verify

| | |
|-------------|--|
| <0 1 2> | Generates additional SPICE decks to verify setup, hold, recovery and/or removal. Default: 0 |
| 0 | Disables generation of additional SPICE decks. |
| 1 | Enables the creation of SPICE decks that can be used to verify the library characterization without actually running the characterization. In addition to the non-constraint-related SPICE decks, special SPICE decks will be output for the constraints of setup, hold, recovery or removal timing constraints. These constraint verification SPICE decks will utilize a sweep search in SPICE. The additional decks are placed in the appropriate sub-directory within the directory defined by the <u>extsim_deck_dir</u> variable and have a "_sweep.sp" suffix. |
| 2 | <p>Note: Minimum pulse width SPICE decks will not be created.</p> <p>Sets <u>extsim_save_passed</u> always to <u>deck</u>, and an additional 'verify' deck containing offset params on both data and clock signals will be generated based on the characterization's final results (setup/hold/recovery/removal values). By adjusting the included SPICE parameter offsets, the user can also regenerate the nominal (undegraded) value as well. The operation of <u>extsim_save_verify=2</u> is equivalent to the operation of <u>extsim_save_verify=1</u> except that it uses the final constraint results instead of a brute force sweep. This mode will also generate MPW decks. Note that this requires that all simulations will still be performed if <u>extsim_save_verify</u> is set to 2.</p> |

When `extsim_save_verify` is set to 2 along with EM characterizations, an extra EM verification deck is generated for every EM arc. This has a spice deck, em config file, and a run script that can be executed to verify the em results. There is one deck for every data type (avg, rms, and peak) with the input toggles reflecting the toggle rates computed. Since the EM toggle rates may be theoretical some of the EM reports may require further processing and indirect verification. There will be comments before the alter blocks such as:

```
* Entry:2 fail+119.0*r203 - Direct Verification
.alter
.param i_t0=0.0000000e+00
.param i_v0=0.0000000e+00
```

```
* Entry:3 pass-49.5*r203 - Indirect Verification
.alter
.param i_t0=0.0000000e+00
.param i_v0=0.0000000e+00
```

The above comments denote which results can be directly looked up from the reports and which one is needed to be numerically computed.

Note: All peak toggle rates have to be indirectly verified because of the min duty based formulas involved in computing the toggle rates.

Output SPICE decks are only saved when the `char_library -extsim` option is enabled.

This variable must be used before the `char_library` command.

Example

```
set_var extsim_deck_dir      ${rundir}/spice_decks
set_var extsim_save_passed   deck
set_var extsim_save_failed   all
set_var extsim_save_verify   1
```

extsim_tar_cmd

| | |
|-----------------------|---|
| <code>"string"</code> | Enables to tar the SPICE decks. Default: "tar zcf" |
|-----------------------|---|

This parameter is used to specify the command used to compress the output SPICE decks. Set this parameter to the NULL string ("") to disable compression.

This variable must be used before the `char_library` command.

Example

```
# Disable SPICE deck compression
set_var extsim_tar_cmd ""
```

extsim_timestep

<value> Sets the time step to use for external SPICE simulation.
Default: `1e-12` seconds (1ps)

This variable must be used before the `char_library` command.

Example

```
# Set the time step for external SPICE
set_var extsim_timestep 2e-12
```

extsim_tran_append

<options> Adds additional options to the `.tran` statement.
Default: " " (none)

This variable must be used before the `char_library` command.

Example

```
# Set conservative mode for Spectre
set_var extsim_tran_append "errpreset=conservative"
```

extsim_use_node_name

<0 | 1> Maps node names to numbers in extsim SPICE decks.
Default and recommended: 1

0 Specifies to map the node names to numbers in extsim SPICE decks.

| | |
|---|---|
| 1 | Specifies that if a port name has an escape character "\", Liberate should remove this character and replace it by '_' by default because SPICE simulators (HSPIC, Spectre) do not support this character in voltage/current source commands. |
|---|---|

This variable must be used before the `char_library` command.

floating_channel_bias

| | |
|----------------------------|--|
| <code><value></code> | Specifies to bias the initial conditions for floating channel nodes in SPICE decks, where, $-0.5 \leq \text{value} \leq 1.5$. Default: 0.5 |
| 0 | Sets the initial condition for floating channel nets to the expected final state for the net. |
| 1 | Sets the initial condition for floating channel nets to opposite rather than the expected final state. |

Use this variable to bias the initial conditions for floating channel nodes in Spice decks when enabled by setting the variable `floating_channel_mode` to 3. Liberate attempts to determine the expected final state for each floating channel node. A value of 0 will bias the initial condition to an expected final state. A value of 1 will bias the initial condition to the state opposite from the expected final state. Any other value will bias the initial condition between the expected final state and its opposite value.

This variable must be used before the `char_library` command.

floating_channel_mode

| | |
|------------------------------------|---|
| <code><0 1 2 3></code> | Enables various algorithms to initialize floating channel nodes. Default: 1 |
| 0 | Initialize floating channel nodes if they drive the gate of a loading transistor. |
| 1 | Never initializes floating channel nodes and leaves it to the simulator to determine the initial condition. (Default and recommended) |

Virtuoso Liberate Reference Manual

Liberate Parameters

- | | |
|---|---|
| 2 | Always initializes floating channel nodes to: $\text{gnd} + 0.5 * (\text{vdd} - \text{gnd}) = 0.5 * (\text{vdd} + \text{gnd})$. |
| 3 | Always initializes floating channel nodes based on the setting of the <u>floating_channel_bias</u> variable. |

Liberate provides initial conditions to nodes that are on the active path. Nodes that are not on the active path are not initialized. It is up to the circuit simulator to provide a DC solution for these nodes. This variable can be used to enable various algorithms to initialize floating channel nodes.

This variable must be used before the `char_library` command.

floating_node_initialize_mode

- | | |
|-------------|---|
| <0 1 2> | Specifies the initialization method for floating nodes. Default and recommended: 0 |
| 0 | Adds $1/\text{gmin}$ gshunt resistance on all floating nodes to a ground supply node. |
| 1 | Adds a <code>.ic</code> (initial condition) set to 0V on all floating nodes. |
| 2 | Adds a <code>.NODESET</code> set to 0V on all floating nodes |

Liberate automatically recognizes floating nodes and adds `.ic` to these nodes to make the simulation results consistent across different simulators. A change to the initialization method may result in different simulation results.

This variable must be used before the `char_library` command.

force_avg_default_select_order

- | | |
|---------|---|
| <0 1> | Enforces a pre-determined order for selecting average default groups. Default: 1 |
| 0 | Liberate will use its internal ordering to decide the default group. |

| | |
|---|---|
| 1 | Enforces a deterministic ordering. (Default and Recommended.) |
|---|---|

When the criteria for selecting the default group is set to the average, Liberate will select the group closest to the statistical median value as the default group. This variable is needed to resolve differences in default groups for cells with exactly two states.

This variable must be used before the `write_library` command.

force_condition

<0 | 1 | 2 | 3 | 4> Controls whether to include conditional `when` and `sdf_cond` attributes for single or binate timing groups.

Default: 1

| | |
|---|---|
| 0 | Disables the <code>when</code> / <code>sdf_cond</code> condition for arcs with a single or binate (an arc with a pair of <code>positive_unate</code> and <code>negative_unate</code> timing groups) timing group. With many sequential cells, the clock to Q path may contain many states. Some of these states may contain only rise or fall arcs while others contain both rise and fall arcs. Liberate models this arc as a single worst case unconditional arc. |
| 1 | Output <code>when</code> for binate arcs (example: XOR) or constraint arcs; this is an aid to Verilog. (Default) |
| 2 | In addition to mode 1, always output <code>when</code> , for example, for AND gate. |
| 3 | In addition to mode 1, provide state dependent clock->output arcs even when they are non-symmetrical between rise and fall. |
| 4 | In addition to 3, an arc with <code>SE*SI</code> rise and <code>when SE*!SI</code> fall is merged into a single timing group with both rise, fall, and <code>when SE</code> . |

This variable must be used before the `char_library` command.

Example

```
# Turn off conditions on single timing groups
set_var force_condition 0
```

force_default_group

| | |
|---------|---|
| <0 1> | Controls whether to include a default group. Default: 0 |
| 0 | Specifies that Liberate should not always output a default group. |
| 1 | Requests Liberate to always output a default group, even if the group is redundant. This can occur when all states are exhaustively enumerated. |

This variable must be used before the `char_library` command.

Example

```
# Require output of default group
set_var force_default_group 1
```

force_edge_timing_type

| | |
|-------------|--|
| <0 1 2> | Controls the <code>timing_sense</code> and <code>timing_type</code> of single-sided edge timing arcs. Default: 1 |
| 0 | Convert <code>rising_edge</code> and <code>falling_edge</code> to <code>combinational_rise</code> and <code>combinational_fall</code> for single sided edge-triggered timing() groups. An appropriate <code>timing_sense</code> (<code>negative_unate</code> and <code>positive_unate</code>) will be output instead of <code>non_unate</code> . |

- | | |
|---|---|
| 1 | Do not change the <code>timing_type</code> of <code>rising_edge</code> and <code>falling_edge</code> for single sided edge triggered arcs. If appropriate, change the <code>timing_sense</code> to <code>negative_unate</code> and <code>positive_unate</code> from <code>non_unate</code> . This is the default and recommended setting. |
| 2 | Convert the <code>timing_type</code> for single sided timing arcs where the <code>related_pin</code> is a clock from <code>combinational_rise</code> and <code>combinational_fall</code> to <code>rising_edge</code> and <code>falling_edge</code> . |

Use this variable to request specific handling of single sided edge triggered timing arcs. Normally, Liberate only outputs timing() groups with an edge timing types (`rising_edge` or `falling_edge`) where both rising and falling data exist in a single group.

If the `define_arc` command is specified in the template, the `timing_type` is not modified.

The following variables (there might be others) might also affect the `timing_type`/`timing_sense` of characterized arcs: `combinational_risefall`, `nonseq_as_recrem`, `merge_related_preset_clear`, and `discard_timing_sense_after_merge`.

This variable must be used before the `char_library` command.

force_leakage_if_no_pg_pin

- | | |
|----------------------------|--|
| <code><0 1></code> | Enables output of leakage when no <code>related_pg_pin</code> is available. Default: 1 |
| 0 | Liberate reverts back to pre-3.0 behavior where leakage groups will not be output if there is no <code>related_pg_pin</code> . |
| 1 | Requests Liberate to force cells, such as antenna cells that do not have related pg pins for leakage power, to generate a <code>leakage_power_group</code> whenever <code>pin_based_leakage</code> is set. |



The `force_leakage_if_no_pg_pin` variable only works with `pin_based_leakage` and it will only affect cells without related pg pins for leakage power.

To be compliant with LC, the `cell_leakage_power` for that cell is output to the library, without regard for the value of the `cell_leakage_power` variable. The default leakage power for that cell will not be output to the library, without regard for the value of the `keep_default_leakage` variable.

This variable must be used before the `write_library` command.

force_related_power_pin

`<0 | 1 | 2>` Forces the output of `related_power_pin` groups for cells.
Default and recommended: 2

If there is no related pg pin found in the netlist of a cell, this variable controls whether to force to output the related power pin or signal level to the default vdd/gnd or the vdd/gnd set by `set_pin_vdd`/`set_pin_gnd`.

- | | |
|---|---|
| 0 | Do not force the output of <code>related_power_pin</code> data. |
| 1 | Forces this behavior only for cells that have more than one pin (Antenna cells are not forced). |
| 2 | Forces this behavior for every cell. |

This variable must be used before the `write_library` command.

force_unconnected_pg_pin

`<0 | 1>` Controls whether `pg_pin` information will be generated for unconnected power supplies.
Default: 0 (no `pg_pin` for supplies with no connections)

- | | |
|---|---|
| 0 | <code>pg_pin</code> information for unconnected power supplies will <i>not</i> be output. |
| 1 | <code>pg_pin</code> information for unconnected power supplies will be output. |

The `set_vdd` and `set_gnd` commands specify the power supplies. Normally, if a subcircuit definition for a cell is not connected to a specified supply, that supply will not be included in the `pg_pin` group for that cell. Set this variable to force all specified supplies to be output in `pg_pin` groups.

This variable must be used before the `char_library` command.

group_attribute

`<"attribute_name">` Creates a cell group based on the named attribute.
Default: " " (if not set, then the library attribute
`cell_footprint` is used)

A cell group may be created explicitly by the `define_group` command, or implicitly via cells having a common cell level attribute as specified by this variable. Cell grouping works with a `read_library` or `write_template` flow, `compare_library -group`, and `compare_structure` (only available in Liberate LV). Cells that have the same value for a selected attribute are grouped together (for example, "cell_footprint").

Example

```
set_var group_attribute "cell_footprint"
```

heartbeat_initial_timeout

`<time>` The time in seconds that the server will wait for the first client to communicate back to the server.
Default: 3600 (1 hour)

This variable is used to specify a time limit to wait for a client process to communicate with the master Liberate jobs. When this timeout is exceeded, the Liberate server will issue a warning that the client has failed to start and then restart the `heartbeat_initial_timeout` timer. This situation could occur, for example, due to network problems.

This variable must be used before the `char_library` command.

Example

```
# Set the heartbeat initial timeout to 2 hours
set_var heartbeat_initial_timeout 7200
```

heartbeat_timeout

<time> The time in seconds that a client machine is inactive before being released by the server machine.
Default: 300 (5 minutes)

This variable is used to enable recovery from machine failures during distributed characterization. It controls how long the server machine should wait for a response from a client before releasing that client. If a client hangs it will not be used for the remainder of the characterization run. In addition, the task (a collection of arc simulations) being performed by the failing client is re-submitted to another client. If the re-submission of this task causes another client to hang then this task is skipped. At the end of the characterization run any cells that are not fully characterized will be reported.

This variable must be used before the `char_library` command.

Example

```
# Set the heartbeat timeout to 10 minutes
set_var heartbeat_timeout 600
```

hidden_power

<0 | 1> Enables conditional hidden power calculations for all cells.
Default: 1

This variable is used to enable or disable "hidden" power calculations for all cells, including input pins of combinatorial gates. When this parameter is disabled hidden power is only calculated for hidden input pins (i.e. pins that have no path to the output such as the D pin of a flip-flop). When `hidden_power` is enabled, conditional hidden power will be calculated for all pins that have can have a logic state that does not toggle an output node - for example, a 2 input NAND gate where one of the side inputs is zero.

This variable must be used before the `char_library` command.

Example

```
# Disable hidden power calculations
set_var hidden_power 0
```

ibis_compensate_odt

<0 | 1> Enables compensation calculation for weak pullup/pulldown On-Die Termination (ODT) I/O cells.
Default: 1 (enabled)

If `ibis_compensate_odt` is set to 1 (default), then Liberate will calculate the PAD pin current `lg` at `VSS`, and current `lp` at `VDD` (from total clamp IV data) and compare them to the `ibis_odt_min_current` variable (Default: `2e-5`; units in mA). If one of the two calculated/measured currents (`lp` or `lg`) is larger than `ibis_odt_min_current`, Liberate re-calculates the power clamp and GND clamp I-V table to remove (compensate) the ODT or weak pull-up/down current (which is already included in pullup/pulldown I-V table.)

See also: [ibis_odt_min_current](#)

This variable must be used before the `char_library` command.

ibis_has_weak_hold

<0 | 1> Specify to request .NODESET instead of .IC in extsim decks for output pin initial condition.
Default: 1

When preparing IBIS simulations, Liberate will use the .IC spice command to initialize Hi-Z pins in the spice deck. If set to 1, in IBIS mode (see `char_library -ibis`) Liberate will use .NODESET instead of .IC in the spice deck.

The initial voltage on a node will be set to a supply voltage. When the output does not swing full rail, this voltage might be incorrect which can cause the spice engine difficulty in finding a DC solution. It is advisable to set this variable when the spice engine complains or takes too long finding a DC solution and the user knows the holding strength on the output pin is weak (no full vdd/vss swing).

This variable must be used before the `char_library` command.

ibis_iv_max_step_factor

<value> Specify the maximum voltage step in the filtering algorithm as a factor of `ibis_iv_step`. Default: 1.5

This variable is used to control the maximum voltage step in the filtering algorithm when writing out I-V tables. IBIS contains up to 3 process corners each with a set of waveforms (I-V table) that have to be combined into a data table with the same rows (the V for I-V table). Also, the total number of rows are limited by different IBIS versions (for example: 100 voltage rows/steps for the I-V table. Liberate needs to reduce up to 1000 rows into 100 rows).

Liberate uses a proprietary algorithm to retain the best waveform points globally. By default, Liberate will use $1.5 * 0.05 = 0.075$ mV (ibis_iv_max_step_factor * ibis_iv_step) to guarantee there should be at least one data point for every such voltage range.

This variable must be used before the `char_library` command.

ibis_iv_mode

`<0 | 1>` Enables differential-pin IBIS characterization.
Default: 0 (Preserve previous behavior)

Set to 1 to enable input differential-pin IBIS characterization; not needed for non-differential input pin model.

This variable must be used before the `char_library` command.

ibis_iv_step

`<value>` Specifies the IBIS voltage step.
Default: 0.05v

Sets the IV voltage step used in the IBIS file when characterizing I-V data such as power_clamp, gnd_clamp and pull_up, pull_down table.

This variable must be used before the `char_library` command.

Example

```
# Set the acceptable glitch output peak to 10%
set_var ibis_iv_step 0.1
```

ibis_odt_min_current

`<value>` User-defined threshold for IBIS weak pullup/pulldown compensation calculation.
Default: 2e-5 mA (20 nA)

See [ibis_compensate_odt](#) for full explanation.

This variable must be used before the `char_library` command.

ibis_sim_duration

`<value>` The IBIS simulation duration.
Default: 20e-9

Set the simulation duration for the IBIS related simulations. Use the `ibis_sim_duration` and `ibis_tend_factor` (default 1.8) variables to make the simulation end time consistent in all 3 IBIS corners (typ/min/max).

This variable must be used before the `char_library` command.

Example

```
# Set the acceptable glitch output peak to 10%
set_var ibis_sim_duration 30e-9
```

ibis_t2b_cmd

`"string"` Command string to be used to change the t2b executable.
Default: t2b

This variable can be used to override the default command used by Liberate and call the external executable. For example, if user wrote a wrapper script for t2b with the path name of `/bin/t2bwrapper.exe`, then setting this parameter to `set_var ibis_t2b_cmd "/bin/t2bwrapper.exe"` overwrites the default setting.

ibis_tend_factor

<value> The IBIS transient simulation factor.
Default: 1.8

Set the IBIS transient simulation end time which is to be multiplied by the original Liberate simulation end time estimation. The final end time will be compared to the end time specified by `ibis_sim_duration` (if greater than 0) and take the larger value to use for the simulation end time.

The `ibis_tend_factor` and `ibis_sim_duration` variables exist to make sure Liberate can see the voltage/time waveform tail. The default Liberate simulation end time estimation might not be long enough for some analog IO pad cells.

This variable must be used before the `char_library` command.

ibis_vt_max_num_pts

<value> Specifies the maximum number of V-t time points.
Default: 1000

Set this variable to the maximum number of time points that will be output in a V-t table in an IBIS model. The value must be ≥ 40 . If the simulation output has fewer data points than the value set by this variable, then all of the available points will be printed. If the simulation output has more points than the value set by this variable, the data will be reduced to have the specified number of points. The value specified should be greater than or equal to the `ibis_vt_min_num_pts` value.

This variable should be used before the `write_ibis_file` command.

Example

```
set_var ibis_vt_max_num_pts 500
set cell "BC335"
set file "bc3350.ibs"
set ldbs { ibis_BC335_typ.ldb.gz ibis_BC335_min.ldb.gz ibis_BC335_max.ldb.gz }
write_ibis_file $file $ldbs $cell
```

The generated rising/falling V-t waveform will have up to 500 time points written in the IBIS file as long as the original V-t data has more than 2 time points stored in the ldb.

ibis_vt_min_num_pts

<value> Specifies the minimum number of V-t time points.
Default: 40

Set this variable to the minimum number of time points that will be output in a V-t table in an IBIS model. The value must be ≥ 2 . If the simulation output has fewer data points than the value set by this variable, then Liberate will use linear interpolation to add data points at a sparse area of the waveform curve. The value specified should be less than or equal to the `ibis_vt_max_num_pts` value.

This variable should be used before the `write_ibis_file` command.

Example

```
set_var ibis_vt_min_num_pts 500
set cell "BC335"
set file "bc3350.ibs"
set ldbs { ibis_BC335_typ.ldb.gz ibis_BC335_min.ldb.gz ibis_BC335_max.ldb.gz }
write_ibis_file $file $ldbs $cell
```

The generated rising/falling V-t waveform will have at least 500 time points written in the IBIS file even if there are fewer V-t time points stored in the ldb.

immunity_glitch_peak

<value> The fraction of the output supply voltage used for characterizing noise immunity curves.
Default: 0.05 (5%)

This parameter is used to set the noise glitch-peak that specifies the failure criteria for the output of the cell when generating noise immunity curves. Default is 5% of the supply voltage.

This variable must be used before the `char_library` command.

Example

```
# Set the acceptable glitch output peak to 10%
set_var immunity_glitch_peak 0.1
```

immunity_noise_skew_ratio

`<value>` Ratio of time to reach noise-peak compared to the noise width.
Default: 0.5

This parameter is used to set the skew of the input noise waveform used for characterization of noise immunity rejection curves. It defines the ratio of the time to reach the noise peak over the noise width. The ratio must be within a range of 0.25 to 0.75, default is 0.5 (the input glitch is represented by an isosceles triangle).

This variable must be used before the `char_library` command.

Example

```
# Set maximum noise peak to occur at 40% of the width
set_var immunity_noise_skew_ratio 0.4
```

init_clock_period_mode

`<0 | 1 | 2>` Specifies initialization sequence for obtaining initial node voltages for simulation.
Default: 0

| | |
|---|---|
| 0 | Apply <code>init_constraint_period</code> as an input pulse to the <code>related_pin</code> and other side pins and capture the initial node voltages. Apply these initial node voltages to subsequent simulations for this vector. (Default) |
| 1 | Get the initial node voltages from #0 above and if these node voltages do not agree with the Inside View prediction of the circuit behavior, then <code>init_constraint_period</code> is silently disabled for this arc. |
| 2 | Use the behavior as in 1 above, but apply the <code>init_constraint_period</code> only to the <code>related_pin</code> . No side pins will be pulsed during the initialization sequence. (Recommended) |

This variable must be used before the `char_library` command.

init_comb_num_cycles

<integer> Number of times the related_pin will be toggled. Default: 1

This variable lets you specify how many times the related_pin in combinational arcs will be toggled. Allowable values are integers starting from 1.

Note: Only when the `init_comb_related_pin_period` variable is set to a positive number will `init_comb_num_cycles` be used. Otherwise, the related_pin will not be toggled.

This variable must be used before the `char_library` command.

Example

```
set_var init_comb_num_cycles 2
```

init_comb_related_pin_period

<integer> Controls related_pin toggling in combinational arcs.
Default: -1

Setting this variable to a positive number (units in seconds) causes the related_pin in combinational arcs to be toggled. The period of the toggle is set by this variable while the number of toggles is set by the `init_comb_num_cycles` variable.

This variable must be used before the `char_library` command.

Example

```
set_var init_comb_related_pin_period 10e-9
```

init_constraint_period

<value> Clock period (in seconds) used for circuit initialization during constraint characterization.
Default: -1 (do not perform extra initialization)

This parameter is used to specify a clock period (in seconds). A clock pulse with a 50% duty cycle will be used to initialize the circuit before constraint circuit simulation. Default: no initial clock pulse is used.

This variable must be used before the `char_library` command.

Example

```
# Use a clock pulse with a 5ns period for initialization
set_var init_constraint_period 5e-9
```

init_constraint_period_binning_mode

| | |
|---------|---|
| <0 1> | Use <code>init_constraint_period</code> for binning simulation. Default: 1 |
| 0 | Behavior of release 12.1.1 and earlier |
| 1 | Use <code>init_constraint_period</code> for binning simulation. (Default and Recommended) |

This variable must be used before the `char_library` command.

init_constraint_period_check_mode

| | |
|---------|---|
| <0 1> | Controls how vector pre-pulsing is applied. Default: 0 |
| 0 | Check to make sure the entire vector is identical before-and-after pre-pulsing. |
| 1 | Loosens the check to just the probe node. |

When `init_constraint_period` is enabled, Liberate checks to make sure the entire vector is identical before and after the pre-pulsing, otherwise pre-pulsing is disabled.

For certain cells with multiple internal latch nodes, pre-pulsing should only be concerned with the state of the setup/hold probe (usually the first latch node.) This results in setup/hold differences after logic constraints are applied, even though all vectors should give similar setup/hold results. The difference can be attributed to pre-pulsing. A vector that has all 0's or all 1's have a higher likelihood of passing Liberate's pre-pulsing check.

This variable must be used before the `char_library` command.

init_delay_period

<value> Clock period (in seconds) used for circuit initialization during delay characterization.
 Default: -1 (do not perform extra initialization)

This parameter is used to specify a clock period (in seconds). A clock pulse with a 50% duty cycle will be used to initialize sequential cells before delay characterization. Affects clock to output delay arcs only. Default: no initial clock pulse is used.

This variable must be used before the `char_library` command.

Example

```
# Use a clock pulse with a 5ns period for initialization
set_var init_delay_period 5e-9
```

init_pin_hidden_period

<value> Time period (in seconds) used for circuit initialization during hidden arc characterization.
 Default: 10e-9 seconds (10ns).

This variable must be used before the `char_library` command.

init_pin_hidden_num_cycles

<integer> Number of times the `related_pin` will be toggled.
 Default: 1

Use this variable to specify how many times the `related_pin` in hidden power arcs will be toggled when using a transient initialization. This can be useful when using `init_pin_hidden_period` to initialize circuits such as synchronizer circuits that require more than one toggle. The valid values are integers starting from 1.

Note: The `init_pin_hidden_period` variable is used only when the `init_pin_hidden_period` variable is set to a positive number. Otherwise, the `related_pin` will not be toggled.

This variable must be used before the `char_library` command.

Example

```
set_var init_pin_hidden_period 1e-9
set_var init_pin_hidden_num_cycles 2
```

init_pin_hidden_period_mode

| | |
|---------|---|
| <0 1> | Additional effort can be made to ensure that the initialization pulse puts the cell in the correct initial state. Default: 1 (Use additional effort) |
| 0 | Standard method. |
| 1 | Extra effort is used to make certain the cell is properly initialized. (Recommended) |

If `init_pin_hidden_period_mode=1` and either `init_combinational_period>0` or `init_delay_period>0`, then additional effort is made to ensure that the initialization pulse puts the cell in the correct initial state for the measurement. The effect of using this option will be seen primarily on hidden power measurements of clock pins on sequential cells.

This variable must be used before the `char_library` command.

input_noise

| | |
|---------------------|--|
| <dc hyper both> | Specifies SI constructs to write for input pins. Default: hyper |
| dc | Causes <code>input_voltage</code> DC noise level attributes to be written |
| hyper | Outputs noise rejection curves (<code>hyperbolic_noise_low</code> , <code>hyperbolic_noise_high</code>). |
| both | Enables both DC noise and hyperbolic constructs to be written. |

This variable is used by `write_library` to specify the SI constructs to output for input pins (including bi-directional pins). This is in addition to any noise-immunity table that may exist between an output pin and that input pin.

This variable must be used before the `char_library` command.

Example

```
# Output DC noise for each input pin
set_var input_noise dc
```

input_output_voltage

| | |
|-------------|--|
| <0 1 2> | Creates <code>input_voltage</code> and <code>output_voltage</code> groups in the library. Default: 0 (Do not create these groups.) |
| 0 | Do not create <code>input_voltage</code> , <code>output_voltage</code> groups. |
| 1 | Create <code>input_voltage</code> , <code>output_voltage</code> groups and use VDD/GND (rail voltages) for vih,voh and vil,vol. For example, for VDD=1.0 and GND=0.0 with 20% and 80% slew thresholds: |
| | <pre>input_voltage(default_VDD_GND_input) { vil: 0.0 vih: 1.0 vimin: 0.0 vimax: 1.0 }</pre> |
| 2 | Create <code>input_voltage</code> , <code>output_voltage</code> groups and scale values by upper and lower slew thresholds. For example, for VDD=1.0 and GND=0.0 with 20% and 80% slew thresholds: |
| | <pre>input_voltage(default_VDD_GND_input) { vil: 0.2 vih: 0.8 vimin: 0.0 vimax: 1.0 }</pre> |

keep_dcap_leakage

| | |
|---------|--|
| <0 1> | Controls whether to keep the default DCAP leakage group Default: 1 (Keep) |
|---------|--|

Virtuoso Liberate Reference Manual

Liberate Parameters

Set this variable to 0 to not keep the default leakage groups for decap cells. (See also, [keep_default_leakage_group](#).)

This variable can be used after the `char_library` command.

keep_default_leakage_group

<0 | 2> Controls output of default leakage power group.
Default: 0 (Disable output of default `leakage_power` groups.)

0: Disables the output of the `leakage_power` groups.

1: *(Reserved)*

2: Liberate will skip all leakage group processing.

The parameters `keep_dcap_leakage` and `keep_default_leakage_group` are related and only affect cells without pins, such as DCAP cells. They have no affect on cells with one or more pins. For cells without pins, the interaction are as follows:

| keep_dcap_leakage | keep_default_leakage_group | Result |
|-------------------|----------------------------|---|
| 0 | 0 | All leakage groups removed |
| 0 | 2 | Leakage groups unchanged |
| 1 | 0 | voltage_map = 0: All leakage groups removed. voltage_map != 0 or CCSP: Only default leakage group remains. |
| 1 | 2 | Leakage groups unchanged |

This variable can be used after the `char_library` command.

keep_empty_cells

<0 | 1> Determines how to handle a cell when the netlist is empty.
Default: 0 (Treat empty cells as an error condition.)

0 Generates an error condition when an empty cell subcircuit is found. The cell will not be included in the `.lib` file. (Default)

1

Generates a warning condition when an empty cell subcircuit is found. The cell will be included in the .lib file, but will not have any data.

A cell is available for characterization when all of the following conditions are true:

- There is a `define_cell` command and a netlist that has been read in using the `read_spice` command.
- The `char_library` command does not have the `-cells` argument, or the `-cells` argument includes the cell name and the `-exclude` argument is not used.

This variable must be used before the `char_library` command.

keep_user_defined_arc_failed_data

`<0 | 1>` Keep user specified arcs (see `define_arc`) even if they cannot be characterized.
Default: 1 (Do not remove bad data.)

When Liberate is provided with a `define_arc` for hidden power and Liberate determines that the arc is not a valid arc, this control variable can be reset (set to 0) to remove the whole failed user defined hidden power arc from the characterized data (Ldb). Once removed, if needed, Liberate will add the missing data as a `scalar 0` data matrix to make LC pass.

This variable must be used before the `char_library` command.

ldb_checkpoint_dir

`<dir>` Directory where the ldb checkpoint file will be stored.
Default: "." (The initial run directory)

Liberate stores the characterization data in a temporary ldb (library database) file called `altos.ldb.<PID>` where PID is the process ID. This parameter is used to specify the directory where the temporary ldb checkpoint file will be stored. The default directory is the directory where the initial run was started.

This variable must be used before the `char_library` command.

Example

```
set_var ldb_checkpoint_dir /home/work/rundir
```

ldb_precision

<positive_integer>

Specifies the precision used in the ldb.
Default: 6 (See note below.)

This specifies the precision used internally in the ldb.

Note: If this variable is not specified, precision defaults to 6 places.

Precision can only be specified with a positive integer. Examples:

```
set_var ldb_precision 8      # Means "%.8g" or "%.8f".  
set_var ldb_precision 0      # Means "%.0g" or "%.0f"  
set_var ldb_precision 08     # Same as "8"
```

Please note these are invalid settings:

```
set_var ldb_precision abc    # Not an integer  
set_var ldb_precision "2 "   # No white space  
set_var ldb_precision ""     # "(nil)" is illegal  
set_var ldb_precision 1e+2   # No scientific notation  
set_var ldb_precision 1e-2   # - same error -  
set_var ldb_precision 8.0    # No floats (integers only)
```

A warning message will be output if invalid settings are encountered.

This variable must be used before the `char_library` command.

ldb_save_all_cells

<0 | 1>

Controls if cells in the input ldb not being re-characterized are written out into the output ldb.
Default: 1

When set to 1, the `ldb_save_all_cells` variable ensures that cells in the input ldb that aren't being re-characterized are still written out into the output ldb. To get the pre 3.0p2 and prior behavior, set this variable to 0. The recommended setting is 1, which is also the default.

This variable must be used before the `char_library` command.

leakage_add_input_pin

`<0 | 1 | 2>` Include or exclude input pin leakage power.
Default: 1 (Include input pin leakage.)

0: Disables the inclusion of input pin leakage with reported leakage. (Set this to achieve the behavior of releases prior to 3.1, which only reported input leakage when `voltage_map=0`.)

1: Include input pin leakage. (Default.)

2: Account for leakage from bi-directional pins.

This variable must be used before the `char_library` command.

leakage_add_missing_group

`<0 | 1>` Prevents Liberate from adding missing leakage groups.
Default: 1

By default, for level shifters, when the input voltage comes from a voltage domain that is not specified as a `pg_pin`, setting this variable to 0 will stop Liberate from adding the missing leakage groups, thus avoiding an LC error.

This variable must be used before the `char_library` command.

leakage_cell_attribute

`<0 | 1>` Output `cell_leakage_power`.
Default: 1

Set this variable to 0 to omit `cell_leakage_power` attributes from the output library.

This variable can be used after the `char_library` command.

leakage_expand_state

<0 | off | 1 | whens | 2 | vectors>

Selects different algorithms for expanding leakage states.

Default: 0 | off.

0 | off Select one vector per when condition, or per define_leakage command, if specified.

1 | whens Characterize all vectors that match the specified "when". The when is adjusted to include all pins included in the vector. The define_leakage command must be used to specify the desired "when" leakage states.

2 | vectors Characterize all vectors that match the specified "when". The original "when" leakage states will be maintained and the combination (worst, best, or average) of the vectors will be modeled according to the criteria specified in the set_default_group command. If mega-mode is used (see mega_enable or the -type option of define_cell), all bundle or bus pins will be removed from the specified "when" expressions and the characterized vectors will be limited to the bundle combinations specified (see mega_bundle_mode, define_bundle_pins, and define_bus).

This variable must be used before the char_library command.

leakage_float_internal_supply

<0 | 1 | 2> Affects leakage measurements for power switch cells.
Default: 1

Set this variable for increased pessimism of leakage characterization for power switch cells.

0: The following behavior applies for all header and footer cell internal supply pins that are specified using define_cell -internal_supply:

- ❑ If the power switch is on, then leave the internal_supply pin floating.
- ❑ If the power switch is off, then connect the internal_supply pin to a created voltage source connected to the opposite supply voltage. The current through this created voltage source will not be monitored. Note: for header (footer) cells, the opposite state would be ground (vdd).

The recommended setting is 0.

1: The internal_supply pin will always be floating when measuring leakage.

2: The behavior will match the setting of 1 with the exception that the current through the created voltage source will be monitored and for a header(footer) cell, the current will be added to the related ground (vdd) pin. The user must specify the related ground (vdd) pin using the set_pin_gnd (set_pin_vdd) command.

Example:

```
define_cell \
  -input { sleep } \
  -internal_supply { vdda } \
  -delay delay_template_7x7 \
  -power power_template_7x7 \
  header

define_cell \
  -input { sleepn } \
  -internal_supply { vssa } \
  -delay delay_template_7x7 \
  -power power_template_7x7 \
  footer

set_pin_gnd -supply_name vss header vdda $vss
set_pin_vdd -supply_name vdd footer vssa $vdd
```

This variable must be used before the `char_library` command.

leakage_force_tristate_pin

| | |
|----------------------------|---|
| <code><0 1></code> | Set to force tristate pins output <i>cell_leakage_power</i> . Default: 0 |
|----------------------------|---|

Set this variable to 1 to force a tri-stated pin to ground during leakage simulations. This variable can be used to help match legacy libraries. When set to 0 (default), Liberate will allow tristate outputs to float.

This variable must be used before the `char_library` command.

leakage_merge_state

| | |
|-------------|---|
| <0 1 2> | Specifies the leakage merge algorithm. Default: 0 |
| 0 | Outputs a single leakage group for each unique state. The value is selected based on the <code>-criteria { leakage ... }</code> setting of <code>set_default_group</code> . With this setting, if the <code>toggle_leakage_state</code> variable is set to 1, the output pins in sequential cells are dropped from 'when'. |
| 1 | Outputs a single leakage group for each unique state. The value is the minimum leakage from the characterized values. |
| 2 | Outputs a single leakage group for each unique state. The value is the maximum leakage from the characterized values. |

Use this variable to tell Liberate how to select the leakage value to report when there are multiple leakage measurements with the same 'when' state. The output library should have only one leakage for each state. Merging of leakage states might be needed, For example, when there are user-specified `define_leakage` commands whose states do not include all of the cell pins.

This variable must be set before the `char_library` command.

leakage_mode

| | |
|-------------|--|
| <0 1 2> | Set the leakage computation mode. Default: 1 (multiply the leakage current by the supply voltage) |
|-------------|--|

This option will specify the method used when computing the DC leakage to report in the .lib. The reported leakage is computed as follows:

Virtuoso Liberate Reference Manual

Liberate Parameters

If `voltage_map=0`, leakage is reported *without related_pg_pin* as follows:

| leakage_mode | Formula |
|---------------------|--|
| 0 | $(-I_{VDD} \times V_{SWING}) - (I_{INPUT_HIGH} \times V_{INPUT_SWING})$ |
| 1 | $(-I_{VDD} \times V_{VDD}) + (I_{VSS} \times V_{VSS}) - (I_{INPUT_HIGH} \times V_{INPUT_HIGH}) + (I_{INPUT_LOW} \times V_{INPUT_LOW})$ If $V_{VSS} = V_{INPUT_LOW} = 0$ then the above formula reduces to: $(-I_{VDD} \times V_{VDD}) - (I_{INPUT_HIGH} \times V_{INPUT_HIGH})$ |
| 2 | $(-I_{VSS} \times V_{SWING}) - (I_{INPUT_LOW} \times V_{INPUT_SWING})$ |

If `voltage_map=1`, leakage is reported *with related_pg_pin* as follows:

| leakage_mode | related_pg_pin | Formula |
|---------------------|-----------------------|--|
| 0 or 2 | VDD VSS | $(-I_{VDD} \times V_{SWING}) - (I_{INPUT_HIGH} \times V_{INPUT_SWING})$ $(I_{VSS} \times V_{SWING}) + (I_{INPUT_LOW} \times V_{INPUT_SWING})$ |
| 1 | VDD VSS | $(-I_{VDD} \times V_{VDD}) - (I_{INPUT_HIGH} \times V_{INPUT_HIGH})$ $(I_{VSS} \times V_{VSS}) + (I_{INPUT_LOW} \times V_{INPUT_LOW})$ |

Note: All current in the formulae above are the actual measured currents. In the case where multiple supply rails (like VDD, VDDL) are associated with a single ground rail (VSS), then power rail selection is random. The I_{INPUT_HIGH} and I_{INPUT_LOW} are the current measured at the input pin when the input is in a logic High or Low State. The V_{INPUT_HIGH} and V_{INPUT_LOW} are the input voltages. The above formulae assume that leakage_add_input_pin is set to 1. When this variable is set to 0, then all input leakage components in above formula are ignored.

This variable must be used before the `char_library` command.

leakage_model_internal_pin

`<0 | 1>` Controls if internal pins are used in leakage "when" state.
Default: 1

When this variable is set to 0, Liberate will remove internal pins from being modeled. The `internal_pin` group and the `internal_pin` in the leakage when condition will not be output.

This variable must be used before the `char_library` command.

leakage_precision

`<value>` Specify leakage precision (Default: "%g")

Set this variable to the desired precision using standard "C" print formats for the values under the *leakage_power*, *gate_leakage* and *pg_current* groups. If not set, *leakage_power* defaults to using "%g" and *gate_leakage* to the **write_library -precision** value.

This variable can be used after the `char_library` command.

leakage_ramp_vsrc

`< 0 | 1 >` Initializes voltage sources to 0V at time zero and ramps to its intended value at the time equal to $\frac{1}{2}$ of **leakage_sim_duration**. (Default: 0)

If this variable is set to 1, Liberate initializes all voltage sources to 0V at time zero, then ramps those voltage sources to its intended value at a time equal to $\frac{1}{2}$ of the **leakage_sim_duration** variable. If **leakage_sim_duration** is not set or if it's set to a value of less than 1ns, then the **leakage_sim_duration** variable will be automatically forced to 1ns. The **leakage_ramp_vsrc** variable is particularly helpful with leakage simulation where HSPICE errors out due to its inability to find a DC solution.

This variable must be used before the `char_library` command.

leakage_sim_duration

`<value>` If non-zero, use transient simulation after the specified delay for leakage. Default: 0 seconds (use dc solution)

Use this variable to enable a transient simulation and to specify the simulation duration to be used when measuring leakage. Setting it to 2e-12(seconds) will run the transient simulation

from 0s to 2e-12s and measure the leakage at the time (value) specified minus 1ps. This variable works with Alspice and with external simulators.

This variable also affects the rise time when ramping the power supplies. See `leakage_ramp_vsrc` and `ramp_vsrc` for more information on power supply ramping. When supply ramping is enabled and this variable is not set, then the power supply ramping for leakage simulations will follow the `sim_init_duration`.

This variable must be used before the `char_library` command.

library_copyright

`<value>` The string to denote copyright in the output library (Default: "")

This parameter specifies the value of the `copyright` attribute in the output library.

Note: The copyright can also be provided in the `write_library -user_data` file.

This variable must be used before the `char_library` command.

Example

```
# Set the copyright line
set_var library_copyright "Cadence Design Systems, 2006-2013"
```

library_revision

`<value>` The string to use for denoting the library revision. Default: 1.0

This parameter sets the value of the `revision` attribute in the output library. **Note:** The revision can also be provided in the `write_library -user_data` file.

This variable must be used before `char_library`. Example:

```
# Set the revision to 2.0
set_var library_revision "2.0"
```

library_revision_mode

`< 0 | 1 >` Affects value of `library_revision` text string. Default 1.

If library_revision is specified as "1.0", then:

0: Inserts library_revision in this form: revision : "\$Revision: 1.0 \$";

1: Inserts library_revision in this form: revision : "1.0"; (Default)

This variable must be used before **char_library**.

lic_max_timeout

<value> Specifies the duration of time, in seconds, to wait for the required licenses to be acquired. Default: 86400 (seconds)

When starting up, Liberate will attempt to check out all licenses that are needed. For a Server, 1 server license is needed. For a client, Liberate will need 1 client license for each thread (see **char_library -thread**). If **ALTOS_QUEUE** is set and if only one license is needed, then Liberate will wait until a license is available and then start running. If **ALTOS_QUEUE** is set and more than 1 license is needed, then Liberate will wait until the timeout or it has all of the licenses it needs for all threads. Set the variable **lic_max_timeout** to specify the number of seconds that Liberate will wait for licenses. When the timeout ends, if Liberate has at least 1 license, then it will stop waiting and start execution with however many licenses it has. If Liberate has no licenses at the end of the timeout, then it will reset the timeout clock and begin waiting again for licenses. After execution begins, Liberate will stop looking for additional licenses.

For example, if **ALTOS_QUEUE** is set to 1 along with **char_library -thread 4**, and if there are only 2 (mix-and-matched Liberate_Client, Variety_LX_Client and Liberate_LX_Client) licenses available at beginning, then Liberate will remain in a wait-and-check queue for an additional 2 licenses. As soon as the additional 2 client licenses are checked out successfully, Liberate will start execution with 4 simulation threads. If, however, there are no additional licenses checked out at the end of the timeout, then Liberate will start execution with only 2 simulation threads.

If **ALTOS_QUEUE** is set to 0 or is not set, then Liberate will not wait for licenses. Instead, it will check out as many licenses as it can (not exceeding the number it needs) and will begin execution. If no licenses are available, then Liberate will terminate.

The shell environment variable **ALTOS_LIC_MAX_TIMEOUT** will override the value set by this variable in the Tcl file. For more information, see [ALTOS_LIC_MAX_TIMEOUT](#).

This variable must be used before **char_library**.

lic_queue_timeout

<value> Specifies the duration of time, in seconds, to wait for the required licenses to be acquired. Default: 60 (seconds)

The shell environment variable ALTOS_LIC_CHECK_ALT_TIMEOUT will override the value set by this variable in the Tcl file. For more information, see [ALTOS LIC CHECK ALT TIMEOUT](#).

This variable must be used before **char_library**.

logic_and

"string" The characters to use for denoting logic AND in library attributes. Default: " * " (*Notice the space on both sides of **)

This parameter sets the logic AND string for attributes that contain logic functions such as *when* conditions. It does not apply to **sdf_cond** attributes where the string for logic AND can be set by the **sdf_logic_and** parameter.

This variable must be used before **char_library**.

logic_not

"string" The characters to use for denoting logic NOT in library attributes. Default: "!"

This parameter sets the logic NOT string for attributes that contain logical functions such as *when* conditions. It does not apply to **sdf_cond** attributes where the string for logic NOT can be set by the **sdf_logic_not** parameter.

This variable must be used before **char_library**.

logic_or

"string" The characters to use for denoting logic OR in library attributes Default: " + " (*Notice the space on both sides of +*)

This parameter sets the logic OR string for attributes that contain logical functions such as *when* conditions. It does not apply to **sdf_cond** attributes where the string for logic OR can be set by the **sdf_logic_or** parameter.

This parameter must be used before **char_library**. Example:

```
# Set the logic AND, OR and NOT string to &&, ~, |
set_var logic_and "&&"
set_var logic_not "~"
set_var logic_or "|"
```

mac_address_query_timeout

<time_in_seconds> Specifies a maximum time window to keep re-trying a given MAC address. Default: 60 (seconds)

The system will keep trying to access a MAC address, and if there is no response within the set time limit, the system will give up on that address. Useful in networks with heavy traffic.

-1: No timeout – keep trying the address until there is a response.

0: No retry – if there is no response on the first try, give up. (Behavior of versions 3.1 and older.)

Positive number: Maximum amount of time to continue re-trying. (Default = 60 seconds.)

This variable must be used before **char_library**. Example:

```
# Set timeout to a minute and a half:
set_var mac_address_query_timeout 90
```

mark_failed_data

< 0 | 1 > Select the warning level for failed data. Default: 1

When Liberate determines that a characterization has failed, it sets the `altos_error_flag` inside the `ldb`. This variable determines how Liberate will deal with the presence of this flag in the `ldb`.

0: Does not issue any warnings. The output library contains the value as stored in the LDB. This value may be a legal number and may pass the library compilation without generating any warnings. In most cases, the constraint value in `.lib` is an exceptionally large value.

1: Issues a warning when **read_ldb** reads in an `ldb` that contains the `altos_error_flag`. In addition, all children groups of the group containing the failed data have their values set to `DBL_MAX/3` (`INF` in `.lib`).

This variable should be set prior to the `write_library` command.

mark_failed_data_replacement

`<string>` Specifies the characters to use as replacement value in the output library for all the failed data. The variable `mark_failed_data` must be enabled.
Default: “1e+31”

If `mark_failed_data` is set to 0, then no non-constraint failed data is replaced. See `constraint_failed_value` for the handling of failed constraint data in this case.

This variable is used before the `write_library` command.

max_capacitance_attr_limit

`<value>` Maximum allowed `max_capacitance` attribute value.
Default: 1 (Farad)

Use this parameter to set a maximum value allowed for all `max_capacitance` attributes. This limit is applied after `max_capacitance_factor`.

This parameter may be used after `char_library`. Example:

```
# Set max capacitance attribute limit
set_var max_capacitance_attr_limit 1000e-15
```

max_capacitance_attr_mode

`< 0 | 1 >` Controls how Liberate selects the `max_capacitance` attribute value. Default: 1

0: Liberate will select the max of the maximum `index_2` values.

1: Liberate will select the min of the maximum `index_2` (`load_index`) values. (Default and recommended)

This variable must be used before `char_library`.

Example:

```
set_var max_capacitance_attr_mode 1
```

max_capacitance_derive_limit_maxload

`< 0 | 1 >` Limits the `max_capacitance` attribute to the largest `index_2` characterized load value. Default: 1

1: Limits the `max_capacitance` attribute to the largest `index_2` characterized load value when `write_library -derive_max_capacitance` is enabled. (Default and Recommended)

0: Allows `max_capacitance` to be larger than the largest `index_2` characterized value.

This variable must be specified before **write_library**.

max_capacitance_factor

`<value>` Multiplication factor applied to all *max_capacitance* attributes
Default: 1

This variable can be used to apply a factor to all *max_capacitance* attributes in the library. This factor does not get applied to indices. Default: 1.

This parameter may be used after **char_library**. Example:

```
# Set max capacitance factor
set_var max_capacitance_factor 0.66
```

max_capacitance_limit

`<value>` Maximum allowable load index value.
Default: 1 (Farads)

This variable controls a global limit that only becomes effective when **char_library -auto_index** is enabled. It specifies the maximum output pin load capacitance in Farads that can be assigned to `index_2`. If the max load capacitance calculated by **auto_index** exceeds this limit, the max load capacitance will be reset to the value stored in this variable. Default: 1 (Farads).

This variable must be used before the `char_library` command.

Example

```
# Set maximum allowed capacitance load
set_var max_capacitance_limit 1e-9
```

max_hidden_vector

<value> Maximum number of hidden power vectors (states). Default: 64

This parameter sets the maximum number of vectors that will be characterized for hidden power. The hidden power vector count is determined from the internal nodes and primary pins such that the hidden power vectors are unique. The input pins, bi-directional pins, output pins, and internal nodes that can have unique states are included in determining the vector count.

Internal nodes such as intermediate nodes in a transistor stack and storage nodes can have unique states depending on their initial values. Default: 64.

This variable must be used before **char_library**.

Example

```
# Set maximum number of hidden power vectors
set_var max_hidden_vector 1000
```

max_leakage_vector

<value> Maximum number of leakage vectors (states). Default: 256

This parameter sets the maximum number of vectors that will be characterized for leakage. The leakage vector count is determined from the internal nodes and primary pin such that the leakage vectors are unique. The input pins, bi-directional pins, output pins and internal nodes that can have unique states are included in the vector count determination. Internal nodes such as intermediate nodes in a transistor stack and storage nodes can have unique states depending on their initial values. Default: 256.

This variable must be used before **char_library**.

Example

```
# Set maximum number of leakage vectors
set_var max_leakage_vector 1000
```

max_noise_width

`<value>` Maximum allowable noise-glitch width (in seconds). Default: -1
(Don't automatically create indices)

This parameter is used to enable automatic creation of the indices for **si_immunity** tables. This parameter sets the maximum allowable noise-glitch width. Setting this parameter overrides indices defined in noise immunity templates. This parameter should always be set when characterizing noise immunity.

This variable must be used before **char_library**.

Example

```
# Set maximum noise glitch width to 4ns
set_var max_noise_width 4e-9
```

max_transition

`<value>` Maximum allowable delay transition time (in seconds). Default: 3.0e-9 (3ns)

This parameter is used to limit the maximum allowable output transition for a cell. Set this parameter when using the **auto_index** option to **char_library**. The **define_max_transition** command can be used to specify a pin based local override for the **max_transition**.

This is the report value in the Liberty model. In the case where **slew_*** parameters differ from their **measure_slew_*** counterparts, Liberate will derive the slew used during simulation by multiplying **max_transition** by the value used for **slew_derate_from_library**.

This variable must be used before **char_library**. Example:

```
# Maximum output transition time allowed
set_var max_transition 1e-9
```

max_transition_attr_limit

`<value>` Maximum allowed *max_transition* attribute value. Default: 1
(Seconds)

Use this parameter to set a maximum value allowed for all *max_transition* attributes. This limit is applied after *max_transition_factor*.

This parameter may be used after **char_library**. Example:

```
# Set max transition attribute limit to 1ns
set_var max_transition_attr_limit 1e-9
```

max_transition_factor

`<value>` Multiplication factor applied to all *max_transition* attributes.
Default: 1

This parameter can be used to apply a factor to all *max_transition* attributes in the library. This factor does not get applied to indices.

This parameter may be used after **char_library**. Example:

```
# Set max transition factor  
set_var max_transition_factor 0.66
```

max_transition_for_outputs

`<-1 | value | max>` Requests **max_transition** on output pins. Default: -1 (output pins do not have the attribute `max_transition`)

Use this variable to request the **max_transition** attribute on output pins. The default is to not output this attribute on output pins. Set this variable to a **value** to force a specific **max_transition** value on an output pin or to **max** to use the maximum value of all the **rise/fall_transition** values from all the arcs ending at that pin.

This variable must be used before **write** **library**.

max transition include power

< 0 | 1 > Includes power arcs when determining **max_transition**.
Default: 0 (max_transition is determined from timing arcs only.)

Liberate will automatically determine the **max_transition** attribute from the characterized timing arcs. Some cells, such as keeper cells do not have any timing arcs. Set this variable to 1 to tell Liberate to include power arcs in the search for determining **max transition**.

This variable must be used before **write_library**.

measure_cap_active_driver_mode

<0 | 1>

Instructs Liberate to begin measuring the pin capacitance from the time corresponding to the last supply crossover.

When an active driver is used to create the input waveform, the waveform shape may transition multiple times around the supply at the very beginning of the input transition. When the pin capacitance measurement has a threshold very close to the initial supply voltage, this ringing may result in a negative pin capacitance value.

Default: 0

This measurement change is applied to both ECSM and CCS pin capacitances.

0

Measures the pin capacitance from the beginning of the input waveform without observing the actual shape of the waveform.

1

Measures the pin capacitance from the last supply crossing at the beginning of the input transition.

This variable must be used before the `char_library` command.

measure_cap_lower_fall

<value>

The % point on the cell input waveform to use for measuring falling input capacitance to. Default: 0.01 (1% of supply)

measure_cap_lower_rise

<value>

The % point on the cell input waveform to use for measuring rising input capacitance from. Default: 0.01 (1% of supply)

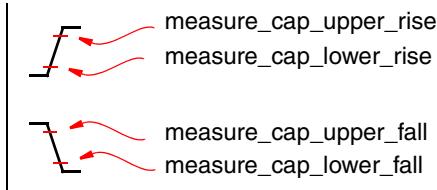
measure_cap_upper_fall

<value> The % point on the cell input waveform to use for measuring falling input capacitance from. Default: 0.99 (99% of supply)

measure_cap_upper_rise

<value> The % point on the cell input waveform to use for measuring rising input capacitance to. Default: 0.99 (99% of supply)

The above parameters are used to control how input capacitance is measured after SPICE simulation.



These variables must be used before **char_library**. Examples:

```
# Set the capacitance measurements to 10-90%
set_var measure_cap_lower_fall 0.1
set_var measure_cap_upper_fall 0.9
set_var measure_cap_lower_rise 0.1
set_var measure_cap_upper_rise 0.9
```

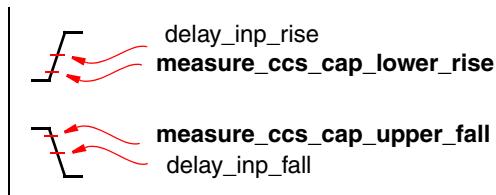
measure_ccs_cap_lower_rise

<value> The % point on the cell input waveform to use for measuring rising input CCS capacitance from. Default: 0.0 (0% of supply)

measure_ccs_cap_upper_fall

<value> The % point on the cell input waveform to use for measuring falling input CCS capacitance from. Default: 1.0 (100% of supply)

These variable are used to measure the first portion (C1) of the CCS receiver capacitance model; used in combination with the delay rise/fall thresholds:



These variables must be used before **char_library**. Examples:

```
# Set the CCS capacitance measurements
set_var measure_ccs_cap_upper_fall 0.3
set_var measure_ccs_cap_lower_rise 0.7
```

measure_em_target_occurrence

<last | integer>

Specifies any legal value that can be used in a spice measure statement to direct the actual measurement to use a specific target (TARG) match. This variable only works with EM characterization (see **-em** of **char_library**).

Default: 1

This variable must be set prior to the **char_library** command.

Example

```
set_var measure_em_target_occurrence last
```

measure_output_range

< 0 | 1 >

Enables measurement of output transition range. Default: 0
(Assume full rail voltage swing)

This parameter impacts the measurement of the initial and final voltages of a pin. If this variable is set then the measurement thresholds are applied to the range between the initial voltage and the final voltage of an output transition. Note: The initial and final voltages of the output must be different by more than 100mV, or the actual rail voltage will be used. The

default is 0 which means the outputs are assumed to swing full rail. This variable only applies when the **io** argument to **char_library** is used.

This variable must be used before the `char_library` command.

measure_output_range_abstol

< minimum range Volts>

Define the minimum output voltage swing range.
Default: 0.100 (Volts)

This variable is used to specify the minimum supported output voltage swing range. It is enabled when the variable `measure_output_range=1`. If the measured output swing range is less than the value specified in this parameter, then the value specified in this parameter is used.

This variable must be set before **char_library**.

measure_slew_lower_fall

<value>

The % point on the cell output waveform to measure falling output transition times to. Default: 0.2 (20%)

measure_slew_lower_rise

<value>

The % point on the cell output waveform to measure rising output transition times from. Default: 0.2 (20%)

measure_slew_upper_fall

<value>

The % point on the cell output waveform to measure falling output transition times from. Default: 0.8 (80%)

measure_slew_upper_rise

<value>

The % point on the cell output waveform to measure rising output transition times to. Default: 0.8 (80%)

The above parameters are used to control how output transition times are measured after SPICE simulation.

These variables must be used before **char_library**. Examples:

```
# Set the transition measurements to 20-80%
set_var measure_slew_lower_fall 0.2
set_var measure_slew_upper_fall 0.8
set_var measure_slew_lower_rise 0.2
set_var measure_slew_upper_rise 0.8
```

measure_target_occurrence

<last | legal_spice_value>

Specify any legal spice measure value. Default: last (assume full rail voltage swing)

When dual outputs are tied together by a resistor, oscillations can occur. Set this parameter to any legal value that can be used in a spice measure statement to direct the actual measurement to use a specific match.

This variable must be used before **char_library**.

mega_bundle_mode

| | |
|---------|---|
| <0 1> | Controls how mega mode should use the information from <code>define_bundle_pins</code> . Default: 1 |
| 0 | Do not use information from <code>define_bundle_pins</code> . |
| 1 | Use a restricted set of combinations of bundle pins for side inputs. |

This variable must be set before **char_library**.

mega_enable

<2 | 3>

Enables the use of an advanced algorithm referred to as "mega" mode. This mode improves inside_view preprocessing of large cells.

Default: 2

2

Enables mega mode for all cells in the library. It is not necessary to use the define_cell -type mega option.

3

Uses the mega mode algorithm on a cell by cell basis and is enabled only when the define_cell -type mega option is used for a specific cell.

Note: 0 and 1 are unsupported modes.

This variable must be set before char_library.

mega_mode_constraint

< minimum | fanout | all >

Enables heuristics that controls the number of combinations (vectors) generated for the side pins when characterizing a constraint arc.

Default: fanout

minimum

Select a single vector for all the side pins. If possible, cell outputs not on the characterized path are set to be non-switching. This gives the best runtime by using one arbitrary combination of side pins.

Virtuoso Liberate Reference Manual

Liberate Parameters

fanout

Select all combinations of side pins that can have some effect on the measured constraint. This includes the nodes that are in the path from input to probe for the arc being characterized and all nodes that are one Channel Connected Component (CCC) away from this path (first order miller effect). This mode covers all loading conditions for the constrained path and provides the best constraint accuracy. For multi-bit cells this mode may produce an exponential number of vectors if not restricted by the -patterns option of the `define_bundle_pins` command.

all

Generate simulation vectors for all combinations of side pins. This mode is intended for maximum compatibility with non-mega (when `mega_enable` is equal to 3) results. For multi-bit cells this mode produces the longest runtime due to the maximum number of vectors applied. The vectors can be restricted by the -patterns option of the `define_bundle_pins` command.

Mega mode (see `mega_enable`) identifies the cone of logic from inputs to probe for each constraint arc to be characterized. The signals that do not impact logically the constraint arc are considered side pins. These side pins may be completely independent of the cone of logic or they may have a parasitic effect on the arc. When possible the `vector_side_input` setting is observed.

This variable must be set before `char_library`.

mega_mode_delay

< minimum | fanout | all >

Enables heuristics that controls the number of combinations (vectors) generated for the side pins when characterizing a delay arc. Default: all

| | |
|---------|--|
| minimum | Select a single vector for all the side pins. If possible, cell outputs not on the characterized path are set to be non-switching. This gives the best runtime by using one arbitrary combination of side pins. |
| fanout | Select all combinations of side pins that can have some effect on the measured delay or power. This includes all nodes that are switching (due to their effect on power). This mode covers all loading conditions for the delay path and provides the best accuracy. For multi-bit cells this mode may produce an exponential number of vectors if not restricted by the <code>-patterns</code> option of the <code>define_bundle_pins</code> command. |
| all | Generate simulation vectors for all combinations of side pins. This mode is intended for maximum compatibility with non-mega (when <code>mega_enable</code> is equal to 3) results. For multi-bit cells this mode produces the longest runtime due to the maximum number of vectors applied. The vectors can be restricted by the <code>-patterns</code> option of the <code>define_bundle_pins</code> command. |

Mega mode (see `mega_enable`) identifies the cone of logic from inputs to output for each combinational (see `define_arc`-type combinational) arc to be characterized. The signals that do not impact the logic for this delay arc are considered side pins. These side pins may be completely independent of the cone of logic or they may have a parasitic effect on the arc. When possible the `vector_side_input` setting is observed.

This variable must be set before `char_library`.

mega_mode_hidden

< minimum | fanout | all >

Virtuoso Liberate Reference Manual

Liberate Parameters

Enables heuristics that controls the number of combinations (vectors) generated for the side pins when characterizing a hidden power arc.

Default: fanout

minimum

Select a single vector for all the side pins. This gives the best runtime by using one arbitrary combination of side pins.

fanout

Select all combinations of side pins that can have some effect on the hidden power. This includes all nodes that are switching (due to their effect on power). This mode covers all loading conditions for the hidden power and provides the best accuracy. For multi-bit cells this mode may produce an exponential number of vectors if not restricted by the `-patterns` option of the `define_bundle_pins` command.

all

Generate simulation vectors for all combinations of side pins. This mode is intended for maximum compatibility with non-mega (when `mega_enable` is equal to 3) results. For multi-bit cells this mode produces the longest runtime due to the maximum number of vectors applied. The vectors can be restricted by the `-patterns` option of the `define_bundle_pins` command.

Mega mode (see `mega_enable`) identifies the cone of logic from inputs internal node for each hidden power arc (see `define_arc -type hidden`) arc to be characterized. The signals that do not impact the logic cone for this hidden power arc are considered side pins. These side pins may be completely independent of the cone of logic or they may have a parasitic effect on the arc. When possible the `vector_side_input` setting is observed.

This variable must be set before `char_library`.

merge_related_preset_clear

<0 | 1 | 2>

Enables the merging of related preset and clear states.

Default: 2

The arc which is caused by the de-assertion of a preset/clear signal may be considered a *combinational_fall/rise*, or a *clear/preset* type. When **merge_related_preset_clear** is set to a 1, the rise and fall tables from an asynchronous preset/clear related pin will be merged into a single timing group with the clear/preset type. When **merge_related_preset_clear** is set to a 2, both the clear and preset timing_type groups will be output.

By default, the preset(clear) will generate a rise(fall) transition, and the timing arc from the de-assertion of the preset/clear will be omitted. If this parameter is **0**, and combinational_risefall is set to **1**, then both the preset/clear arc and their associated de-assertion arc will be output in separate timing groups.

This variable can be used after **char_library**.

Example

```
# Allow the merging of combinational timing types with
# preset/clear timing types.
set_var combinational_risefall 1
set_var merge_related_preset_clear 1
```

min_capacitance_for_outputs

<0 | 1> Set this variable to enable *min_capacitance* attribute for output pins. Default: 0

Set this variable to a **1** to enable the output of the *min_capacitance* attribute on output pins. A setting of **1** is useful to match legacy libraries. When set to **0** (default), Liberate will not output the *min_capacitance* attribute on output pins.

This variable must be set before **char_library**.

min_output_cap

<value> Minimum allowable output capacitance (in Farads). Default: -1 (use min input cap found in library)

This parameter is used to set the minimum output capacitive load when using the **auto_index** option to **char_library**. If not specified the minimum input pin capacitance found in the library is used.

This variable must be used before **char_library**.

Example:

```
# Set the minimum load index
set_var min_output_cap 5e-16
# Set the minimum output transition time index
set_var min_transition 1e-11
char_library -auto_index
```

min_period

`<0 | 1 | 2>`

Controls the calculation of the `minimum_period` timing group for clock and asynchronous pins. The `minimum_period` is calculated as $2 * \max(\text{mpw_high}, \text{mpw_low})$
Default: 0

Note: Modeling of `min_period` is dependent on the MPW data that is characterized and stored in the ldb. If MPW is not characterized, then `min_period` cannot be computed.

- | | |
|---|---|
| 0 | No minimum period calculation. |
| 1 | Calculates minimum period for clock (flip-flops) and enable (latch) pins only |
| 2 | Calculate minimum period for clock/enable pins plus async pins. |

This variable must be used before the `write_library` command.

min_period_when

`"string"`

Specifies the logic *when* condition for the `min_period` constructs. Default: ""

Setting this variable adds both a *when* condition and an equivalent `sdf_cond` condition to the `minimum_period` group. Only `min_pulse_width` groups that overlap with the user-defined *when* condition will be used to calculate the `minimum_period`.

This variable works only in conjunction with the `min_period` variable.

This variable must be used before `write_library`.

min_transition

<value> Minimum allowable delay transition time (in seconds).
Default: -1 (automatically calculate)

This parameter is used to set the minimum output transition when using the `auto_index` option to `char_library`. Default is to automatically calculate the minimum transition index.

This variable must be used before **char_library**.

min_transition_attr_limit

<value> Minimum allowed `min_transition` attribute value.
Default: 1e-15 (seconds)

Use this parameter to set a minimum value allowed for all `min_transition` attributes. This limit is applied after `min_transition_factor`. (See [write_min_transition_attr](#) to enable writing the `min_transition` attribute into the library.)

Example:

```
# Set min transition attribute limit
set_var min_transition_attr_limit 1e-12
```

This parameter may be used after the `char_library` command.

min_transition_factor

<value> Multiplication factor applied to all `min_capacitance` attributes. Default: 1

This parameter can be used to apply a factor to all `min_transition` attributes in the library. This factor does not get applied to indices. Default: 1.

Example:

```
# Set min transition factor
set_var min_transition_factor 0.66
```

This parameter may be used after **char_library**.

min_transition_for_outputs

`<-1 | "min" | value>` Use this variable to add the `min_transition` attribute on output pins.
Default: `-1` (Do not add this attribute.)

-1: Do not put the attribute `min_transition` on outputs.

min: Use the minimum value of all the `rise/fall_transition` values from all the arcs ending at that pin.

value: Forces a specific `min_transition` value on an output pin.

This variable must be used before **write_library**.

min_transition_include_power

`<0 | 1>` Includes power arcs when determining `min_transition`.
Default: `0` (`min_transition` is determined from timing arcs only.)

Liberate will automatically determine the `min_transition` attribute from the characterized timing arcs. Some cells, such as keeper cells do not have any timing arcs. Set this variable to 1 to tell Liberate to include power arcs in the search for determining `min_transition`.

This variable must be used before **write_library**.

mpw_criteria

`<delay | glitch>` Specifies MPW failure criteria as delay or glitch.
Default: `delay`

This parameter can be used to specify the failure criteria to be used when measuring the minimum pulse width. To measure Minimum Pulse Width (MPW) Liberate applies a narrowing pulse to the input pin until the failure criteria is met. The circuit is sensitized to enable a CLK->Output delay measurement. Legal values are: `delay` and `glitch`. When `delay` is specified, the `constraint_delay_degrade` variable specifies the relative delay degrade value for the CLK->Output arc. When `glitch` is specified, the CLK->Output delay is measured at the `constraint_glitch_peak` ratio of supply and the searches for a pass/fail of the CLK->Output arc, that is, the CLK->Output path returns a measurement (it toggles)

or the measurement fails (the Output does not toggle) when measured at the `mpw_glitch_peak` threshold.

This variable must be used before the `char_library` command.

mpw_glitch_peak

`<ratio>` Specify the MPW glitch peak failure threshold.
Default: 0.95

Set this variable to the desired measurement threshold when the Minimum Pulse Width (MPW) measurement uses the glitch criteria (see [mpw_criteria](#)). This variable has no effect if the `mpw_criteria` is set to `delay`.

This variable must be used before the `char_library` command.

mpw_input_threshold

`<value>` Specifies the minimum height allowed when the clock pulse becomes triangular. The `min_pulse_width` constraint is deemed to be violated if this occurs before.
Default: 0.9 (90% of Vdd)

This variable must be used before the `char_library` command.

mpw_linear_waveform

`<0 | 1>` Defines the input threshold.
Default: 1

Set this parameter to request that Liberate use a linear waveform on the input pin when characterizing the minimum pulse width. By default, the linear input waveform is used for mpw as is used for delay.

- 0:** Use the same input waveform as for delay.
- 1:** Use the linear input waveform.

This variable must be used before `char_library`.

mpw_search_bound

`<value>` Controls the initial search bound for MPW characterization.
Default: $5e-9$

This variable must be used before the `char_library` command.

mpw_search_mode

`<0 | 1 | 2>` Enables fast algorithm for determining MPW.
Default: 1

0: Default algorithm using bisection techniques.

1: Enables a fast algorithm for determining minimum pulse width. This may produce a slight change in results, but they should be within tolerance. (See [constraint_search_time_abstol](#).) (Default)

2: This mode enables a search method which requires that the Minimum Pulse Width (MPW) exceeds the `related_pin` to probe delay. Use this method to increase risk aversion while significantly decreasing the run time. For this method to be successful, the [mpw_criteria](#) must use delay and the probe must be selected so the leading edge of the pulse is the active edge. This search method measures the `related_pin` to probe delay and adjusts the pulse width so that the pulse trailing edge transition aligns with the probe transition. The search terminates when the difference between the two edges is less than the constraint tolerance. The probe is selected from the nodes that transition due to the pulse leading edge but not the trailing edge, as this is necessary for the search criterion to work. The probe nodes that are triggered from the pulse trailing edge, or with both edges, are not monitored. For table based MPW (see [mpw_table](#)), the fastest slew is characterized for MPW. The rest of the slews are calculated as the fastest slew MPW plus the difference between the slews. The variables [constraint_check_final_state](#) and [constraint_check_rebound](#) can be used with this method.

This variable must be used before `char_library`.

mpw_skew_factor

`<value>` Enables skewing the ratio of the input rise versus input fall slew for calculating *min_pulse_width* constraints.
Default: 1.0 (rise and fall input slews are identical)

This variable must be used before the `char_library` command.

mpw_slew

<value | min | mid | max>

Defines the input slew for minimum pulse width characterization.
Default: min

This variable specifies the value for the input slew to be used when determining the minimum pulse width timing constraint of a clock or async signal. The value can be any floating point number in seconds. If **min**, **mid** or **max** is used then the minimum, middle or maximum input slew value is taken from the input slew indices of the first delay arc where this clock signal is a related pin (e.g. a clock to Q delay arc on a flip-flop).

This variable must be used before **char_library**. Examples:

```
# Set the MPW glitch height criteria to 30% of Vdd
set_var mpw_glitch_height 0.3

# Set the slew for MPW to 200ps
set_var mpw_slew 200e-12

# Set fall to rise slew ratio to be 0.8
set_var mpw_skew_factor 0.8

# Set the MPW input threshold to 70% of Vdd
set_var mpw_input_threshold 0.7
```

mpw slew clock factor

<value> Defines a ratio of the mpw_slew to apply to all clock nets.
Default: 1 (use mpw_slew)

This variable specifies the ratio to apply to the **mpw_slew** when characterizing the minimum pulse width value of clock nets. By default, clock nets will use the slew determined when applying **mpw_slew**. When this variable is set, the slew applied to clock nets will be: **mpw_slew_clock_factor * mpw_slew**.

This variable must be used before **char_library**. Example:

```
# Set the slew for MPW for clocks to 100ps when
# other nets are using 200ps
set var mpw slew 200e-12
```

```
set_var mpw_slew_clock_factor 0.5
```

mpw_table

<0 | 1> Enables generating a one dimensional table of mpw values.
Default: 1

Use this parameter to enable the generation of a minimum pulse-width data table. When this parameter is set the mpw table will use the *index_1* table as specified by the **constraint_template** associated with the **define_cell** command for the cell. Characterizing mpw tables will increase runtime.

This variable must be used before **char_library**.

mpw_vector_bin_mode

<0 | 1 | 2 | 3> Controls pre-check index point for binning vectors.
Default: 1 (First index point)

Set this to reduce the number of vectors used to characterize MPW on an input pin (which should reduce the characterization run time). Liberate performs the simulation needed to fill in a selected index in the table, and uses that to determine how vectors can be "binned" (consolidated). This variable specifies which index point in the table should be used to determine the binning of vectors. Legal values are:

- 0:** No binning
- 1:** First index point (Default)
- 2:** Middle index point
- 3:** Last index point

This variable should be used before **char_library**. Example:

```
set_var mpw_vector_bin_mode 2
```

msg_level

<0 | 1> Controls the verbosity of error and warning messages.
Default: 0

0: Output error messages and useful warning and informational messages.

1: Output all messages. Caution: this setting can output a lot of messages (some of which may not be helpful) making it difficult to determine which messages are important.

This variable must be used before **char_library**.

msg_level_user_data_override

<0 | 1> Controls the verbosity messages related to inserting user data in the library.
Default: 0

0: No messages are displayed. (Default)

1: Display a message when an attribute named in the **user_data_override** variable is overridden by data in the **write_library -user_data** file.

This variable must be used before `write_library`.

msg_limit_per_type_per_cell

<integer> Controls the output limit of the maximum number of messages for each message type.
Default: 5

This variable should be set before the `char_library` command.

net_batch_mode

<0 | 1> Specifies Netbatch as a queuing system. Cannot be used together with qsub_no_shell_mode.
Default: 0 (Do not use Netbatch.)

nochage_mode

<0 | 1 | 2> Enables modelling and characterization of nochange timing type arcs.
 Default: 0
 Recommended: 1

- | | |
|---|---|
| 0 | The nochange arcs are not characterized but can be modeled with user-specified delay values. You can specify the nochange timing arcs with the <code>define_arc</code> command. The value is specified by the <code>-value</code> argument of the <code>define_arc</code> command or by the variable <code>nochange_value</code> . This setting is compatible with previous releases of Liberate. |
| 1 | Note: If a nochange arc cannot be characterized, the mode 0 can be used to specify the desired value. |
| 2 | The nochange arcs are characterized by Liberate if a <code>define_arc</code> command is provided for the arc. |
| 3 | Similar to 1. However, both pull-in and push-out are measured for both the leading and trailing output pulse edges instead of push-out for the leading edge and pull-in for the trailing edge. |

This variable must be specified before the `char_library` command.

nochange_value

`<value>` Specify a default nochange arc delay in seconds.
Default: `300e-12` seconds

When `nochange_mode` is set to 0, this variable provides the default delay value to be specified for the nochange arcs. This can be overridden by the `-value` argument of the `define_arc` command.

normalized_driver_waveform

`{ list }` Use a given PWL waveform as the input driver waveform.
Default: " " (Unused).

This variable enables using a user-defined PWL waveform as the reference waveform for all input pins. The reference waveform is linearly scaled to match the slew and voltage range required for a given input slew.

The PWL must be entered in a normalized, ordered list of time-voltage pairs { t0 v0 t1 v1 ... tN vN } where t0=v0=0 and vN=1.0. Liberate automatically derives the reference slew from the reference waveform using the derating from given measure_slew_*_rise and slew_*_rise settings.

The default is an empty list. If set, it overrides any set_driver_cell commands and the predriver_waveform variables.

This variable must be used before **char_library**. Example:

Characterize with a user-define waveform

```
set_var normalized_driver_waveform { \
0      0      \
1.00E-12  0.0001  \
1.00E-11  0.001  \
2.00E-11  0.01  \
3.90E-11  0.05  \
5.20E-11  0.1  \
6.00E-11  0.15  \
6.50E-11  0.2  \
7.00E-11  0.25  \
7.50E-11  0.3  \
8.00E-11  0.35  \
8.50E-11  0.4  \
9.00E-11  0.45  \
9.50E-11  0.5  \
1.00E-10  0.55  \
1.05E-10  0.6  \
1.10E-10  0.65  \
1.15E-10  0.7  \
1.20E-10  0.75  \
1.25E-10  0.8  \
1.30E-10  0.85  \
1.36E-10  0.9  \
1.46E-10  0.95  \
1.55E-10  0.97  \
1.65E-10  0.98  \
1.85E-10  0.99  \}
```

```
2.05E-10  0.999    \
2.25E-10  0.9999   \
2.50E-10  1         \
}
```

non_seq_copy_dst_pin

`<pin_name>` Name of pin for copy/transpose operation

These two variables work together, and instruct Liberate to copy and transpose the following:

- **non_seq_setup** values under copy-source pin into corresponding **non_seq_hold** values under copy-destination pin
- **non_seq_hold** values under copy-source pin into corresponding **non_seq_setup** values under copy-destination pin.

This variable must be used before **write_library**.

Example

```
set_var non_seq_copy_src_pin CDN
set_var non_seq_copy_dst_pin SDN
```

non_seq_copy_src_pin

`<pin_name>` Name of pin for copy/transpose operation

non_seq_pin_swap

`<0 | 1>` Controls swapping of the constrained pin and related_pin when characterizing **nonseq_setup/hold**. Default: 1

0: Liberate will swap the pin and the related_pin.

1: Liberate will not swap the pin and related_pin. (Default)

This variable must be used before **char_library**.

nonseq_as_recrem

`<0 | 1>` The parameter converts arcs with timing-type *nonseq_setup/hold* into recovery/removal. Default: 0

When this variable is set, all arcs with the timing-type of *nonseq_setup/hold* arcs are converted into recovery/removal. By default, these arcs are not converted.

This variable can be used after **char_library**.

Example

```
# Convert non_seq_setup and non_seq_hold timing types
# to 'recovery and removal'
set_var nonseq_as_recrem 1
```

output_internal_pin

`<0 | 1>` The parameter requests the internal probe pins to be included in the .lib file.
Default: 0 (Do not include internal pins.)

When this variable is set, all internal probe pins will be output in the .lib file with the type *internal_pin*. By default, these pins are not output into the .lib file.

This variable must be used before the **char_library** command.

packet_arc_notification_interval

`<value>` Specifies the minimum time interval between two informational notifications (see **packet_arc_notification_list**). The range is between 0 to 72000.
Default: 600 (in Seconds = 10 minutes)

This variable must be used before the **char_library** command.

Example

```
set_var packet_arc_notification_interval 3600
```

The above example requests no more than one informational notification per hour.

packet_arc_notification_limit

<value> Specifies the maximum number of informational notifications per run. This variable is effective when the `packet_arc_notification_list` has been set. The range is between 0 to 100.
Default: 10

This variable must be used before the `char_library` command.

Example

```
set_var packet_arc_notification_limit 5
```

The above example limits the notifications to no more than 5.

packet_arc_notification_list

<string> Sets the e-mail addresses or SMS equivalent e-mail addresses that can receive notifications. Multiple e-mails or SMS numbers can be specified by using a comma-separated list. By default, no notifications are sent. You can set this variable to a valid e-mail address to enable notifications to that address.
Default: " " (empty list)

Requirements:

- The main Liberate AMS job must run on a machine that is able to send e-mails.
- Any SMS numbers provided for notifications should be able to receive messages by e-mail. Some carriers block this ability to prevent spam messages.

This variable must be used before the `char_library` command.

Example

```
set_var packet_arc_notification_list "111111111@mms.att.net, \  
222222222@messaging.sprintpcs.com, 333333333@tmomail.net, abc@def.com"
```

The above example has three SMS numbers (ATT/Sprints PCS/TMobile) and one e-mail address.

packet_arc_optimize_idle_clients

<0 | 1> This parameter reduces the number of idle clients.
Default: 0

Set this variable to 1 to reduce the number of idle clients. This could however increase wall time if there is a large variation in CPU time required to characterize each cell. This variable is best used when CPU time required to characterize all cells are similar.

This variable must be used before **char_library**.

For example:

```
set_var packet_arc_optimized_idle_clients true
```

packet_arcs_per_thread

`<number>` Sets the number of arcs each thread will characterize during arc-based distribution.
Default: 10 (Each thread characterizes 10 arcs)

In the arc-based flow, Liberate can distribute the arcs of a cell across multiple machines. This variable controls how many arcs are simulated by each thread. Larger values mean more arcs will be run on each thread, reducing the preprocessing and distributed overhead but increasing the runtime of that client. Cells with short preprocessing time but long simulation time will benefit from a smaller value for `packet_arcs_per_thread`, while cells with a longer preprocessing time and shorter simulation time benefit from larger values.

The default and recommended setting is 10. For a large number of standard or base cells (cells with mixed simulation times), a performance gain can be seen by using a setting of 40.

For a small number of cells with longer simulation time (for example, block or multibit), a setting of 3 will yield greater distribution of jobs across clients.

This variable must be used before `char library`.

packet_arcs_per_thread_auto_adjust

| | |
|---------|---|
| <0 1> | Adjusts automatically the <u>packet_arcs_per_thread</u> to fit the number of <u>packet_clients</u> and the specified threads. Default: 0 (off) |
| 0 | Honors the user-provided setting for <u>packet_arcs_per_thread</u> . |
| 1 | Adjusts <u>packet_arcs_per_thread</u> based on <u>packet_clients</u> and <u>-thread</u> settings. |

Setting this variable causes packet_arcs_per_thread to automatically be increased (if necessary) so that all arcs can be distributed into a minimum number of jobs. This can help improve performance in cases where the pre-processing time comprises a significant amount of run time, or one cell is run by itself. The downside is that jobs with a few extremely long arcs might get bundled together; thus, lengthening the wall time of the entire run.

This variable must be set before the char_library command.

packet_require_spectre_char_opt

| | |
|---------|--|
| <0 1> | Defines how the packet flows (see <u>packet_mode</u>) handle the <u>Spectre_char_opt</u> licenses. Default: 0 (Use <u>Spectre_char_opt</u> licenses when they are available, otherwise use <u>MMSIM</u> licenses.) |
| 0 | Uses <u>Spectre_char_opt</u> licenses if they are available. If unavailable, let Spectre check out an <u>MMSIM</u> license. The <u>packet_client</u> continues even if it fails in checking out the required number of <u>Spectre_char_opt</u> licenses, relying on Spectre to manage simulation licenses to be used for characterization. This provides the most flexibility in terms of license utilization at the risk of stability problems for large installations. |

Virtuoso Liberate Reference Manual

Liberate Parameters

1

Uses only Spectre_char_opt licenses. The packet_client waits until it checks out the required number of Spectre_char_opt licenses, with one license used per thread for normal jobs and two licenses used per thread for electromigration jobs. This setting is recommended if enough Spectre_char_opt licenses are available or for installations with more than 500 clients.

When starting a packet_client, Liberate has the option of acquiring and holding all licenses needed for characterization at once by checking out both client and Spectre_char_opt licenses. One client license is checked out per thread. If Spectre_char_opt licenses are not checked out for characterization, then Spectre acquires licenses each time the Spectre process starts, allowing for utilization of any valid Spectre license (Virtuoso_Spectre, Virtuoso_Multi_mode_Simulation, Virtuoso_Acceler_Parallel_sc), as required.

The flexibility in using any available Spectre license may have an impact on license server stability in large installations. For SKI runs, there are one or more check out events over the course of the job. For Spectre runs, there is one check-out event for each simulation. FlexLM license servers can only handle between 1000 and 2000 simultaneous accesses per second. If the installation has more than 500 client licenses, then there is a risk of overwhelming the license server if Spectre_char_opt licenses are not used.

This variable must be used before the char_library command.

packet_cell_max_fets

<value>

Controls the threshold below which the packet arc flow does not splits a cell.

Default: 25

This parameter is only used when packet_mode is "arc". For example, if the value of packet_cell_max_fets is set to 100 then all cells with less than 100 devices will not be divided into smaller work units by the arc packet flow.

Min value: 0 (all cells are divided)

Max value: infinity (no cells are divided)

This variable must be used before **char_library**.

packet_clients

<0 | integer> Enables Parallel Packets mode and specifies the number of machines to be used for distributed processing.
Default: 0 (Packet-mode off)

0: Parallel Packet mode off. (Default)

<integer>: Enables Parallel Packet Mode and sets the number of machines to be used.

Note: If your flow uses write_vdb, you must set this to **0**.

This variable must be used before **char_library**.

packet_client_idle_count

<number> Specifies the allowed number of idle clients.
Default: -1 (Number controlled by internal heuristics)

Set this variable to reduce the number of clients available for characterization near the end of the run. The default and recommended value is -1, which allows the tool to internally control the client release mechanism.

While characterizing statistical delay, setting this variable to 2 helps to release idle clients in a more timely manner.

This variable must be used before **char_library**.

packet_client_resubmit_count

<number> Specifies the number of times a failed LSF job should be resubmitted.
Default: 0

This variable specifies the number of times a failed LSF job should be resubmitted for simulation. (Note: Liberate will also check the ldb to make sure it contains data from the job, and will resubmit if necessary.)

This variable must be used before **char_library**.

packet_client_timeout

<value> Sets a timeout value in seconds for client machines on the network. The valid values are 600 (10 minutes) to 86400 (1 day).
Default: 86400 seconds (1 day)

This variable specifies a timeout limit (in seconds) for client machines on the network. If a packet client log file has not been updated for more than the number of seconds specified, Liberate will assume that packet has died. (This can occur because of a machine crash, or a signal such as "kill -9" that can't be trapped.) If a packet client is determined to be "dead", then the server will not wait and move on to the next client.

This variable must be used before **char_library**.

packet_client_timeout_action

<value> Timeout action is not triggered unless the **packet_client_timeout** variable is set to a non-negative value. The default action is **warning** .

This variable has no effect unless **packet_client_timeout** is set to a valid value by the user. The two possible values are, **warning** and **error**.

error: The master marks the job as failed (and the associated cell). If using LSF, the client job is killed via bkill.

warning: Output a warning into the log file when the **packet_client_timeout** is exceeded. The timer is reset and the warning is repeated again and again after the timeout is exceeded.

This variable must be used before **char_library**.

packet_log_filename

<file name> Name of log file. Default: "log"

Must set this to match the log file specified in the **rsh_cmd** to report characterization statistics. We recommend using the default name “**log**” and also setting **rsh_cmd** to use “**/log**” as stdout and stderr filenames.

Note: "%L" is not allowed as a string in **packet_log_filename**.

packet_mode

<cell | arc> Controls the Parallel Packet Distribution Mode.
Default: **arc**

Liberate can distribute Parallel Packets in cell-based mode or arc-based mode.

cell: Cell-based mode.

arc: Arc-based mode. (Default and recommended)

This variable must be used before **char_library**.

packet_rdb_mode

<0 | 1> Enable RDB in parallel packets.
Default: 0 (disabled).

The RDB is a recovery data base that saves raw SPICE data in a compact form. The RDB is used to store data for incremental or arc-based flows when using a parallel packet mode. If kept, re-characterizations will take much less time, as Liberate can skip a simulation if the results are already present.

0: Disables RDB in parallel packets. (Default)

1: Enables RDB in parallel packets.

This parameter will be automatically set to 1 when **packet_mode=arc**. The recommended setting is 0 in cell packet flow.

This variable must be used before **char_library**.

packet_rsh_mode

`<lsf | nc | custom>` Instructs the server during the arc packet flow (see [packet_mode](#)) to automatically kill client jobs if the server job is interrupted.
Default: `custom`

This variable is automatically set for the following batch submission commands (see [rsh_cmd](#) and [set_rsh_cmd](#)):

- `bsub`
- `nc`

This variable may need to be explicitly set to the correct value if the [rsh_cmd](#) is pointing to a wrapper script instead of using the commands mentioned above.

This variable must be set before the **char_library** command.

parenthesize_not

`< 0 | 1 >` This puts parentheses around variable names that are negated using the exclamation mark (!).
Default: 0

Liberate treats the *not* expression as represented with parentheses around the variable name in logic functions. Example: `!(A)`

0: Parentheses will not be added, i.e.: `!A` (Default)
1: Put parentheses around variable, i.e.: `!(A)`

This variable must be used before **char_library**. Example:

```
# Remove () from variable names
set_var parenthesize_not 0
```

parenthesize_sdf_cond

`< 0 | 1 >` Puts parentheses around all `sdf_cond` statement in output library.
Default: 0

0: Do not put parenthesis around sdf_cond statements. (Default)

1: Put parentheses around sdf_cond statements.

This variable can be used after **char_library**.

parse_auto_define_leafcell

| | |
|-------------|--|
| <0 1 2> | Controls whether Liberate should recognize leafcells when parsing a netlist, if yes, then how should it happen. Default: 1 |
| 0 | Does not generate <code>define_leafcell</code> commands. This setting is provided for backward compatibility to LIBERATE 14.1 ISR3 and prior releases. |
| 1 | Recognizes leafcells automatically and prints <code>define_leafcell</code> commands into a log file. All of the user-provided <code>define_leafcell</code> commands are preserved and honored. |
| 2 | Detects leafcells automatically, but does not apply the detected leafcells to the analysis. Print <code>define_leafcell</code> commands into a log file so that you can examine and fill in the missing information. For example: <code>l</code> , <code>w</code> , <code>nfin</code> , <code>cjsw</code> , and so on. |

A leafcell is the lower most instance that is found when flattening a netlist that is outside of the model file. The `define_leafcell` command can be used to manually specify leaf cells. Also, the auto-detection of leafcells is supported only when using the Spectre Front End (SFE) parser (see `read_spice -format spectre`) and the variable `extsim_model_include` is set.

This variable must be set prior to the `char_library` command.

parse_space_bang_is_comment

| | |
|-----------|---|
| < 0 1 > | Specifies whether to treat bang (!) preceded by a space as a comment. Default: 0 |
|-----------|---|

When set to the default of 0, this variable parses the netlist and does not treat bang (!) preceded by a space as a comment. Setting it to 1 accepts bang preceded by a space as a comment when parsing the netlist.

This variable must be used before **read_spice**.

pin_based_leakage

<0 | 1> Controls whether or not to output pin based leakage_power groups.
Default: 1

Supported values are:

- 0:** Output only one leakage_power per state condition.
- 1:** Output leakage_power groups based on state and power.

This variable must be used before **char_library**.

pin_based_power

<0 | 1 | 2> Controls how pin-based power is calculated and characterized.
Default: 1

| | |
|---|--|
| 0 | Models only the power in the positive supply nodes (see <code>set_vdd</code>). This setting is not compatible with advanced power formats such as CCSP and ECSMP. |
| 1 | Models the power for all of the power pins of the cell (see <code>set_vdd</code> and <code>set_gnd</code>). |

- 2 This is similar to mode 1 with the addition that Liberate monitors the Miller capacitance current to each cell input. If the input is set to a positive or negative voltage, that current is added to the vdd or gnd power. Typically, this Miller capacitance current is not accounted for in any other power measurement. This is because when the driving cell is characterized, the receiving cell is not included, and even if it is, it may not be switching in the correct way to trigger the Miller capacitance currents. See the [Liberate Details](#) section for information on power characterization.

The variable `pin_based_power` when set to 0 can co-exist with `voltage_map` 1. When this specific condition is encountered, the positive supply power will have CV^2 subtracted. When this variable is greater than 0, both positive and negative supplies will have $\frac{1}{2}CV^2$ subtracted from them, depending on the value of `power_subtract_output_load`. The VSS power will not be output when using `pin_based_power` 0 in NLDM format libraries.

This variable must be used before the `char_library` command.

Example

```
# Set the power calculation to monitor only VDD power
set_var pin_based_power 0
```

pin_based_signal_level_mode

- <0 | 1> Enables both `input/output_signal_level` and `related_power/ground_pin` to be written out to the `.lib` file.
Default: 0
- 0 Includes either `related_power/ground_pin` or `input/output_signal_level` under pin group based on the `voltage_map` setting.
- 1 Includes both `related_power/ground_pin` and `input/output_signal_level` when `voltage_map` is set to 1.

This variable must be used before the `write_library` command.

pin_capacitance_matching_mode

`<0 | 1>`

Set `pin_capacitance_matching_mode` to 1 for the behavior of version 3.0p2 (or older) where `set_pin_capacitance` expects a complete match between the characterization "when" conditions and the specified condition while considering capacitances. Setting it to 0 will have it consider overlapping "when" states as well.

Default: 0

This variable must be used before the `write_library` command.

pin_type_order

`{ pin_order_list }` Specifies the grouping of pins for a cell when written out to the library.
Default: `internals inouts outputs inputs`

The pins in a cell may be sorted into groups based on pin type as they are written out to the library. For example, all output pins may be put in one `group` and all input pins may be put in another group.

When pin sorting is enabled, this variable sets the pin group order to be applied while writing the pins to the library; for example, it sorts input and output pins and places them in two separate groups. To enable pin sorting, the parameter `sort_pins` must be enabled. The currently supported list of keywords include:

- `internals` - All internal pins
- `inouts` - All bidirectional pins
- `outputs` - All output pins
- `inputs` - All input pins

Set this variable to "" (the empty string) to request that if pin sorting is enabled (see `sort_pins`) then all pins are sorted alphanumerically.

Example

```
set_var sort_pins 1
set_var pin_type_order {internals inouts outputs inputs}
```

This variable must be set before **write_library**.

pin_vdd_supply_style

<0 | 1 | 2> The parameter controls the pin supply style.
Default: 0 (Do not output a vddName)

This variable controls how the pin supply is reported in the Liberty model. The vddName are specified in the set_pin_vdd and/or set_pin_gnd supply_name options.

0: No vddName will be output (Default)

1: Behavior depends on setting of voltage_map:

voltage_map=0: Use related_power_pin
voltage_map=1: Use input/output_signal_level

2: vddName will be output in the .libs following the format as specified by voltage_map.

This variable must be used before **char_library**.

power_add_input_pin

<0 | 1> Specify whether to include non-switching side input pin current.
Default: 1 (include input pin current)

Set the variable to **0** to ignore the power contribution from non-switching side input pins regardless of the pin_based_power setting. When set to **1** (Default) Liberate will add non-switching side input pin power according to the pin_based_power settings of **1** or **2**.

This variable must be used before **char_library**.

power_adjust_for_pin_load

<0 | 1> Set this variable to remove the harness power from the internal_power.
Default: 0

Set this variable to **1** to separate the power consumed by the pin_load in the harness from the cell internal_power. When set to **0** (default), the power will not be separated.

This command must be used before **char_library**.

power_binate_arc

< "separate" | "merge" >
Splits the power into multiple groups based on a WHEN condition.
Default: "merge"

Allows the power to be split into multiple groups based on a WHEN condition. Used when multiple timing tables exist based on the timing_sense.

Set to **separate**: Creates state-dependent power tables.

Set to **merge**: Merges power into a single table that is not state dependent. (Default)

This command must be used before **char_library**.

power_combinational_include_output

<0 | 1> Specify whether to include output pin in when.
Default: 1 (Include output)

Controls whether the output pin is included in combinational cell leakage and hidden power when conditions.

0: Excludes the output pin from the when state.

1: Includes output pin in leakage when states. (Default)

This command must be used before **char_library**.

power_divide_num_switching_mode

< 0 | 1 | 2 > Specifies that the power is divided by the number of outputs and inputs switching simultaneously. Simultaneous switching inputs are enabled using define_arc -vector and/or simultaneous_switch_from_cell_when and/or simultaneous_switch. Simultaneous switching outputs are controlled by the design itself but can be indicated using the define_arc -vector option.
Default: 0

0: The hidden power (input pin power) is divided by the number of switching inputs, and the switching power (output pin power) is divided by the number of switching outputs. This is the default and recommended value.

1: The hidden power is divided by the number of switching inputs, and the switching power is divided by the number of switching inputs after dividing by the number of switching outputs.

2: Liberate reports the full power (no division) on each pin and is used solely to verify the measured power without any processing. It may be desirable to also use power_subtract_hidden=0 and power_subtract_leakage=0 in order to start with the raw power numbers and then introduce each post-process in turn. This setting is used for debugging reported power.

This variable must be set prior to the **char_library** command.

power_info

<0 | 1 | 2> Print additional messages regarding power calculations.
Default: 0 (no extra information)

Validating power calculations is one of the challenging aspects of library qualification, especially since the relevant power data may not be contained in a single SPICE simulation. Enabling this feature outputs the additional information that may be used for power debugging or validation purposes.

For each power table, the additional data consists of the following:

- Cell
- Arc
- When
- Vector space covered by the arc and vector selected by Liberate
- Deck location
- Leakage power state selected

- Hidden power state selected
- Equation used

Note that some of the power calculations are done prior to **write_Idb**, while other calculations are performed after **read_Idb** or **write_Idb** but prior to **write_library**. Therefore, it may be necessary to consider both the options in order to arrive at the final power value. The separation should help to improve clarity and understanding of how the final numbers are generated. See **power_info_filename** for additional information.

This feature is not supported for distributed jobs. It is only meant to be run locally for debug purposes.

0: Print no extra information. (Default and recommended for production flows)

1: Add calculations for the first point in each power table.

2: Add calculations for all points in every power table.

This command must be specified before **char_library**.

power_info_filename

| | |
|--------------------------------|--|
| <code><file_name></code> | Name of log file. Default: <code>powerInfo.log</code> . |
|--------------------------------|--|

If **power_info** is greater than 0, then the information is written to the log files corresponding to this parameter setting. Because power is calculated at two places, two log files are generated:

- **powerInfo1.log:** Contains Spice to LDB equations (leakage and load power).
- **powerInfo2.log:** Contains LDB to Library equations (hidden power).

These logfiles are saved under the `extsim_deck_dir` directory unless a full path is given. The name given in this parameter is changed to separate logs for each step.

This variable must be specified before **char_library** for all calculations.

power_minimize_switching

| | |
|---------|---|
| <0 1> | Enables an alternative algorithm for power vector selection. This is only needed for multi-output cells where the outputs do not always switch together. For example: multi-bit seq, some single-bit seq with async control, and multi-bit comb. Once enabled, this is not automatically disabled for single-output cells. Default: 0 |
| 0 | Delay and power vectors are the same. |
| 1 | Delay and power vector selection are decoupled. The power vectors are selected so that the minimum number of outputs are switching. This gives the most reliable total power when the internal behavior and the behavior at each switching output are combined by power analysis tools. This increases the total number of simulations (original delay vectors + switching power vectors), but does not affect constraints or hidden power. While the number of simulations increases, it does not double. This is because the setting restricts power to vectors with the minimum number of switching outputs. This setting is only required for multi-output cells where the outputs do not always switch together (multi-bit seq, some single-bit seq with async control, multi-bit comb). |

This variable must be set before the `char_library` command.

power_model_gnd_waveform_data_mode

| | |
|---------|--|
| <0 1> | Controls modeling of <code>leakage_power</code> and <code>internal_power</code> for rails having 0v. Default: 0 |
| 0 | Allows modeling of <code>leakage_power</code> and <code>internal_power</code> for rails having 0v. |

| | |
|---|--|
| 1 | Disables modeling of <code>leakage_power</code> and <code>internal_power</code> for rails having 0v. |
|---|--|

This command must be used before the `write_library` command.

power_multi_output_binning_mode

| | |
|---------|--|
| <0 1> | Control binning methodology for cells with multiple output pins. Default: 0 Recommended: 1 |
|---------|--|

For cells with multiple outputs, it is possible that the switching condition of one output might be completely unrelated to the switching condition of another output. For example, one output pin might rise while another could rise, fall, or not change. We recommend that binning take into account timing and power, especially where hidden power and leakage power are subtracted from separate decks.

| | |
|---|---------------------------------------|
| 0 | Bin for timing effects only |
| 1 | Bin for both timing and power effects |

This variable must be used before the `char_library` command.

power_sequential_include_complementary_output

| | |
|---------|--|
| <0 1> | Specify whether to include a complementary output pin is included in sequential cell leakage and hidden power <code>when</code> conditions. Default: 1 (Include output) |
| 0 | Excludes the complementary output pin from the <code>when</code> state. |
| 1 | Includes output pin in leakage <code>when</code> states. |

This command must be used before the `write_library` command.

power_sim_estimate_duration

| | |
|-------------|---|
| <0 1 2> | Allows the user to specify a different method for the power tend. Default: 2 |
| 0 | Not implemented. |
| 1 | Estimates the power tend using the slew measurement threshold. |
| 1 | Used in 12.1 ISR3 and earlier behavior. This setting is for backward compatibility. |

This variable follows the `tran_tend_estimation` mode for consistency.

power_subtract_leakage

| | |
|---------------------|--|
| <0 1 2 3 4> | Determines what method is used to subtract leakage power from the energy numbers in the <code>internal_power</code> tables. (See Leakage Power for more information.). Default: 1 |
|---------------------|--|

Supported modes are:

0: No leakage subtract is done.

1: Leakage is determined from the separately computed leakage values found in the `.lib`. The leakage subtracted is the average of the leakage matching the initial (pre-transition) and final (post-transition) states. **Note:** With this setting, the leakage is only subtracted from `internal_power` tables when `voltage_map=0` and `pin_based_power=0`. (Default)

2: Leakage is determined from the final (post-transition) state current as measured in the energy measurement deck. The leakage will be subtracted from all `internal_power` tables (All combinations of `voltage_map` and `pin_based_power`).

3: Leakage is computed as in mode 1, but is subtracted from all `internal_power` tables (All combinations of `voltage_map` and `pin_based_power`).

4: Leakage is determined from the final (post-transition) state current as measured in a separate "leakage" measurement deck. It is subtracted from all `internal_power` tables (All combinations of `voltage_map` and `pin_based_power`). Note that an incomplete set of `leakage_groups` may lead to incorrect leakage subtraction. This option should not be used if

the separate leakage simulations do not contain complete coverage for all input and output pin post-transition states simulated for a cell.

This variable must be used before **char_library**.

power_subtract_leakage_msg_level

<0 | 1> Enable messages for state matching during leakage subtraction.
Default: 0 (Off)

This variable enables an additional check when **power_subtract_leakage** is enabled. If the leakage state matching the pre- or post-transition criteria is not an exact match, Liberate reports which state was used for subtraction.

As **power_subtract_leakage=2** is the only method where in-deck leakage values are used, settings 1, 3, and 4 can have a partially-expanded set of leakage states, which would result in subtraction of the default leakage power. It is recommended to enable this check if full leakage state coverage is expected as warnings indicate missing states.

0: No warnings reported, even when mismatches occur. (Default)

1: Warnings for mismatched or missing leakage states reported. (Recommended)

This variable must be used before **char library**.

power_subtract_leakage_mode

<0 | 1 | 2> Controls the method Liberate uses to subtract leakage power from internal_power tables.
Default: 2

0: Behavior in 2.5p2 and prior releases. Multiple outputs and WHEN states are not considered.

1: Switching power is summed for cells with multiple outputs, leakage is subtracted, and then the remainder is divided by the number of switching output pins.

2: Behavior in 1 plus consideration for the leakage WHEN state in hidden power calculations for table-based default internal_power groups. Does not apply to bitwise default groups. (Default and Recommended)

Example:

```
# Apply power_subtract_leakage_mode=2 properly.
set_var power_subtract_leakage 3
set_var power_subtract_leakage_mode 2
set_default_group -criteria {power max leakage min} -method {default table}
```

This variable must be used before **char_library**.

power_subtract_output_load

`<none | all | one | one_rise_full | one_both_half | all_rise_full | all_both_half>`

Specifies the method to be used to subtract the output load from the internal power.

Default: `all` (Subtract output load energy for all switching outputs from internal power.)

| | |
|----------------------------|--|
| <code>none</code> | Do not subtract output load power. |
| <code>all</code> | If <code>pin_based_power=0</code> , subtract CV^2 for all rising outputs. If <code>pin_based_power=1</code> , subtract $0.5CV^2$ for all rising or falling outputs. |
| <code>one</code> | If <code>pin_based_power=0</code> , subtract CV^2 for the rising pin. If <code>pin_based_power=1</code> , subtract $0.5CV^2$ for the rising or falling pin. |
| <code>one_rise_full</code> | Subtract CV^2 for the rising pin. |
| <code>one_both_half</code> | Subtract $0.5CV^2$ for the rising or falling pin. |
| <code>all_rise_full</code> | Subtract CV^2 for all rising outputs. |
| <code>all_both_half</code> | Subtract $0.5CV^2$ for all rising or falling outputs |

NOTE: When `all`, `all_rise_full`, or `all_both_half` are selected, then the internal power for all arcs originating at the switching `related_pin` will be divided by the total number of output pins. This assumes the power tool will add the internal power from all of the arcs of the `related_pin`. For example, a DFF with one clock and 2 outputs will have 2 arcs in the library, one from clock to Q and one from clock to QB. The internal power from a single clock transition will be divided between the 2 outputs.

This command must be used before **char_library**.

power_subtract_output_load_mode

<0 | 1 | 2> Specify to use final simulation output voltage or the supply rail value.
Default: 0 (Use the simulation final output voltage)
Recommended: 2

When computing the switching `internal_power`, Liberate will subtract the power used to drive the output load. Liberate will subtract CVV or 0.5CVV depending on the setting of the **power_subtract_output_load** variable. The C in CVV is the output load capacitance.

Use this variable to determine the values represented by the Vs in the CVV.

- 0 Alspice: $0.5 * C * V_{supply} * V_{final}$ (alspice)
extsim: $0.5 * C * V_{final} * V_{final}$
- 1 $0.5 * V_{supply} * V_{supply}$
- 2 $0.5 * V_{supply} * V_{final}$ (All simulators)

This variable must be set before **char_library**.

power_tend_match_tran

<0 | 1> Force power_tend to match tran_tend in SPICE simulations when using external simulators (see `char_library -extsim`) without the SKI interface.
Default: 0 (Do not force a match).

When setting up SPICE decks, Liberate can measure power to the end of the transient simulation or to a prior point. In well-behaved circuits, the current should stabilize prior to the end of the simulation so that either of the settings gives the same result. However, if a circuit exhibits trapezoidal ringing, and the purpose is to match results from different simulators or methodologies (for example, correlate stand-alone Spectre with SKI); it is recommended to use matching measurement times.

Note that trapezoidal ringing in the current waveforms lead to both inconsistent and incorrect power results. Setting this variable only addresses the inconsistent part of the problem. This problem should be corrected by updating the extracted netlist or using different simulation options (gear or related integration methods).

0: Do not match power_tend to tran_tend (Default).

1: Match power_tend to tran_tend (Recommended).

Note: When using the internal Alspice or integrated Spectre-SKI simulation engines, Liberate measures power by integrating the current to the end of the transient simulation (tran_tend). When using an external simulator, Liberate integrates the current to the estimated end of the transition (power_tend). There are cases where tran_tend or power_tend are different which can cause power correlation issues between Alspice, SKI runs, and external simulator runs. This variable forces power_tend to be equal to tran_tend, which improves power correlation. However, trapezoidal ringing in the current waveforms can lead to both correlation problems and incorrect power results. Setting this variable only addresses the correlation part of the problem. A more appropriate solution might be to update the extracted netlist or use different simulation options (gear or related integration methods).

This variable must be set before **char_library**.

predriver_waveform

<0 | 1 | 2>

Set a PWL waveform as the input driver based on averaging a linear ramp and the equivalent exponential response from an RC network.

Default: 0 (Use a linear map as the input slew.)

This parameter enables using a PWL waveform which is generated by averaging a linear input ramp and a step response of an RC network as the pre-driver.

0: Disabled (Default)

1: The linear ramp used will be limited to the supply voltage rails. Over-rides any set_driver_cell commands.

2: The linear ramp used will not be limited by the supply rail, but will continue in a linear fashion. This setting is recommended when characterizing CCS format data.

Use this analytical waveform to give a good approximation for real waveforms over a large variety of different input driver/receiver combinations, including fast slews on short wires and slow slews on long wires.

Important

1. This variable is disabled when the **predriver_waveform_ratio** variable is set to 0.
2. For library validation (LV) if a normalized waveform exists in the library, it will override the predriver_waveform setting.

This variable must be used before **char_library**. Example:

```
# Use a PWL pre-driver derived from an RC network
set_var predriver_waveform 2
```

predriver_waveform_mode

<0 | 1> When set to 1, force the slew measurement thresholds to be included in the predriver waveform.
Default: 1

This parameter instructs Liberate to include the exact transition (see `measure_slew_*`) threshold values in the input waveform. The trade-off is that the accuracy of the best curve-fit waveform may be sacrificed in order to include the exact slew measure threshold values.

0: Use best curve fit result

1: (Default) Include the exact slew measure threshold values in the input waveform. Use curve-fitting to determine the remaining points. This is also the recommended setting because it removes the interpolation errors of the input waveform from the delay measurements.

This variable must be used before **char_library**.

predriver_waveform_npts

<value> Set this variable to the minimum number of points used to store each driver waveform.
Default and Recommended: 17

This variable must be set before the `char_library` command.

predriver_waveform_ratio

<value> Specify a ratio of the linear to the exponential waveforms being merged.
Default: 0.5 (50%)

The CCS predriver waveform is created from a summation of a linear ramp and an exponential waveform. Use this variable to control the weight applied to the linear ramp waveform versus the exponential waveform. (Default value is 0.5, acceptable values are in 0.0

- 1.0). This variable is works anytime the predriver_waveform is set to a non-zero (enabled). When **predriver_waveform_ratio** is set to 0, predriver_waveform is disabled.

```
v_weighted = v_linear * predriver_waveform_ratio +  
v_exponential * (1-predriver_waveform_ratio);
```

This variable must be used before **char_library**. Example:

```
# Use a PWL pre-driver waveform with a more linear behavior  
set_var predriver_waveform_ratio 0.65
```

preserve_user_function

| | |
|---------|---|
| <0 1> | Used during Verilog/Vital modeling to preserve the function given previously instead of regenerating it from AND/OR/INV constructs. Default: 1 (preserve the function) |
|---------|---|

Liberate has the ability to generate a logic function from basic building blocks: AND, OR, INV. This is used during Verilog and Vital modeling.

0: Generate the logic function internally. This is the behavior of pre-3.1 versions of Liberate. Under certain circumstances, this can introduce additional pessimism during X-propagation tests.

1: Do not regenerate the logic function. Instead, the function must come from user_data given during Liberty modeling, or a function read in during **read_Idb** or **read_library**. This is the default and recommended behavior as of release 3.1. (Default and recommended.)

prevector_period

| | |
|---------|--|
| <value> | Sets the period used by the prevector option of the define_arc command. Each vector listed in the prevector is simulated for the time period specified by this variable. Default:1e-8 (10ns) |
|---------|--|

This variable must be used before the **char_library** command.

prevector_slew

```
<value | "min" | "mid" | "max" | "index_n">
```

Specifies the prevector slew rate to apply to the input and bidi pins in the `-prevector_pinlist` of the `define_arc` command.

Default: `0.1e-9` (100pS)

`value`

Specify the exact slew to use, in seconds.

`min`

Uses the minimum slew from the slew index (usually `index_1`).

`mid`

Uses the middle slew from the slew index.

`max`

Uses the max (largest) slew from the slew index.

`index_n`

Uses the positional slew (beginning with position 0) from the slew index.

Example

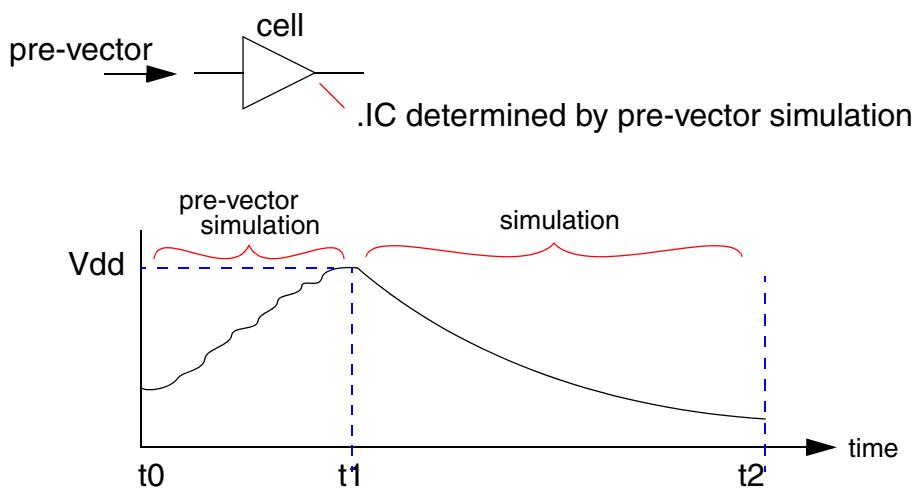
```
index_0 = min = first slew
index_1 = second slew
index_2 - third slew
```

This variable must be set prior to the `char_library` command.

prevector_voltage_waveform_mode

<0 | 1 | 2>

Selects which part of the pre-vector waveform will be stored in the Library.
Default: 1



- 0: Store waveform from **t0** to **t1** ("Incorrect" behavior, prior to release 3.2)
1: Store waveform from **t1** to **t2** (Default and Recommended)
2: Store waveform from **t0** to **t2** (Might be needed for some analog circuits.)

This variable must be set before **char_library**.

process_match_pins_to_ports

<0 | 1>

Set this to require strict port checking for the **define_cell** and **define_arc** commands.
Default: 0

0: (default) The **define_cell** command may have cell pins that are not in the subckt port list. If a -pin or -related_pin name of **define_arc** is not in the **define_cell** command for the specified cell, Liberate skips the **define_arc** command. The warning message is not printed out.

1: This enables strict pin to port mapping. Every pin in the **define_cell** command must map to a port in the subckt for the cell and every non-supply (see **set_gnd** / **set_vdd**) port must

map to a pin in the **define_cell** command. If a pin does not map one to one with a port, Liberate outputs an error message and skip the cell for characterization.

Note: In a **read_library** followed by **write_template** flow, Liberate sets this variable to 1 in the output template file to ensure that the **define_cell** pins and subckt ports match.

This variable must be used before **char_library**.

process_node

<string> This parameter is typically set by the foundry to load custom settings for a certain process technology. Access to some reporting features might be restricted.
Default: ""

This variable must be used before the **char_library** command.

ramp_vsrc

<0 | 1> Enables supply ramping.
Default: 0

Set this variable to enable supply ramping. This can help with DC convergence related issues when using an external simulator. When set to 1, this variable will tell Liberate to always ramp the supply. This ramping will be enabled for all data acquisition and will override the **leakage_ramp_vsrc** variable. The power supply will ramp from 0s to ½ sim_init_duration.

This variable must be used before **char_library**.

rc_floating_cap_mode

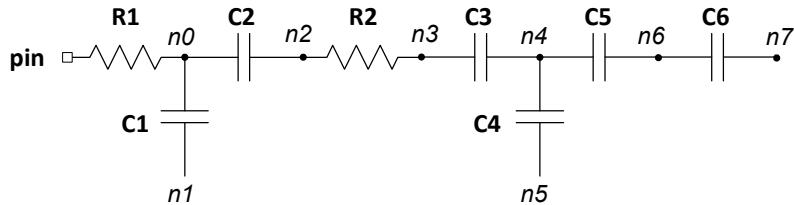
<0 | 1 | 2 | 4> Controls the handling of floating-cap nodes.
Default: 1

To assist with DC convergence and prevent a drastic increase in simulation run-time, Liberate can add terminating resistors to floating capacitor nodes. A floating node or network is one that is resistively connected: it has coupling capacitors to other nodes, but has no path (leakage or driving) to power or ground.

Virtuoso Liberate Reference Manual

Liberate Parameters

In the following circuit example, dc-path (1/gmin) resistors will be tied from node "0" to nodes: n1, n2, n4, n5, n6, n7 (n0 is not a floating node and n3 is current equivalent with n2.)



0: Disable automatic detection of floating nodes.

1: Adds a large dc-path resistor (fixed at 1e+14 ohms) from floating cap nodes to node 0. (Default)

2: Write out (and overwrite any existing) floating cap net names to a file named `<cell_name>.ics` in the current working directory. The user can examine and include this file to set .IC on floating nodes.

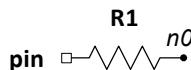
For Spectre, the file contents will look like this:

```
ic n1 = 0
ic n7 = 0
```

For other simulators:

```
.ic v(n1) = 0
.ic v(n7) = 0
```

4: Remove all floating caps and their direct connected RC-network during simulation. Using the example above, the result will be this:



This variable must be used before **char_library**.

rcp_cmd

`<scp | rcp | cp>` The file copy command to use for copying files from the host to the client machine when using distributed parallel processing.
Default: `scp`

This variable must be used before the `char_library` command.

rdb_checkpoint_dir

`<directory_name>` Full path to the directory where the rdb checkpoint file is stored.
Default: (there is no default)

If set, Liberate can store per-arc characterization data into a recovery database (RDB). This parameter is used to specify the directory where the RDB checkpoint files will be stored. This enables a per-arc recovery flow at the cost of additional drive space and runtime. By default, this data is not saved.

The `rdb_checkpoint_dir` can be useful when characterizing extremely large IO cells that have very long runtimes. This is because Liberate will load the characterized arc data and proceed to characterize the missing arcs. However, there is no version control to prevent Liberate from mixing simulation data created using different settings.

We recommend *not* using this variable.

This variable must be set before **char_library**.

Example

```
set_var rdb_checkpoint_dir /NFS/work/rundir/RDB
```

rdb_exit_if_source_differ

`<0 | 1>` Exit if mismatched PVT corners are found.

The RDB flow is enhanced to ensure that exactly the same script/settings are used before restoring characterized values from RDB. This ensures data consistency is maintained from run to run even if the user erroneously sets the `rdb_checkpoint_dir` variable to the same location.

The default and recommended value is 1.

This variable must be used before **char_library**.

Example:

```
# To enable this check:  
set_var rdb_exit_if_source_differ 1
```

rechar_chksum

`<string>` Name of the Linux file checksum utility. Must be in \$PATH.
Default: " " (Unset).

This variable specifies the Linux utility that Liberate uses to generate a unique checksum for each netlist file read by Liberate. The checksum utility is used during the incremental recharacterization flow to determine if a new netlist has been loaded for a cell. The **rechar_chksum** utility is enabled in the incremental recharacterization flow when the **read_Idb** command is called with the **-incremental** option and is enabled by the **read_Idb** command arguments **-incremental** and **-check_spice**.

The recommended checksum utility is md5sum. There is no default; the user must set this.

This variable must be used before **char_library**.

Example

```
set_var rechar_chksum md5sum
```

removal_glitch_peak

`<value>` Glitch height as a ratio of supply used in characterizing removal constraints.
Default: -1 (Use the same value as constraint_glitch_peak)

This parameter is used to specify the maximum size of a voltage glitch permitted on the constraint output pin before an arriving signal is deemed to fail a removal constraint. When this variable is not set, then removal constraint characterization will follow the constraint_glitch_peak value.

This variable must be used before **char_library**. Example:

```
# Set removal glitch peak to 5%
set_var removal_glitch_peak 0.05
```

res_merge

<0 | 1> Allows Liberate to merge resistors to improve simulation run time.
Default: 1 (on)
Set to 0 to disable.

This variable must be used before the `char_library` command.

res_open_tol

<value> Controls to filter out (and leave the connection open) any resistor that is larger than this variables setting, only if `res_open_tol` is set to a value larger than 0.

This variable must be used before the `char_library` command.

res_tol

<value> Threshold value to short resistors.
Default: 0.0011

Any resistors below this limit will be shorted (set to zero ohms), filtering out resistors that are suspected of being extraction artifacts.

A netlist containing very small resistors could produce an unstable simulation matrix, resulting in unnecessarily long simulations or inconsistent results. Therefore it is important that the extractor be set up to generate netlists that are representative of the actual cell as it will be used in the design, and are well-behaved in SPICE simulation.

For certain extraction methodologies, it may be necessary to set this parameter to 0.1 for consistent results. Setting this to 0 will disable this functionality, *however*, the quality of the simulation results may deteriorate if the netlist contains the instabilities mentioned. If some simulations fail after changing this parameter, try setting minimum resistance control options in an external simulator to see if the results improve (e.g. `resmin=0.001` in HSPICE).

This variable must be used before `char_library`.

reset_leakage_current_mode

`<0 | 1>` Turn any negative leakage value to zero.
Default: 0 (Zero negative leakage for Library Compiler compliance)

Controls when leakage will be reset to 0 (zero) amps.

0: Reset leakage current greater than 0 amps to 0 for `primary_ground` or `backup_ground` pg pins. Reset leakage current less than 0 amps to 0 for all other pg pins. (Default and recommended)

1: Do not reset leakage current when the `pg_type` is `primary_power`, `primary_ground`, `backup_power`, or `backup_ground`.

This variable can be used after **char_library**.

reset_negative_leakage_power_value

`<value>` Specifies the value (in Watts) that the negative leakage power value is reset to. The reasonable value for this variable is zero or a small positive value at the leakage power level. This variable is applied when the variable `reset_negative_leakage_power` is enabled.
Default: 0.0

Note: The value should be non-negative.

This variable must be set prior to the `write_library` command.

reset_negative_constraint

`<0 | 1>` Turn any negative constraint value to zero.
Default: 0)

By default, Liberate will output a negative constraint (setup, hold, recovery, and removal) value if the simulator measurement returns a negative value. Setting **reset_negative_constraint** will cause the output library to have the negative values replaced by zeros. This reset will be applied after **set_constraint -margin**. Note: This variable will not be applied to mpw constraints.

This variable can be used after **char_library**.

reset_negative_delay

<0 | 1> Turn any negative delay or transition values to zero.
Default: 0

By default Liberate will output negative delay and transition values if the simulator measurement returns a negative value. Setting **reset_negative_delay** will cause the output library to have the negative values replaced by zeros.

This parameter may be used after **char_library**. Examples:

```
# Turn all negative delays and constraints to zeros
set_var reset_negative_delay 0
set_var reset_negative_constraint 0
```

reset_negative_leakage_power

<0 | 1> Turn any negative power sum to zero.
Default: 0

Set this variable to **1** to avoid using negative leakage currents in power calculation when the leakage has been zeroed out for LC compliance. The recommended setting is **1**.

When the variable **reset_negative_power** is set to **1**, the variable **reset_negative_leakage_power** will have no effect and all negative power, including leakage will be reset.

This variable must be set before **char_library**.

reset_negative_power

<0 | 1> Turn any negative power sum to zero.
Default: 1

By default, Liberate will reset negative power so that the **rise_power + fall_power** should always be greater than or equal to zero.

Dynamic power entries are processed when "power pairs" are identified. (Power pairs have matching *when* conditions, same input/output pins, but different directions, ie: rise vs. fall.) The sum of corresponding entries between rise/fall pairs is checked, and if it is less than zero (e.g., $R2+F2 < 0$), then one of the following actions is performed:

```
( Rn < 0 ) & ( Fn < 0 ) then Rn = Fn = 0  
( Rn < 0 ) & ( Fn > 0 ) then Rn = ( -1 * Fn )  
( Rn > 0 ) & ( Fn < 0 ) then Fn = ( -1 * Rn )
```

If an arc does not have a matching rise/fall "power pair", then the above procedure is not performed.

Note: when characterizing CML logic, it may be necessary to disable the resetting of negative power. Also, negative *leakage_power* entries will be reset to 0.

0: Do not reset negative power; allow negative values for power.

1: If the sum of rising and falling power is negative, whichever power is negative will be adjusted so the sum of the rising and falling power is zero. If they are both negative, they both will be set to zero. *See algorithm above.* (Default)

2: Any power that is negative will be set to zero.

This variable must be used before **write_library**.

resolve_collision

| | |
|-------------|---|
| <0 1 2> | Controls simulation engine used to resolve collision vectors with the requested spice engine. Default and Recommended: 2 (Use <code>-extsim</code> if specified) |
| 0 | Disables simulation-based collision resolution. |
| 1 | Use Alspice to resolve collisions. |
| 2 | Use the external simulator if <code>-extsim</code> is used in <code>char_library</code> . |

Collision vectors occur when a node has a path to vdd and to gnd. In these cases, Liberate can simulate the vector in spice to determine if the output results in a digital behavior.

This variable must be specified before the `char_library` command.

retry_count

<value> Specifies the number of retries for a particular arc, pausing 5 seconds between retries.
Default: 1 (Instructs Liberate to retry failing simulations one time)

This variable must be set before the `char_library` command.

retry_count_file_operation

<value> Specifies how many times to retry when an attempt to move, copy, or rename a file fails due to network or LSF issues. A valid value is an integer between 0 and 60, both inclusive.
Default: 10

This variable must be set before the `char_library` command.

rsh_cmd

<ssh | rsh> The shell command to use for accessing a remote client when using distributed parallel processing.
Default: ssh

The above parameters are used to control the interface to remote clients when using distributed parallel processing. Before using parallel processing, make sure that the server machine (the machine Liberate is to be invoked from) can perform either an **rsh** or an **ssh** and an **rcp** or a **scp** to each client machine without requiring a password or passphrase. The **rsh_cmd** command string can reference the current client and the command to invoke Liberate with by using **%M** (machine) and **%C** (command). (See [Distributed Processing](#) for list of all "percent" (%) variables.) In addition, command line options that appear after the Liberate Tcl file name can be passed into the **rsh_cmd** string by using **%O** (options). These substitutions can be useful if your system is using a job queuing system.

This variable must be used before `char_library`.

Example

```
# Set the shell and copy variables for distributed runs
# set remote file copy to rcp instead of scp
```

```
set_var rcp_cmd rcp
# set remote shell to rsh instead of ssh
set_var rsh_cmd rsh
```

scale_load_by_template

`<0 | 1>` Set this parameter to scale the load indices using the values defined in the template when generating indices using the **auto_index** argument of **char_library**.
Default: 0 (off)

This parameter turns on scaling of the load indices created by the **auto_index** option of **char_library**. Normally, when the **auto_index** option of **char_library** is enabled, the intermediate indices are determined using a geometric sequence. When this control variable is set, the indices are determined using the following formula:

$$\text{index_value}(i) = (\text{max_load} - \text{min_load}) * \text{template_load_index_value}(i) + \text{min_load}$$

where:

$i = 2, \dots, \text{number_of_indexes} - 1$

template_load_index_value is given in a **define_template** command where the index values are treated as a ratio between 0 and 1. See **scale_tran_by_template** for an example.

This variable must be used before **char_library**.

scale_tran_by_template

`<0 | 1>` Set this parameter to scale the transition indices using the values defined in the template when generating indices using the **auto_index** argument of **char_library**.
Default: 0 (off)

This parameter turns on scaling of transition indices created by the **auto_index** option of **char_library**. Normally, when the **auto_index** option of **char_library** is enabled, the intermediate indices are determined using a geometric sequence. When this control variable is set, the indices are determined using the following formula:

$$\text{index_value}(i) = (\text{max_tran} - \text{min_tran}) * \text{template_tran_index_value}(i) + \text{min_tran}$$

Where:

i=2,...,number_of_indexes-1

The *template_tran_index_value* is given in a **define_template** command where the index values are treated as a ratio between 0 and 1.

This variable must be used before **char_library**. Example:

```

define_template -type delay \
  -index_1 {0 0.1 0.2 0.4 0.7 0.9 1} \
  -index_2 {0 0.2 0.4 0.8 1} delay_template_7x7

define_cell \
  -input { A } -output { X } \
  -delay delay_template_7x7 INV_1

set_var min_transition 6.6e-12
set_var max_transition 0.6e-9
set_var scale_load_by_template 1
set_var scale_tran_by_template 1
char library -auto index

```

scan_dummy_include_leakage_power

<0 | 1> Request leakage power in the scan dummy.
Default: 0

Liberate can output leakage group data into the scan dummy cell. Set this parameter to **1** to enable this feature. Default: **0** (do not output leakage power in the scan dummy)

This variable must be used before `char library`.

sdf cond equals

`<string>` Specify the `sdf_cond` attribute style.
Default: " "

Use this variable to specify the format for the *sdf_cond* construct to be used in output models such as Liberty, Verilog, and Vital models. This variable supports the following values: "==" , "== logical" , "== binary" , "===" , "===" logical" and "===" binary" . This variable is identical to

and will replace the **sdf_cond_equals** argument to **write_library**. It allows the **sdf_cond_equals** modification to be applied to all output modeling routines including **write_library**, **write_verilog** and **write_vital** allowing all models to use the same modeling format for *sdf_cond*.

The **write_library -sdf_cond_equals** command will continue to be supported for backward compatibility, but if used, it will set this variable such that any subsequent **write_verilog** or **write_vital** commands will also have this modification. This variable is sequence dependent. It only impacts the models written after setting this variable.

This variable can be used after **char_library**.

Example

```
# The following will result in an inconsistency between the .lib and .v model files.

write_library no_sdf_equals.lib
set_var sdf_cond_equals "==" 
write_verilog sdf_equals.v

# The following will create consistent .lib and .v model files.

set_var sdf_cond_equals "==" 
write_library sdf_equals.lib
write_verilog sdf_equals.v
```

sdf_cond_postfix

| | |
|-----------------------|--|
| <string> | The postfix to use for complex conditional sdf_cond attributes on sequential cells. Default: " " |
|-----------------------|--|

This parameter sets the postfix that Liberate uses for naming complex **sdf_cond** attributes on sequential cells (flip-flop or latch). If an **sdf_cond** attribute is complex (that is it has two or more operands) it will be replaced by a name of the format \${sdf_cond_prefix}#\${sdf_cond_postfix} (see the variable **sdf_cond_prefix**) where the # is a unique number for each **sdf_cond** within the cell. The **sdf_cond** attributes are used for writing conditional timing arcs in Liberty, Verilog and Vital formats (see **write_library**, **write_verilog** and **write_vital**).

This variable must be used before **char_library**.

Example

```
# Set the sdf_cond postfix
set_var sdf_cond_postfix "int_cond"
char_library
write_library my.lib
```

sdf_cond_prefix

| | |
|-----------------------------|---|
| <code><string></code> | The prefix to use for complex conditional <code>sdf_cond</code> attributes on sequential cells. Default: adacond |
|-----------------------------|---|

This parameter sets the prefix that Liberate uses for naming complex **sdf_cond** attributes on sequential cells (flip-flop or latch). If an **sdf_cond** attribute is complex (i.e. it has two or more operands) it will be replaced by a name of the form `${sdf_cond_prefix}#${sdf_cond_postfix}` (see the variable `sdf_cond_postfix`) where the # is a unique number for each **sdf_cond** within the cell. The **sdf_cond** attributes are used for writing conditional timing arcs in Liberty, Verilog and Vital formats (see **write_library**, **write_verilog** and **write_vital**).

This variable must be used before **char_library**.

Example

```
# Set the sdf_cond prefix
set_var sdf_cond_prefix "int_cond"
char_library
write_library my.lib
```

sdf_cond_style

| | |
|--------------------------------|--|
| <code><0 1 2></code> | Specify the <code>sdf_cond</code> style. Default: 0 |
|--------------------------------|--|

The `sdf_cond` attribute for complex conditional constraint arcs must be represented as a "variable" that is the functional equivalent to the `when` condition.

0: (Default). Liberate generates a unique variable for each different complex condition (has merged states as in "state1 || state2") as follows:

`"${sdf_cond_prefix}#${sdf_cond_postfix}"` (see the `sdf_cond_prefix` and `sdf_cond_postfix` variables)

- 1:** Liberate generates an SDF variable name from the "when" for all complex conditions and for all conditional constraints such that the logic and operators will be replaced with `_AND_`, logic `or` with `_OR_`, logical `not` with `_NOT_`, `("` with `OP_`, and `")` with `CP_` (see the variable `sdf_cond_variable_map`). This constructed variable name is prefixed with the value of the variable `sdf_cond_prefix` and postfixed with the value of the variable `sdf_cond_postfix`.
- 2:** Liberate generates a constructed SDF variable name for all arcs using the naming convention as described in the setting of 1 above.

Example:

`"A * (B | !C)"` becomes `"adacond_A_AND_OP_B_OR_NOT_C_CP"`

Note that **sdf_cond_style 1** applies to all conditional constraints while **sdf_cond_style 0** only applies to complex `when` conditions that have multiple operands.

This variable can be set after the `char_library` command.

sdf_cond_variable_map

{list of paired strings}

Specifies a list of string pairs that map strings in the arc "when" to alternative strings in the arc `sdf_cond`.

Default: { " " " " (OP_) _CP ! _NOT_ && _AND_ || _OR_ }

This variable must be set before the `char_library` command.

sdf_logic_and

`"string"` Sets the characters to use for denoting logic AND for `sdf_cond` attributes.
Default: " & " (*Notice the space on both sides of &*)

This variable must be set before the `char_library` command.

sdf_logic_not

`"string"` Sets the characters to use for denoting logic NOT for `sdf_cond` attributes.
Default: " ~ "

This variable must be set before the `char_library` command.

Examples

```
# Set the logic AND, OR and NOT to &&, || and !
set_var sdf_logic_and "&&"
set_var sdf_logic_or "||"
set_var sdf_logic_not "!"
```

sdf_logic_or

| | |
|----------|--|
| "string" | Sets the characters to use for denoting logic OR in <code>sdf_cond</code> attributes. Default: " " <i>(Notice the space on both sides of)</i> |
|----------|--|

This variable must be set before the `char_library` command.

server_timeout

| | |
|---------|---|
| <value> | Specify unresponsive client message interval. When the server is waiting for a response from a client, it prints out a message when it has waited for the timeout specified by this variable. The message repeats every timeout duration. Default: 10800 (seconds) |
|---------|---|

This variable must be used before the `char_library` command.

set_var_failure_action

| | |
|-------------------|---|
| <warning error> | Notifies the <code>set_var</code> command how to consider a failure. Default: warning |
| error | When a <code>set_var</code> fails, an error message is issued and subsequent commands which would result in characterization or library generation (for example, <code>char_library</code>) are suppressed. Subsequent <code>set_var</code> commands are still allowed so they can be checked for correctness. |

| | |
|---------|--|
| warning | A warning is issued when <code>set_var</code> fails. The failed <code>set_var</code> is ignored and execution continues. |
|---------|--|

This variable must be set prior to any other `set_var` command.

Sample Message:

The simulation on client <client> has been running for 180 minutes. The simulation might be stalled, waiting for a license, or terminated. If the simulation has terminated and the client is not responding, you might need to restart the characterization job. Check the simulation status, for example, check `sim.lis` file if using external simulator.

sim_default_engine

| | |
|----------|---|
| <string> | Enables you to set the default engine for simulations. If you want to keep using als spice, you can set it to "alspice". The variable is ignored if <code>extsim</code> option is specified in the <code>char_library</code> command. Default: "ski" |
|----------|---|

This variable must be set before the `char_library` command.

sim_duration

| | |
|---------|---|
| <value> | Sets to the maximum allowed simulation time to be used during spice simulation. This variable puts a limit on the total simulation duration. It acts as a global override. Default: 1e-7 |
|---------|---|

This variable must be used before the `char_library` command.

Example

```
# Set the parameter 'sim_duration' to 1e-5 seconds
set_var sim_duration 1e-5
```

sim estimate duration

<0 | 1> The maximum simulation duration.
Default: 1 (Estimate the simulation duration)

Use this variable to disable the algorithm in Liberate that will estimate the expected simulation duration. Set this variable to **0** to disable the estimation. Once disabled, the simulation will run for exactly the time specified by the control variable **sim_duration**.

This variable must be used before **char library**.

Example

```
# Set the parameter 'sim_duration' to 1e-5 seconds
set_var sim_estimate_duration 0
```

sim init condition

```
<hybrid | ic | ic_except_dc | ic_except_dc_plus_leakage>
```

Specify initial condition method when presenting spice decks to the simulator.
Default: hybrid

The **ic** option requests Liberate to always use the spice *.ic* command in generated spice decks. This option is not compatible with Leakage and CCS DC sweep data characterization. The **ic_except_dc** option requests Liberate to use *.ic* for all generated spice decks except those which use a DC analysis. The **ic_except_dc_plus_leakage** option requests Liberate to use *.ic* to perform leakage characterization. The **hybrid** option is the default. When set to **hybrid**, Liberate will use *.ic* for any floating nodes and *.nodeset* everywhere else.

This variable must be used before **char_library**. Example:

```
set var sim init condition ic except dc
```

sim init condition estimation mode

<0 | 1> Specify initial condition method.
Default: 1

Use this variable to select the algorithm that Liberate will use to initialize internal nets. When set to **0**, Liberate may set incorrect signal swing rails for internal nets when a cell has multiple supplies. When set to **1**, a modified algorithm seeks to remedy this issue by propagating supplies twice. In the first pass, vdd/gnd will only propagate thru pmos and nmos channels. In the second pass, Liberate will propagate vdd/gnd thru all channels for those nets that didn't get set in the first pass. (Default: **1**). To revert to release 2.3p2 and prior release behavior, set this variable to **0**.

This variable must be used before **char_library**.

sim_init_duration

<value> The initialization simulation duration.
Default: 500e-12 (500ps)

Liberate can provide initial conditions, as needed for a vector to function properly to spice. The spice engine will simulate for the specified duration and report the internal node voltages to Liberate to use during subsequent spice simulations of the vector. This parameter will only have an effect when used in combination with **sim_use_init_duration**.

When the **ramp_vsrc** variable is set, the **sim_init_duration** variable will determine the amount of time Liberate will simulate to determine the internal node voltages. If **ramp_vsrc** is set, then this variable must also be set. The **leakage_sim_duration** variable will override this variable for all leakage simulations.

This variable must be used before **char_library**.

Example

```
# Set the parameter 'sim_init_duration' to 1e-9 seconds
set_var sim_init_duration 1e-9
```

sim_power_duration_extend

<value> Specify an increment to add to the power simulation.
Default: 0 (Specified in seconds)

Use this variable to increase the spice dynamic power simulation duration by a user defined increment in seconds. This extension is only applied when combined with **power_subtract_leakage** set to **2**. The power simulation duration will be capped at the value of the **sim_duration**.

This variable must be used before **char_library**.

Example

```
# Increase the power simulation duration by 10pS
set_var sim_power_duration_extend 100e-12
```

sim_use_init_duration

<0 | 1 | 2> Turn on the **sim_init_duration** algorithm.
Default: 0

Liberate can provide initial conditions to spice, as needed, for a vector to function properly. The spice engine can be used to validate the initial conditions using a transient simulation. The spice engine will simulate for the **sim_init_duration** and report the internal node voltages to Liberate which will use them during subsequent spice simulations of the vector. Set this variable to **1** to use this algorithm. This can help to work around issues where the spice engine is having problems finding a dc solution. When set to **2**, the pre-simulation for capturing internal conditions for all internal nodes will have its supply ramped from 0 to vdd, similar to **leakage_ramp_vsrc 1** – except this is done for timing simulations. Default: **0** (do not use spice transient solution)

This variable must be used before **char_library**.

Example

```
# Set the parameter 'sim_use_init_duration'
set_var sim_use_init_duration 1
```

simultaneous_switch

<0 | 1> Enables simultaneous input switching analysis.
Default: 0

Use this variable to enable analysis for simultaneous switching inputs. Any cell that has any of the following will **not** be analyzed for simultaneous switching inputs: *ff* group, *latch* group, *statetable* group, *clock* attribute, *three_state* attribute, *direction* attribute of *inout*, *rise_constraint* group, and *fall_constraint* group.

Set to one ("1") to enable simultaneous input switching for combinational gates. When enabled, Liberate automatically finds one or more worst or best case simultaneous input

switching vectors for each arc. Large fan-in cells will have a substantial performance penalty. It is recommended to set the variable `conditional_arc` to 0 to substantially reduce simultaneous switching characterization time.

Use the **set_default_group** command to set the default timing criteria to a minimum or maximum during (simultaneous switching) characterization to control the reported timing values.

This variable must be used before **char_library**.

Example

```
# Enable simultaneous switching analysis.  
set_var simultaneous_switch 1
```

simultaneous_switch_from_cell_when

| | |
|---------|---|
| <0 1> | Enables automatic dual input switching. Default: 1 |
|---------|---|

When a cell has a set of inputs with logic relationships between them, such as one_hot or dual inputs, Liberate can automatically find one or more arcs with simultaneous switching inputs that satisfy the `-when` argument of the **define_cell** command for each arc. You must specify the `-when` argument of the **define_cell** command to define the logical relationships of the cell pins.

0: Do not look for simultaneous switching side pins.

1: Look for simultaneous switching side inputs such that the **define_cell** and **define_arc** `-when` states are met both before and after all of the pins switch.

If `simultaneous_switch_from_cell_when` is set, Liberate determines a minimum set of pins necessary to switch to allow the primary pin to switch while still adhering to the `-when` argument of the **define_cell** command. This is be as many pins as necessary. There are no additional controls required or provided.

Two commands are available to assist with adding the when states to the **define_cell** command. They are: **one_hot** and **one_cold**.

This variable must be used before **char_library**.

Example

```
define_cell -when "ck^ckb" ...
```

```
# Enable simultaneous switching analysis.  
set_var simultaneous_switch_from_cell_when 1
```

simultaneous_switch_offset

<value> Offset to be used between the related_pin and the simultaneously switching side inputs.
Default: 0 (seconds)

Set this variable to specify a timing offset to be applied between related_pin and the simultaneous switching side inputs. The offset must be greater than or equal to zero so that the side inputs will switch with or after related_pin. All switching side inputs will use the same transition time.

This parameter will be honored at all times, even when the parameter **simultaneous_switch** is not enabled. This parameter has a default value of **0** so Liberate will not offset MIS (Multiple Input Switching) signals automatically.

This variable must be used before **char_library**.

Example

```
# Set the side inputs to switch with an offset of 5pS.  
set_var simultaneous_switch_offset 5e-12
```

simultaneous_switch_use_arc_when

<0 | 1 | 2> Defines how the **-when** argument of the **define_arc** command is applied to simultaneous input switching detection.
Default: 0 (ignore **define_arc when**)

- if **simultaneous_switch_from_cell_when** is enabled (1 or 2).
- if **simultaneous_switch_from_cell_when** is 0, this variable has no effect.
- if **simultaneous_switch_from_cell_when** is enabled, then this variable works as follows:

0: (Default). In this case, the following two conditions apply:

If `simultaneous_switch_from_cell_when` is 1, then no pin listed in the `define_arc` -when is permitted to switch.

If simultaneous_switch_from_cell_when is 2, then the **define_arc** -when is completely ignored if the side pin is switching.

- 1: The **define _arc** -when will be observed before the side pins switch.
 - 2: The **define arc** -when will be observed after the side pins switch.

Note: The recommended value of this variable is 0.

This variable must be used before **char** library.

Example

```
# Set the parameter 'sim_use_init_duration'  
set var simultaneous switch use arc when 1
```

simultaneous switch worst vector

<1 | 4> Number of slew/load simulations used in determination of worst simultaneous switch vectors.
Default: 1 (use mid-point of slew/load indices for binning).

Set this variable to 4 to revert to release version 2.3 (and earlier) behavior where 4 slew/load simulations were used. These 4 slew/load simulations are min_slew/min_load, min_slew/max_load, max_slew/min_load and max_slew/max_load.

This variable must be used before **char** library.

Example

```
# Use 4 slew/load simulations for binning worst simultaneous
# switching vectors
set var simultaneous switch worst vector 4
```

ski alter mode

<1 | 2 | 3 | 4> Controls the method for generating SPICE decks used in constraint acquisition.
 Default: 3

Liberate can employ several different methods for generating SPICE decks during the bisection search for constraint acquisition. Due to numerical differences within the meta-stable region, these can result in slightly different constraint results.

This variable has an effect only if `ski_enable=1`. See [External Simulator Options and Settings](#) for recommended settings.

Note: This setting might affect performance. Generally, `ski_alter_mode=2` should be faster.

- 1:** Use external simulator compatible settings (recommended for matching Spectre, HSPICE, etc.)
- 2:** Use Alspice-compatible settings (recommended for matching Alspice)
- 3:** Use external simulator compatible settings (recommended for matching Spectre, HSPICE among others) similar to the setting of 1 above, but with optimizations to improve runtime (default).
- 4:** Use Alspice-compatible settings (recommended for matching Alspice) similar to the setting of 2 above, but with optimizations to improve runtime.

This variable must be used before **char_library**.

ski_clean_mode

| | |
|--------------------------------|---|
| <code><0 1 2></code> | Enables clean-up of the shared memory environment used by SKI. Default: 0 (not enabled). |
|--------------------------------|---|

Enabling `ski_clean_mode` directs Liberate to clean up semaphore resources that are owned by you but are not in active use. This feature can augment or replace similar capabilities enabled in `altos_init` files. The clean-up is performed by running the script in `$ALTOSHOME/bin/clean_sm.sh`, which may be independently modified or updated, if required.

When `ski_enable=1`, Liberate and Spectre communicate through the Spectre Kernel Interface (SKI) using shared memory and semaphores. Because these resources are both system-wide and limited, if insufficient resources are available to run SKI, then Liberate disables SKI and use the standard Spectre interface. While this yields accurate results, there is a performance penalty.

- 0:** Perform no clean-up. (Default)
- 1:** Perform standard cleanup via adding "-a" to the clean-up command. (Recommended)
- 2:** Same as 1, but log additional debugging information via adding "-d" to the clean-up

command. This is useful to debug situations where SKI is consistently being disabled on certain farm machines.

Because these resources follow the standard rules of Unix permissions, a user cannot clean up resources belonging to another user. The resources must either be cleaned up by the owner or by root.

This variable must be used before **char_library**.

ski_compatibility_mode

<0 | 1> Enables consistency between standalone Spectre and Spectre-SKI.
Default: 0

0: Each individual SKI control variable is set manually.

1: Set the following variables to corresponding values:

- ❑ ski_power_subtract_output_load_match_extsim 1
 - ❑ alspic_e_power_subtract_output_load_match_extsim 1
 - ❑ ski_alter_mode 3
 - ❑ ski_mdlthreshold_exact 1
 - ❑ capacitance_save_mode 0

The dependent variables listed above are under Liberate control and cannot be overridden while ski compatibility mode is set to 1.

This variable must be used before **char library**.

ski_enable

| | |
|----------------------------|---|
| <code><0 1></code> | Enables the Spectre Kernel Interface, a tight integration between Liberate and Spectre that provides a substantial performance improvement. See External Simulator Options and Settings for recommended settings. SKI can be enabled or disabled for a specific arc (see <code>set_var</code>) as long as it is enabled globally. If SKI is globally disabled, it cannot be enabled locally using <code>set_var</code> . Also, then the setting overrides any arc-specific settings with a warning Default: 0 (off) |
| 0 | Disables the Spectre high-performance (SKI-based) interface. |
| 1 | Enables the Spectre high-performance (SKI-based) interface. |

Important:

- Requires Spectre version **MMSIM 10.1 ISR20**, or **MMSIM 11.1 ISR8**, or equivalent newer versions. See the `${ALTOSHOMEDIR} / README` file for the currently supported version of Spectre.
- To use Spectre as the circuit simulator, the "**char_library -extsim spectre**" option must be set.
- SKI is only supported in the parallel packet flow.

Example

```
set_var ski_enable 1
set_var -cell DFF -type mpw -pin RN ski_enable 0
```

This variable must be used before the `char_library` command.

ski_mdlthreshold_exact

| | |
|----------------------------|---|
| <code><0 1></code> | Enables the Spectre option <code>mdlthresholds=exact</code> when <code>ski_enable=1</code> . Default: 0 (not enabled). |
|----------------------------|---|

SPICE waveforms normally use given relative and absolute tolerances to determine the time-steps in a waveform. The Spectre option `mdlthresholds=exact` has the ability to place a

time-point exactly on a threshold crossing (for example, delay and slew thresholds) at the expense of some performance. This option is set to `exact` by default in Spectre, but is set by default to `interpolated` in SKI.

0: SKI will use a setting similar to Spectre's `mdlthresholds=interpolated` behavior. (Default)

1: SKI will use a setting similar to Spectre's `mdlthresholds=exact` behavior. (Recommended for consistency with standalone Spectre default behavior). For more details and additional settings, see the [Correlation between stand-alone Spectre & SKI](#) section.

This variable must be used before **char_library**.

ski_power_subtract_output_load_match_extsim

`<0 | 1>` Lets you switch between charge-based or voltage-based power calculation.
Default: 0 (use voltage-based power for subtraction)

Liberate can calculate power based on charge or voltage. External simulations always use voltage-based power calculations. A well-designed circuit should yield similar results when switching between the two methods.

To remove power correlation differences on sensitive circuits based on power calculation methodology, it is recommended to use the same approach for both SKI and external simulations.

0: SKI will use charge-based power calculation. (Default)

1: SKI will use voltage-based power calculation. (Recommended)

This variable must be set prior to **char_library**.

ski_reset_cnt

`<integer>` Force reset of Spectre-SKI memory usage.
Default: 50000 (Reset after 50,000 iterations)

When SKI is enabled, Liberate will start a separate Spectre simulation process. The memory footprint of this process can change over time (both increasing and decreasing). If the process grows too large (more than available memory), then performance can degrade. This variable will force a memory reset after the specified simulation iteration count. This variable

should be only used as a safety measure in the case when memory usage grows extremely fast or for memory debugging. Small values will increase runtime. Values below 1000 are not recommended.

This variable only has an effect if **ski_enable=1** (Spectre-SKI is enabled). See [External Simulator Options and Settings](#).

This variable must be used before **char_library**.

ski_use_large_memory

<value> Set this variable to a higher value if you get a message similar to the following.

Number of simulation time points at t=8.48005e-08 (s) exceeds size of pre-allocated waveform data storage.

Note: This variable is used when SKI flow is enabled. (see [ski_enable](#)).

The allowed values for this variable are 0 to 10.

Default: 3

These variables must be used before the **char_library** command.

skip_nfs_sync

<0 | 1> Forces an NFS file system synchronization by including multiple commands separated by a semicolon (;) on the command line.
Default: 0

0 Submit runs to the batch tool using:

`ls -l dirname; liberate`

1 Submit runs to the batch tool using:

`liberate`

This variable may be needed when using a batch system that cannot accept a batch command containing multiple commands separated by a semicolon

This variable must be set before the **char_library** command is run.

slew_lower_fall

`<value>` The % point on the cell output waveform to output falling output transition times to.
Default: 0.2 (20%)

These variables must be used before the `char_library` command.

slew_lower_rise

`<value>` The % point on the cell output waveform to output rising output transition times from.
Default: 0.2 (20%)

These variables must be used before the `char_library` command.

slew_normalize

`<0 | 1>` Report the normalized or measured slew.
Default: 1 (Report normalized slew)

By default, Liberate will measure the output slews using the `measure_slew*` parameter values and then normalize this measured value to the `slew*` reporting values. To have Liberate report the actual measured value, set this parameter a **0**.

This variable must be used before `char_library`.

slew_upper_fall

`<value>` The % point on the cell output waveform to output falling output transition times from.
Default: 0.8 (80%)

slew_upper_rise

<value> The % point on the cell output waveform to output rising output transition times to.
Default: 0.8 (80%)

The above parameters are used to control how output transition times are stored in the delay tables. These parameters can be set differently from the equivalent measurement thresholds, in which case the measurements are normalized to fit the appropriate output transition slew thresholds.

If these parameters are set symmetrically, the output library that is generated will include the **slew_degrade_from_library** attribute and the **slew_*_threshold_pct_*** attributes will be set to the equivalent **measure_slew_*** variables (x 100). If they are not symmetrical then the **slew_degrade_from_library** attribute will be omitted and the **slew_*_threshold_pct_*** attributes will be set to the equivalent **slew_*** variables (x 100).

These variables must be used before **char_library**.

Example

```
# Set the output transition thresholds to 10-90%
set_var slew_lower_fall 0.1
set_var slew_upper_fall 0.9
set_var slew_lower_rise 0.1
set_var slew_upper_rise 0.9
```

sort_cells

<value> Set this to a 0 to disable sorting cells alphabetically.
Default: 1 (Sort the cells.)

This parameter controls the sorting of cells into the *output.lib* file. The default is to sort all cells alphabetically.

The pins within a cell are sorted with outputs first, then bidirectional pins followed by input pins. The pins are sorted alphanumerically within each of those groups.

This parameter may be used after **char_library**.

Example

```
# Do not sort cells
set_var sort_cells 0
```

sort_groups_under_pin

<value> Set this to a 0 to disable sorting pin timing groups by "when" alphabetically.
Default: 1 (Sort alphabetically.)

Use this parameter to request Liberate to sort the timing groups alphabetically using the "when" state description. By default, the timing groups under a pin are sorted alphabetically. This command is useful when an arbitrary order is desired for the timing groups under a pin.

This variable can be used after **char_library**.

Example

```
# Sort the timing groups
set_var sort_groups_under_pin 1
```

sort_pins

<0 | 1 | 2> Specifies the sorting of pins for each cell when writing the library.
Default: 1 (Sort)

Pins for each cell may be sorted in forward or reverse alphabetical order.

0: Don't sort pins.

1: Sort pins in alphabetical order. (Default)

2: Sort pins in reverse alphabetical order.

This variable can be used together with **pin_type_order**, to specify that pins be sorted within groups, ie: keep all input pins together and sort them; keep all output pins together and sort them; etc.

This variable must be used before **write_library**.

sort_pins_under_when

<0 | 1> Enable sorting of pins in a WHEN.
Default: 1 (Sort alphabetically)

Liberate can sort the pins in a *when* condition in the output library. Default: **1** (Sort pins). When set to **1**, Liberate sorts pin names within the *when* string as follows:

input pins - alphabetical, followed by
bidi pins - alphabetical, followed by
output pins - alphabetical

This variable must be used before **char library**.

Example

```
# Disable sorting the pins  
set var sort pins under when 0
```

spectre_dash_log

<0 | 1> Enables the Spectre -log command line option when set to 1.
Default: 1

Set this variable to 0 to disable the automatic addition of `-log` to the `extsim_cmd` option when the `-extsim` Spectre option of the `char` library command is used.

When using Spectre, Liberate automatically adds `-log` to the `extsim_cmd_option`. This is done to minimize disk usage because the `stdout/stderr` is saved by Liberate into a file called `sim.lis` and an additional log file is not needed. This parameter controls whether the Spectre `-log` option will be added when Spectre is used. It may be desirable during debug to view the Spectre log file because it contains additional messages that do not print to `stdout/stderr`. Set this parameter to 0 to save both the Liberate `sim.lis` and the Spectre `sim.log` files.

This variable must be set before **char library**.

spectre use char opt license

<0 | 1> Controls use of Spectre_char_opt license feature.
Default: 1 (use the licence feature if it exists)

This variable enables Liberate to checkout Spectre_char_opt license features upfront to minimize overheads associated with checking out Spectre licenses. Spectre_char_opt license feature is restricted for Liberate characterization flows and can not be used for other simulations. This license feature must be purchased and your license file must contain the Spectre_char_opt feature.



Important
If your license file does not contain the Spectre_char_opt feature, Liberate resets `spectre_use_char_opt_license` to 0. This will prevent a slow down in performance due to Spectre checking for a nonexistent license feature.

0: Do not use the `Spectre_char_opt` feature.

1: Use the `Spectre_char_opt` feature. (Default if the license feature exists)

This variable must be set before **char_library**.

spice_character_map / spectre_character_map

"list of character pairs"

List of characters to be mapped when building SPICE decks.
Default: " " (No character mapping)

Use this variable to tell Liberate to map characters in the netlist to alternate characters in the SPICE decks. The string is comprised of character pairs, where the first character is replaced with the second character. Multiple mappings are supported.



Spectre-SKI does not support this feature.

The list must not contain whitespace.

The list must contain an even number of "paired" characters.

This is independent of the external simulator used. The mapping is applied during `read_spice`. If "read_spice -format spectre" is used, then the `spectre_character_map` is applied. Otherwise, the `spice_character_map` is applied.

We highly recommended modifying the extraction tool to not use undesirable characters as part of any net name. It is possible that the resulting mapped deck could have name

collisions. This can be especially true if using SPICE-formatted netlists with characters used in bitwise operations (e.g. &, |) with Spectre.

This variable must be used before **read_spice**.

Example

```
# map the character '/' to '_' ... then map '&' to 'a'.
set_var spice_character_map "/_&a"
```

spice_delimiter

| | |
|----------|---|
| "string" | Character(s) used to indicate hierarchy in the input SPICE netlists. Default: ". " |
|----------|---|

This variable specifies the hierarchy delimiter in the SPICE format netlists loaded into **read_spice**. It can be a single or multiple character string. Every character in this variable is treated as a hierarchical delimiter. In the SPICE decks that are written out, the first character in this string will be used as the hierarchical delimiter.

This variable must be used before **char_library**.

Example

```
# Set the SPICE delimiter to |
set_var spice_delimiter "|"
```

spice_delimiter_replacement

| | |
|----------|--|
| "string" | Character used to replace hierarchy in the input SPICE netlists. Default: "_" (Underscore character.) |
|----------|--|

By default the parameter **extsim_use_node_name** is **0**, and the node id's in the spice decks for the external simulator are represented in numerical form. If **extsim_use_node_name** is set to **1**, some characters (for example, : / .) could be interpreted has a *hierarchical delimiter* even if the deck itself has been flattened. In some cases there are characters that cannot be allowed by the SPICE netlist interpreter (for example, as voltage source name or node). Use this variable to specify the character that will be used to replace those delimiter characters. Default "_".

This variable must be used before **char_library**.

Example

```
# Set the SPICE delimiter replacement character to #
set var spice delimiter replacement "#"
```

spice_instance_name_require_x_prefix

<0 | 1> Specifies that SPICE instance names must begin with the character "X".
Default: 1 (on)

Instance names in SPICE netlists are typically required to begin with the character "X". However, Spectre does not enforce this, so turn this variable off to allow netlists with instance names not prefixed by "X".

This variable must be used before **read spice**.

Example

```
set var spice instance name require x prefix 0
```

spice_logical_netname_mode

<0 | 1> Use logical wire name instead of shortest node name.
Default: 0

Note: The default value is overwritten to 1 in the `write_vdb` flow.

When **spice_logical_netname_mode** is set to 1, instead of setting the logical netname to the shortest node name, Liberate will set the logical wire name using the name of the attached instance.

In cases where a VDB flow is used with switch cells and multiple netlist extractions, setting this variable will resolve a certain class of errors. The recommended value is 1.

This variable must be used before **char library**.

subtract_hidden_power

<0 | 1 | 2> Control the subtraction of hidden power from internal power.
Default: 0 (Do not subtract hidden power.)

Use this control variable to cause hidden power to be subtracted from `internal_power`.

- 0:** Do not subtract hidden power. (Default)
- 1:** Subtract hidden power in sequential cells.
- 2:** Subtract hidden power for all cells.

This variable must be used before `char_library`.

subtract_hidden_power_consider_all_supplies

<0 | 1> Controls whether to subtract all of the hidden power groups that overlap with the switching power if the cell has multiple supplies such as in a level shifter. This variable can be used only when the variable `subtract_hidden_power_use_default` is set to 2.
Default and recommended: 1

| | |
|---|--|
| 0 | Subtracts only the last hidden power group among the hidden power groups that overlap (usually have no when condition). This can result in wrong results if there are multiple supplies in the cell because every supply corresponds to one such hidden power group. This setting is provided for backward compatibility to release 15.1 ISR5. |
| 1 | Subtracts all hidden power groups corresponding to all supplies that overlap with the switching power. |

This variable must be set before the `char_library` command.

subtract_hidden_power_use_default

<0 | 1 | 2> Enables subtraction of the default hidden power, if available.
Default: 0 (Disable hidden power subtraction)

Use this variable to control what value of hidden power should be used when subtracting hidden power from switching internal_power.

0: (Default) Liberate chooses one of the non-conflicting state dependent hidden power groups to subtract. If all hidden states conflict with the switching state, hidden power is not subtracted. Following are examples of:

| Conflict | Non-Conflict |
|---|--|
| CK->Q when " ! SE" with CK hidden when " SE&D". | CK->Q when " ! SE" with CK hidden when " ! SE&D" |
| | CK->Q when " " with CK hidden when " ! SE&D" |

1: The default hidden power value is used for subtraction from the switching power. If no default hidden power is found, a warning is issued and the related subtraction is skipped. When using this setting, it is recommended that the **force_default_group** variable is also set to a 1.

2: (Recommended) Liberate uses the following priority for subtraction of the default hidden power:

- Subtract the hidden power matching the simulation vector, if it exists
- Subtract the default hidden power, if it exists

If no default hidden power is found, a warning is issued and the related subtraction is skipped. When using this setting, the WHEN state must also have exact or partial overlap with the switching power state because the power tool uses this while adding the default hidden power.

This variable will have no effect if **subtract_hidden_power** is set to 0.

This variable must be set before **write_library**.

supply_define_mode

<0 | 1>

Changes the way supplies are recognized. When set, there are no default supplies. Therefore, all supplies must be identified using the **set_vdd** and **set_gnd** commands.
Default: 0

0 Determines the supplies by tracing the connectivity. All supplies must connect to a transistor.

Note: The default supplies are: VDD, VCC, and VSS.

1 Determines the supplies from module wires instead of tracing connectivity to transistors. This setting is useful in matching legacy libraries and when there are unconnected supplies that must be modeled in the library.

Note: There are no default supplies. All supplies must be identified using `set_vdd` and `set_gnd`.

This variable must be set before the `set_operating_condition` command.

switch_cell_bounded_dc_current

<0 | 1> Enable bounding the dc_current sweep between supplies.
Default: 1

Set this variable to limit the dc current sweep for power switch cells. When this variable is set to **0**, the dc current sweep reported for the power switch cells will use the same voltage range as the CCSN `dc_current`. When this variable is set to **1**, the `dc_current` sweep will be limited to the gnd and vdd rail voltages.

This variable must be used before **char_library**.

switch_cell_dc_current

<0 | 1 | 2> Includes or omits `dc_current` from switch cells.
Default: 1 (Include `dc_current`.)

| | |
|---|--|
| 0 | Omits the <code>dc_current</code> from switch cells in an NLDM library. If CCSN is requested, the <code>dc_current</code> is not omitted. This can be useful to prevent characterization of the <code>dc_current</code> for switch cells when using a separate Liberate run to characterize leakage. |
| 1 | Includes <code>dc_current</code> in the switch cells when the <code>-internal_supply</code> option of the <code>define_cell</code> command is provided. |
| 2 | Always omit the <code>dc_current</code> from switch cells. Neither NLDM nor CCSN style libraries will include <code>dc_current</code> . |

This variable must be used before the `char_library` command.

switch_cell_dc_current_output_offset

`<value>` Specify voltage offset value used in switch cell `dc_current`.
Default: 0

Use this variable to specify an absolute voltage value that will be used to offset the output swing range of the `dc_current` constructs of switch cells. The default offset value is 0. The formula applied when sweeping the `dc_current` is described below.

- input sweep range: `gnd(input), vdd(input)`
- output sweep range: `gnd(output)+offset, vdd(output)-offset`

This variable must be used before the `char_library` command.

switch_cell_internal_node_timing_arc

`<0 | 1>` Generates an input to internal switch node timing arc.
Default: 0

0 Skips generation of the timing arc of the internal supply pin. The timing arc of the internal switch node is copied from the timing arc of the output pin.

| | |
|---|---|
| 1 | For power switch cells, generates input to internal switch node timing arc. When this variable is set to 1, the conditional <code>ccsn_*_stage</code> is not outputted. |
|---|---|

This variable must be set before the `char_library` command.

switch_cell_powerdown_function

| | |
|----------------------------|--|
| <code><0 1></code> | Specifies whether to generate the <code>powerdown_function</code> on switch cells. Default: 1 |
|----------------------------|--|

Setting **`switch_cell_powerdown_function`** to 1 will generate the **`powerdown_function`** on switch cells.

This variable must be set before the `char_library` command.

switch_function_attr_mode

| | |
|-------------------------------------|--|
| <code><-1 0 1 2></code> | Selects the algorithm for determining the switch cell function. Default and recommended: 0 |
| -1 | Form <code>switch_function</code> by primary input and unateness in the <code>ccsn_last_stage</code> group(s) of the internal supply pin in <code>ldb</code> |
| 0 | Use the <code>three_state</code> function of the internal supply pin in <code>ldb</code> as the <code>switch_function</code> . |
| 1 | Form <code>switch_function</code> by generating logic function with a CCC function identification algorithm and adding logic 'not' to invert the function; |
| 2 | Generates <code>switch_function</code> by using BDD to invert and simplify the identified function for the internal supply pin. |

Notes:

- Setting of -1 can be incorrect when stacked MOS are between real power and virtual power.
 - Setting of 0 can be effective both in the read_ldb and write_library flow and in the char_library and write_library flow. This mode is the default mode.
 - Setting of 1, 2 can be effective only in the char_library then the write_library flow, but not in the read_ldb then the write_library flow. This is because the function identification algorithm requires circuit module.

This variable must be set before the `char_library` command.

template_unique_power_mode

<0 | 1> Default: 0

Note: This variable should never be set manually. It is automatically set in a template file that is created when the `-unique_power` argument of the **write_template** command is used. When the `-unique_power` argument is used, depending on the reference library, additional power arcs may be generated. In some libraries, this can result in the following:

- Many more power arcs are generated because the modeled states ("when" conditions) differ between delay and power.
 - Some power arcs are invalid because the reference library contains data duplication between rise and fall power that did not come from a valid simulation.

When this variable is set to 1, the above issues are addressed, resulting in correct power values and runtimes as fast as without `-unique power` argument.

This variable must be set before **char_library**.

test cell at end

<0 | 1> Control the position of the test_cell.
Default: 1 (move the test_cell)

Set this variable to **0** to not move the *test_cell* group to the end of each cell in the library output by write_library rather than the current position which is after the leakage data and before the pin data.

This variable must be set before `write_library`.

timing_group_unateness

`<merge | separate>` Controls merging of positive/negative unate arcs into single non-unate group.
Default: `merge` (compatible with 3.2 behavior)

separate: Liberate will *not* merge `pos_unate` and `neg_unate` arcs into a single non-unate timing group. This variable applies to timing groups that are not default timing groups. For control of merging for default groups, see the `define_arc_ignore_mode` variable.

merge: Merge two timing groups under the following conditions: they have the same `related_pin` and `when` condition, the `timing_type` is `combinational`, and one is `positive_unate` and the other is `negative_unate`. (Default)

The `timing_type` will be changed to the appropriate edge. (For example, if the `related_pin` is a clock, then change the `timing_type` to `rising_edge` or `falling_edge` accordingly, otherwise, set it to `combinational`.)

The `timing_sense` will be removed if `discard_timing_sense_after_merge` is true, otherwise, it will be set to `non_unate`.

Note: Setting this variable to `separate` can *adversely impact* model generation for all `define_arc/verbose` flows.

This variable must be set before `char_library`.

tmpdir

`<string>` Specify a temporary working directory.
Default: `none`

This variable specifies a temporary working directory for Liberate to store extsim run data.

Either the Tcl variable `tmpdir`, or the environment variable `TMPDIR` can specify a temporary working directory. If both variables are set, the Tcl variable `tmpdir` takes precedence. Liberate follows this rule for creating temporary directories:

1. Use `tmpdir` (create dir if missing).
2. If `tmpdir` is not set use environment variable `TMPDIR` (create if missing).
3. Error out if either `tmpdir` or `TMPDIR` point to files.

4. If neither `tmpdir` nor `TMPDIR` are available use the current working directory (cwd).

This variable must be set before **char_library**.

toggle_leakage_state

`< 0 | 1 >` Toggle the clock before measuring state dependent leakage on sequential cells.
Default: 0 (Off)

This parameter will ensure that the clock pin is toggled for a cycle to store a logic state in a sequential cell before measuring state-dependent leakage. Turn this on to disable fully-specified state-dependent leakage that includes output pins for all cells. That is, when this variable is set to 1, the *when* specified in the leakage groups will not include the output pin state.

This variable must be used before **char_library**.

Example

```
# Disable leakage conditions that depend on Q
set_var toggle_leakage_state 1
```

tran_tend_estimation_mode

`<0 | 1>` Switch between different algorithms for estimating the transient simulation end time.
Default: 1

0: The estimation method uses 99.5% of the transition threshold which in some corner cases can take a long time to complete simulation, resulting in a longer simulation time for power simulations. Subsequently, the leakage power that is being subtracted from internal power calculations may become inaccurate.

1: Estimate `tran_tend` using the slew measurement threshold.

tristate_disable_transition

`<0 | 1>` Output disable transition values in the library.
Default: 1 (copy delay to transition)

Use this parameter to request Liberate to output full transition tables for the tristate disable arcs. By default, Liberate will output a scalar value for the *three_state_disable* arcs.

0: Liberate outputs a scalar value of 0 in the transition tables for *three_state_disable* arcs.

1: Liberate output sfull transition tables for *three_state_disable* arcs. The transition disable arc data is copied from the corresponding delay tables.

This variable must be set before **write_library**.

tristate_pin_cap_always_on_res_mode

| | |
|---------|---|
| <0 1> | Adjusts the behavior of <i>adjust_tristate_load</i> where the pin capacitance is very large due to pull-up and pull-down resistances that are always on. Default: 1 (on) |
| 0 | Behavior of release 12.1.1 and earlier. |
| 1 | Corrects the effect of <i>adjust_tristate_load</i> when pin capacitance is very large. |

This variable must be set before the `char_library` command.

unique_pin_data

| | |
|---------|---|
| <0 1> | Requests unique pin data under all bundles and buses. Default: 0 |
| 0 | Bundle and bus has one data table applied to the entire bus. Each <code>write_library</code> command may be generated with unique pin data by using the <code>-unique_pin_data</code> option of the <code>write_library</code> command. |

1

The bus and bundle data is modeled with unique characterized data for each bundle and bus element.

This setting does require that all bits of the bus are characterized and unique arc data exists. The setting is equivalent to the `-unique_pin_data` option of the `write_library` command.

This variable is set for a specific cell, pin, related_pin, type by using the `set_var` command.

Example

To break-out per-pin data for power only, specify the following:

```
set_var -type power unique_pin_data true
```

This variable must be set before the `write_library` command.

use_pid_tmpdir

< 0 | 1 >

Add the process ID to the tmp directory name.
Default: 1 (Add pid)

Add the process ID to the tmp directory name. (Default.).

Example

```
/tmp/<user>/altos.<pid>.0  
/tmp/<user>/altos.<pid>.1
```

This variable must be used before `char_library`.

user_data_attr_order

< 0 | 1 >

Request user_data attribute order.
Default: 1

Set this parameter to control whether to keep the complex attributes in the exact order found in the user_data or to follow the default order used by Liberate when writing out a library.

0: Liberate outputs attributes from the user_data file (see [write_library](#)) when writing out a library.

1: Liberate outputs the complex attributes in the exact order found in the user_data file.

This variable must be used before **write_library**.

user_data_override

{attribute_list} A list of attributes that will override characterized values during write_library -user_data.
Default: {area function three_state state_function}

This variable allows the user to control which entries in the user_data file will override the corresponding data in the **ldb** when using the **user_data** option of the **write_library** command. It specifies a list of items which, if found in the user data file, will replace the value in the ldb. This list will append values to the built-in list of overrides.

Currently, the following elements are valid for the override list: max_capacitance (a simple attribute under the pin() group), min_pulse_width (for both min_pulse_width_high and min_pulse_width_low), min_pulse_width_high, and min_pulse_width_low, cell_leakage_power (a simple attribute under the cell() group), voltage_map and pg_pin.

The built-in default list of attribute overrides currently includes:

```
{area function three_state state_function}
```

This variable can be used after **char_library**.

Example

```
set_var user_data_override \
{max_capacitance cell_leakage_power}
```

user_data_quote_attributes

{ list } Specifies a list of attributes whose values need to have double quotes surrounding them.
Default: { } (No attributes surrounded with quotes.)

This variable must be used before the `write_library` command.

user_data_quote_simple_attr

`<0 | 1>` Specifies whether simple, non-numeric attributes from the user data are wrapped in quotes.
Default: 1

When **user_data_quote_simple_attr** is set to 1 and **user_data_attr_order** is set to 1, simple, non-numeric attributes from the user data are wrapped in quotes. Set this variable to 0 for 2.5p2 and prior behavior.

This variable should be used before `char_library`.

variation_mean_nominal_model_mode

`<0 | 1>` Enables modeling of mean shift, standard deviation, and skewness.
Default: 0 (not enabled)

Use this variable to save additional data for mean shift, standard deviation, and skewness into the output library when LVF modeling is requested (see `write_library - sensitivity_file`) to write additional data that is used by Tempus into the output library.

To add this additional data into the output library, the `sensitivity_file` created by Variety must include the required data in the `sensitivity_file`. The variable `variation_mean_nominal_model_mode` must be set to 1 in the Variety model generation run and the variable `variation_mean_nominal_mode` must be set to 4 in the Variety characterization run.

0: Write the `ocv_sigma` data contained in the `sensitivity_file` into the output LVF formatted Library. (Default).

1: In addition to the `ocv_sigma` from 0, also write the mean shift, standard deviation, and skewness for delay and transition into the LVF formatted output library. The additional data is outputted as new groups after the nominal group but before the `ocv_sigma` groups. The additional groups are named:

| | | |
|--|---------------------------------------|---------------------------------------|
| <code>cell_rise_mean_shift,</code> | <code>cell_rise_std_dev,</code> | <code>cell_rise_skewness</code> |
| <code>cell_fall_mean_shift,</code> | <code>cell_fall_std_dev,</code> | <code>cell_fall_skewness</code> |
| <code>rise_transition_mean_shift,</code> | <code>rise_transition_std_dev,</code> | <code>rise_transition_skewness</code> |
| <code>fall_transition_mean_shift,</code> | <code>fall_transition_std_dev,</code> | <code>fall_transition_skewness</code> |

This variable must be set before the **write_library** command.

vector_check_mode

| | |
|------------------|--|
| < 0 1 2 3> | Controls the algorithm used to determine the cell behavior for a particular vector. This variable helps in rare cases where Liberate fails to find an arc which is known to exist. This may occur in those cases where there is an internal race condition. Default:3 Recommended: 1 |
| 0 | Less precise than 1 but independent of internal delays. |
| 1 | Determines the cell behavior for a particular vector. |
| 2 | More precise than 1 but slower. Use only if a cell is very sensitive to internal conditions. |
| 3 | Uses improved internal logic simulator for checking vector validity. |

This variable must be set before the **char_library** command.

vector_estimate_dump

| | |
|---------|--|
| <0 1> | Prints the result of the estimation decks for use in vector debugging. Default: 0 (Off) |
|---------|--|

For each `define_arc` with more than one possible vector, Liberate prints a list of vectors along with the estimated delay, power, and so on. It also prints which vectors are chosen for full simulation. This only includes vectors for which an estimate is done. If `define_arc` has only one vector, then the results of the full simulation should be reviewed.

0: Specifies not to print vector estimate information. (Default and recommended)

1: Prints vector estimate information intended for template debugging purposes.

This variable must be set before **char_library**.

vector_side_input

<0 | 1> Set unknown side inputs to 0 or 1.
Default: -1 (expand all X)

This parameter can be used to force side inputs whose values are not specified in a "when" or vector to all zeros or all ones. By default, Liberate will automatically decide what to set the side inputs to. This includes leaving them floating. It is recommended to always set all side inputs. Set to 0 for 0:X and set to 1 for 1:X.

This variable must be used before **char_library**.

verilog_cg_filter_edge

<0 | 1> Enable filtering of extra timing constraints in Liberty and Verilog.
Default: 1

If this variable is set to 1, extra timing constraints will be filtered out of Liberty and Verilog for cells that contain the attribute **clock_gating_integrated_cell**, according to the following table:

| clock_gating_integrated_cell = | posedge | negedge |
|--------------------------------|---|---|
| Filter these out: | min_pulse_width_hig h constraint_high | min_pulse_width_lo w constraint_low |

The default is 1. Set this variable to 0 to disable this filter.

This variable must be set before **write_library** or **write_verilog**.

voltage_map

<0 | 1 | 2> Generate a voltage map in the output library.
Default: 1 (on)

This parameter can be used to generate a set of voltage map attributes in the output library. This should be used when multiple voltage levels are used within a cell, for example, a level shifter.

0: Disable generation of **voltage_map** attributes in the library.

1: Generate **voltage_map** attributes at the library level along with **pg_pin** groups for each cell and **related_power_pin** / **related_ground_pin** attributes for each pin.
(Liberty 2006.06 syntax).

2: Generate a **power_supply** group along with **rail_connection** attributes for each cell and **input_signal_level** / **output_signal_level** attributes for each pin.
(pre-Liberty 2006.06 syntax).

If **voltage_map** is set to **1** or **2** an *operating_conditions* group will be generated in the output library with the appropriate *process*, *temperature*, *voltage* and *signal_level* attributes. The name of the operating group defaults to "*PVT_<process>_<voltage>V_<temp>C*" (with decimal points (.) being replaced by the character P). This can be overwritten by providing a named *operating_conditions* group as **user_data** to the **write_library** command. The *process* attribute within the *operating_conditions* group will also be overwritten by the value in the **user_data** file.

Note: If the **char_library -ccsp** option is enabled, this parameter will be overridden. A CCS-Power library requires the *voltage_map*, *related_power_pin*, and *related_ground_pin* attributes and the *pg_pin* and *power_supply* groups.

Liberate will support multiple power formats in the following way:

2005.12 format:

```
set_var voltage_map 2
char_library -ccsp
write_library
```

2006.06/9 format:

```
set_var voltage_map 1
char_library -ccsp
write_library
```

CCSP format:

```
char_library -ccsp
write_library -ccsp
```

This variable must be set to a non-zero value before **char_library**.

This variable can be set to 0 after **read_Idb** and before the first **write_library** command.

Example

```
# Use multiple supply voltages
set_operating_condition -voltage 1.08 -temp 25
set_vdd VDDL 0.84
set_gnd VSSL 0.02
set_var voltage_map 1
```

The following constructs will appear in the library :

```
default_operating_conditions : PVT_TT_1P08V_25C;
voltage_map (VDD_VSS, 1.08);
voltage_map (VDDL_VSSL, 0.82);

.
cell (BUF_1) {
    pg_pin (PP1) {
        pg_type : primary_power;
        voltage_name : VDD;
    }
    pg_pin (GP1) {
        pg_type : primary_ground;
        voltage_name : VSS;
    }
    pg_pin (PP2) {
        pg_type : primary_power;
        voltage_name : VDDL;
    }
    pg_pin (GP2) {
        pg_type : primary_ground;
        voltage_name : VSSL;
    }
    pin (X) {
        direction : output;
        function : "A";
        max_capacitance : 0.089809;
        related_ground_pin : GP2;
        related_power_pin : PP2;
        timing () {
            ....
        }
    }
}
```

vsrcl_slope_mode

| | |
|---------|---|
| <0 1> | Improved Alspice PWL breakpoint handling. Default: 1 |
|---------|---|

This allows for improved Alspice PWL breakpoint handling. The default and recommended values are 1. (Note: 0 is not recommended; this is only for compatibility with releases prior to 3.1.

This variable must be used before **char_library**.

waveform_report

<0 | 2> Enable saving of driver waveform.
Default: 2

Set this control variable to 0 to disable the saving of the driver input waveform into the .vdb file (see **write_vdb**). Do not use this variable if the **write_library_driver_waveform** option is to be used. Default: 2 (The driver input waveform will be saved into the .vdb and the .lbd).

This variable must be set before **write_vdb** and/or **char_library**.

wnflag

| | |
|---------|---|
| <0 1> | Instructs Liberate to use W/nf for model selection. If the user has .param wnf=0, they must set the variable <code>wnflag</code> to 0 in their Tcl run script to inform Liberate to use W <u>instead</u> of W/nf to select model binning. Default: 1 |
| 0 | Use W for model selection. |
| 1 | Use W/nf ratio for model selection. |

Example

```
set var wnf1ag 0
```

This variable must be set before the `read spice` command.

write library is unbuffered

| | |
|---------|--|
| <0 1> | Controls the output of <code>is_unbuffered</code> attribute in .lib. Default: 0 |
| 0 | Outputs the Liberty <code>is_unbuffered</code> attribute for CCSN arcs. |
| 1 | Outputs the Liberty <code>is_unbuffered</code> attribute for non-CCSN arcs. |

This variable must be set before the `char_library` command.

write_library_mode

| | |
|---------|--|
| <0 1> | Enables faster method for writing a library in an arc packet flow. Default: 0 |
| 0 | Writes a library in the server containing all characterized cells |
| 1 | The <code>write_library</code> command is distributed to each client in an arc-packet flow (see packet_mode). |

When enabled, a directory is created in the LDB called `LIBS`. This directory contains a unique library file for each characterized cell. These individual libraries are appended together in the server. This results in the library creation being distributed across the clients.

This flow requires that the `write_library` command executes in the server and in the client. Any existing script that protects the `write_library` command so it cannot execute in the client (see `packet_slave_cells` and `ALAPI_active_cell`) must be updated. If the `write_library` command does not execute in the client, the library is written on the server as when `write_library_mode` is set to 0.

This variable must be set prior to the `write_library` command. Writing a library with a setting of 1 followed by writing a library with a setting of 0 is not supported.

write_library_sync_ldb

| | |
|---------|---|
| <0 1> | Forces Liberate to read the ldb on disk prior to writing. Default: 0 |
|---------|---|

0: Liberate will not read the ldb prior to writing the library. (Default)

1: Forces Liberate to read the ldb on disk prior to writing the library. Used to address possible precision issues between the `char_library` and `read_ldb/write_library` flow. (See also [write_library -sync_ldb](#).)

This variable must be set before **write_library**.

write_logic_function

<0 | 1> Enable/Disable logic function generation.
Default: 1 (Enable)

Enables/disables writing of the **logic function** when using Inside View. Disabling this feature is useful when Liberate has difficulty generating the correct function for complex cells. In that case, the user can provide the function in the **user_data** file to write_library, which will override any Liberate generated function. Using **user_data** can also reduce runtime.

0: Turn off logic function determination when using Inside View.

1: Automatically write function and three_state attributes. (Default)

This parameter must be used before **char_library**.

write_logic_function_group at end

<0 | 1> Specifies the desired position for the function groups in the library. It instructs Liberate where to place the function groups (ff, latch, state_table) in each constrained cell in the library. Default: 1 (ff/latch after the pins and state_table before the pins).

0 Places the function groups before the first pin in the cell.

1 Places the ff and latch function groups after the last pin in the cell and the state_table (see write_logic_function_mode) before the pins.

This parameter must be used before the `char` library command.

write logic function mode

<0 | 1 | 2> Enables an alternate algorithm for determining logic function, designed to minimize X-propagation.
Default: 2 (on)

Note: This algorithm does *not* guarantee matching the X-propagation behavior of all circuits.

0: Standard algorithm for generating logic function.

1: Alternate algorithm for generating logic function.

2: Alternate algorithm for generating logic function with redundant terms to reduce X-propagation. (Default)

This variable must be set before **write_library**.

write_logic_function_statetable_limit

<value> Sets the maximum number of sequential elements allowed when generating a statetable format function when the write_logic_function_statetable_mode variable is enabled.
Default: 16

This variable must be set before the **char_library** command.

write_logic_function_statetable_mode

<0 | 1 | 2> Enables generation of statetable format function.
Default: 0

Enables the generation of a cell function using statetable format in the output library for complex cells. This algorithm requires mega mode to be enabled (see the mega_enable variable and the define_cell -type command). The **-io** option must not be used with the **char_library** command because it disables all inside-view including mega mode algorithm.

0: Do not generate statetable format functions. (Default)

1: Generate statetable format functions for cells with more than one sequential element.

2: Generate statetable format functions for all cells with sequential elements.

This variable must be set before **char_library**.

write_min_transition_attr

<0 | 1> Write **min_transition** pin attribute to library.
Default: 0

0: Do not output **min_transition**. (Default)

1: Output **min_transition**.

This parameter must be used before **write_library**.

write_template_force_power

<0 | 1> Output power templates whenever there is characterized power data.
 Default: 1

The **write_template** command will output a reference to a power template for every cell that has any power data. There are cells (ex.:DECAP and Filler cells) that do not have any timing or internal power arcs, but that only have leakage. Liberate must have a **define_cell -power** option referencing a power template (see [define_template](#)) for these types of cells to be characterized. Set this variable to **0** to get release v3.2p3 and prior behavior where **write_template** would not include a **-power** option in the **define_cell** command for these types of cells.

This variable must be set before **write_template**.

Virtuoso Liberate Reference Manual
Liberate Parameters

Library Comparisons

This chapter describes the Liberate utilities for comparing libraries.

Liberate provides ways to easily compare two libraries. This is useful, for example, to understand how new SPICE models may change library characteristics such as leakage power. The comparison can also be used to compare Liberate-generated libraries against existing libraries.

Both textual and graphical comparisons can be generated. The **compare_library** command can be used to generate a text comparison report highlighting outliers that exceed the defined absolute and relative tolerances. Use the **gui** option with **compare_library** to generate an intermediate file for graphical comparisons. The graphical comparison utility is called **lcplot**.

lcplot

<gui comparison file> File generated by **compare_library –gui** option

The **lcplot** graphical comparison plots make it very easy to pin-point library entities that have significant differences, or to confirm expected trends such as slower delays when comparing a library generated at a slow corner versus a fast corner.

Panel Buttons

| | |
|--------------------|---|
| Fit X | Expand the current graph in the X range to full scale while keeping the Y range fixed. |
| Fit Y | Expand the current graph in the Y range to full scale while keeping the X range fixed. |
| Fit All | Expand the current graph in both the X and Y ranges to full scale. |
| Log - X | Change the X-axis in an X/Y style graph into a logarithmic scale. |
| Log - Y | Change the Y-axis in an X/Y style graph into a logarithmic scale. |
| Timing | Select only Timing Data to display (delay, setup, hold, recovery, removal and trans). |
| Power | Select only Power Data to display. |
| Leakage | Select only Leakage Data to display. |
| Capacitance | Select only Capacitance Data to display (capacitance, fall_capacitance and rise_capacitance). |
| Style | Select the style of the graph to display. More details can be found below. |
| Cell Sel | Brings up 'Cell Selection' form to select cells to display |
| Data Sel | Brings up 'Data Selection' form to select data type to display |
| Direction | Brings up 'Direction Selection' form to select data toggle direction (rise, fall) |
| Close | Closes the Icplot window. |

Pull-Down menus

| | |
|-----------------------------|---|
| File -> Print | Brings up Print form to print to file (postscript format) or printer in gray scale or color. |
| File -> Exit | Close Icplot |
| View -> ZoomOut 2 | Zoom out by 2X (or, click the Right Mouse Button in graphic display window to zoom out by 2X) |
| View -> ZoomIn 2 | Zoom in by 2X (or, use the Left Mouse Button in graphic display window to select a box to automatically zoom into). |

| | |
|-------------------------|--|
| Redraw | Redraw the window to clean up any cursor ghosts. |
| Help -> About | Display Version and Copyright notice. |

Graphical Library Comparisons

There are 3 styles of plots available: X/Y, Accuracy, and Errorbound.

Style: X/Y

The following graphical library comparison shows a scatter plot where the values from the reference library are given on the X-axis and the values from the comparison library are given on the Y-axis. When the values in the two libraries match, the plotted data points will fall exactly on the 45-degree axis. Note: By default, this plot type will display all four data types: Timing, Power, Leakage and Capacitance.

An example is shown below:

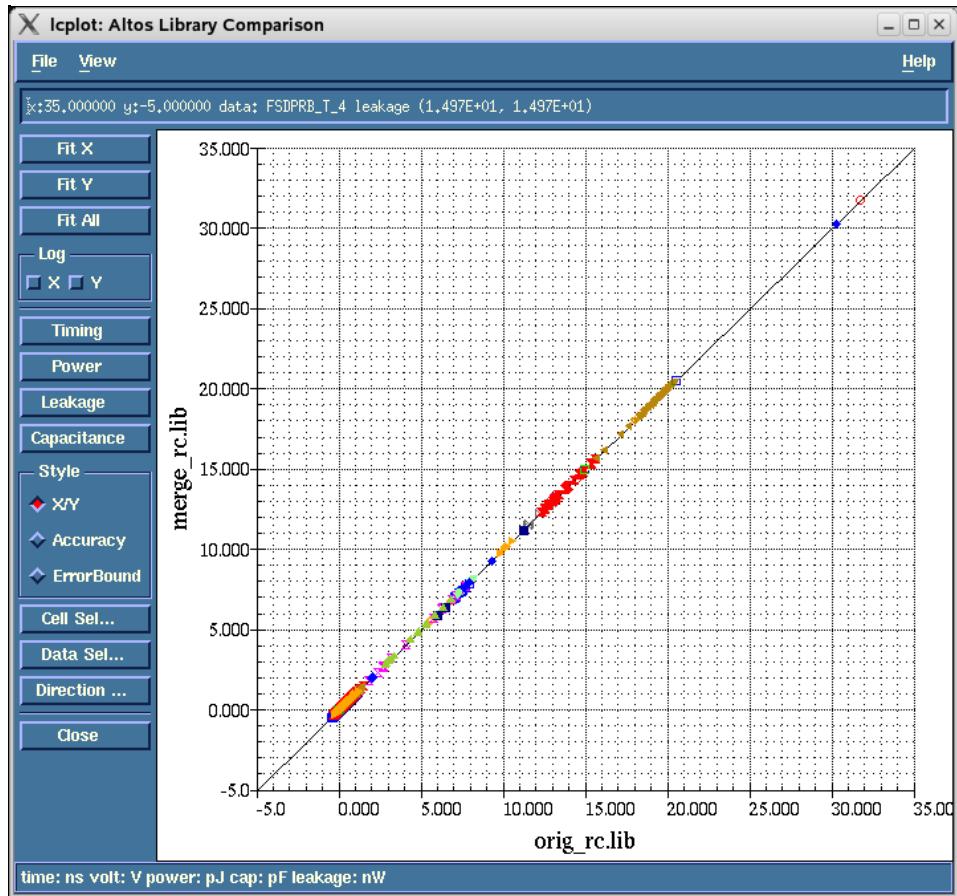


Figure 5: LCPlot GUI: X/Y Library Comparison

Style: Accuracy

The following graphical library comparison shows an Accuracy plot where the existing value in the reference library is given on the X-axis and the absolute difference in values (scaled up by 100) between the comparison library and the reference library are given on the Y-axis. This plot also includes a +45 degree and a -45 degree axis. These two axes form four triangular regions. The left and right regions represent the data points that fall within 1%. The upper and lower triangular regions represent the data points that are greater than 1% of difference. This plot is useful in determining which data points are greater than 1%. When the mouse is positioned directly over a data point, the actual data from both libraries will be displayed in the frame directly above the graph. Note: This plot style only applies to timing comparisons.

An example is shown below:

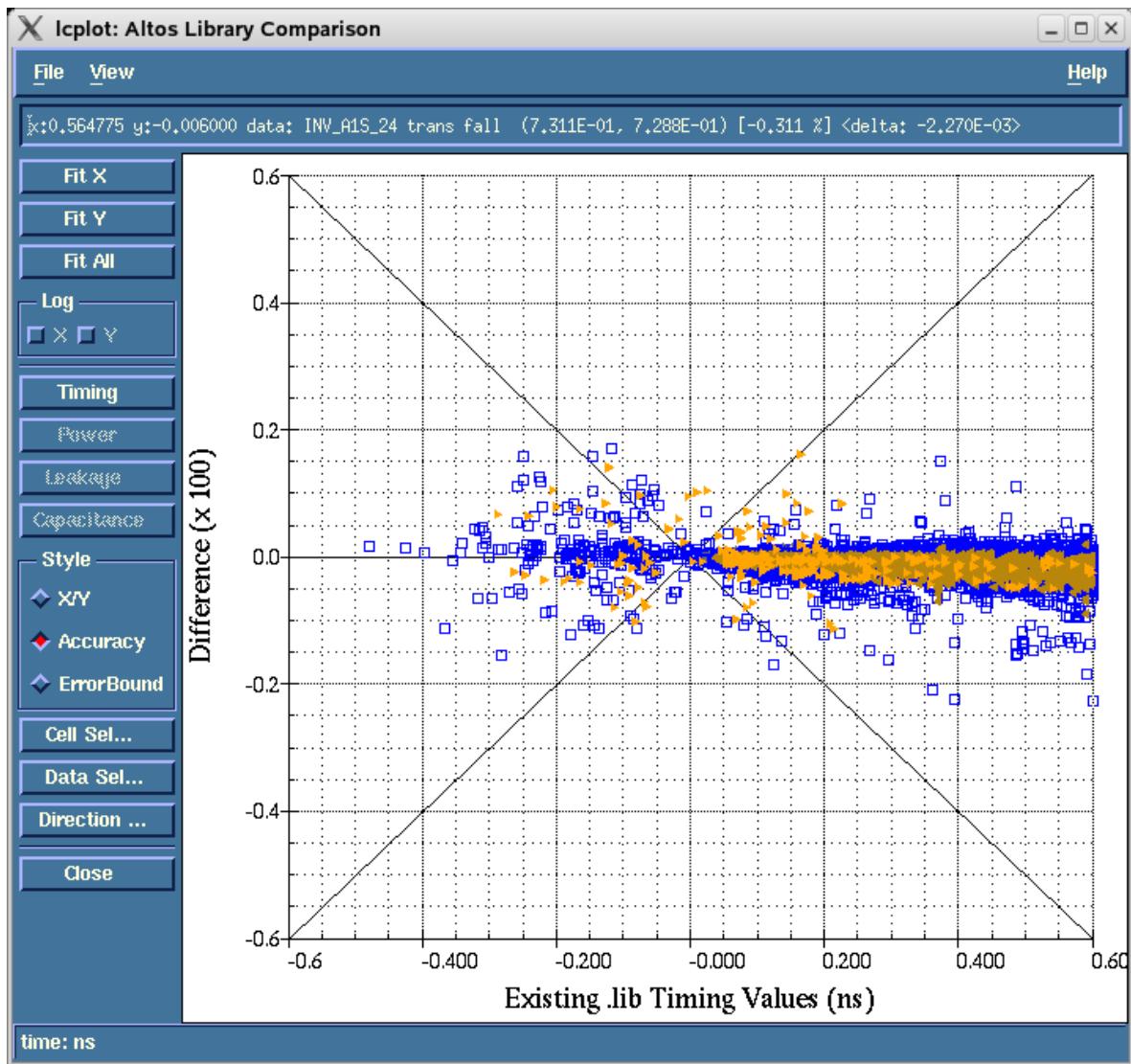


Figure 6: LCPlot GUI: Accuracy Library Comparison

Style: ErrorBound

The following graphical library comparison shows an ErrorBound plot where the absolute difference in values between the comparison library and the reference library is given on the X-axis, and the Difference Ratio $((compVal-origVal)/origVal)*100$ of the comparison library to the reference library is given on the Y-axis.

Virtuoso Liberate Reference Manual

Library Comparisons

- If the data point on the graph falls close to the X-axis zero reference ($X=0$), then the absolute error is small so the relative error can be ignored, even if it is large.
- If the data point falls close to the Y-axis zero reference ($Y=0$), then the ratio of difference is small and even if the absolute difference is large, this difference can be ignored.
- When data points do not fall near either of the zero reference axes, the difference may be significant and should be reviewed.
- When the mouse is positioned directly over a data point, the actual data from both libraries will be displayed in the frame directly above the graph.

Note: By default, this plot type will display all four data types: Timing, Power, Leakage and Capacitance. An example is shown below:

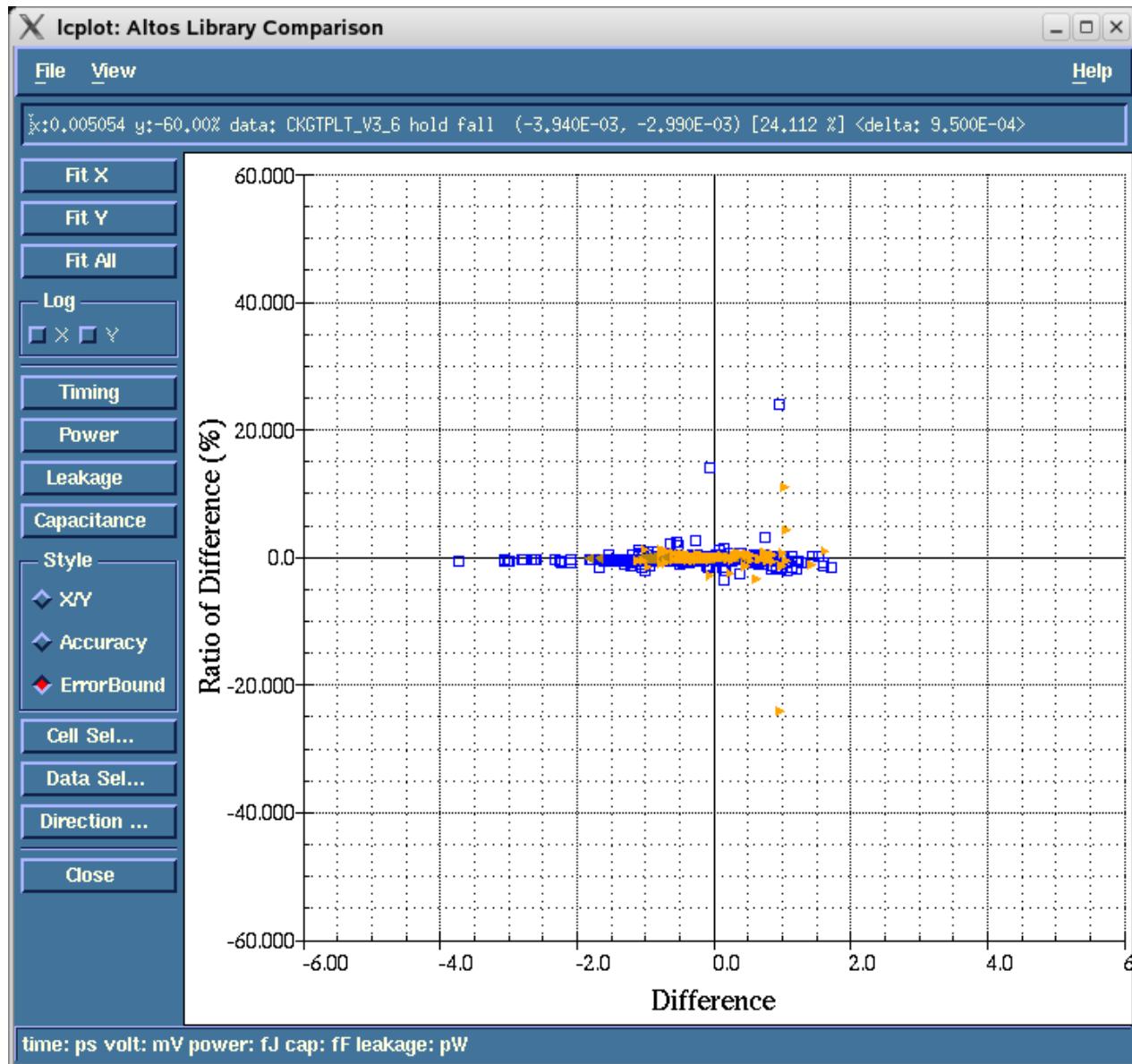


Figure 7: LCPlot GUI: ErrorBound Library Comparison

Data Selection

There are 2 ways to select data: Cell Type, and Data Type.

Cell Type Selection

To select the cells to compare, click on the "**Cell Select**" button. This pops up a selection menu that lists all of the cells in the library. An example is shown below:



Figure 8: Cell Selection Menu

To select a cell from the list click on the cell name and then click the "**Apply**" button. To select multiple cells hold down the "**Shift**" key, select the cell name followed by "**Apply**". To add cells to the selected set hold down the "**Ctrl**" key, select the cell names and click on "**Apply**". To select all cells click the "**All**" button followed by "**Apply**". To unselect all cells click on the "**None**" button followed by "**Apply**". To quit the cell selection menu use the "**Close**" button.

Data Type Selection

To select the type of library data to compare, click on the "**Data Select**" button. This pops up a selection menu that lists all of the available data types. Select a data type such as "**delay**" from the list and click on the "**Apply**" button. Multiple data types can be compared at once by holding down the "**Shift**" key and selecting the data types followed by "**Apply**". The units for

each of the values displayed are defined by the reference library. The following data types are available for comparison (assuming they are present in the reference library).

| | |
|-------------------------|---|
| trans | Transition times |
| capacitance | Input pin capacitance |
| power | Switching and hidden power |
| leakage | Leakage power |
| fall_capacitance | Pin capacitance for falling transitions |
| rise_capacitance | Pin capacitance for rising transitions |
| delay | Transition delays |
| recovery | Recovery time constraints |
| hold | Hold time constraints |
| setup | Setup time constraints |
| removal | Removal time constraints |



Figure 9: Data Type Selection Menu

Virtuoso Liberate Reference Manual

Library Comparisons

As the cursor moves across the plot window, the corresponding data represented by each comparison point is highlighted in the window pane header. This shows both the reference and comparison values, and their percentage difference.

Holding down the left mouse button and dragging the rectangle over the plot area will zoom into the selected area (when the cursor is in the plot window). A single click of the right mouse key will zoom out by 2X (when the cursor is in the plot window). The "**fit_x**", "**fit_y**" and "**fit_all**" buttons can be used to fit the data within the window after zooming.

Liberate Details

This chapter details how Liberate performs various characterization tasks. Liberate characterizes the following constructs:

Delay Models

- Non-linear Delay Model (NLDM)
- Composite Current Source (CCS) delay model
- Effective Current Source Model (ECSM)

Pin Capacitance

- Non-linear Delay Model (NLDM)
- CCS Receiver capacitance
- ECSM capacitance

Timing Constraints

- Setup & Hold
- Recovery & Removal
- Minimum Pulse Width

Power Models

- Leakage Power
- Switching & Hidden Power
- Power Subtraction
- Power Validation
- Common Usage Models for Power and Leakage

Signal Integrity Models

- Steady State Current
- Noise Immunity Curve
- Hyperbolic Input Noise
- DC Noise Margin

Composite Current Source Noise (CCSN) Models

- CCSN DC current
- CCSN Output Voltage
- CCSN Miller Capacitance
- CCSN Propagated Noise

IBIS Models

Virtuoso Liberate Reference Manual

Liberate Details

- Power Clamp & Ground Clamp Current
- Pull_up & Pull_down V-I Tables
- Rising & Falling Waveforms

Electromigration Models

- Electromigration Maximum Toggle Rate
-

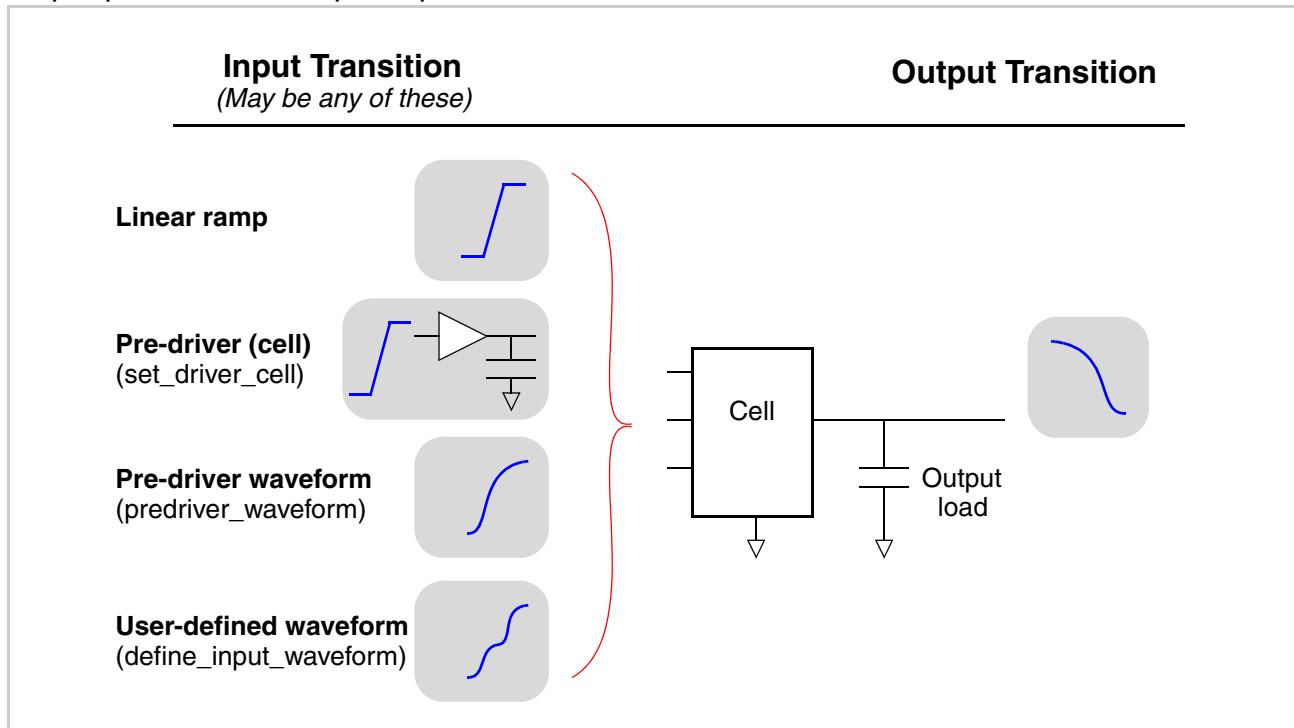
Delay Models

NLDM

The NLDM delay model is characterized by measuring the delay and output transition when simulating a given range of different combinations of input transitions and output loads. The input transition can be any of these:

- linear ramp (from the defined template)
- output of a pre-driver (set_driver_cell)
- analytical driver waveform (predriver_waveform)
- user-defined waveform created using the `define_input_waveform` command.

The input pin is triggered by either a ramp or a smooth waveform (output of pre-driver). The output pin load is a simple capacitance.



Circuit used for delay and output transition measurements

Delay is measured from the input crossing the delay measurement point (`delay_inp_rise`/`delay_inp_fall`) to the output crossing the delay measurement point (`delay_out_rise`/`delay_out_fall`, default 50% of supply). Output transition time is measured from the lower-to-

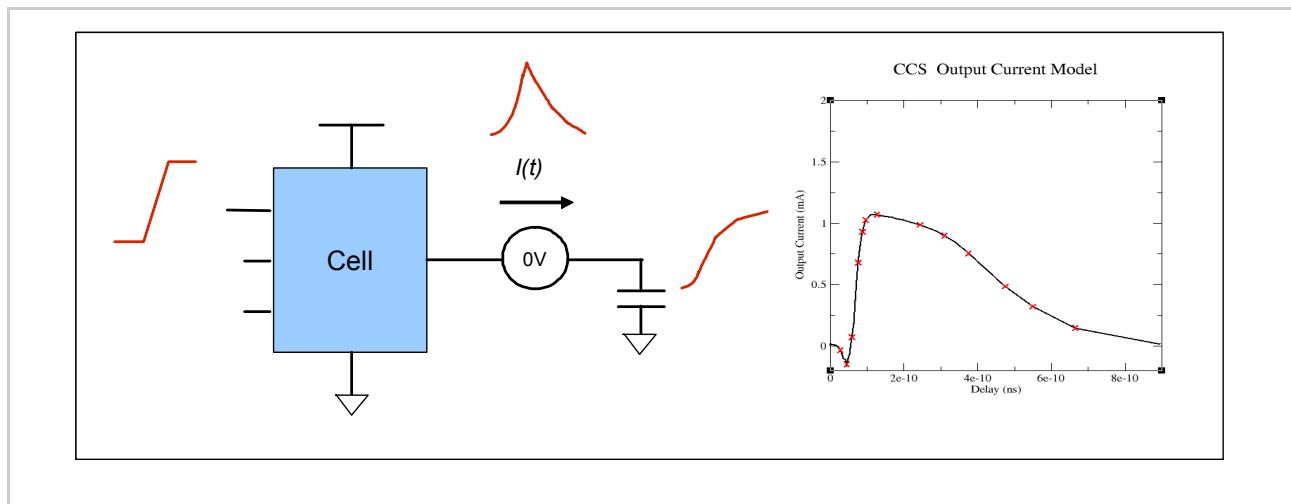
upper slew measurement points (default 30% to 70% of supply) for rising transitions and from the upper-to-lower measurement points (default 70% to 30% of supply) for falling transitions. The slew measurement points can be reset using the **measure_slew_***.

All timing arcs are automatically and exhaustively characterized. For each arc, every side input combination is considered. If an arc has conditional delays, then all conditional tables will be generated, as well as the unconditional table which contains the worst-case results for each table entry.

Input transition and output load indices may be specified uniquely for each arc, cell or group of cells using the **define_index** and **define_template** commands.

CCS

The CCS timing model includes output current characterization using a similar circuit to the circuit used in NLDM delay and transition measurements. In addition, a voltage source is attached to the output pin before the load capacitor to measure current flowing out of the pin.



Circuit used for CCS Driver Model

The current waveform is automatically simplified into a number of time/value pairs, which accurately model the original current waveform. Liberate will verify the accuracy of the simplified waveform and make the necessary adjustments to create the optimal number of points that maintain the required accuracy.

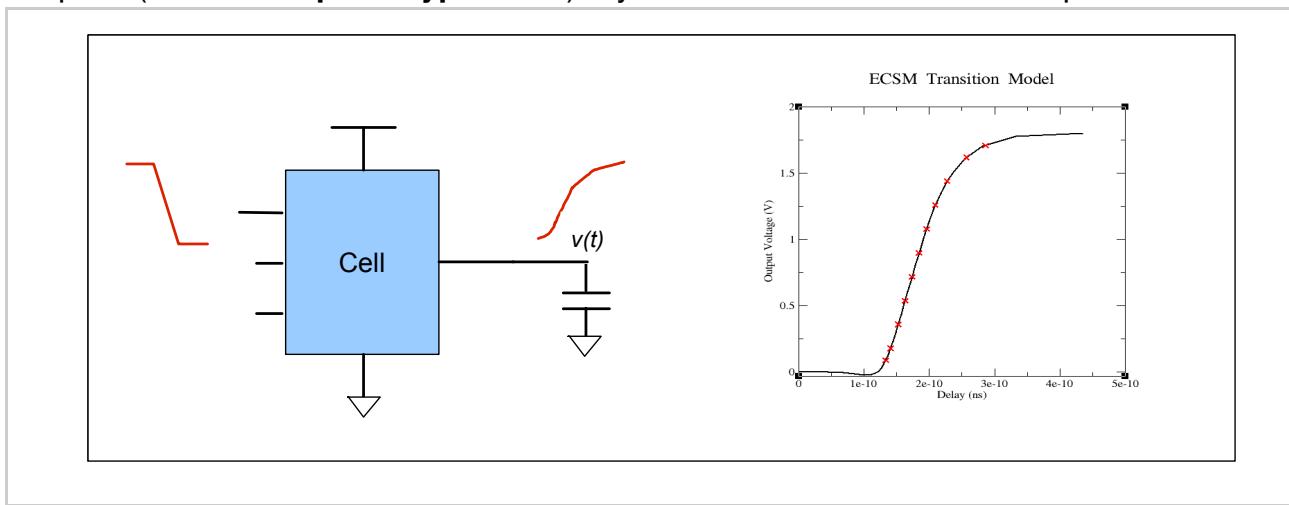
Dynamic selection of points ensures accuracy with minimal data size. The current waveform is measured such that the final voltage reaches within 1% of Vdd or Ground, and integrated

Voltage is calculated within 2%/2ps of NLDM. CCS receiver capacitance is measure for two values above and below the delay threshold. No performance overhead is incurred over NLDM model characterization, as both models are calculated in the same simulation.

Liberate samples currents, then integrates them back to voltage. A comparison is done between this "reconstructed" voltage and the original voltage. Liberate will stop choosing more points when any one of the following criteria is satisfied. Liberate will select up to `ccs_max_pts` or until the `ccs_abs_tol` or the `ccsp_rel_tol` has been met. Note that increasing the number of points and lowering the `ccs_abs_tol/rel_tol` will increase the library size. It has no impact on characterization runtime.

ECSM

The ECSM delay model includes output voltage characterization using the same circuit as in NLDM delay and transition measurements. In addition, many more time points are captured during the output transition. The voltage thresholds for these time points are set by the ECSM template (`define_template -type ecsm`). By default Liberate measures 11 points.

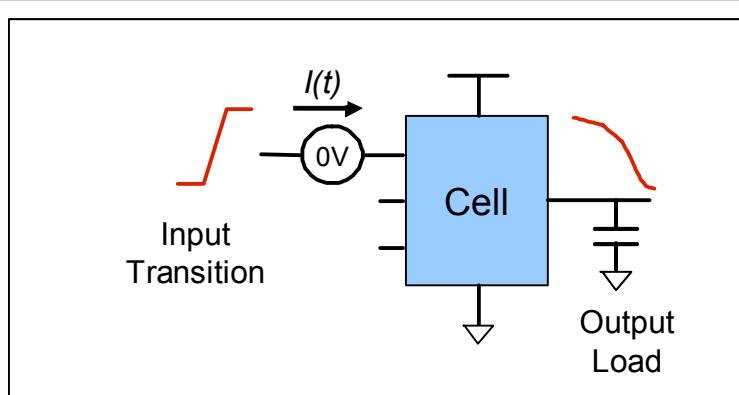


Circuit used for ECSM Driver Model

Pin Capacitance

NLDM Capacitance

Pin capacitance is characterized by measuring the current injected into the input pin over a time period, i.e. charge divided by the voltage change on the input pin over that same duration of time. See circuit below:



Circuit used for pin capacitance measurement

Pin capacitance is measured at the same time as delay and transition. The reported capacitance is the worst capacitance value, as measured from each timing arc originating from the pin and for each table entry in the arc.

For rising transitions, pin capacitance is measured when the input transitions from the **measure_cap_lower_rise** threshold (default ground) to the **measure_cap_upper_rise** threshold (default 50% of supply). For falling transitions, pin capacitance is measured when the input transitions from the **measure_cap_upper_fall** threshold (default supply) to the **measure_cap_lower_fall** threshold (default 50% of supply).

Note: If the input pin always has an on parasitic pull-up or pull-down resistance, the pin capacitance measurement reports a false (large) pin capacitance. In this case, the **user_data_override** variable can be used in combination with the **-user_data** option of the **write_library** command to override the characterized pin capacitance with a specified value.

CCS Receiver Capacitance

The CCS receiver capacitance measurement utilizes the same circuit as the NLDM pin capacitance measurement, where current flowing into the input pin is captured during the input transition.

For rising transitions, current is measured from the input at 0 Volts to the delay measurement point (default 50% of supply) for the receiver_capacitance1, (C1) and from the delay measurement point (default 50% of supply) to the slew upper measurement threshold (default 70% of supply) for receiver_capacitance2 (C2). For falling transitions, current is measured from the input transitioning from VDD to the delay measurement point (default 50% of supply) for the receiver_capacitance1 (C1), and from the delay measurement point (default 50% of supply) to the slew lower measurement threshold (default 30% of supply) for receiver_capacitance2 (C2). The actual capacitance values are determined by integrating the current over the range of the curve and dividing that by the change in voltage.

ECSM Capacitance

ECSM capacitance characterization utilizes the same measurement circuit as the NLDM model. In ECSM capacitance measurement, the capacitance values for all input-slew/output-load combinations are reported in a table, not just the worst-case.

For rising transitions, current is measured from the input at 0 Volts to the value listed in the "threshold_pct" attribute in the Liberty model. For one-piece models, this is usually 0-50% of the supply. For n-piece models, the measurements start at 0 Volts and continue to the threshold_pct listed in each table. Capacitance measurements for falling transitions start at the upper rail and end at the threshold_pct value. The actual capacitance values are determined by integrating the current over the range of the curve and dividing that by the change in voltage.

Timing Constraints

Liberate automatically determines timing constraints on sequential cells. Liberate does not require the logic function or stimulus to be specified, merely that each pin be classified into one of the following types using the **define_cell** command: **input**, **output**, **bidi**, **clock**, **async** (set/reset).

Setup and Hold

In general, Setup and hold measurements involve sweeping the data-pin transition with respect to the clock-pin transition. Sometimes the data sweeps toward the clock causing the

delay or slew to degrade, and sometimes the clock transitions first and the data sweeps backwards toward the clock until a glitch is detected on the output. The failure criteria can be a degradation in delay greater than **constraint_delay_degrade** (10%), a degradation in slew greater than **constraint_slew_degrade** (50%), or a glitch on the output greater than **constraint_glitch_peak** (10% of Vdd). By default, flip-flops use delay degradation for both setup and hold, and latches use delay degradation for setup and glitch peak for hold. Flip-flops can use glitch peak for hold by setting variable **constraint_glitch_hold**.

In some cases, the preferred metric cannot be used to measure a constraint. In such a case the required metric is used instead. For example, a setup for an enable pin on a flip-flop going inactive can only be measured using the glitch peak metric rather than the normal delay degradation.

There are several algorithms available for measuring Setup and Hold time built into Liberate including: Pass/Fail Bisection, Delay based Bisection and combinational. Liberate will use the following flow to measure constraints:

Pass/Fail bisection: This method requires a passing and a failing logical simulation (switching and non-switching output) to bound the constraint measurement. If this method cannot produce any vector for the constraint arc that passes the constraint check, and if **constraint_combinational**=0, then output 0 values for the constraint arc.

If **constraint_combinational**=1 and the pass/fail bisection fails, then use a delay based linear search for constraint characterization. In this method, a linear search is performed using a step size specified by the control variable **constraint_combinational_step_size** where a pass and a fail are defined strictly by the delay degradation (without regard for logical pass/fail). If this method fails to characterize the constraint, then Liberate will output 0 values for the constraint arc.

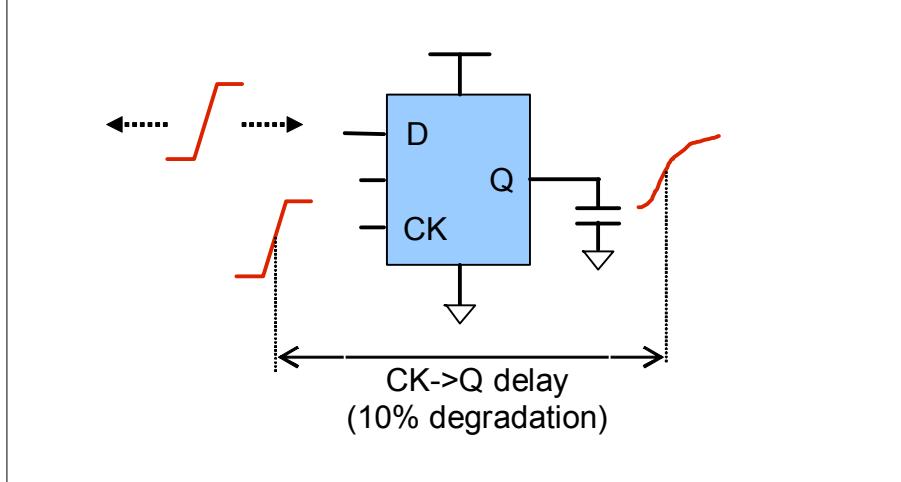
If **constraint_combinational**=2 and the pass/fail bisection fails, then use a formula based method. This algorithm uses a formula that is based on the delay difference of the pin and related pin transition values. The formulae used are:

```
setup_rising rise_constraint value =  
related_pin_transition - constrained_pin_transition + ${constraint_margin}  
if < 0 then output "0" into the library  
hold_falling fall_constraint value =  
constrained_pin_transition - related_pin_transition + ${constraint_margin}  
if < 0 then output "0" into the library
```

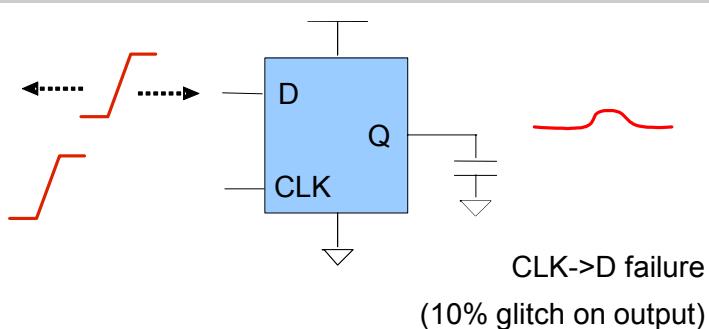
Note: constraint_margin defaults to 2ps

By default, the smallest output capacitance is selected as a load. This can be changed by using the **constraint_output_load** variable to load the output pin to ensure conservative setup

and hold values. If a cell contains multiple output pins, such as Q and QN, then each output pin will be loaded with the same capacitance value.



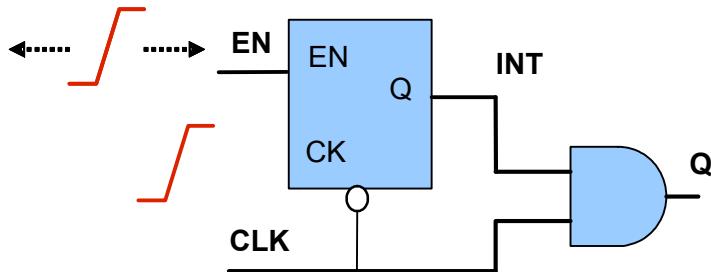
Setup and Hold Calculation



Hold Calculation for Latches

A smart binary search is used to find setup and hold. The search range is automatically determined by Liberate, based on proprietary circuit analysis techniques. Which output pin to

monitor is set by the **constraint_output_pin** variable. It can be an external pin or an internal node.



Typical Clock Gater Circuit

Clock-gating circuits such as the example shown in the figure above require special setup and hold measurement criteria.

For Setup-1, EN rising is swept toward a rising CLK. Then EN->INT and CLK->Q degrade, but CLK->Q is the critical path, so choose CLK->Q degrade as the failure criteria.

For Hold-1, EN Falling sweeps backwards toward CLK rising. Then INT glitches and eventually falls, and then Q delay degrades. But since INT glitches first, use that as the failure criteria.

For Setup-0, EN falling is swept toward a rising CLK. Then EN->INT degrades, but depending on the internal race between INT falling and CLK rising, output Q may glitch, so monitor both EN->INT delay degradation and Q glitch.

For Hold-0, EN rising is swept backwards to a rising CLK. Then INT will at first glitch and eventually rises, and then Q glitches. But since INT glitches first, use that as the failure criteria.

These special checks for clock-gater circuits can be disabled using **constraint_clock_gater** set to a "0".

Dependant constraint characterization

Liberate can characterize dependant setup and hold. An existing ldb can be used to re-characterize the dependant setup (or hold) time. Use the **constraint_dependent_setuphold** variable to enable dependant setup and hold characterization. A value of **1** requests re-characterization of setup. A value of **2** requests re-characterization of hold. To re-characterize for dependant setup and hold, an ldb must be loaded using **read_ldb** that includes standard setup and hold data. Default = 0 (standard setup and hold characterization). Use the

constraint_dependent_setuphold_margin variable to add a margin to the hold (setup) time when re-characterizing for dependent setup (hold) constraints. Use the **constraint_dependent_setuphold_margin_ratio** variable to add a margin to the hold (setup) time when re-characterizing for dependent setup (hold) constraints.

Usage:

```
read_ldb existing.ldb.gz# from a previous run
# Specify which constraint. Default: 0, 1=rechar setup, 2=rechar hold
set_var constraint_dependent_setuphold 2
char_library
write_ldb dependent.ldb
```

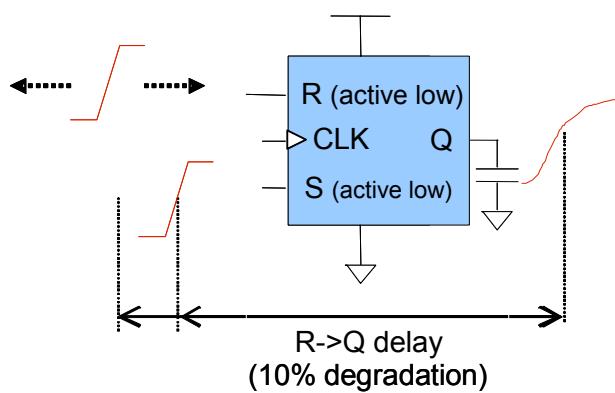
Recovery and Removal

Recovery and removal measurements are analogous to setup and hold, with the data pin replaced by an asynchronous set or reset pin. Recovery and removal between two asynchronous set and reset pins are also captured in a similar fashion. These models appear in Liberty format as `timing_type` recovery and removal. Control of the tolerances used in acquiring removal constraints can be controlled separately from hold tolerances by using the parameter **removal_glitch_peak**.

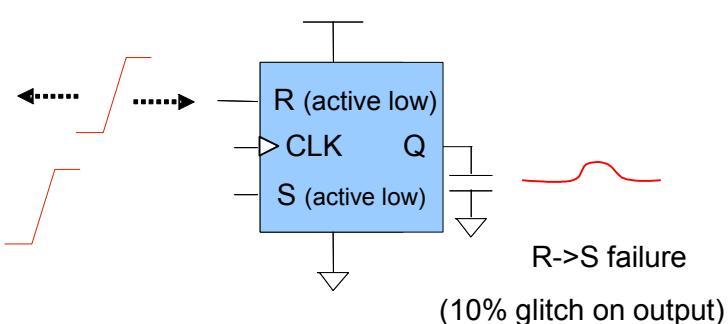
Non-sequential Setup and Hold

Non-sequential Setup and Hold measurements are analogous to setup and hold, but involve two asynchronous set or reset pins. These models appear in Liberty format libraries as `timing_type non_sequential_setup` and `non_nonsquential_hold`. To change the `timing_type` to recovery and removal, set the parameter **nonseq_as_recrem** to 1. All Non-sequential Setup and Hold measurements are done using the de-assertion edges of both asynchronous pins.

The figures below illustrate how constraints are measured for two asynchronous pins. In this illustration, the Reset (R) pin is dominant, and the constraint is S->R. If a constraint is measured for a pin that is not dominant, such as R->S, it may not be possible to measure a change in the output pin. In such a case, Liberate forces the measurement of S->Q by finding an internal node that toggles as S toggles.



Non-sequential Setup acquisition



Non-sequential Hold acquisition

Min Pulse Width

Minimum pulse width is measured for all clock and asynchronous set and reset pins. The slew to use for the minimum pulse-width characterization can be defined using the `mpw_slew` variable. This slew is used for both rising and falling transition of the `clock` or `async` signal. The `mpw_slew_clock_factor` can be used to change the slew for clocks to a ratio of the `mpw_slew`. The ratio of the fall to the rise slew can be changed using the `mpw_skew_factor` variable. Setting this to less than 1.0 (default) will decrease the fall slew and increase the rise slew.

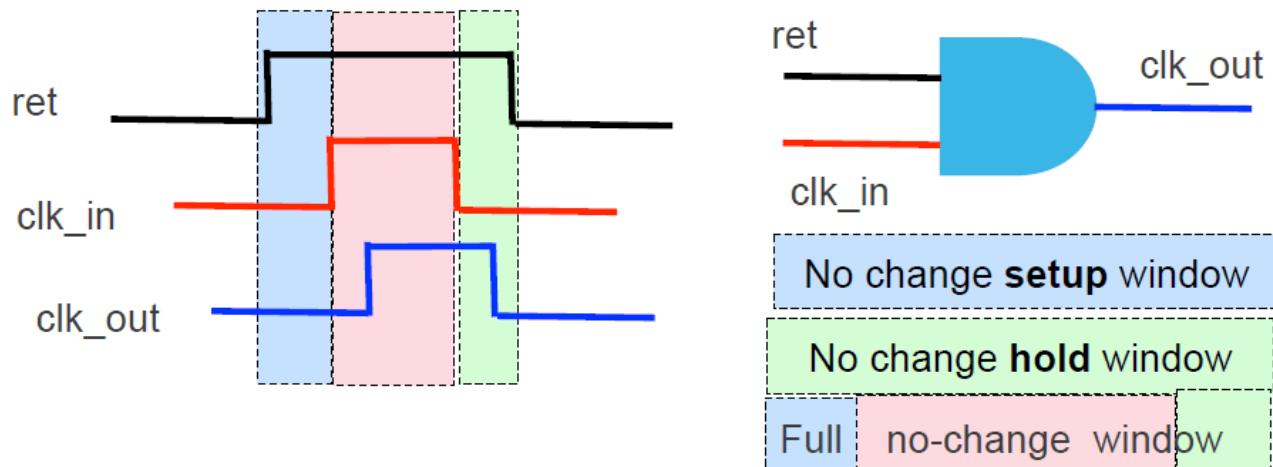
The minimum output load (taken from all the delay arcs related to this pin) is used as the load for calculating the minimum pulse-width. The minimum pulse-width is determined by a binary search, wherein the pulse-width will shrink until an appropriate failure criterion is met. There are 2 different cases. In one case, the output will normally switch once and the failure criterion in this case is met when the output fails to switch. In another case, the output will pulse like the input and the failure criterion is met when the output peak voltage fails to meet the glitch-peak as set by the `mpw_glitch_peak` control variable. In both cases, the final output voltage is checked that it is at the correct voltage. The `constraint_output_pin` variable specifies which output pin or internal node to monitor. The minimum pulse height permitted when the clock pulse becomes triangular can be set using `mpw_input_threshold`. If the input waveform peak drops below this threshold before the `mpw_glitch_peak` failure threshold is reached, then the characterization will end. The reported `min_pulse_width` value will be determined from the final input waveform.

By default, Liberate generates state-dependent `min_pulse_width` attributes. Set the `mpw_table` variable to generate tables of min pulse width values based on the input slew of the clock. Note using tables will increase run-time.

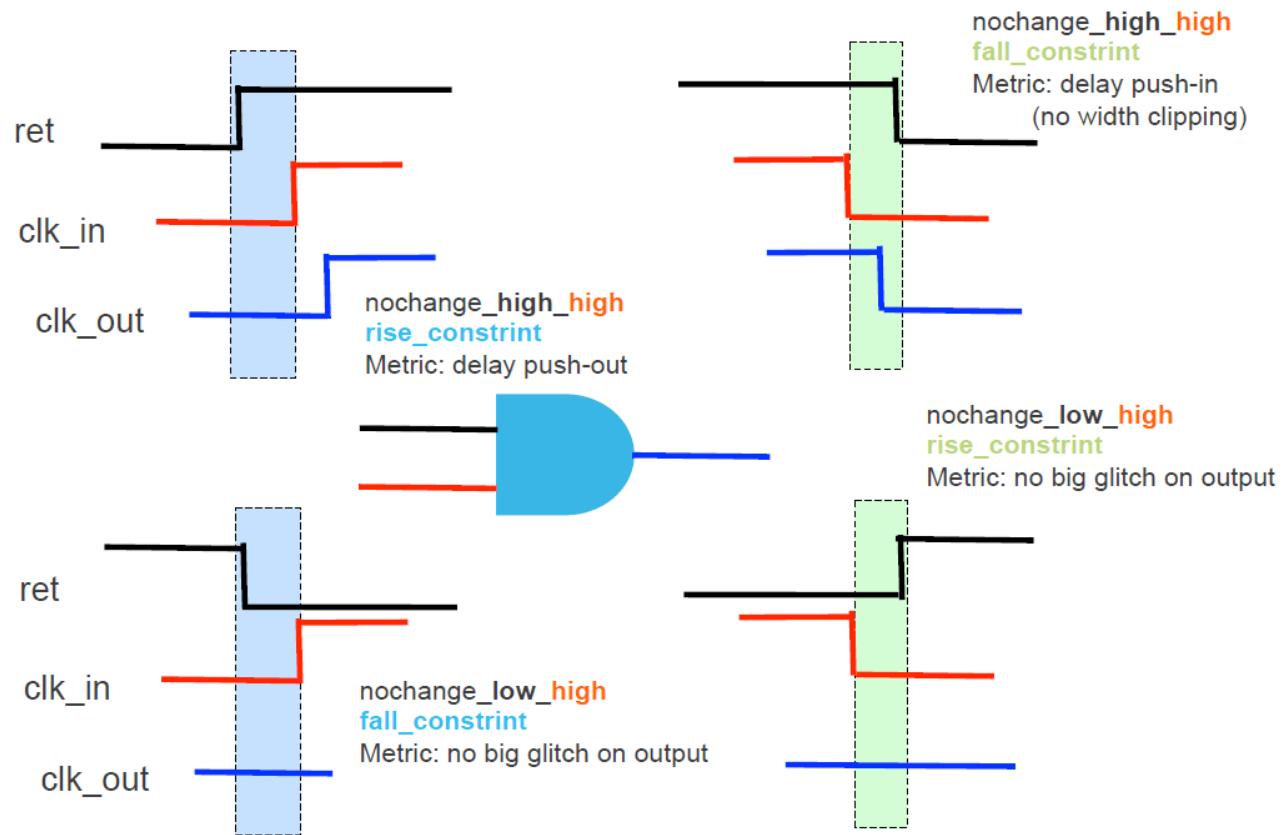
The mpw constraint will be filtered by direction for clock gater circuits, depending on the setting of the `clock_gating_integrated_cell` attributes.

No Change Arcs

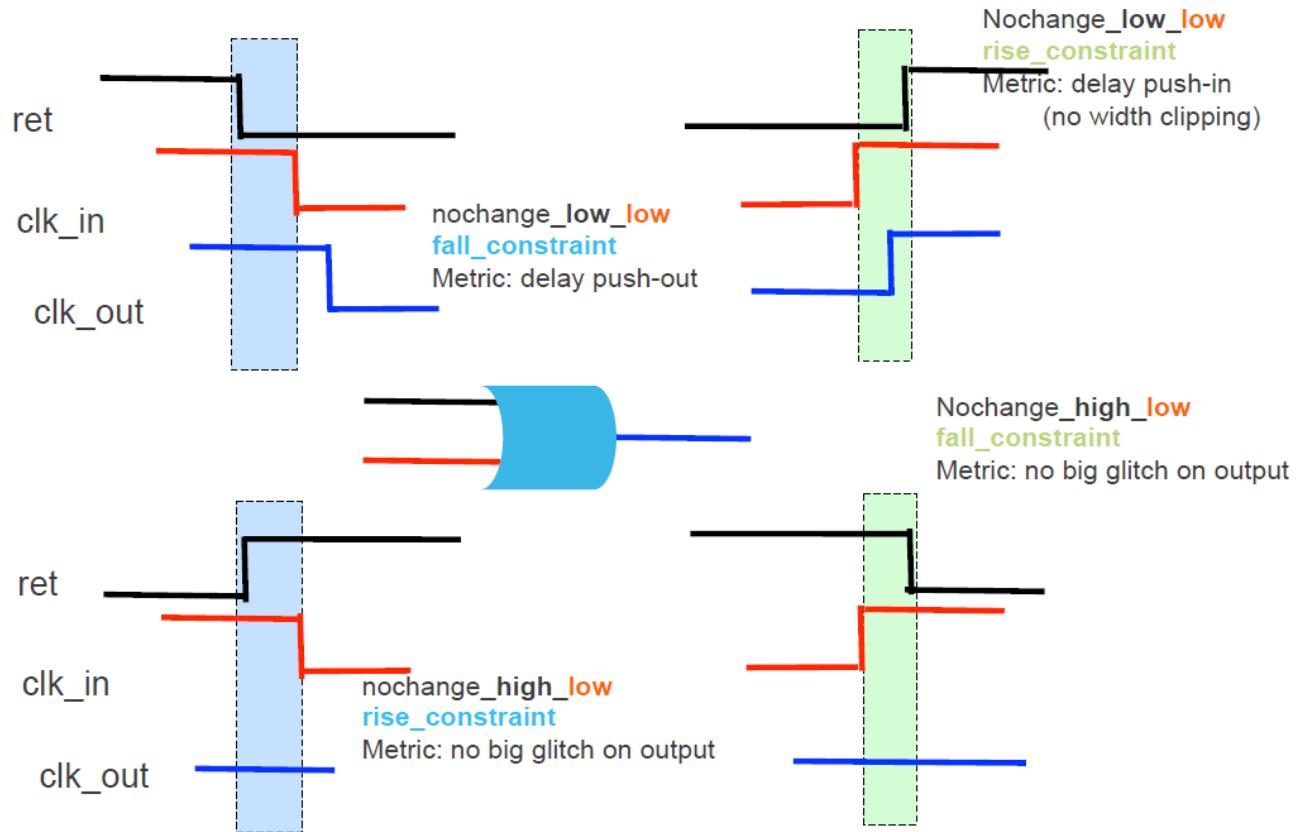
No change timing constraint checks a constrained signal against a level-sensitive related signal. The constrained signal must remain stable during an established setup period, for the width of related pulse, and during an established hold period. The following figure shows an example of no change timing window.



No Change Constraint: Types (AND - gater cell)



No Change Constraint: Types (OR - gater cell)



You do not need to specify `-metric` on `define_search` explicitly. It is automatically identified by the tool. The following table mentions how to choose `-type` for no change arc and how `-metric` is used in characterization.

Table 7-1 No Change Arcs

| Gater logic | | Table - How to chose nochange specific -type on define_arc & metric used for char | | | |
|----------------------------|--------|---|-------------------|-------------------|------------------|
| AND | / NAND | nochange_high_high | nochange_low_high | nochange_high_low | nochange_low_low |
| en & clk / !(en & clk) | | delay | glitch | x | x |
| len & clk / !(len & clk) | | glitch | delay | x | x |
| en&!clk / !(en&!clk) | | x | x | delay | glitch |
| len&lclk / !(len&lclk) | | x | x | glitch | delay |
| | | | | | |
| OR | / NOR | nochange_high_high | nochange_low_high | nochange_high_low | nochange_low_low |
| en + clk / !(en + clk) | | x | x | delay | glitch |
| len + clk / !(len + clk) | | x | x | glitch | delay |
| en + lclk / !(en + !clk) | | delay | glitch | x | x |
| len + lclk / !(len + !clk) | | glitch | delay | x | x |

No Change Constraint: Char Methodology

- The characterization methodology remains the same for setup and hold characterization.
- The characterization is based on a single edge of data or clock and there is no pulse passed in simulation.
- The clock-to-out delay push-in method is exactly the same as delay push-out method except that the delay change expected is in the other direction (that is lesser than the reference delay).

No Change Constraint: Control/Setup

- Set the following variable before `char_library`.

```
set_var nochange_mode 1
```

If this variable is set to 0 and you have `nochange* define_arc` specified then you can control the modeled value.

- Relevant nochange arcs should be added in template file:

```
define_arc -type nochange_high_high -related_pin CK -pin EN $cell
define_arc -type nochange_low_high -related_pin CK -pin EN $cell
```

- All nochange types supported are:

- nochange_high_high , nochange_low_high
- nochange_high_low , nochange_low_low

- How to chose one of the above two sets for `define_arc`

See [Table 7-1, “No Change Arcs,” on page 652](#) for using the correct combination based on your cell function and logic.

No Change Constraint: Modeling

```
pin (EN) {
  direction : input; timing () { related_pin : "CK";
  timing_type : nochange_high_high;
  rise_constraint (constraint_template_3x3) {
  }
  fall_constraint (constraint_template_3x3) {
  }
}

timing () {
  related_pin : "CK";
  timing_type : nochange_low_high;
  rise_constraint (constraint_template_3x3) {
  }
  fall_constraint (constraint_template_3x3) {
  }
}

documented for CCR No.
```

Power Models

Power is modeled in a Liberty file in separate sections. Power is divided into leakage, hidden (internal_power under the input pin), and power (internal_power for switching outputs that is found under an output or bidi pin). The following sections will discuss how Liberate models these power types and the variables available to control the modeling.

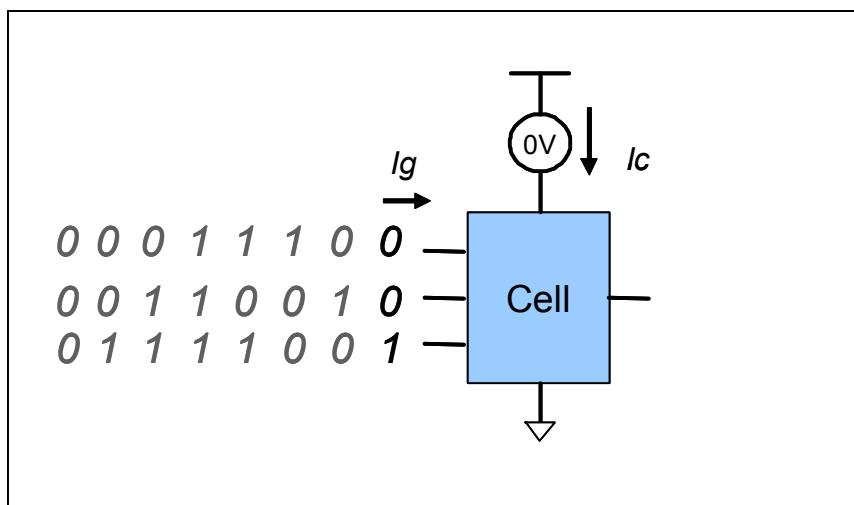
Liberate can completely ignore the power contribution for a specific supply pin. To ignore the power for a specific supply pin, add the **ignore_power** option to the [set_gnd / set_vdd](#) commands.

Leakage Power

All possible input logic combinations are evaluated for state-dependent leakage characterization. Both channel leakage, **lc**, (the dominant leakage component in today's technology) as well as gate leakage, **lg**, (projected to grow significantly in future technologies)

are characterized. The reported leakage power is the sum of the quiescent current from the supply and the current from logic high inputs, multiplied by the supply voltage. Use the **define_leakage** command to specify the desired leakage states to characterize.

The correct internal voltage is determined by the circuit simulator with **.nodeset** applied to each internal and output node. The simulator performs the DC solution of the circuit, after which the currents **Ic** and **Ig** are measured. The circuit used to measure leakage is shown below:



Circuit Used for Leakage Power Measurement

The **leakage_mode** variable can be used to control the voltage applied when computing the leakage. By default, the measured dc current will be multiplied by the supply voltage. This will result in a leakage value of 0 for all gnd supplies operating at 0 volts. For other supported modes, see the **leakage_mode** variable.

The **power_subtract_leakage** variable can be used to enable the subtraction of the leakage power from the internal and hidden power. This variable defaults to 1 which will trigger the subtraction of the average of the pre and post transition leakage power when **voltage_map** is not enabled. See the **power_subtract_leakage** command for the other modes that are available.

The **pin_based_leakage** variable can be used to output state dependent leakage without the related power pin. This can help reduce the overall size of the output library. The **toggle_leakage_state** variable tells Liberate to toggle all clock pins once before measuring leakage. The result is that the internal states will be initialized and the *when* in the leakage group will not include the output pin. The **power_combinational_include_output** variable can

be used to directly control if the when in the output leakage group for combinatorial cells will include the output pin. The power_sequential_include_complementary_output variable can be used to request that the when in the output leakage group for sequential cells will include complementary output pins.

Liberate will include the input pin DC leakage (Lpin_dc) in the value in the leakage groups. To not include the Lpin_dc in the leakage, set the variable leakage_add_input_pin to **0**.

Liberate will limit the number of leakage states to a maximum of 256 states (vectors). This limit can be changed by setting the variable max_leakage_vector to the desired maximum number of leakage states to characterize. If the default_leakage command is specified, then the *Liberate Inside View* will be disabled for leakage characterization, and only the user specified leakage when states will be characterized.

The conditional_leakage variable can be used to disable the output of state dependent leakage groups.

The keep_dcap_leakage variable can be used to request the characterization and subsequent reporting of leakage for decap cells.

Switching and Hidden Power

Switching power (see `define_arc -type power`) measures the energy dissipated by the cell when one or more inputs switch which causes one or more outputs to switch. It includes shortcircuit power plus internal switching power that is dissipated during the charging and discharging of internal capacitive loads. The energy dissipated via the non-switching input pins is also measured as it contributes to the total dissipated energy inside the cell.

Hidden power (see `define_arc -type hidden`) measures the energy dissipated inside the cell when one or more inputs transition and do NOT result in any output transition. Hidden power is reported in the Liberty file as an `internal_power` group, usually state dependent, in an input pin group. If more than one input is switching, as might be the case when the cell has dual inputs, the hidden power reported will be divided by the number of switching inputs. The parameter hidden_power can be used to enable or disable hidden power characterization for all combinational cells. Sequential cell inputs will always have `hidden_power`. By default, hidden power will be calculated for all input pins including combinational gate inputs and will be reported only with full state dependency. No default hidden power group will be output. The user can disable state dependency for hidden power with the variable conditional_hidden_power. When this variable is set to 0, no state dependent hidden power will be reported. Instead, only a default (state independent) hidden power will be reported. It is sometimes desirable to subtract hidden power from switching power. To enable this, use the subtract_hidden_power variable. When this feature is enabled, and hidden power has

been characterized, then Liberate will choose the first matching hidden power value to subtract from the switching power in the output pin group.

Current is continuously monitored through all Vdd (see **set_vdd**) and Gnd (see **set_gnd**) supplies from the beginning of the input transition to the end of all the internal and output pin transitions (0.5% of supply for falling transition and 99.5% of supply for rising transition). For both rising and falling transitions, the energy dissipated during the charging or discharging of the output load capacitor is subtracted from the total energy calculation (see [power_subtract_output_load](#)). For cells with multiple output pins, the energy dissipated by other switching outputs is also accounted for. The parameter **pin_based_power** can be set to 0 to only include power dissipated via the positive power supplies (see **set_vdd**), excluding the power in the non-switching input pins. This can be useful for correlating power results between libraries from other characterization tools. Alternatively, when this parameter is set to a 2, Liberate will monitor the current (including Miller capacitance) to each cell input. If the input is tied to a positive/negative voltage, that current is added to the vdd/gnd power. This non-switching input pin current is not accounted for in any other power measurement. This is because when the driving cell is characterized, the receiving cell is not included, and even if it was, it may not be switching in the correct way to trigger the Miller capacitance currents.

Liberate will normally output power that is not related to a specific power pin. Set the variable **voltage_map** (default=0) and the variable **pin_based_power** (default=1) to request that power be reported by power pin. With these variables set, the output library will have **voltage_map** groups, **pg_pin** groups, and the power will be reported with the **related_pg_pin** attribute.

Liberate will check that the sum of the **rise_power** and the **fall_power** will not be negative. If the sum is negative, then Liberate will adjust the more negative value such that the sum becomes zero. The variable **reset_negative_power** can be set to 0 to disable this check.

Power Subtraction

Non-Linear Power Models (NLPM) are often generated with the understanding that downstream power tools will sum the data from multiple power groups when reporting power for a specific power event. To prevent power from being over-reported, Liberate can subtract leakage power and/or hidden power. See [power_subtract_leakage](#) and [subtract_hidden_power](#) for more details.

When Liberate's Inside View generates the characterization plan or if all arcs are manually specified (see **define_arc**) with full state dependency, then power subtraction can be a straight-forward process to characterize, model and validate:

Switching Power (internal_power under an output pin) = Simulation value - hidden power - leakage power

Hidden Power (internal_power under an input pin) = Simulation value - leakage power

However, if the characterization plan does not include complete state coverage then selecting the correct hidden or leakage power can be problematic. The most common cause is a verbose template that does not have full state coverage. This does not mean that the power in the library is incorrect, but it may mean that it is more difficult to validate the reported power values.

If an exact leakage state, including the states of output pins, is not included in the characterization plan, then it will not be included in the output library. If leakage power subtraction is enabled, then Liberate will select either a leakage state that has a full or partial overlap or a default leakage power. For example, if the define_leakage commands only include WHEN states for input pins, then any combination of states on the output pins would still allow for a match.

When power tools add leakage power, they will select the best match for the actual state that is represented in the model. On large cells, there can be a difference of 60% leakage power from state to state, so it is important to have Liberate subtract the same leakage vector that the power tool will add back in, even if this means that the power number represented in the model does not necessarily match the SPICE simulations.

The problem is more complicated when subtracting hidden power from switching power arcs. It can cost a prohibitive amount of runtime to characterize and model all possible combinations of input and output pin states for hidden power in order to be able to validate any given state of switching power. To reduce this burden and maintain accuracy, some users will write specific define_arc statements that align vectors used for hidden power and switching power. If the vectors are not aligned, the power values will be more difficult to validate.

Power Validation

Validating power calculations is one of the challenging aspects of library qualification, especially since relevant power data may not be contained in a single SPICE simulation. To reduce the burden, Liberate can print power equations and associated values for validation purposes. See power_info for details on how to enable this feature.

Common Usage Modes for Power and Leakage

No PG-pin syntax

1/2 of Vdd, Gnd, input energies, subtract 1/2 of output load energy

| | | |
|-------------------------|---|-------------|
| set_var voltage_map | 0 | # (default) |
| set_var pin_based_power | 2 | |

```
set_var power_subtract_leakage      1      # (default)
set_var power_subtract_output_load all    # (default)
set_var leakage_add_input_pin      1      # (default)
set_var leakage_mode               1      # (default)
```

PG-pin syntax

1/2 of Vdd, Gnd, input energies, subtract 1/2 of output load energy

```
set_var voltage_map                1
set_var pin_based_power           2
set_var power_subtract_leakage    1      # (default)
set_var power_subtract_output_load all    # (default)
set_var leakage_add_input_pin    1      # (default)
set_var leakage_mode              1      # (default)
```

No PG-pin syntax

Vdd energy, subtract output load energy on rise

```
set_var voltage_map                0      # (default)
set_var pin_based_power           0
set_var power_subtract_leakage    1      # (default)
set_var power_subtract_output_load all    # (default)
set_var leakage_add_input_pin    1      # (default)
set_var leakage_mode              1      # (default)
```

PG-pin syntax

Vdd energy, subtract output load energy on rise

```
set_var voltage_map                1
set_var pin_based_power           0
set_var power_subtract_leakage    0      # (default)
set_var power_subtract_output_load all    # (default)
set_var leakage_add_input_pin    1      # (default)
set_var leakage_mode              1      # (default)
```

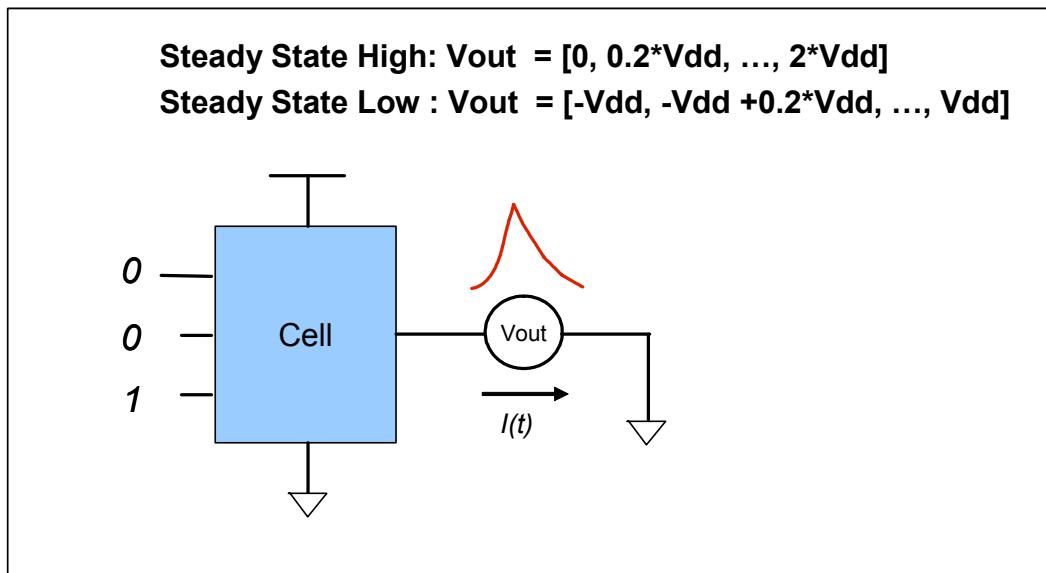
Signal Integrity Models

Liberate supports both Liberty SI models and composite current-source noise models (CCSN). The Liberty SI model contains constructs for steady state current, DC noise margins, hyperbolic noise curves and noise immunity tables. To characterize Liberty SI models, use the **si** option to **char_library**.

CCSN models are characterized for each channel-connected collection of transistors (stage) that connects to each input, inout or output pin. For timing arcs that have two or less stages between the input and the output pin, the CCSN data is stored on each timing arc. For timing arcs with more than two stages between input and output, such as flip-flops, the CCSN data is stored on the appropriate input, output or inout pin. There are four components to the CCSN model: DC current, output voltage, Miller capacitance and propagated noise. To characterize CCSN models use the **ccsn** option to **char_library**.

Steady State Current

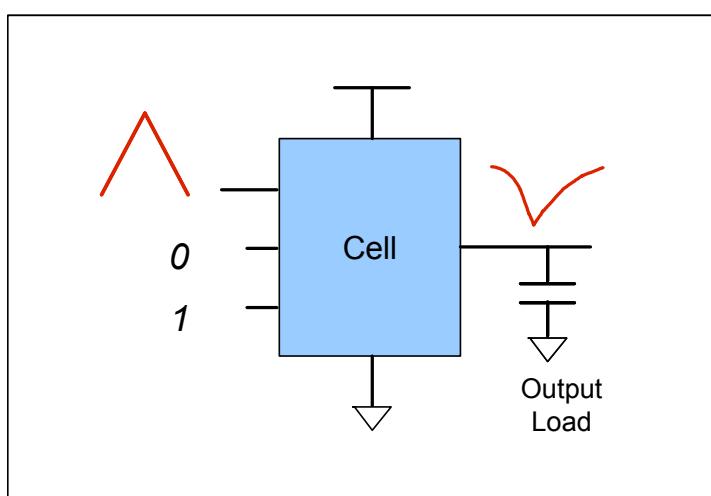
Steady state current is characterized for each cell output/inout pin by putting the output into steady state, connecting a voltage source to that pin and varying the voltage level of the voltage source from 0 volts to $2*Vdd$, for steady-state high, and from $-Vdd$ to Vdd , for steady-state low. The current through the voltage source is captured in the steady-state current table. This is performed for each distinct logic state where the output goes low or high. The intermediate voltage values to be tabulated can be specified by defining a template of type `si_iv_curve`.



Circuit used for Steady State Current Calculation

Noise Immunity Curves

Noise immunity is characterized for each cell input/inout pin by injecting input glitches of various widths and heights at that pin for a set of output loads. For each width/load combination, a search is performed to determine the input glitch- height that causes a noise failure (where a failure is defined by a propagated noise pulse whose height exceeds the **immunity_glitch_peak**, default 5% of Vdd). The noise-immunity curve is a table of failure heights for the widths and loads defined by the given template of type **si_immunity**. A unique noise-immunity curve is generated for each valid logic state.



Circuit used for Noise Immunity

The input noise-glitch used is assumed to be an isosceles triangle, unless skewed using the **immunity_noise_skew_ratio** parameter. Characterizing for noise-immunity is enabled by the **si** option to **char_library** and by defining a **si_immunity** template for that cell.

Hyperbolic Input Noise, DC Noise Margin

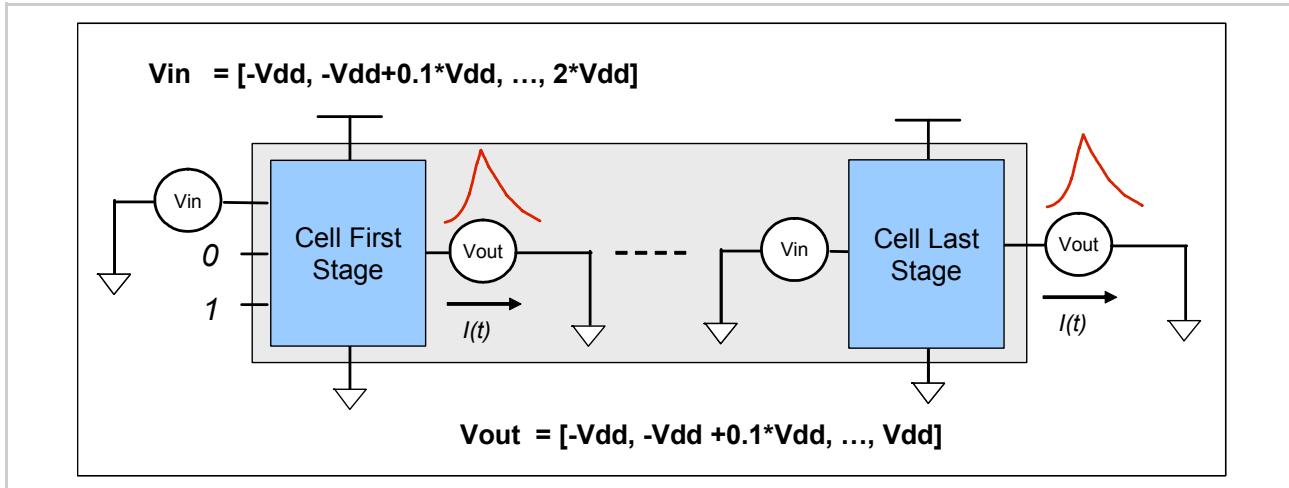
In addition to the noise-immunity curve, hyperbolic input noise (low and high) and DC noise-margins can be generated in the output library by setting the parameter **input_noise**. These values are extracted from the data captured by calculating the noise immunity curve. The hyperbolic area, height and width are determined by fitting the noise-immunity curve for the minimum load to a hyperbolic function using a least-squared fit. The DC noise margin is the height of the hyperbolic curve.

Composite Current Source Noise (CCSN) Models

CCSN DC Current

The CCSN DC current model is calculated for both the input (first) and output (last) stages of a cell. For each stage, under every logic condition, a simulation is performed where the input to the stage (input/inout pin or internal node) and the output of the stage (output/inout pin or internal node) are both connected to a voltage source. Each voltage (input and output) is varied from $-V_{dd}$ to $2*V_{dd}$ and the DC current through the output voltage source is recorded.

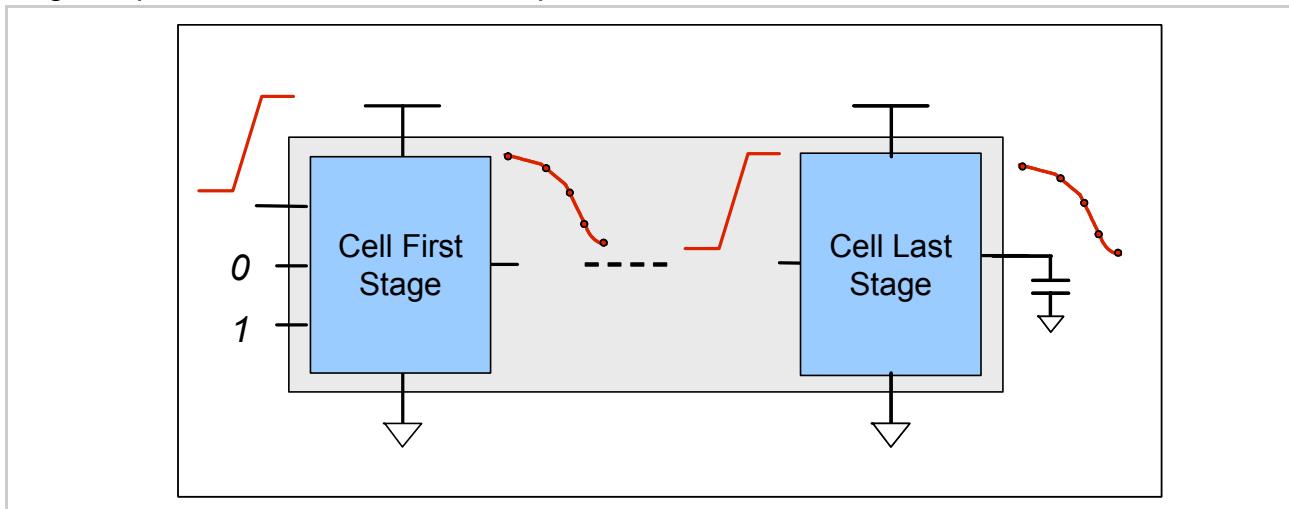
The voltage steps between $-V_{dd}$ and $2*V_{dd}$ are determined by a pre-defined formula, such that 29 voltage points are sampled for each input and output voltage resulting in a 29X29 DC current table.



Circuit used for CCSN DC Current

CCSN Output Voltage

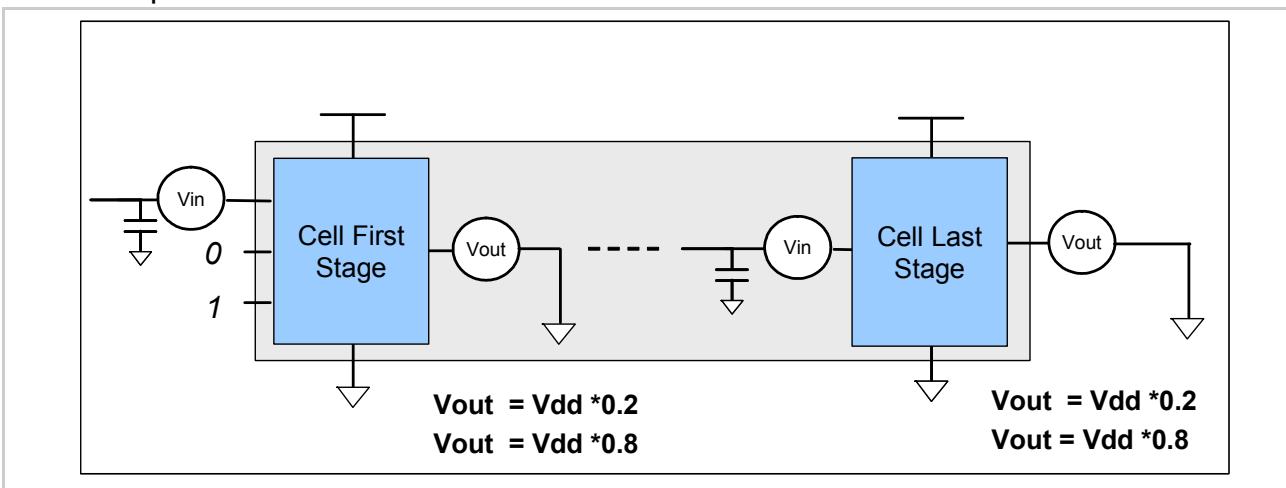
The CCSN output voltage model is calculated by simulating each stage using two linear slews from the input slew indices, and either two loads from the output load indices (if the stage output is a pin) or zero additional load capacitance (if the stage output is an internal node). Consequently, each stage is simulated either two or four times. The output voltage at the stage output is measured at five time-points.



Circuit used for CCSN Output Voltage

CCSN Miller Capacitance

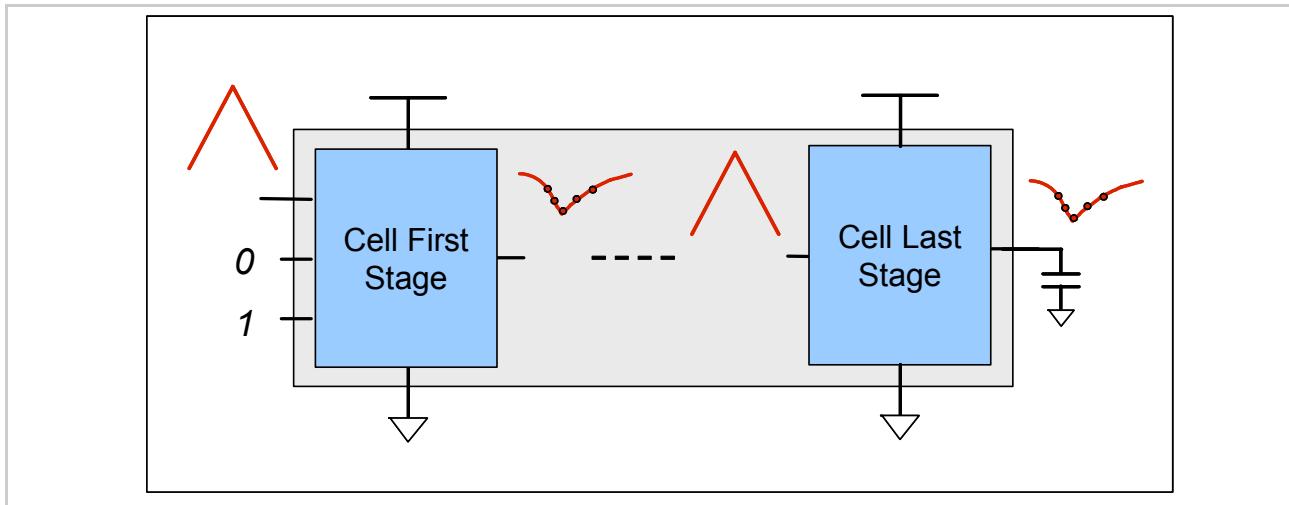
The CCSN Miller capacitance is calculated by placing a voltage source at the output of each stage and a capacitive load at the input pin of the stage. The Miller capacitance is measured by comparing the change in input voltage against a 20% change in output voltage for two different input loads.



Circuit used for Miller Capacitance

CCSN Propagated Noise

The CCSN propagated noise model is calculated by simulating input noise glitch triangles, of various widths and heights, over a small range of capacitive loads for each stage. If the output of a stage is a pin, two loads are used, otherwise a single zero load capacitance is used. Input glitches of the appropriate height and width are automatically determined, to ensure that there is a propagated noise pulse at the stage output. The resulting output glitch voltage is captured using 5 points - 50% of the leading edge, 80% of the leading edge, the glitch peak, 80% of the trailing edge, 50% of the trailing edge.



Circuit used for CCSN Propagated Noise

IBIS Models

IBIS model content

Liberate supports the generation of models for IBIS versions 3.2 and 4.2. The process flow for characterizing IBIS models is described below.

The input/output buffer information specification (IBIS) is an approved standard within the Electronic Industry Alliance (EIA), and is also known as ANSI/EIA-656. Specifications of the IBIS format may be found The IBIS Open Forum website: www.eda.org/ibis

IBIS is a behavioral-modeling specification. It is a standard for describing the analog behavior of the buffers of a digital device using plain ASCII-text formatted data files. IBIS model files are used to perform signal integrity (SI) simulations of printed circuit boards. The information needed to perform this simulation are the voltage-current (V-I) characteristics and switching (output voltage versus time) characteristics of the buffer.

Each IBIS model file contains a general title section and six model sections containing behavioral model data for each pin:

1. Power_clamp

Table of DC current values through the diode connected to Vcc rail, when a DC Voltage is applied to the output pin.

2. GND_clamp

Table of DC current values through the diode connected to Gnd rail, when a DC Voltage is applied to the output pin.

3. Pull_up

The V-I characteristics of the pullup circuit. This data is referenced to the pullup reference voltage: $V_{table} = [Pullup Reference] - V_{in}$, where V_{in} is the voltage referenced to ground.

4. Pull_down

The V-I characteristics of the pulldown circuit. This data is referenced to the pulldown reference voltage: $V_{table} = V_{in} - [pulldown reference]$, where V_{in} is the voltage referenced to ground

5. Rising Waveform

- a.** With resistive load connected to pulldown reference
- b.** With resistive load connected to pullup reference

6. Falling Waveform

- c.** With resistive load connected to pullup reference
- d.** With resistive load connected to pulldown reference

The Ramp value in the IBIS file is always taken from 20% - 80% and will overwrite the following parameters:

```
measure_slew_lower_rise
measure_slew_upper_rise
measure_slew_lower_fall
measure_slew_upper_fall
slew_lower_rise
slew_upper_rise
slew_lower_fall
slew_upper_fall
```

The Liberate IBIS modeling flow has been validated using examples of output, input, and tristate buffer cells. The current version of Liberate does not support the following IBIS constructs. These may be supported in future releases:

- Define Package Model
- '.pkg FILE' and '.ebd FILE'
- Series Pin Mapping
- Dynamic_clamp

Note: the content of IBIS files is case sensitive, except for reserved words and keywords.

Valid scaling factors allowed for IBIS are:

| <u>Symbol</u> | <u>Meaning</u> | <u>Power of 10</u> | <u>Decimal</u> |
|---------------|----------------|--------------------|--------------------|
| t | tera | 10^{12} | 1,000,000,000,000. |
| g | giga | 10^9 | 1,000,000,000. |
| M | Mega | 10^6 | 1,000,000. |
| k | Kilo | 10^3 | 1,000. |

| | | | |
|----------|-------|------------|------------------|
| m | milli | 10^{-3} | .001 |
| u | micro | 10^{-6} | .000001 |
| n | nano | 10^{-9} | .000000001 |
| p | pico | 10^{-12} | .000000000001 |
| f | femto | 10^{-15} | .000000000000001 |

Base units are: Volts, Amperes, Ohms, Farads, Henries, Celsius, and Seconds.

Full abbreviations for the units are allowed, (e.g., pF, nH, mA, mOhm). In addition, scientific notation is supported (e.g., 1.2345e-12)

IBIS modeling flow

Liberate supports the generation of IBIS models using a flow similar to the Liberate Liberty library generation. An IBIS template must be written for each PVT corner IBIS-specific definition. Each PVT is then characterized using the Liberate command "**char_library -ibis -io**". A separate Liberate database file (ldb) is created for each. These characterized results are then merged into a single IBIS file using the Liberate command "**write_ibis_file \$file \$ldbs \$cells**".

The first step is to run liberate to generate one to three set of IBIS enabled LDBs to cover typ, min and max corners. The min corner refers to the simulation setting with minimum voltage. The max corner refers to the simulation setting with maximum voltage. If only one ldb is used, then it is assigned as the 'typ' corner. If two set of LDBs are used, then they are 'typ' and 'min' corners.

The following are required to enable IBIS mode in ldb.

1. Describe the IBIS structure to Liberate:

(required)

```
ibis_define_header
```

Virtuoso Liberate Reference Manual

Liberate Details

```
ibis_define_component
ibis_define_model
ibis_define_pin
```

(depends on IBIS model type)

```
ibis_define_waveform_template
ibis_define_model_selector
```

2. Provide `define_arc` commands to instruct Liberate to generate I/V and V/t simulation data for IBIS. In the `char_library` command, specify the `io` and `ibis` flags to run Liberate in IBIS mode using user defined arcs.

Example:

```
set_operating_condition -voltage 3.3 -temp 25
source template_arcs.tcl
source template_ibis.tcl
char_library -ibis -io -thread 2 -extsim hspice
write_ldb MyCell_typ.ldb
```

After all corner LDBs are generated, run Tcl command 'write_ibis_file' from Liberate to write out IBIS file.

Example :

```
set cells "MyCell"
set file "MyCell.ibs"
set ldbs { MyCell_typ.ldb.gz MyCell_min.ldb.gz MyCell_max.ldb.gz }
write_ibis_file $file $ldbs $cells
```

A three corner IBIS model flow will look like this:

| <u>char_library</u> | <u>write_ibis_file</u> |
|--|------------------------|
| {typ PVT IBIS.tcl} -> Liberate -> [typ.ldb] \ | |
| {min PVT IBIS.tcl} -> Liberate -> [min.ldb] --> Liberate --> [cell.ibs] | |
| {max PVT IBIS.tcl} -> Liberate -> [max.ldb] / | |

IBIS-specific templates must be prepared for each cell. These are defined using these IBIS-specific commands: `ibis_define_header`, `ibis_define_waveform_template`, `ibis_define_component`, `ibis_define_model`, `ibis_define_pin`, `ibis_define_model_selector`. Each of these IBIS-specific commands is described in detail in the "Liberate Commands" chapter.

Example: Liberate command sequence for an IBIS cell characterization:

```
# characterize 3 corners, typ, min, max
set_operating_condition -voltage 3.3 -temp 25

# defines all of the measurement arcs
source template_arcs.tcl

# defines headers, waveform templates, components, pins,
# and models
source template_ibis.tcl
```

Virtuoso Liberate Reference Manual

Liberate Details

```
# characterize using IBIS mode and save the database
char_library -ibis -io
write_ldb lvds_typ.ldb

# After all corners are characterized, write the
# IBIS cell model
set_file "OBuff.ibs"
set_ldbs { OBuff_typ.ldb.gz \
OBuff_min.ldb.gz \
OBuff_max.ldb.gz }
set_cells "MyOBuff"
set_versions {4.3 3.2}
set_revision "1.1"
write_ibis_file -version $versions -rev $revision \
$file $ldbs $cells

##### ****
### Liberate template example for Tri-state IO cell ****
##### ****

## Measure [Ramp] using 20% and 80% threshold
set_var slew_lower_rise 0.20
set_var slew_lower_fall 0.20
set_var slew_upper_rise 0.80
set_var slew_upper_fall 0.80

set_var measure_slew_lower_rise 0.20
set_var measure_slew_lower_fall 0.20
set_var measure_slew_upper_rise 0.80
set_var measure_slew_upper_fall 0.80

## Define input slew (1ns, and no loading cap)
define_template -type delay \
    -index_1 {1.0} \
    -index_2 {0.0} \
    delay_template_1x1

define_template -type power \
    -index_1 {1.0} \
    -index_2 {0.0} \
    power_template_1x1

## Define cell
define_cell \
    -input { A EN PUPD } \
    -output { Y } \
    -bidi { PAD } \
    -pinlist { A EN PUPD PAD Y } \
    -delay delay_template_1x1 \
    -power power_template_1x1 \
    MyCell

## Define pin pull_up and pull_down (must be consistent
## with R_load and V_fixture from IBIS)
define_pin_load \
    -pullup_voltage 3.3 \
    -pullup_resistance 50 \
    load_template_up

define_pin_load \
```

Virtuoso Liberate Reference Manual

Liberate Details

```
-pulldown_resistance 50 \
load_template_down

# These define_arc commands generate I/V and V/t
# waveforms for model when condition '!PUPD'.

# Si-IV steady_state_current_low : PAD pull_down
define_arc \
    -vector {R00RX} \
    -when "!PUPD" \
    -related_pin A \
    -pin PAD \
    MyCell

# Si-IV steady_state_current_high : PAD pull_up
define_arc \
    -vector {F00FX} \
    -when "!PUPD" \
    -related_pin A \
    -pin PAD \
    MyCell

# These 4 define_arc commands generate rising/falling
# waveform V/t table

# PAD Rising Waveform Pullup On
define_arc \
    -vector {R00RX} \
    -when "!PUPD" \
    -pin_load load_template_up \
    -load_dir U \
    -related_pin A \
    -pin PAD \
    MyCell

# PAD Rising Waveform Pulldown Off
define_arc \
    -vector {R00RX} \
    -when "!PUPD" \
    -pin_load load_template_down \
    -load_dir D \
    -related_pin A \
    -pin PAD \
    MyCell

# PAD Falling Waveform Pulldown On
define_arc \
    -vector {F00FX} \
    -when "!PUPD" \
    -pin_load load_template_down \
    -load_dir D \
    -related_pin A \
    -pin PAD \
    MyCell

# PAD Falling Waveform Pullup Off
define_arc \
    -vector {F00FX} \
    -when "!PUPD" \
    -pin_load load_template_up \
    -load_dir U \
```

Virtuoso Liberate Reference Manual

Liberate Details

```
-related_pin A \
-pin PAD \
MyCell

# These 2 define_arc commands generate waveform for
# Power_clamp and GND_clamp I/V table

# Tri-state Enable, steady_state_current_low :
# Power_clamp/GND_clamp
define_arc \
    -type enable \
    -vector {1F0RX} \
    -related_pin EN \
    -pin PAD \
MyCell

# Tri-state Enable, steady_state_current_high :
# Power_clamp/GND_clamp
define_arc \
    -type enable \
    -vector {0F0FX} \
    -related_pin EN \
    -pin PAD \
MyCell

# ****
# The following are define_arc commands to model an OBuff
# output pad. A subckt file: 'termres.sp' is included
# during characterization. A custom output load (harness)
# defined in termres.sp, is applied to OBuff to model
# PAD and PADN pin loading. The 'mid' voltage is taken
# from a previous measurement of the cross voltage of PAD
# and PADN signals with a 100ohm resistor connected as
# termination. 1.2V is used here as typ value.
# For standard timing characterization (delay, transition),
# a global pin cap 'PAD_cap' and 'PADN_cap' is defined in
# the top level, which must be connected here.
# ****

## an example of termres.sp file
.subckt termres_typ pad padn
R1 pad mid 50
R2 padn mid 50
Vmid mid 0 1.20
.ends
.subckt termres_min pad padn
R1 pad mid 50
R2 padn mid 50
Vmid mid 0 1.195
.ends
.subckt termres_max pad padn
R1 pad mid 50
R2 padn mid 50
Vmid mid 0 1.205
.ends
.subckt termres pad padn
R1 pad padn 100
C1 padn 0 'PADN_cap'
C2 pad 0 'PAD_cap'
.ends
```

Virtuoso Liberate Reference Manual

Liberate Details

```
# The define_cell command includes this harness cell
define_cell \
    -input {A SEL PWRDN} \
    -output {PAD PADN} \
    -pinlist {A SEL PWRDN PAD PADN} \
    -harness "termres_typ" \
    -delay delay_template_DL_1x1 \
    -power power_template_DL_1x1 \
    MyOBuff

# A set of define_arc commands for generating
# pullup/pulldown I/V table

# Si-IV steady_state_current_high : PAD pull_up
define_arc \
    -vector "F00FR" \
    -when "!SEL" \
    -related_pin A \
    -dual_pin PADN \
    -pin PAD \
    MyOBuff

# Si-IV steady_state_current_low : PAD pull_down
# ECSCM : PAD Rising Waveform Differential
define_arc \
    -vector "R00RF" \
    -when "!SEL" \
    -dual_pin PADN \
    -related_pin A \
    -pin PAD \
    MyOBuff
```

Example:

The Tcl command for Liberate IBIS :

```
#####
# The following commands can be used in the header:
#
# [IBIS Ver]    version      <-- required
# [Comment Char] comment token <-- optional
# [File rev]    rev          <-- optional
# [Source]      source        <-- optional
# [Notes]       notes         <-- optional
# [Disclaimer]  disclaimer   <-- optional
# [Copyright]   copyright    <-- optional

## ibis_define_header {}
-ibis_ver      : IBIS version, default '4.2'
-file_rev      : user IBIS file version
-comment_char  : IBIS comment character, default "|"
-source        : User specified source description
-notes         : User specified note
-disclaimer    : User specified disclaimer text
-copyright     : User specified copyright text
filename       : IBIS output file name

## ibis_define_component {}
-manufacture   : Component's manufacturer name
```

Virtuoso Liberate Reference Manual

Liberate Details

```
-package          : a list of package R/L/C at typ/min/max condition
-waveforms       : a list of pullup/pulldown waveform templates
-net_typ_vdd     : a list of {net typ_vdd} pairs
-end_time        : The maximum waveform end time to write to IBIS rise/fall
table
components      : IBIS component name(s) list

## This command defines waveforms for use in the ibis_define_component command
## ibis_define_waveform_template {}
  -type            : 'rise' or 'fall'
  -vfix            : V_fixture
  -rfix            : R_fixture
  -lfix            : L_fixture
  -cfix            : C_fixture
  -pins            : list of pins to apply this waveform
  name             : Waveform template name

## ibis_define_pin {}
  -net              : Pin net/signal name from Spice netlist
  -model            : Pin model name
  -enable           : Enable pin signal (port) name from Spice netlist
  -function         : Logic function to set pin to logic 1 (high) state (the
when logic)
  -inv_pin          : The corresponding inverting pin name for I/O output
  -inv_pin_net      : The inverting pin signal (port) name
  -vdiff            : The differential receiver threshold voltage between pin
and inv_pin
  -tdelay_typ       : The typ launch delays of the inv_pin relative to the pin
  -tdelay_min       : The min launch delays of the inv_pin relative to the pin
  -tdelay_max       : The max launch delays of the inv_pin relative to the pin
  -R_pin            : Pin package R value
  -L_pin            : Pin package L value
  -C_pin            : Pin package C value
  -components       : IBIS component name(s) list which use this pin definition
  pin               : Pin name

## ibis_define_model {}
  -model_type        : Pin model type, supported type : Input, Output, I/O, 3-state
  -when              : When condition to match waveforms' when logic
  -polarity          : "Non-Inverting" or "Inverting"
  -enable             : "Active-High" or "Active-Low"
  -vinl              : Maximum lower threshold voltage
  -vinh              : Minimum upper threshold voltage
  -vmeas             : Reference voltage for timing measurements
  -cref              : Timing specification test load capacitance
  -rref              : Timing specification test load resistance
  -vref              : Timing specification test load voltage
  -cref_diff         : Timing specification differential capacitance
  -rref_diff         : Timing specification differential resistance
  -cfix_f            : Falling waveform C_fixture
  -r_load            : Loading resistor for [Ramp] measurement
  -component         : this model is defined for these component(s) list
  modelName          : IBIS model name

## ibis_define_model_selector {}
  -model_desc         : list of model and description in pair(s)
  -component          : this model is defined for these component(s) list
  msName             : model selector name
```

Example :

```
set cells {MyLVDS}

set mslist { \
LOW_VOLT_PUPD    "Low Voltage Pull_up enabled"  \
LOW_VOLT_!PUPD   "Low Voltage Pull_down enabled" \
}

ibis_define_model_selector \
-model_desc $mslist \
-component $cells \
LVDS_OUT

## write_ibis_file {}
file      : IBIS file to write to
ldbs      : List of ldb file for typ/min/max corners
cells     : List of cell names
```

For example:

```
set cells "MyCell"
set file "Cell.ibs"
set ldbs { MyCell_typ.ldb.gz MyCell_min.ldb.gz \ MyCell_max.ldb.gz }
write_ibis_file $file $ldbs $cells
```

Truth Table Example for IBIS

The Truth Table based IBIS flow contains the following steps :

A) Use an existing truth_table template or start from scratch to compose a truth table file with IBIS specific commands to describe the cell pin behavior and IBIS information. This file should contain at least the following structure:

Example:

```
* LIB=MyLib
* CELL=buffer
* TABLE= A OEN PAD @ PAD C
  0 0 - @ 0 -
  1 0 - @ 1 -
  - 1 0 @ - 0
  - 1 1 @ - 1
  - 1 Z @ - X
* TABLE_END
* CELL_END
#
* IBIS_BEGIN
* IBIS_VER=4.2
* IBIS_FILE_REV=0.1
* IBIS_FILE=buffer.ibs
* CORE_TEMPERATURE={25,125,-40}
* CORE_VOLTAGE={1.1,0.99,1.32}
.....
.....
* IBIS_END
* CELL_END
```

B) Create a script to generate template files.

Example:

```
% cat > gen.tcl
    read_truth_table buffer.tt
    write_template -truth_table -ibis -ibis_slew 0.1 template_arc.tcl
```

A) Run Liberate

```
% liberate gen.tcl
```

B) Review the message generated by Liberate for any warning and error.

If there is no error and if all three corner data are provided, then you can find the following files been generated:

| | |
|----------------------------|---|
| char_all.csh | Shell script file to run all char works and generate IBIS file |
| common.tcl | An empty file for users to define simulator and model/netlist path and common variable settings |
| ibis_buffer_max.tcl | Char script for max voltage corner |
| ibis_buffer_min.tcl | Char script for min voltage corner |
| ibis_buffer_typ.tcl | Char script for typ voltage corner |
| igen_buffer.tcl | Script to read in ldb.gz file and generate IBIS file |
| predef.tcl | An empty file for users to define variables used in template_arc.tcl and template_ibis.tcl. |
| postdef.tcl | An empty file for users to define variables used in char_library section |
| template_arc.tcl | Template file contains define_cell{} and define_arc{} commands |
| template_ibis.tcl | Template file contains IBIS related define_ibis command. |

If the files **common.tcl**, **predef.tcl**, and **postdef.tcl** exist they will not be overwritten.

Sample run script showing loading sequence (ibis_buffer_typ.tcl):

```
...
set cells {}
set corner typ

source predef.tcl

source ${run_dir}/template_arc.tcl
source ${run_dir}/template_ibis.tcl

source common.tcl
```

```
read_spice $modelfiles
read_spice $spicefiles

source postdef.tcl

char_library
```

Electromigration Models

Electromigration Model Content

When high-density current passes through a thin metal wire, the high-energy electrons exert forces upon the atoms causing them to "migrate". This "electromigration" can drastically reduce the lifetime of an electrical circuit, by causing increased resistivity or a break in the metal wire or by creating a short circuit between adjacent lines. Electromigration can be controlled by establishing an upper boundary or maximized for the output toggle rate. This is achieved by annotating cells with electromigration characterization tables that represents the net's maximum allowable toggle rates.

In Liberty libraries, the electromigration pin-level group attribute contains the `em_max_toggle_rate` group, which specifies the maximum toggle rate, and the `related_pin` and `related_bus_pins` attributes. The `related_pin` attribute associates the electromigration group with a specific input pin. The `related_bus_pins` attribute associates the electromigration group with the input pin or pins of a specific bus group.

Electromigration Modeling Flow

Electromigration Characterization Flow Overview

This Electro-Migration (EM) flow in Liberate requires the use of Cadence Spectre APS/EMIR, an EM configuration file and spice subcircuit descriptions in DSPF format in addition to the other commands normally found in a Liberate characterization setup.

Summary of EM Characterization and Modeling Flow

Creation of an EM liberty model in Liberate can be accomplished in the following five steps:

1. Create a valid Liberate setup for all cells requiring EM models. The spice subcircuits loaded into `read_spice` are required to be in DSPF format. The setup must use Spectre-

APS and have sufficient MMSIM licenses (compatible with Spectre EMIR) available
(Note: the Spectre_Char_Opt license is not usable with EM characterization)

2. Provide the additional EM commands to the Liberate setup. At a minimum, this includes the maximum clock frequency (see [em_clock_freq](#)) and the QRC technology file (see [em_tech_file](#)).
3. Execute Liberate using the currently supported version of Spectre (see `${ALTOSHOMEROOT}/README`) or a newer version.
4. Generate a liberty library with EM data by adding the command line argument `-em` to the [write_library](#) command.
5. If needed, merge the EM data into another library that has all of the data formats, including the more advanced formats such as CCS and ECSV.

Gathering Required Data for EM Characterization and Modeling

The enhanced EM capability in Liberate was developed to take advantage of the Spectre APS/EMIR capability. As such, Liberate requires that the data needed by Spectre APS/EMIR be provided in the Liberate Tcl file. The data needed consists of a DSF format netlist file for each cell to be characterized, process models for the process/voltage/temperature (PVT) corner of interest, an EM data file or QRC tech file and a version of Spectre APS that supports EMIR (version MMSIM 13.1 ISR13 or later). In addition, the user must provide a maximum clock frequency for the library.

The interface between Liberate and Spectre APS/EMIR was designed such that the creation of the spice decks, the simulation, and parsing of the simulation output is fully automated.

Creating the Run Script

Obtain a fully validated Liberate characterization script for the library. To enable EM characterization, add the following two variables to the script prior to the [char_library](#) command. They will define a location for the EM QRC technology file required by Spectre APS EMIR and a maximum clock frequency to use for the spice level simulation. The syntax and usage is as follows:

```
# EM related clock frequency in Hertz
set_var em_clock_freq 20e6
# Specify the full path and filename of QRC tech file
set_var em_tech_file /proj/work/tech.qrc
# Request Spectre to use APS
set_var extsim_cmd_option "+aps -mt +spice"
# Tell Liberate to use Spectre
char_library -extsim spectre
```

Note: There are other variables that can be used to tune the EM characterization. For more information, see the variables prefixed by "em" in the [Liberate Parameters](#) chapter of the Liberate manual.

Creating a Liberty Library with EM Data

To create a library with EM data, add the option `-em` to the **write_library** command.

Example:

```
read_ldb MyEM.ldb
write_library -em -filename My.lib typical
```

Merging EM Data into an Existing Liberty Library

To merge a library with EM data into another library, use the **merge_library** command.

Example:

```
merge_library My.lib MyEM.lib
```

EM Characterization Example

The liberty syntax for EM data consists of input slew/output load related toggle rates for an output pin. They can either be a scalar value or a table of toggle rates based on a variety of input slews. The syntax of the Electromigration data in the Liberty format is as given in the following:

Example:

```
pin (O) {
    electromigration () {
        related_pin : i0;
        em_max_toggle_rate (em_lut_template_7x7) {
            index_1 ("slew1 ... slewN");
            index_2 ("load1 ... loadN");
            values ( \
                togglerate1 ... \
                togglerateN ... )
        }
    }
    electromigration () {
        related_pin : i1;
        em_max_toggle_rate (em_lut_template_7x7) {
            index_1 ("slew1 ... slewN");

```

```
    index_2 ("load1 ..loadN");
    values ( \
        togglerate1 ... \
        togglerateN ... )
    }
}
}
```

Notice that each input pin has a separate electromigration group. For cells with multiple input pins, these groups can also be state dependent. Liberate will automatically create an electromigration group for each delay arc characterized for each cell.

Data Table Index Determination

The size of the characterized data tables are predetermined by the user from the `define_index` command. The `define_template` command must be specified and referenced in the `define_cell` command for each cell. All arcs for the cell will have the number of slew and load indices as specified in the `define_template` command.

Liberate has 3 methods available for determining the actual data table index values, also known as **index_1** (slew index) and **index_2** (load index).

Method 1: Liberate uses the `define_template` index values.

The user specifies the exact indices desired using the `define_template` and `define_index` commands. This method gives the best runtime since Liberate will not spend any CPU time determining the index values. The `write_template` command can be used to extract the exact index values from an existing library.

Method 2: Liberate computes the index values.

The user specifies the `max_transition`, `min_transition` and `min_output_cap` values. These values can be extracted from an existing library by `write_template`. Liberate determines the `max_capacitance` by simulating the arc for each cell from each input to each output (all "when" states are observed). The input pin will be driven using the `max_transition`. Liberate will adjust the output load until the `max_transition` is reached as measured at the `measure_slew_*` thresholds. Although it may not be required, the user should provide the `min_transition` (smallest input transition) and `min_output_cap` (smallest output load) values. If these not provided, then the `min_transition` will be set by Liberate to the smallest/fastest slew as estimated for all outputs in the library when unloaded. If not provided, the `min_output_cap` will default to an estimation of the smallest input cap (the transistor port cap + wire cap) for all input pins in the library. These min transition and load values must be provided when using a packet based flow because only one cell will be evaluated at a time

resulting in tables for a cell that may not extend over the desired range for the library. Once the min/max transition and min load are determined, then the intermediate values in the table will be determined using a simple geometric series.

Method 3: The user wants to maintain a curve shape as exists in a current library, but wants to change the max_transition, min_transition and max_output_cap.

Liberate can read in the existing library (see read_library), and can output a template where the values in the define_template and define_index commands are a ratio of the index_n(max)-index_n(min) (see scale_tran_by_template and scale_load_by_template for details). The template file will have index_1/2 values that are ratios of the original index with the min value being 0 and the max value being 1. This flow can be useful if you want to increase the range of the data table. You can change the max value to a number greater than 1 or add additional values (increasing the number of values). By adding to or changing the values in the index, the data table can be made to extend beyond the design rules. For example, if the index_1 template has a max value of 1.5, then the data table will be characterized to a slew that is 1.5 times the max_transition value.

In methods 2 and 3 above, if the simulation method for determining the max_capacitance as described above should fail for some reason, then Liberate will set the max_capacitance using the method described in the documentation of the variable max_capacitance_auto_mode.

Liberate will characterize the max_capacitance in methods 2 and 3 above using a single output transition. The specific arc used will be determined by a proprietary algorithm that looks at all input to output arcs and states. Set the variable auto_index_distinct_risefall to have Liberate compute a separate max_capacitance for rise and fall transitions.

NOCHANGE Arcs

Truth Table Format

Notation recognized in Liberate Truth Table format:

-- comment line, when # appears at the beginning of line
* -- command line, when * appears at the beginning of line

If '#' appears as the first non-white character of a line, then this line is a comment and all text in this line is ignored.

Truth Table Format – Basic

The following are basic truth table commands:

ASYNC_PINS={Pin1 Pin2...}

Specifies the **define_cell async** pins that are included in the **INTERNAL_PINS**. If there is a conflict between the **ASYNC_PINS** and the truth table specified pins, Liberate will honor the information from the truth table.

For example:

* ASYNC_PINS={A B C}

BIDI_PINS={Pin1 Pin2...}

Specifies the **define_cell bidi** pins that are included in the **INTERNAL_PINS**. If there is a conflict between the **BIDI_PINS** and the truth table specified pins, Liberate will honor the information from the truth table.

For example, if pin PAD is specified as:

* BIDI_PINS={PAD}

but in the truth table, we have pin PAD defined as an output:

| | | | | | | | | |
|---|-------|---|---|----|----|-----|---|-----|
| * | TABLE | = | A | GZ | T0 | SR0 | @ | PAD |
| | | | 0 | 0 | 0 | 1 | @ | 0 |
| | | | 1 | 0 | 0 | 1 | @ | 1 |
| | | | - | 1 | 0 | 1 | @ | Z |

the pin PAD will be added to the output option of the **define_cell** command:

```
define_cell -output {PAD} ...
```

CELL=<cell_name>

Specifies the cell name. A cell table starts with this command and ends with the **TABLE_END**. All commands in this range will apply to this cell only.

CELL_END

This is the same as **TABLE_END**. If it is used at the end of the last table definition, it can provide readability by indicating the end of the cell definition.

CLOCK_PINS={Pin1 Pin2...}

Specifies the **define_cell** **clock** pins that are included in the **INTERNAL_PINS**. If there is a conflict between the **CLOCK_PINS** and the truth table specified pins, Liberate will honor the information from the truth table.

CONSTRAINT_TEM=<name>

Specifies the constraint template name to be inserted to **define_cell**.

DELAY_TEM=<name>

Specifies the delay template name to be inserted to **define_cell**

DUAL_IN= {<pin>:<dual_pin>}

DUAL_IO= {<pin>:<dual_pin>}

DUAL_OUT={<pin>:<dual_pin>}

Specifies a pair of pins that are dual-related pins. They must be of the same type (both are output or both are input) and have different (non-unate) toggle direction. (For example, if one pin transitions R -> F then the dual pin must transition F -> R.)

This is needed for IBIS template generation for dual input/output model type cells such as LVDS cells. When this argument is present, Liberate adds **-dual_pin** and **-dual_dir** options in the **define_arc** command in the generated template file.

The two dual pins must switch at the same time.

```
# Example
* DUAL_OUT={PADP:PADN}
* DUAL_IN={IPADP:IPADN}
* DUAL_IO={BPADP:BPADN}
```

ENABLE_PINS={out_pin:enablePin[,sideInputPin ..] out_pin:enablePin[...]}...

Specifies the enable pins of a tri-state output pin. Currently, Liberate only recognizes one enable pin per tri-state in/output pin (A tri-state bidi pin has to have a Z state, and it appears in both the input and output columns of the truth table). This command is used to help specify two or more enable pins. Multiple commands can be used, or a single command can have one or more lines. (In that case, use curly braces to group pin the list.) Example:

```
# if A1 -> PAD1 has enable pin HHV and EN1, and A2->PAD2 has
# enable pins HHV and EN2, then the following can be used:
* ENABLE_PINS=PAD1:EN1,HHV
* ENABLE_PINS=PAD2:EN2,HHV
      or
* ENABLE_PINS={PAD1:EN1,HHV PAD2:EN2,HHV}
```

IGNORE_ARC=< pin:related_pin [, Related_Pin ...] >

Specifies to Liberate which arc(s) (Related_Pin to Pin) should be ignored. Multiple related pins can be grouped together with ',' for the same output pin. This command is useful if there are ambiguous arcs in the truth table that are not required to be characterized with a define_arc command. Example :

```
* IGNORE_ARC={PAD:PE,IE AI:IE DI:IE}
```

INPUT_PINS={Pin1 Pin2...}

Specifies the **define_cell input** pins that are included in the **INTERNAL_PINS**. If there is a conflict between the **INPUT_PINS** and the truth table specified pins, Liberate will honor the information from the truth table.

INTERNAL_PINS={Pin1 Pin2...}

INTERNAL_PINS={Pin1:value Pin2:value...}

INTERNAL_PINS={Pin1:value Pin2:value Pin3 Pin4 Pin5:value...}

The **INTERNAL_PINS** command has been modified to set pins which are not listed in the truth table but need to be added to the **define_cell pin_list**. The user can set values for these pins. The values will be added to the **define_arc vector** option. Only one value per pin is supported at this time. Supported values are: 0, 1, and X. Any pin in the list without a value will be set to X. Example:

```
* INTERNAL_PINS={UP_N:0 UP_P:0 SCANOUT:X Y1 Y2 PI:1 PO:1}
* TABLE= A GZ T0 SR0 @ PAD
      0 0 0 1 @ 0
```

Virtuoso Liberate Reference Manual

Truth Table Format

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | @ | 1 |
| - | 1 | 0 | 1 | @ | Z |

This would be the resulting **define_arc** command:

```
define_arc \
  -vector "F001F00XXX11" \
  -when "!GZ&!T0&SR0" \
  -related_pin A \
  -pin PAD \
  Mycell
```

Note: Any pin defined as an INTERNAL_PINS in the truth table command must also be specified as either an INPUT_PINS, OUTPUT_PINS, BIDI_PINS, CLOCK_PINS, ASYNC_PINS or SLEEP_PINS. Otherwise the internal pin will be ignored and the simulation run might fail.

LEAKAGE_ENUM_PINS= {Pin1, Pin2, ...}

Defines the pins to be enumerated for generating **define_leakage** 'when' states. If not specified, then all input pins and tri-state pins are utilized. By default, up to 8 pins are accepted for enumeration. This is controlled by the parameter **max_leakage_vector**, which defaults to 256 leakage states.

For example:

```
* LEAKAGE_ENUM_PINS= {PAD data_out data_out_en st_en}
```

LIB=<library_name>

Specifies the library name. Currently this is only used when generating IBIS format output. The library name specified is used as the ID for .ldb file. Example:

In buffer.tt we have:

```
* LIB=MyLib
```

Then the final IBIS ldb files would be:

```
ibis_MyLib_typ.ldb.gz
ibis_MyLib_min.ldb.gz
ibis_MyLib_max.ldb.gz
```

These names will be used in igen_<cellname>.tcl file as well.

MAP_IGNORED_BIT=<character>

Maps a "-" to another supported character in the **define_arc** vector. The valid data is a single character of **0**, **1**, **x**, or **X**. For example, a vector of "0-0-RR" becomes "0X0XRR" in the **define_arc** command when the following is used:

* MAP_IGNORED_BIT=X

The default is "X" (map to an "X").

Note: The above mapping is applied only to input fields.

MODE = < ENUM_ALL | ENUM_ARC | ENUM_LEAKAGE | NO_ENUM | STD >

ENUM_ALL

Liberate will enumerate all pins or only those pins specified by the **LEAKAGE_ENUM_PINS** command to generate define_leakage commands in the template. It will also enumerate all pins to generate define_arc commands in the output template.

ENUM_ARC

Liberate will follow the truth table entries to generate define_leakage commands in the template. All pins will be enumerated to generate define_arc commands.

ENUM_LEAKAGE

Liberate will enumerate all pins or only those pins specified by the **LEAKAGE_ENUM_PINS** command to generate define_leakage commands in the template. The define_arc commands will follow the truth table entries.

NO_ENUM | STD

Liberate will follow the truth table entries to generate define_leakage and define_arc commands in the output template file. (Default.)

Note: **STD** and **NO_ENUM** are equivalent.

OUTPUT_PINS={Pin1 Pin2...}

Specifies the **define_cell output** pins that are included in the **INTERNAL_PINS**. If there is a conflict between the **OUTPUT_PINS** and the truth table specified pins, Liberate will honor the information from the truth table.

PINLOAD_DIR=dir

Specifies the pin load direction in pin load template, where dir is "U", "D" and "B" (up, down, both). If a direction is not specified, no load will be added. (Default)

PINPAIR=<pin1:pin2,...>

Identifies the differential pin pairs. Multiple pin pairs can be specified when separated by a comma (",").

POWER_TEM=name

Specifies the power template name to be inserted to **define_cell**.

PULLUP=<pins>

PULLDOWN=<pins>

Specifies the pins that can only pull-up or pull-down. For the pull-up/down pins, only the pin-caps are characterized. Note: These commands are not supported at this time and will be supported in a future release.

PULLUP_RES=<resistance>

PULLDOWN_RES=<resistance>

Specifies the pull-up/down resistance (float, in Ohms) for the pull-up/down pins. If this is not specified, the resistance will not be output.

PULLUP_VOLT=<voltage>

Specifies the pull-up voltage (float, in Volts)

RELATED_PINS={Pin:Related_Pin [,Related_Pin ...] }

Specifies to Liberate which related pin(s) are associated with a given output pin. Multiple related pins can be grouped together with ',' for the same output pin. This command is useful if there are ambiguous arc's in the truth table for which Liberate might fail to find a related pin for an output pin.

Example:

```
* RELATED_PINS={PAD:DO AI:DO DI:DO}
```

SERIES_RES=<resistance>

Specifies the series connected resistance (float, in Ohms) to add to the bi-directional pin.

SI_IM_TEM=name

Specifies the SI Immunity template name to be inserted to define_cell

TABLE=Inputs@Outputs

<state_values>

TABLE_END

Specifies the actual truth table. The Input and Output pin arcs are specified as input pins, a separator, then the output pins. The pin names and the values must be separated by whitespace. The values are specified one vector at a time. The table continues until the command **TABLE_END** is encountered. Bi-directional pins are specified as both inputs and outputs.

Example:

```
* TABLE OEN I PAD @ PAD C
      1   -   @   1   1
* TABLE_END
```

TABLE_SEP=<value>

Specifies the default table separator value.

(Default: '@'). Specify this command at the very beginning of the input file (not inside any *CELL) to specify separator globally.

WRITE_HIDDEN_POWER = <0|1>

Controls the output of hidden power arcs into the template. Hidden arcs are arcs where an input toggles, but no output toggles. Default is 0 (do not write hidden power.)

WRITE_WHEN = <0|1>

Adds the '-when' option to the **define_arc** commands. When in IBIS mode, the '-when' is always added. This option can be applied globally when used before the CELL command, or locally for individual cells when used after the CELL command. Default = 0.

Note: If the '-when' is present in a define_arc command, it will be carried into the lib.

Example 1:

```
* CELL= MUXI31
* MODE=ENUM
* WRITE_WHEN=1
* LEAKAGE_ENUM_PINS={S0 S1 S2}
* TABLE= I0 I1 I2 S0 S1 S2 @ X
      1   x   x   1   0   0   @ 0
      0   x   x   1   0   0   @ 1
* TABLE_END
```

Example 2:

Virtuoso Liberate Reference Manual

Truth Table Format

In file gen.tcl :

```
read_truth_table mux.tt
write_template -truth_table template_mux
```

In file mux.tt :

```
* CELL= MUXI31
* MODE= ENUM
* WRITE_WHEN=1
* LEAKAGE_ENUM_PINS={S0 S1 S2}
* TABLE= I0 I1 I2 S0 S1 S2 @ X
  1 x x 1 0 0 @ 0
  0 x x 1 0 0 @ 1
  x 1 x 0 1 0 @ 0
  x 0 x 0 1 0 @ 1
  x x 1 0 0 1 @ 0
  x x 0 0 0 1 @ 1
* TABLE_END
* CELL_END
```

The generated define_leakage and define_arc commands in template_mux.tcl file :

```
# define_leakage for 3 pins :
# S0 S1 S2
define_leakage -when "S0 S1 S2" {MUXI31}
define_leakage -when "!S0 S1 S2" {MUXI31}
define_leakage -when "S0 !S1 S2" {MUXI31}
define_leakage -when "!S0 !S1 S2" {MUXI31}
define_leakage -when "S0 S1 !S2" {MUXI31}
define_leakage -when "!S0 S1 !S2" {MUXI31}
define_leakage -when "S0 !S1 !S2" {MUXI31}
define_leakage -when "!S0 !S1 !S2" {MUXI31}

# Generate define_arc by enumerating all possible vector combination.
# Pins : I0 I1 I2 S0 S1 S2 X
define_arc -vector "FXX100R" -when "S0&!S1&!S2" -related_pin {I0} -pin X
MUXI31
define_arc -vector "10XFR0R" -when "I0&!I1&!S2" -related_pin {S0 S1} -pin X
MUXI31
define_arc -vector "1X0F0RR" -when "I0&!I2&!S1" -related_pin {S0 S2} -pin X
MUXI31
define_arc -vector "RXX100F" -when "S0&!S1&!S2" -related_pin {I0} -pin X
MUXI31
define_arc -vector "01XFR0F" -when "!I0&I1&!S2" -related_pin {S0 S1} -pin X
MUXI31
define_arc -vector "0X1F0RF" -when "!I0&I2&!S1" -related_pin {S0 S2} -pin X
MUXI31
define_arc -vector "01XRF0R" -when "!I0&I1&!S2" -related_pin {S0 S1} -pin X
MUXI31
define_arc -vector "XFX010R" -when "!S0&S1&!S2" -related_pin {I1} -pin X
MUXI31
define_arc -vector "X100FRR" -when "I1&!I2&!S0" -related_pin {S1 S2} -pin X
MUXI31
define_arc -vector "10XRF0F" -when "I0&!I1&!S2" -related_pin {S0 S1} -pin X
MUXI31
define_arc -vector "XRX010F" -when "!S0&S1&!S2" -related_pin {I1} -pin X
MUXI31
define_arc -vector "X010FRF" -when "!I1&I2&!S0" -related_pin {S1 S2} -pin X
MUXI31
define_arc -vector "0X1R0FR" -when "!I0&I2&!S1" -related_pin {S0 S2} -pin X
MUXI31
```

```
define_arc -vector "X010RFR" -when "!I1&I2&!S0" -related_pin {S1 S2} -pin X
MUXI31
define_arc -vector "XXF001R" -when "!S0&!S1&S2" -related_pin {I2} -pin X
MUXI31
define_arc -vector "1X0R0FF" -when "I0&!I2&!S1" -related_pin {S0 S2} -pin X
MUXI31
define_arc -vector "X100RFF" -when "I1&!I2&!S0" -related_pin {S1 S2} -pin X
MUXI31
define_arc -vector "XXR001F" -when "!S0&!S1&S2" -related_pin {I2} -pin X
MUXI31
```

Liberate Truth Table Format

Input Field

```
'0' = Low
'L' = low (higher or equal to 0. Currently is treated as '0')
'1' = High
'H' = high (lower or equal to 1. Currently is treated as '1')
'X' = Don't Care. Can be '0' or '1'
'-' = ignored
```

Output Field

```
'0' = Low
'L' = Low
'1' = High
'H' = High
'X' = unknown
'Z' = High Impedance
'-' = Not Specified
```

For a three-state-enabled pin (PAD), there should be at least one 'Z' state specified in the output field. Since the PAD pin appears at both the input and the output field, for any logic value row (vector), there should be a '-' at one of the field. This is required since a bidirectional signal should not be both input and output in the same vector.

Example

When the enable pin is set (PAD is output enabled), the following are valid (PAD @ PAD) logic values:

```
PAD @ PAD
=====
- @ 0    <-- enable
- @ 1    <-- enable
```

When the enable pin is unset (PAD is input), the following are valid (PAD @ PAD) logic values:

```
PAD @ PAD
=====
0 @ -    <-- disable
1 @ -    <-- disable
Z @ -    <-- disable (Liberate converts to "- @ Z")
- @ X    <-- disable
X @ -    <-- disable
```

Note: For any 'Z' found at the input of a bi-directional pin, if the output is 'L', Liberate changes the output 'L' to 'Z'.

Template Generated

The following Liberate commands are written to the generated template file: define_cell, define_pin_load, define_leakage and define_arc.

The define_pin_load command is added when there is a three-state enable pin in the truth table and at least one of PULLUP_RES, PULLDOWN_RES, PULLUP_VOLT, or SERIES_RES commands are specified.

See the **read_truth_table** command for an example.

Truth Table Flow

Use the Tcl command **read_truth_table** to read in truth table file(s). All loaded cell information is attached to the library specified in this command.

Use the Tcl command **write_template** with the **-truth_table** option to write out a template using the truth table data that was read in with the **read_truth_table** command.

After the template has been generated, it should be inspected and modified as required to suit the circuit.

Example:

To read in a truth table file (io_cells.tab), which contains several truth tables, and generate a Liberate template with 5-by-6 delay and power indices.

```
read_truth_table io_cells.tab
write_template -auto_index \
  -index_delay 5x6 \
  -truth_table template_io_cells
```

Truth Table Format – IBIS

Important

This section covers information related to the old flow of IBIS generation that has been included here only for backward compatibility. Instead, Cadence recommends use of Sigrity™ Transistor-to-Behavioral (T2B) model conversion product for IBIS generation. For information about IBIS generation using Sigrity T2B, contact your Cadence Customer Support representative.

Notes on command syntax

Some data lines might be too long to fit in one line. In this case, quotes can be used to continue the command onto multiple lines. For example:

```
* DISCLAIMER="This software contains information  
confidential and proprietary  
to ABC Corporation."
```

Use quotes if there are spaces in the data:

```
* MANUFACTURER="Cadence Design Systems, Inc."
```

For data fields such as {25,-40,125}, ***don't put a space*** in between the values. *Example of incorrect formatting:* {25, -40, 125}.

A space (or ;) is to separate data fields such as {P1:PAD P2:COUT}.

A global command, like CORE_VOLTAGE, SOURCES, and IBIS_FILE can appear only once per table.

Some commands to set IBIS pin data may be utilized many times in a section.

Example:

```
* SET_POWER_NET_VDD={VDDS:1.8,1.62,1.98 BIASFET:1.8,1.62,1.98}  
* SET_POWER_NET_VDD={BIAS1:1.25,1.25,1.25 BIAS2:1.25,1.25,1.25}  
  
* MODEL_WHEN={Buffer_Output:!LOPWRA&!LOPWRB}  
* MODEL_WHEN={Buffer_Output:LOPWRA&!LOPWRB }  
* MODEL_WHEN={Buffer_Output:!LOPWRA&LOPWRB }  
* MODEL_WHEN={Buffer_Output:LOPWRA&LOPWRB }
```

For IBIS Truth Table commands, a PIN refers to the IBIS pin name and a NET refers to the Spice netlist net name.

For example, the user may have these PIN/NET/MODEL in the file:

```
# IBIS_PIN  PIN_NET MODEL  
#  P1        PAD    Buffer_Output  
#  P2        PTST   Buffer_TriState  
#  PW1       VDDS   POWER
```

Virtuoso Liberate Reference Manual

Truth Table Format

```
#  GW1      VSSS  GND
```

Liberate uses the following information to link them together:

- * PIN_MODEL_NAME={P1:Buffer_Output}
- * PIN_NET_NAME={P1:PAD}
- * MODEL_TYPE={Buffer_Output:Output}
- * PIN_MODEL_NAME={P2:Buffer_TriState}
- * PIN_NET_NAME={P2:PTST}
- * MODEL_TYPE={Buffer_TriState:3-state}

Example of dual related output pins (PAD and PADN):

- * PIN_NET_NAME={P1:PAD}
- * PIN_INV_PIN={P1:P2}
- * PIN_INV_PIN_NET={P2:PADN}

IBIS Truth Table Commands

C_PKG={typ,min,max}

L_PKG={typ,min,max}

R_PKG={typ,min,max}

Defines component pins default packaging values, C/L/R in Farads/Henrys/Ohms.

COPYRIGHT="<string>"

Specifies the copyright string to be included in the IBIS output file.

CORE_TEMPERATURE={typ[,min[,max]]}}

Specifies the 3 PVT temperatures that the IBIS will be characterized at. The value is in a list: {typ min max}.

CORE_VOLTAGE={typ[,min[,max]]}}

Specifies the 3 PVT voltages that the IBIS will be characterized at.

DISCLAIMER="<string>"

Specifies the disclaimer string to be included in the IBIS output file.

IBIS_BEGIN
IBIS_END

Specifies the beginning and ending of an IBIS section.

IBIS_FILE=<filename>

Specifies the IBIS output file name. The file name should be in all lowercase. If not, Liberate will automatically convert it to lower case to fulfill the IBIS requirement. If the suffix is not ".ibs", Liberate will append ".ibs" to the end.

IBIS_FILE_REV=<revision>

Specifies the output file revision.

IBIS_VER=<version>
(or *IBIS_VERSION*)

Specifies the IBIS version in the output IBIS file. Default is "4.2". There is a limitation of up to 100 data points in some sections in a version 3.X file. This limit is changed to 1000 data points in version 4.X. (Also accepted: *IBIS_VERSION*)

MANUFACTURER=<string>

Specifies a manufacturer string to be included in the IBIS output file.

MODEL_C_REF/MODEL_R_REF/MODEL_V_REF={model_name:pin_name ...}

Defines the timing specification test load (Cref, Rref, Vref) in the [Model] section. Please see the IBIS specification for more details.

Example:

```
* R_PKG={0,0,0}
* PIN_R_PIN={Out:200m PAD:210m In:50m}
* MODEL_R_REF={Buffer_Output:50}
```

MODEL_ENABLE_PIN_NET={model_name:pin_name ...}

Clarifies the Enable attribute of an IBIS [Model] section. If not provided, Liberate will attempt to determine the enable pin from the truth table, provided there is only one enable pin/net. Use this command to set the enable pin netname for a IBIS model. For example, assume the

cell TriBuffer has an enable pin, a pad pin and one output pwrndn pin. The pin used to enable/disable the output pin PAD of the cell TriBuffer is OEN (OEN -> PAD). In order to help Liberate to choose the OEN pin (instead of PWRDN) is the enable pin for PAD, the user can set this command:

```
* MODEL_ENABLE_PIN_NET={TriBuffer:PAD}
```

MODEL_RELATED_PIN_NET={model_name:pin_name ...}

Specifies the model pin to net mapping. Use this command to set the related pin netname for an IBIS model. For example, assume a cell Buffer_Output has multiple input pins and control pins, and the main input netname to drive the output DOUT of the model Buffer_Output is A (A -> DOUT). In order to tell Liberate that control pins LOPWA and LOPWB are not directly related pins for DOUT, use the following command:

```
* MODEL_RELATED_PIN_NET={Buffer_Output:A}
```

This command is also used to determine the Polarity attribute of an IBIS [Model] section. Liberate determines the polarity direction (Inverting / Non-Inverting) from the truth table and puts it into the IBIS model Polarity attribute.

MODEL_TYPE={model_name:type}

Specifies the type for each model.

Fully supported model types:

```
Input
Output
I/O
3-state
Open_drain
Open_sink
Open_source
I/O_open_drain
I/O_open_sink
I/O_open_source
POWER
GND
```

Note: The truth table feature does *not* support **open_drain/open_sink** and **open_source** types. Users should use the standard (I/O, Output) model first, then manually modify the generated template_arc.tcl and template_ibis.tcl files to remove the pull up/rising arcs that are not needed.

Example:

```
* MODEL_TYPE={Buff_I:Input Buff_O:Output Buff_PAD:3-state}
```

MODEL_VINH={pin_name:Vinh ... }

Specifies the input logic high DC voltage. The Vinh can accept one, two, or three entries with the voltage values mapping to: "typ", "min", and "max". The "typ" value is listed in [Model] section. If more than one voltage is specified, the typ/min/max values are listed in the [Model Spec] section when there is a difference between them.

Example 1:

```
* MODEL_VINH={Buff_PAD:2.0 Buff_In:2.0}
```

Example 2:

```
* MODEL_VINH={Buffer_In:2.31,2.08,2.54}
```

MODEL_VINL={pin_name:Vinl ... }

Specifies the input logic low DC voltage. The Vinl can accept one, two, or three entries with the voltage values mapping to: "typ", "min", and "max". The "typ" value is listed in the [Model] section.

If more than one voltage is specified, the typ/min/max values are listed in the [Model Spec] section when there is a difference between them.

Example 1:

```
* MODEL_VINL={Buffer_PAD:0.8 Buffer_In:0.8}
```

Exmaple 2: input buffer :

```
* MODEL_VINL={Buffer_In:1,0.89,1.09}  
* MODEL_VINH={Buffer_In:2.31,2.08,2.54}
```

Exmaple 2: input buffer :

Liberate command generated from truth table

```
ibis_define_model \
  -model_type Input \
  -vinl {1.0 0.89 1.09} \
  -vinh {2.31 2.08 2.54} \
  -when "(!PI&!PUPDSEL)" \
  -component $ibisCellName \
  P1_!PI_!PUPDSEL

( input buffer IBIS )
[Model] P1_!PI_!PUPDSEL
Model_type Input
Vinl = 1.0
Vinh = 2.31

[Model Spec]
Vinl 1.0 0.89 1.09
Vinh 2.31 2.08 2.54
```

MODEL_VMEAS={pin_name:Vref ... }

Specifies the reference voltage for timing measurements. The Vref can accept one, two, or three entries with the voltage values mapping to: "typ", "min", and "max". The typical value is listed in the [Model] section.

If more than one entry is given, the typ/min/max values are listed in the [Model Spec] section when there is a difference between them.

Example 1:

```
* MODEL_VMEAS={Buff_PAD:1.65 Buff_O:1.65}
```

Example 2: output buffer

```
* MODEL_VMEAS={Buffer_Out:1.21,1.195,1.215}
```

Liberate command, generated from truth table

```
ibis_define_model \
    -model_type Output \
    -polarity Non-Inverting \
    -r_load 50 \
    -vmeas {1.21 1.195 1.215} \
    -rref 101M \
    -cref 15pF \
    -vref 0.0 \
    -when "(!LPSEL&!LOPWRA&!LOPWRB)" \
    -component $ibisCellName \
    P1_!LPSEL_!LOPWRA_!LOPWRB

( output buffer IBIS )
[Model] P1_!LPSEL_!LOPWRA_!LOPWRB
Model_type Output
Polarity Non-Inverting
Vmeas = 1.21

[Model Spec]
Vmeas 1.21 1.195 1.215
```

MODEL_WHEN={model_name:<resistance>,...}

Specifies the *when* condition(s) used in each model. If multiple MODEL_WHEN= commands exists for a model, then Liberate will group them together as one [Model Selector], and create a name for each when condition. The user may change the name and add description of the generated template_arc.tcl file. Example:

```
* MODEL_WHEN={Buff_O: PU&PD, PU&!PD, !PU&PD, !PU&!PD}
```

NOTES=<string>

Specifies a note string to be included in the IBIS output file.

PIN_C_FIXTURE={pin_name:capacitance}

Specifies the C_fixture (test fixture) for an IBIS pin.

PIN_L_FIXTURE={pin_name:inductance}

Specifies the L_fixture (test fixture) for an IBIS pin.

C_fixture and L_fixture can be used to produce waveforms which describe the typical test case setups for reference. However, they are generally not recommended.

(Note, there is no PIN_R_FIXTURE command since PULLUP_RES/PULLDN_RES is the R_Fixture.)

PIN_C_PIN = {pin_name:<capacitance> ...}

PIN_L_PIN = {pin_name:<inductance> ...}

PIN_R_PIN = {pin_name:<resistance> ...}

Defines the package R/L/C value in that pin, which overwrites the default value listed in the [Package] section. Example:

```
* PIN_C_PIN={P3:2.9pF P2:3.4pF P1:2.0pF}
* PIN_L_PIN={P3:3.0nH P2:5.0nH P1:NA}
* PIN_R_PIN={P3:200m P2:209m P1:198m}
```

PIN_INV_PIN={pin_name:net_name ...}

Specifies the corresponding inverting pin (and net) name for I/O output. Example:

```
* PIN_INV_PIN={P2:PADN}
```

PIN_MODEL_NAME={pin_name:model_name ...}

Specifies the pin name to model name mapping. Note: POWER, GND and NC are reserved model names that will be set to model_type. Example:

```
* PIN_MODEL_NAME={VD33:POWER VSSPST:GND}
```

PIN_NET_NAME={pin_name:net_name ...}

Specifies the IBIS pin name to netlist net (signal) name mapping. Example:

```
* PIN_NET_NAME={P3:DOUT P2:PAD P1:DIN}
```

PIN_TDELAY={pin_name:<time> ...}

Contains launch delays (in seconds) of the non-inverting pins relative to the inverting pins.

Example:

```
* PIN_TDELAY={P2:1.0ns,NA,NA}
```

PIN_VDIFF={pin_name:<voltage> ...}

Specifies the differential receiver threshold voltage between the inverting and non-inverting pins for Input or I/O model types. Example:

```
* PIN_VDIFF={P2:200m}
```

PULLDN_RES={pin_name:typ,min,max ...}

(or *PULLDOWN_RES*)

Specifies the pulldown R_load in Ohms for each pin.

PULLDN_VOLT={pin_name:typ,min,max ...}

(or *PULLDN_VOLTAGE* or *PULLDOWN_VOLT* or *PULLDOWN_VOLTAGE*)

Specifies the pulldown termination corner voltages for each pin.

PULLUP_RES={pin_name:typ,min,max ...}

Specifies the pullup R_load in Ohms for each pin. This should be consistent with the R_fixture.

PULLUP_VOLT={pin_name:typ,min,max ...}

(or *PULLUP_VOLTAGE*)

Specifies the pullup termination corner voltages for each pin.

RUN_DIR={/run_dir_typ[/run_dir_min[/run_dir_max]]}

Specifies between 1 to 3 directories where Liberate can find and execute Tcl commands associated with the corners: min, typ, and max. This command must be set between IBIS_BEGIN & IBIS_END statements.

Important: You must set the RUN_DIR when using a job scheduler such as LSF or Sun Grid Engine (SGE). If this is not set correctly, the path to the Tcl scripts sourced within _ibis_<cell>_<corner>.tcl will not be found.

For usage, reference the following examples below:

If RUN_DIR is set to {/run_dir_typ,/run_dir_min,/run_dir_max} then the generated scripts for the 3 corners will have their run directories set as follows:

```
_ibis_<cell>_typ.tcl  use:  /run_dir_typ
_ibis_<cell>_min.tcl  use:  /run_dir_min
_ibis_<cell>_max.tcl  use:  /run_dir_max
```

If only 2 RUN_DIR's are set {/run_dir_typ,/run_dir_min} but three corners are specified the run directories will be set as follows:

```
_ibis_<cell>_typ.tcl  use:  /run_dir_typ
_ibis_<cell>_min.tcl  use:  /run_dir_min      # These 2
_ibis_<cell>_max.tcl  use:  /run_dir_min      # repeated
```

If only 1 RUN_DIR is set {/run_dir_typ} but three corners are specified the run directory will be set as follows:

```
_ibis_<cell>_typ.tcl  use:  /run_dir_typ      #
_ibis_<cell>_min.tcl  use:  /run_dir_typ      # These 3 repeated
_ibis_<cell>_max.tcl  use:  /run_dir_typ      #
```

If RUN_DIR is not specified, then a `set run_dir [exec pwd]` is added in the run script ibis_<cell>_<corner>.tcl

SET_CONTROL_PARAM={<variables>}

Recognizes two variables: **no_header** and **append_file**. They accept the values 1 (or true) and 0 (or false). **append_file** has no effect if **no_header** is not set to 1 (or true). This is used to create an IBIS file with multiple [Model] sections for a single device. (See [Creating an IBIS file with multiple \[Model\] sections for a single device](#).)

Example:

```
* SET_CONTROL_PARAM={no_header:1 append_file:1}
```

This will create the following line in igen_<CELL>.tcl :

```
write_ibis_file -no_header -append_file $file $ldbs $cell
```

SET_NET_GND={net_name:typ[min,max]}

Specifies the net name of a pin and the ground voltage levels applied to each corner.

SET_NET_VDD={net_name:typ[,min[,max]]}

Specifies the net name of a pin and the supply voltage levels applied to each corner.

SET_POWER_NET_GND={supply_net_name:typ[,min[,max]]}

Specifies the name of a GND power source net name and the voltages applied to each corner.

SET_POWER_NET_VDD={supply_net_name:typ[,min[,max]]}

Specifies the name of a VDD power source net name and the voltages applied to each corner.

SET_TCL_VARIABLE={Tcl_variable_name:typ_val[,min_val[,max_val]]}

This command is to set harness (subckt cell name) variables which is to be used in **template_arc.tcl**.

Example:

```
* SET_TCL_VARIABLE={TERMRES:termres_typ,termres_min,termres_max}
* HARNESS=$TERMRES
```

Then in the generated (typical corner) char script file there will be the following:

```
set TERMRES termres_typ
```

and in the generated template_arc.tcl, there will be

```
define_cell \
    -input {A LOPWRA LOPWRB} \
    -output {PAD} \
    -pinlist {A LOPWRA LOPWRB PAD} \
    -harness $TERMRES \
    ...

```

The user should have a harness spice netlist file available.

Example: In the termres.sp file :

```
.subckt termres_typ pad_altos_out padn_altos_out
R1 pad_altos_out mid 50
R2 padn_altos_out mid 50
Vmid mid 0 1.21
.ends
```

In the predef.tcl file, there should be the following line: (note, there are 3 harness cell termres_typ/termres_min/termres_max for the three corners)

```
if {$corner == "typ"} {
    set TERMRES "termres_typ"
} elseif {$corner == "min"} {
```

Virtuoso Liberate Reference Manual

Truth Table Format

```
    set TERMRES "termres_min"
} elseif {$corner == "max"} {
    set TERMRES "termres_max"
}
lappend spicefiles "termres.sp"
```

By doing so, the `-harness` argument in `define_cell` command in `template_arc.tcl` file can be functional.

```
define_cell \
    -input {A LOPWRA LOPWRB LPSEL PWRDN} \
    -output {PAD PADN} \
    -pinlist {A LOPWRA LOPWRB LPSEL PWRDN PAD PADN} \
    -harness $TERMRES \
    -delay delay_template_1x1 \
    -power power_template_1x1 \
MyCell
```

Note, harness is not needed for liberate to characterize IBIS differential IO cells. This example is only for demo purpose.

NOTE: The rising dV is measure with an `R_load` (default 50 Ohm) termination to ground and the falling is measured with an `R_load` (default 50 Ohm) termination to `Vdd`. The following are voltages for pullup/down measurements and should be consistent with `V_fixture`.

Template generated :

The following Liberate commands are written to the generated `template_arc.tcl` file:

```
define_template
define_cell
define_pin_load
define_leakage
define_arc.
```

The following Liberate commands are written to the generated `template_IBIS.tcl` file:

```
ibis_define_header
ibis_define_waveform_template
ibis_define_component
ibis_define_model
ibis_define_pin,
```

The following commands take IBIS model name as leading variable:

```
MODEL_C_REF
MODEL_R_REF
MODEL_V_REF
MODEL_TYPE
MODEL RELATED PIN NET
MODEL_ENABLE_PIN_NET
MODEL_WHEN
MODEL_VINL
MODEL_VINH
MODEL_VMEAS
```

The following commands take IBIS pin name as leading variable:

```
PULLUP_VOLT
```

```
PULLDN_VOLT
PULLUP_RES
PULLDN_RES
PIN_MODEL_NAME
PIN_NET_NAME
PIN_INV_PIN
PIN_INV_PIN_NET
PIN_R_PIN
PIN_L_PIN
PIN_C_PIN
PIN_L_FIXTURE
PIN_C_FIXTURE
PIN_VDIFF
PIN_TDELAY
```

The following commands take SPICE net name as leading variable:

```
SET_POWER_NET_VDD
SET_POWER_NET_GND
SET_NET_VDD
SET_NET_GND
```

Creating an IBIS file with multiple [Model] sections for a single device

The user may create an IBIS file from different sets of IBIS ldb files by using the options **-no_header**, and **-append_file**. The example below shows how these options are used in a series of three Tcl script to add [Model] data to a single IBIS file.

1. Prepare IBIS ldb files and 3 Tcl scripts

script_01.tcl

```
set cell "IOBUF12S"
set file "iobuf12s.ibs"
set ldbs { ibis_IOBUF12S_typ.01.ldb.gz ibis_IOBUF12S_min.01.ldb.gz \
           ibis_IOBUF12S_max.01.ldb.gz }
write_ibis_file $file $ldbs $cell
```

script_02.tcl

```
set cell "IOBUF12S"
set file "iobuf12s.ibs"
set ldbs { ibis_IOBUF12S_typ.02.ldb.gz ibis_IOBUF12S_min.02.ldb.gz \
           ibis_IOBUF12S_max.02.ldb.gz }
write_ibis_file -no_header -append_file $file $ldbs $cell
```

script_03.tcl

```
set cell "IOBUF12S"
set file "iobuf12s.ibs"
set ldbs { ibis_IOBUF12S_typ.03.ldb.gz ibis_IOBUF12S_min.03.ldb.gz \
           ibis_IOBUF12S_max.03.ldb.gz }
write_ibis_file -no_header -append_file $file $ldbs $cell
write_ibis_end $file
```

2. Run scripts sequentially

```
liberate script_01.tcl
liberate script_02.tcl
```

Virtuoso Liberate Reference Manual

Truth Table Format

liberate script_03.tcl

3. Edit IBIS file to add [Model Selector] section manually and validate file integrity.

Virtuoso Liberate Reference Manual
Truth Table Format

Constraint-related Delay and Clock Path Measurement

It is possible to have Liberate measure the clock and data paths for a given constraint arc. These path measurements can be reported in the ldb as described below, or can be used to compute the setup/hold time in lieu of the standard bisection search algorithm (see **define_arc -metric path_delta**). The measurements must be completely specified by the user. A path delay can be measured from a **define_arc pin to pin_probe** and from **related_pin to related_probe**. The following changes were made to support this functionality:

Liberate will perform an additional "final simulation" (last iteration + 1) with the requested clock and data path measurements possibly with added margin (back-off).

If **set_constraint -margin** is applied before **char_library**, the specified margin will be used as a back-off adjustment for the final simulation.

The **define_arc** command has the following options: **pin_probe**, **pin_probe_dir**, **pin_probe_threshold**, and **related_probe_threshold**. The threshold variables accept values between 0 and 1 and default to a value of 0.5.

The following attributes will be written into the ldb: **altos_pin_probe_rise**, **altos_pin_probe_fall**, **altos_related_probe_rise**, and **altos_related_probe_fall**.

Example:

```
define_arc \
  -type setup \
  -metric path_delta
  -pin {D} -pin_dir -R \
  -related_pin {CK} -related_pin_dir R \
  -pin_probe_threshold 0.4 -related_probe_threshold 0.4 \
  -pin_probe {INT_1} \
  -related_probe {INT_2} \
DFFX1
```

The ldb will have:

```
timing () {
  altos_pin_probe_fall : "0.121926 0.125029 0.126677 0.131257 0.124245
  0.13021 0.133014 0.133937 0.134979 0.12743 0.141288 0.143912 0.14479 0.142544
  0.134338 0.222769 0.225745 0.227327 0.223372 0.214846 0.299318 0.301962 0.30321
  0.295557 0.287447 ";
```

Virtuoso Liberate Reference Manual

Constraint-related Delay and Clock Path Measurement

```
    altos_pin_probe_rise : "0.0133118 0.0131942 0.0135056 0.0166397
0.0189255 0.00918533 0.00933199 0.00917782 0.0123258 0.0152624 0.00113986
0.00152897 0.00089008 0.00424648 0.00767286 -0.088543 -0.0885724 -0.0886433 -
0.0878743 -0.0834282 -0.211588 -0.211584 -0.211563 -0.211518 -0.208395 ";
    altos_related_probe_fall : "0.0451655 0.05435 0.059468 0.0610231
0.0395994 0.044886 0.0541113 0.0594423 0.0620969 0.0419871 0.0448538 0.0541429
0.0596396 0.0636134 0.0446303 0.0439837 0.0530142 0.0582266 0.061947 0.0438122
0.0437301 0.0526872 0.0577312 0.0607178 0.042053 ";
    altos_related_probe_rise : "0.0419939 0.0505755 0.0553957 0.0564752
0.036051 0.0422824 0.0509746 0.0557763 0.0572595 0.0367968 0.0422869 0.0510556
0.0560625 0.0580703 0.0379491 0.0427997 0.0520553 0.0574491 0.0617163
0.0433321 0.0426693 0.0519382 0.0572984 0.0616856 0.0444859 ";
    related_pin : "CK";
    timing_type : setup_rising;
    rise_constraint (constraint_template_5x5) {
        index_1 ("0.021898, 0.0788762, 0.135854, 0.591678, 1.16146");
        index_2 ("0.021898, 0.0788762, 0.135854, 0.591678, 1.16146");
        values ( \
            "0.0361599, 0.0288945, 0.0275717, 0.0445843, 0.0765158", \
            "0.0402818, 0.0325445, 0.0305551, 0.0464705, 0.0783839", \
            "0.0438295, 0.0357188, 0.0333824, 0.0494603, 0.0821265", \
            "0.0489605, 0.0411358, 0.0384973, 0.0587414, 0.0995835", \
            "0.0363275, 0.0290285, 0.027267, 0.0504022, 0.0963014" \
        );
    }
    fall_constraint (constraint_template_5x5) {
        index_1 ("0.021898, 0.0788762, 0.135854, 0.591678, 1.16146");
        index_2 ("0.021898, 0.0788762, 0.135854, 0.591678, 1.16146");
        values ( \
            "0.0269794, 0.0171525, 0.00999429, -0.000490503, 0.0155645", \
            "0.0338654, 0.0240763, 0.016751, 0.00486099, 0.0201764", \
            "0.0438901, 0.0341262, 0.0267148, 0.0132729, 0.0277857", \
            "0.118124, 0.108111, 0.100687, 0.082254, 0.0935043", \
            "0.192684, 0.182188, 0.174187, 0.149656, 0.156948" \
        );
    }
}
```

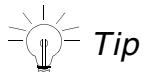
External Simulator Options and Settings

Spectre

Spectre reads options from various locations including the `.options` statement, the `deck header`, the `.tran` statement, and the command line. (See the [Virtuoso Spectre Circuit Simulator Reference](#), "Command Options section for a full list of options.")

Passing options to Spectre requires setting the appropriate Liberate variable so options appear in the expected location:

| | Spice compatible | Native Spectre options | Transient control | Command line |
|---------------------------|---|---------------------------------|---|--|
| Liberate variables | <code>extsim_option</code> <code>extsim_leakage_option</code> | <code>extsim_deck_header</code> | <code>extsim_tran_append</code> | <code>extsim_cmd_option</code> |
| Spectre options | <code>gmin</code> <code>method</code> <code>rabsshort</code> <code>save</code> | <code>reltol</code> | <code>errpreset</code> <code>Iteratio</code> | <code>-64</code> <code>+liberate</code> <code>+aps</code> <code>-cmiconfig</code> <code>+lorder</code> <code>+lqt</code> <code>+modellib</code> <code>+mt</code> <code>-mt</code> <code>+spice</code> |



Tip

The following is a series of examples showing how various options can be set to support different configurations or achieve different results.

General Spectre Settings for Accuracy and Performance

These represent a baseline set of options for Spectre when used with Liberate. The settings are accurate and recommended for all geometries.

Virtuoso Liberate Reference Manual

External Simulator Options and Settings

```
set_var extsim_cmd_option      "+spice +lorder MMSIM:PRODUCT"
set_var extsim_deck_header     "simulator \
                                lang=spectre\nOpt1 options reltol=1e-4\nsimulator \
                                \
                                lang=spice"
set_var extsim_leakage_option  "method=gear gmin=1e-15 redefinedparams=ignore
rabsshort=1m"
set_var extsim_option          "method=gear gmin=1e-15 redefinedparams=ignore
rabsshort=1m"
set_var extsim_tran_append     "lteratio=10"
set_var extsim_reuse_ic        3
```

Note: Liberate Spectre Aging flow was supported by Liberate 13.1 ISR1 along with MMSIM12.1 ISR16 or later versions. It is recommended to use the latest production version of Liberate along with the currently supported version of MMSIM (see `${ALTOSHOME} / README`).

Spectre Kernel Interface (SKI)

~~To enable the high-performance private API to Spectre (SKI), add these to baseline settings. The recommended version of MMSIM is 13.1 ISR5 or newer, in order to gain the full benefit of SKI. To enable SKI (the high-performance private API to Spectre), use the following settings. The recommended version of MMSIM can be found in the `${ALTOSHOME} / README` file.~~

Note: These settings may change over time.

```
# Enable SKI
set_var ski_enable 1
set_var ski_alter_mode 3
# Clean up semaphores
set_var ski_clean_mode 2
# Enable "spectre +modellib" flow. Use extsim_model_include with a comment as the
# fist line.
set_var ski_enable_modellib 1
# Set SKI mdlthreshold to exact for best accuracy.
set_var ski_mdlthreshold_exact 1
# Set SKI power subtraction to match Spectre flow.
set_var ski_power_subtract_output_load_match_extsim 1
# Use the original name in the simulation.
set_var extsim_use_node_name 1
# Do not save simulation decks with SKI as the disk storage requirement is very
# high.
set_var extsim_save_passed none
```

```
# Write SKI temporary files in an NFS partition in the working directory instead
of /tmp
set env(TMPDIR) "$env(PWD)/ski_temp"
```

 *Important*

To use SKI, changes must be made to your altos_init file. An example is provided in the <release>/examples/ directory. We strongly recommend you use this altos_init as-is. Users who are familiar with modifying the altos_init file may merge their existing file with the example provided. The new altos_init should be placed in one of the following locations, in order of priority:

- run directory
- \$HOME
- \$ALTOSHOM/etc/

With SKI's current implementation, Liberate communicates with Spectre via shared memory and semaphores. One caution is that if a Liberate job is killed via bkill or qdel, shared memory may not be cleaned up properly. The altos_init script that is provided will perform this clean up automatically.

Note: When extsim_model_include is set, Liberate will check the format of the file and if compatible (ie: first line must not be a legal SPICE command), it will automatically enable the Spectre +modellib option. When using a full EMI flow, this can increase performance by ~30%.

Correlation between stand-alone Spectre & SKI

To achieve correlation between Spectre stand-alone, SKI, and Alspice add the following options. These are recommended as they will ensure consistent methodologies are used between the different simulations.

```
# Standardize transient window for Alspice/SKI and extsim decks
set_var power_sim_estimate_duration 1
set_var power_tend_match_tran 1
set_var tran_tend_estimation_mode 1
# Match SKI methodologies to Spectre
set_var ski_alter_mode 1
set_var ski_mdlthreshold_exact 1
set_var ski_power_subtract_output_load_match_extsim 1
# Match Alspice methodologies to Spectre
set_var alspice_power_subtract_output_load_match_extsim 1
```

Special Licensing

Cadence offers a characterization-only license for Spectre to increase cost-performance of library characterization known as the `spectre_char_opt` license. It is recommended to use this license if available.

Newer versions of Liberate automatically disables this feature if no licenses are available from the license server. Older versions would wait until the license server timed-out before switching to a different Spectre license, resulting in a delay at the start of every simulation. If no `spectre_char_opt` licenses are included, this feature may be disabled by setting the following parameter:

```
set_var spectre_use_char_opt_license 0
```

HSPICE

Accuracy settings for 28nm and below

```
set_var extsim_cmd < path to HSPICE >
set_var extsim_leakage_option "accurate=1 brief=1 runlvl=6 method=gear
gmindc=1e-15 gmin=1e-15 kcltest=1"
set_var extsim_option "accurate=1 brief=1 runlvl=6 autostop gmindc=1e-15
gmin=1e-15"
```

ECSM Accuracy and Correlation Settings for 20nm and Below

Use these settings to produce:

- Best-practices ECSM Libraries
- Production ECSM libraries for 20nm and below

Driver Cell

If the library is to be used only in an ECSM flow, then an active driver (or series of active drivers) has shown to provide the best accuracy. If the library is to be used in a mixed CCS/ECSM flow, then studies need to be done by the end-user to determine which driver yields the best overall results.

Recommended Liberate Settings for ECSM

```
# Set ECSM modeling variables for N-piece
set_var ecsm_version 2.1
set_var ecsm_cap_mode 1

# Receiver capacitance thresholds need to be balanced with respect to rise/fall
# thresholds
set_receiver_cap_thresholds \
    -rise [list 0.1 0.3 0.5 0.6 0.7 0.8 0.9 0.9999] \
    -fall [list 0.9 0.7 0.5 0.4 0.3 0.2 0.1 0.0001]

# For ECSM waveform thresholds, the range is extended from 2% to 98%
define_template -type ecsm -index_1 {0.02 0.05 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8
0.9 0.95 0.98} ecsm_13

# Change the measured voltage range from fixed to dynamic.
set_var ecsm_measure_output_range 1

# Disable transition tables for tristate disable arcs
set_var tristate_disable_transition 1

# Update the ECSM-N model
set_var ecsm_arctype_enable 0
set_var ecsmn_loadcap_mode 1
set_var ecsmn_mode 1
```

See ecsm_measure_output_range for important considerations.

CCS Accuracy and Correlation Settings

Use these settings to produce:

- Best-practices CCS Libraries
- Production CCS libraries

Driver Waveform

The 2013.12 CCS Characterization Guidelines recommend using the released version of the CCS predriver waveform.

Recommended Liberate Settings for CCS

```
## Driver Waveform Settings
set_var predriver_waveform 2
set_var predriver_waveform_mode 1
set_var predriver_waveform_npts 17
set_var predriver_waveform_ratio 0.5

## CCS Timing Settings
set_var ccs_abs_tol 9e-13
set_var ccs_max_pts 50
set_var ccs_rel_tol 0.009
set_var ccs_voltage_tail_tol 0.955
set_var ccs_voltage_tail_trim_tol 0.999

## CCS Noise Settings
set_var ccsn_floating_init_mode 1

## CCS Power Settings
# Use 0 for correct CCSP behavior of bundle cells
set_var ccsp_related_pin_mode 1

## Compact CCS/CCSP
set_var ccs_base_curve_points 15
set_var ccs_base_curve_share_mode 2
set_var ccsp_base_curve_points 15
set_var ccsp_leakage_current_compensation_mode 1
```

D

Deprecated and Legacy Commands and Variables

This section lists all the variables that are either deprecated, or included for backward compatibility. We would like to discourage users from relying on these variables, and instead find alternate variables and settings to achieve the best results from Liberate.

Deprecated Commands

These commands are being phased out, and have been replaced by either new commands, new variables, or new behaviors of the tool.

add_lib_attribute

{attributes} Specifies attribute(s) to add to a library.

add cell attribute

{cells} {attributes}
Specifies attribute(s) to add to cell(s) in a library.

add pin attribute

{cells} {pins} {attributes}
Specify attribute(s) to add to the pin(s) of cell(s)

These commands provide the ability to add user-specified attributes to the ldb. Attributes can be applied to a library, a cell, or a pin, and can be simple or complex.

Example of adding attributes to a **library**:

```
add_lib_attribute {  
    wire_load ("5000") {  
        resistance : 2.500000e-03;  
        capacitance : 0.001000;  
        area : 3.000000;  
        fanout_length("1.0000000", "5.0000000");  
    }  
}
```

Example of adding attributes to a **cell**: (notice the use of wild cards for cell names)

```
add_cell_attribute {AN??D* AO3*} {  
    test_cell () {  
        ff (IQ,IQN) {  
            clocked_on : "cp";  
            next_state : "d";  
            clear : "rn";  
        }  
    }  
}
```

Example of adding attributes to a **pin**: (wild cards used to match cell groups and pin names)

```
add_pin_attribute lv_buffer* a* {  
    level_shifter_data_pin : true;  
    input_signal_level : "dvdd2v8to1v8";  
}
```

Sample usage:

```
# Read in ldb file  
read_ldb test.ldb.gz  
  
# Apply user data  
add_lib_attribute {myAttribute1}  
  
# Write .lib  
write_library -ccs -overwrite test.lib
```

define_arc

In the **define_arc** command, the **-constraint** argument has been deprecated. Instead use the **-logic_constraint** argument.

define_em

Defines the cells and resistors and ports that will be characterized for electromigration models.

Options

| | |
|---------------------|--|
| -mode | Model mode, can be {ac}, {dc} or {ac dc}. |
| -resistor {R, Port} | Specifies the resistor names and output port names for EM acquisition. |
| {cellnames} | List of cells for EM characterization. |

The resistor name can be a wildcard ("*") and supports a trailing wildcard ("name*"). The `ac` and `dc` modes are used in the proprietary calculation of `max_toggle_rate` library values. In the `dc` mode, the internal currents are summed for both rising and falling toggles in the same simulation. In `ac` mode, each edge is modeled separately. In Chapter 7, the section on [Electromigration Models](#) describes these modes in detail. This command supports wildcards.

This command must be used before `char_library`.

Example

```
define_em -mode {ac dc} -resistor {R1 Y} INVX1
define_em -mode {dc}      -resistor {R2 } {INVX1 NANDX1}
```

remove_pin_attribute

Removes automatically generated pin-level attributes from the output Liberty file.

Options

| | |
|------------------|---------------------|
| {cells} | List of cells. |
| {pins} | List of pins. |
| {attribute list} | List of attributes. |

This command supports wildcards. Pin-level attributes can be provided in the `write_library -user_data` file or automatically added by Liberate.

Note: To remove pin-level attributes that are supplied in the `user_data` file, modify the `user_data` file.

Examples

```
# Remove input_signal_level from pins Q and CK on cell DFF
remove_pin_attribute {DFF} {Q CK} {input_signal_level}

# Same as above except all pins of all cells
remove_pin_attribute {*} {*} {input_signal_level}
```

set_ccs_retry_thresholds

Allows users to override the default behavior of Liberate with respect to voltage thresholds checked during CCS-T characterization.

Options

| | |
|---------------------------|--|
| <code>-rise {list}</code> | List of voltage thresholds specified as a percentage in decimal. (for example, <code>.5</code> = 50%) |
| <code>-fall {list}</code> | List of voltage thresholds specified as a percentage in decimal. (for example, <code>.5</code> = 50%) |

The default is for Liberate to check the following:

- `rise: measure_slew_lower_rise, delay_out_rise, measure_slew_upper_rise`
- `fall: measure_slew_upper_fall, delay_out_fall, measure_slew_lower_fall`

This command has no effect if `ccs_retry_mode=0`. If `ccs_retry_mode` is set to 1 or 2, Liberate checks the reconstructed voltage waveform derived from the output current waveform at these thresholds against the `ccs_abs_tol` and `ccs_rel_tol` criteria. If the tolerance criteria are not met, the simulation will be retried with more accurate options.

This command must be used before `char_library`.

Example

```
# Specify 20%, 30%, %50, 70%, 80%
# for both rising and falling arcs
```

```
set_ccs_retry_thresholds \
  -rise [list 0.2 0.3 0.5 0.7 0.8] \
  -fall [list 0.2 0.3 0.5 0.7 0.8]
```

set_client

Options

-n <number_of_clients>

Specifies the number of clients to use.

If the **-n** argument is specified, Liberate submits the jobs to the specified number of clients with the specified queue name. The directory name (**-dir**) can be created using some or all of the following values:

- **%N** (denotes the client number)
- **%U** (denotes the user name)
- **%S** (denotes the server name)
- **%P** (the Liberate server process ID)

Important

The **-n** argument explained above is deprecated. Use the packet mode instead of the **set_client** mode to use a queuing system. For more details on distributed parallel processing, see [Chapter 3, “Parallel Processing.”](#)

-dir <directory_name>{%N%U%P%S}

Defines a directory on the client machine to use as a temporary workspace for simulation jobs performed on that machine. Liberate creates the directory if it does not exist. (REQUIRED)

<Queue_name> Name of the batch queue.

These can also be used to create unique scratch directories for each individual Liberate characterization run.

Note: When the **-n** argument is used, all file names within the Tcl file must be full path names.

It is also recommended that Liberate is executed with a Tcl file specified using a full path name.

The `-n` argument is only supported when an external job-queuing system is used. When Liberate is managing the clients using its built-in queuing system and the `rsh_cmd` variable is set to `rsh` or `ssh`, the `-n` argument should not be used. Alternatively, Liberate can perform distributed processing by explicitly defining the names of each of the client machines.

To specify multiple machines, use multiple `set_client` commands. The network port number to be used can also be set using the `set_network_port` command.

set_em_imax

Specifies the maximum current value and the resistors or output pins to measure. The `-resistor` and `-cells` options both accept a wildcard ("*"). For more information, see [Electromigration Models](#).

Options

| | |
|--|--|
| <code>-cells {list}</code> | List of cell names. |
| <code>-resistor {resistor output_names}</code> | List of resistor names or output pins. |
| <code>-type <"avg" "rms"></code> | Average or RMS. |
| <code><value></code> | IMAX value. |

This command must be used before `write_library`.

Example

```
# Set the man current for resistor R1 to 5mA
set_em_imax -type avg \
  -resistor { R1 } \
  -cells { inv and2 }
  5e-3
```

Deprecated Variables

These variables are being phased out, and have been replaced by either new commands, new variables, or new behaviors of the tool.

bundle_count

`<value>` Specifies a specific number of bundles. Default: follow `bundle_mem_limit`.

Bundle mode is enabled when `bundle_mem_limit` is exceeded or when this variable is set to a number of bundles. Use this option to override the bundle count that is automatically determined when bundle mode is enabled by `bundle_mem_limit`. For example, if there are 100 cells and `bundle_count` is set to 2, there will be 2 bundles with 50 cells in each bundle.

By default, the bundle mode is not enabled. This feature has been deprecated in favor of the packet mode.

Note: `bundle_count` is not compatible with either the cell-based or arc-based packet mode. If `packet_clients>0`, do not set `bundle_count>0`.

This variable must be used before `char_library`.

ccsn_allow_probe_driven_by_feedback

`< 0 | 1 >` There could be a latch up condition in the Channel Connected Component/Region (CCC) that is driving the out probe (for example, through a passgate). This variable checks if the out probe is connected to any such loop wire and tries to choose an alternate CCC output, if available. Default: 0

| | |
|---|---|
| 0 | Avoids using such probes for CCSN characterization. |
| 1 | For backward compatibility. |

ccsn_overlap_ccr_include_mode

< 0 | 1 > Updates the handling of CCB when the input is (also) connected to the output through a passgate thus providing an alternate path. Set this to 1 for backward compatibility. Default: 0

cell_use_both_ff_latch_groups

< 0 | 1 | 2 > Enables support for multiple *ff* and *latch* groups in the output library. Default: 1

0 Liberate includes one of the latch groups in the output library if the `write_library user_data` file contains both *ff* and *latch* groups.

Note: This setting is provided for backward compatibility.

1 Liberate enables the modeling of both the *ff* and *latch* groups, but keeps only one of each if there are multiple *ff* and *latch* groups in the `user_data` file.

2 Liberate enables the modeling of multiple *ff* and *latch* groups. This is the recommended setting if multiple *ff* and *latch* groups are needed to correctly model the cell function.

Some complex cells, such as retention DFFs use the simultaneous modeling of *ff* and *latch* groups in the Liberty file for correct function modeling. Use this variable to enable support for modeling multiple function groups.

The *ff*, *latch*, *ff_bank*, and *latch_bank* must be provided with the `write_library user_data` file.

To model *ff_bank* and *latch_bank* groups, the `define_bus` and/or `define_bundle` commands must also be used. The *ff*, *latch*, *ff_bank*, and *latch_bank* groups can be modeled under a cell, as described below.

| | <i>ff/latch</i> | <i>ff_bank/</i> <i>latch_bank</i> |
|---------------|-----------------|--------------------------------------|
| buses/bundles | yes | yes |

no buses/bundles yes no

This variable must be set before `write_library`.

default_capacitance

`<min | avg | max>` Sets the desired selection criteria for the *capacitance* attribute. Use the minimum (`min`), average (`avg`), or maximum (`max`) rise/fall capacitance value as the default pin rise/fall capacitance. Default: `max`

Note: This variable has been deprecated. Use of the `set_default_group` command is recommended instead of this variable.

This variable must be set before `char_library`.

default_group_method

`< 0 | 1 >` Specifies the method to use for creating default groups. Default: `1` (use the whole table with the worst bit)

Note: As this is a deprecated variable, use of the `set_default_group` command instead is recommended.

`0` The worst value (`min` or `max`) from all the relevant tables on a bitwise basis will be individually used for all data types. If `default_power` is set to `avg`, a point-to-point average value from all the power groups is used for the default power table.

`1` Find the delay, power and cap table that has the worst value (reviewed bitwise) and to use that complete table, in its entirety, in the default group. This variable does not apply to constraints. Constraint default groups are always selected on a bitwise basis. (Default)

This variable must be set before `char_library`.

default_leakage

`< min | avg | max >` Use the minimum (`min`), average (`avg`), or maximum (`max`) leakage current value as the default leakage current. Default: `avg`

Note: This variable has been deprecated. It is recommended that the [set_default_group](#) command be used instead of this variable.

This variable must be used before `write_library`.

default_power

`< off | min | avg | max >`

Either ignore (`off`), use the minimum (`min`), the average (`avg`) or the maximum (`max`) power as the value for the default power group. Default: `max`

Note: This variable has been deprecated. It is recommended that the [set_default_group](#) command be used instead of this variable.

This variable must be used before `char_library` if it is to have any effect.

default_timing

`< off | min | max | force_off >`

Either ignore (`off` or `force_off`), use the `min` or `max` delay, transition and constraint values as the value for the default timing group. Default: `max`

Note: This variable has been deprecated. It is recommended that the [set_default_group](#) command be used instead of this variable.

This variable must be used before `char_library`.

default_unateness

< merge | separate >

Controls whether default `positive_unate` and `negative_unate` timing groups should be merged into a single `non_unate` default timing group. Default: `merge`

Note: This variable has been deprecated. It is recommended that the [set_default_group](#) command be used instead of this variable.

These parameters control the creation of default library group data for pin capacitance, leakage, power and timing. The default group data is created from taking the `max` or `min` of the values from the relevant conditional groups. The default timing or power groups can be omitted by setting the `default_timing` or `default_power` to `off`. Setting this parameter to `separate` will keep these groups distinct.

Note: If all `timing_sense` attributes are identical, then the original timing sense will remain unchanged during merging. If there are both `positive_unate` and `negative_unate` timing sense group data, then the merged timing sense will be `non_unate`.

This variable must be used before `char_library`. Examples:

```
# Set defaults for the fast corner
set_var default_power min
set_var default_capacitance min

# Ignore default timing groups
set_var default_timingoff

# Disable merging for binate default groups
set_var default_unateness separate
```

By default Liberate will create a single default timing group by merging `positive_unate` and `negative_unate` default groups into a single `non_unate` timing group. Setting this parameter to `separate` will keep these groups distinct. This command typically has an impact on complex combination cells, such as adders.

em_period

`< value >` Specifies the clock period (in seconds) at which to characterize electromigration (EM). The value specified must be a number in seconds. Default: " "

The `em_clock_freq` overrides `em_period`. It is recommended that only one of these two variables is specified. For more information, see the [Electromigration Models](#) section in the Liberate Details chapter.

This variable must be used before `char_library`.

Example

```
# Set the EM Clock Period to 50ns
set_var em_clock_freq 50e-9
```

extsim_interactive

`<0 | 1>` Specifies to enable the external simulator when it is enabled with `char_library -extsim` to operate in an interactive mode.
Default: 0 (disabled)

0 Corrects the settings of the previously mentioned commands if you have problems with hanging jobs in your environment.

Running the external simulator in interactive mode reduces the external simulator start-up time and can be useful in network environments that are not configured for efficient cell characterization runs. One currently known issue is that using Eldo in interactive mode can leave orphaned processes that will need to be manually killed if the Liberate/Variety run exits abnormally.

Note: This functionality only supports Eldo 2008.06 or later. Liberate uses the interactive mode that starts Eldo only once upfront instead of starting Eldo for each simulation deck. You need to take care not to start more interactive jobs than available license tokens by using appropriate settings for `set_client`, `char_library -thread`, and simultaneous job submission. Failure to do this may result in hanging Liberate and Eldo jobs. If you have problems with hanging jobs in your environment, correct the settings of the previously mentioned commands or disable `extsim_interactive` by setting it to 0.

This variable must be used before the `char_library` command.

Example

```
set_var extsim_interactive 1
```

leakage_accuracy_mode

`< 0 | 1 >` Improves leakage accuracy. Default: 1

0 Liberate uses pre-processed netlist for leakage simulation.

1 Liberate retains all R/C.

This variable is relevant for netlists flattened by Liberate.

If the netlist is not flattened by `Liberate` (`extsim_model_include` and `define_leafcell` are specified and `extsim_flatten_netlist` is set to 0) then the netlist and model are included in the deck and `leakage_accuracy_mode` is essentially non-operational. This is the recommended flow for FinFET technologies.

This variable must be used before `char library`.

max capacitance auto mode

< 0 | 1 | 2> Selects the computation mode for tiehi and tielo cells.
Default: 1

0 Selects the computation mode for tiehi and tielo cells.

1 The `char_library` and `write_vdb` arguments
-auto_index or -auto_max_capacitance
calculates the `max_capacitance` pin attribute
for TieHi and TieLo cells by dividing the
max transition by the Rds.

2 Calculates maximum capacitance for TIE cell without using `-auto_max_capacitance` option of the `char library` command.

To measure the Rds, connect a 100kOhm resistor to the opposite supply and compute the Rds (Resistance drain to source) as follows:

```

TieHi_Ion = (Vdd -Vth)/Rds = Vth/Rl
TieLo_Ion = (Vdd -Vt1)/Rl = Vt1/Rds
Max_cap = max_transition / Rds

```

Where:

- v_{th} = The TieHi steady-state output pin voltage.
 - v_{tl} = The TieLo steady-state output pin voltage.

This parameter must be used before the `char_library` command.

mpw_delay_use_active_edge

< 0 | 1 > Specifies which clock edge to use when characterizing MPW.
Default: 1

Set this variable to enable the measurement of mpw related delay degradation from the active edge of the circuit clock instead of the leading edge of the clock. By default, when `mpw_criteria = 1` (delay degradation), Liberate will measure the delay degradation from the leading edge of the clock to the transition on the probe node. This can lead to incorrect delay pushout if the delay should be measured from the trailing (active) edge of the pulse. The default and recommended setting for this variable is `1`.

This variable must be used before `char library`.

packet arc licensing mode

```
< default | wait_for_clients >
```

Controls license-handling for arc-based packet mode. Default: default

Note: This variable has been deprecated and should not be used any longer.

default All Liberate clients will release their Liberate and Spectre licenses as soon as characterization has completed on that job. Recommended for installations with one server or where the load balancing software manages the tool license usage. (Default and Recommended)

wait_for_clients

The Liberate server performs block check-outs of `Liberate_client` and `Spectre_char_opt` licenses based on client license availability. If insufficient licenses are available, the server continues to query the license server for additional licenses.

The Arc-based Packet Mode provides flexible usage of the `Liberate_client` and SPICE licenses. The optimal method depends on the number of Liberate servers, Liberate clients, Spectre licenses, and demand from other users for the Spectre licenses. For flows not using Spectre, only the number of Liberate licenses needs to be considered.

This variable must be used before `char_library`.

packet_arc_require_spectre_char_opt

<0 | 1> Defines how the arc packet flow handles `Spectre_char_opt` licenses.
Default: 0

Virtuoso Liberate Reference Manual

Deprecated and Legacy Commands and Variables

0

The packet_client continues even if it fails in checking out the required number of Spectre_char_opt licenses. However, each characterization task on this packet_client checks out Spectre_char_opt licenses when the task starts and checks them back in at the end of the task. If packet_client fails to check-out Spectre_char_opt licenses, it will let Spectre handle its licensing and check-out other licenses such as, Virtuoso_Multi_mode_Simulation, Virtuoso_Spectre, or Virtuoso_Acceler_Parallel_sc, as required. Therefore, excessive license check-out and check-in occurs if Spectre_char_opt licenses are insufficient. This is because each packet_client typically runs 10s of tasks.

1

The packet_client waits until it checks out the required number of Spectre_char_opt licenses. The packet_client during its initialization phase checks out the required Spectre_char_opt licenses upfront according to the number of threads and simulation type. Simulations involving electromigration require 2 Spectre_char_opt licenses per CPU thread. However, other simulations require 1 Spectre_char_opt license per CPU thread. In this mode, Liberate checks out Spectre_char_opt licenses and does not attempt to check out other Spectre licenses such as, Virtuoso_Multi_mode_Simulation, Virtuoso_Spectre, or Virtuoso_Acceler_Parallel_sc. (Recommended if enough Spectre_char_opt licenses are available).

This variable must be used before the `char_library` command.

rc_sort_mode

`< 0 | 1 >` Enables sorting of RC elements. Default: 1

This variable lets you decide how to add parasitics to Alspice or external simulator decks to help prevent consistency issues when running a single cell versus an entire library. By default, Liberate enforces a consistent order for both single and multi-cell runs. If set to 0, Liberate reverts to release 3.0p2 prior behavior where RC elements are never sorted.

Note: This variable is deprecated with release 3.2p4 and will be ignored; RC's will always be sorted for consistency. Using this variable will generate a warning.

Backward Compatibility Variables

These variables invoke older behaviors of the tool. We generally discourage using these variables because many of the older algorithms have since been corrected or improved.

capacitance_force_hidden

| | |
|-----------|--|
| < 0 1 > | Enables characterization of input capacitance for hidden power arcs. Default: 1 |
| 0 | In IO mode (see the -10 option of the char_library command), Liberate does not characterize input pin capacitance for hidden power arcs when the pin does not have any forward arcs. The following warning is printed: *Warning* (char_library): Capacitance acquisition is disabled for hidden arc on pin PD_EN. |
| 1 | Enables input capacitance characterization on hidden power arcs. No warning is printed since the input capacitance is characterized. |

This variable must be used before the `char_library` command.

ccsn_active_ccr_recognition_mode

| | |
|-----------|--|
| < 0 1 > | Specifies whether to do a more aggressive search for CCSN arcs. Default: 1 |
|-----------|--|

If this variable is set to 1, Liberate will more aggressively search for CCSN arcs. It will search for the existence of ccsn paths from input to outputs, based on the actual vector being sensitized to remove inactive CCSN paths. The recommended setting is 1.

This variable must be used before the `char_library` command.

ccsn_check_valid_noise_prop

| | |
|-----------|---|
| < 1 2 > | Checks if a CCB can propagate noise prior to characterization. Default: 1 |
|-----------|---|

- | | |
|---|---|
| 1 | Check for a tristated CCB output. |
| 2 | Use this setting for backward compatibility to release 13.1ISR4 and prior releases. |

Use this variable to enable an algorithm that screens the CCB for vectors that cause the CCB output to become tristated.

This variable must be set before the `char_library` command.

`ccsn_compatibility_mode`

- | | |
|----------------------------------|--|
| <code>< 0 1 2 ></code> | Controls checks to more thoroughly analyze CCB topologies and electrical paths to identify and characterize CCSN stages at the timing and pin level. Default: 1 |
| 0 | Use standard level of CCB topology checking. |
| 1 | Use additional checks that correct cross-PVT compatibility issues. (Default and recommended if voltage/temperature scaling will be used on the resulting libraries.) |
| 2 | Use the <code>char_library -io</code> mode CCSN algorithm when the delay-based vector generation algorithm does not produce the expected results. |

This variable must be set before the `char_library` command.

`ccsn_compatibility_multi_corners`

- | | |
|------------------------------|--|
| <code>< 0 1 ></code> | Effects CCSN structure in library. Default: 1 |
| 0 | Backward compatible to release 3.2p4_1cs where different PVTs can have structure differences in CCSN data. |
| 1 | Maintain structure equivalence across PVTs. (Default) |

ccsn_dc_estimate_mode

| | |
|---------------|--|
| < 0 1 2 > | Controls algorithm for estimating the end time for CCSN simulations. Default: 1 |
| 0 | Backward compatible to release 12.1. The original algorithm for estimating simulation end times. |
| 1 | The end time is estimated from the previously simulated dc current results to get a more realistic end time. This helps in getting voltage waveforms that reach the final expected limits. (Default and recommended) |
| 2 | Use this setting to enable a simulation duration time estimation for ccsn output voltage simulations for use with some corner cases with large transition times. |

Occasionally, the end time for voltage rise/fall CCSN simulations is not long enough and the final expected voltage was not reached. This variable causes the estimated end time to be extended for such simulations.

This variable must be set before the `char_library` command.

ccsn_dc_sweep_mode

| | |
|-----------|--|
| < 0 1 > | For CCSN characterization compatibility when running an external simulator. Default: 1 |
| 0 | External simulator compatibility setting for CCSN characterization. |
| 1 | (Default) |

ccsn_fanout_select_mode

| | |
|-----------|---|
| < 0 1 > | Sets CCSN groups on test pins. Default: 1 |
|-----------|---|

When this variable is set to 1, Liberate includes CCSN groups on all test pins. Previously, all test pins did not have CCSN data.

This variable must be set before the `char_library` command.

`ccsn_io_skip_channel_inputs`

| | |
|------------------------------|--|
| <code>< 0 1 ></code> | Changes the order in which the channel-connected wires are considered for probing when CCSN stages are determined. This has an effect on CCSN only when the inside view cannot automatically drive CCSN characterization and the fallback approach of automatic vector determination is used. Default: 1 |
| 0 | No distinction about channel connectivity is made. This setting is used for backward compatibility to 13.1 ISR4 and prior releases. |
| 1 | Consider wires that are not channel-connected (does not connect to the transistor source or drain) first. |

This variable must be set before the `char_library` command.

`ccsn_input_xfr_probe_mode`

| | |
|---------------------------------|--|
| <code>< 0 1 3></code> | Controls CCSN modeling of input pins connected directly to pass-gates. Default: 1 Recommended: 3 |
| 0 | Do not generate <code>ccsn_first_stage</code> groups for input pins that connect directly to transistor pass gates. |
| 1 | Generate <code>ccsn_first_stage_groups</code> for input pins that connect directly to transistor pass gates. |
| 3 | Enables extra analysis of pass gate channel control activity before writing out the spice deck. This is done to improve corner cases to sensitize the CCSN vector. |

This variable must be set before the `char_library` command.

ccsn_mega_mbit_hidden_mode

| | |
|--------------|--|
| < 0 1 2> | Controls how Liberate Mega mode (see mega_enable) generates hidden power arcs when characterizing CCSN for multi-bit FF cells. Default: 1 |
| 0 | Setting provided for backward compatibility to LIBERATE14.1 ISR3 and prior releases. |
| 1 | Output hidden power arcs in sync with pin based ccsn_first_stage arcs in mega mode. |
| 2 | Uses faster method for determining CCSN vectors during preprocessing. |

This variable must be set before the `char_library` command.

ccsn_model_unbuffered_output

| | |
|---------------|---|
| < 0 1 2 > | Specifies how to handle unbuffered outputs. Default: 2 |
| 0 | Do not model unbuffered output pins. |
| 1 | Cleanup or disconnect feedback loops on first stages of pins having output ports. |
| 2 | Cleanup or disconnect feedback loops on first stages of pins having output ports and enhanced handling of channel connected input/output CCBs for unbuffered pins. (Default and recommended). |

When a cell has an output driven through a transmission gate (it is unbuffered), and noise can propagate in the output and corrupt internal stored states, then it might be desirable to model this behavior in the CCSN noise constructs in the Liberty model.

This variable must be set before the `char_library` command.

ccsn_pin_unconditional

`< 0 | 1 >` Set to 1 to stop Liberate from generating conditional pin-based CCSN stages. Default: 0

When set to 1, Liberate will only create a single CCSN pin stage when there are multiple CCSN pin stages possible. This can help reduce the number of CCSN stages in an entire library by about 10%. Recent versions of PrimeTime require more complete CCSN models for improved correlation when noise-on-delay techniques are employed.

The setting 1 should only be used for backwards-compatible purposes.

This variable must be set before the `char_library` command.

ccsn_prefer_two_sided_stages

`< 0 | 1 | 2 >` Sets to prefer two-sided stages. Default: 2

Liberate will output two sided (`stage_type: both`) ccsn stages whenever possible. This helps to address errors in 2008 versions of LC (LC200809-SP3). Reset this variable to 0 to revert back to pre-v2.3 behavior. When set to 2, Liberate will always try to generate CCSN stages that have `stage_type` set to *both*, regardless of whether the sensitization is feasible in the cell. Setting this variable to 2 should address any LBDB-898 Synopsys LC error messages.

This variable must be set before the `char_library` command.

ccsn_probe_mode

`< 0 | 1 | 2 >` Specifies the method for selecting a probe node for noise measurements. Default: 2 (Balanced)

- | | |
|---|--|
| 0 | Strict interpretation of the guidelines (Behavior of Liberate version 12.1.4 and earlier.) |
| 1 | Sorting method. |
| 2 | Balanced method. (Default and recommended) |

Controls how Liberate will select the observation or probe point, based on section 2.5 of the CCS Noise Characterization Guideline version 1.05, which details the considerations for selecting an observation point and handling parasitics during CCS Noise measurements.

This variable must be set before the `char_library` command.

`ccsn_prop_retry_duration_incr`

`< 0 | 1 >` Automatically extends the CCSN simulation duration on failures. Default: 1

Set this variable to 1 to automatically extend the simulation duration for CCSN measurements if the original simulation fails. If you see warnings regarding "Unable to obtain reasonable CCS noise propagated waveform", then set this variable and re-run characterization.

The default and recommended setting is 1.

This variable must be set before the `char_library` command.

`ccsn_redundant_pin_stages`

`< 0 | 1 >` Set to duplicate certain CCSN stages on both arc and pin.
Default: 0 (Do not duplicate)

| | |
|---|--|
| 0 | Do not allow redundant stages. (Default and recommended) |
| 1 | Allow redundant stages (3.2p4_lcs behavior). |

Prior to the 12.1 release, Liberate would duplicate some CCSN stages such that they would show up as both pin-based and arc-based CCSN groups. These may be considered as redundant groups, since the STA and SI tools have rules regarding priority of pin-based or arc-based noise stages.

This behavior was fixed in 12.1. For backwards-compatibility purposes, this parameter was introduced in 12.1.2.2 to allow the redundant stages.

This variable must be set before the `char_library` command.

ccsn_skip_mod_vdata

| | |
|------------------------------|---|
| <code>< 0 1 ></code> | Liberate generates vecdata for CCSN arcs. Some of this vecdata is associated with tristate paths. The vecdata is global, and may be visible when building other CCSN stages. However, in some cases, the vecdata from tristate paths leads to false CCSN stages. Default: 1 |
| <code>0</code> | Get all matching vecdata when characterizing CCSN. The vecdata may originate from a tristate arc and then be modified for use with CCSN characterization. This setting is provided for backward compatibility to LIBERATE14.1 ISR3 and prior releases. |
| <code>1</code> | Ignore simulation vectors that originate from tristate arcs when characterizing CCSN. |

This variable must be set before the `char_library` command.

cell_leakage_power_legacy_mode

| | |
|----------------------------------|---|
| <code>< 0 1 2 ></code> | Determines when <code>cell_leakage_power</code> will be recalculated, based on the setting of <code>voltage_map</code> . Default: 0 |
| <code>0</code> | <code>cell_leakage_power</code> will be re-calculated for <i>all</i> <code>voltage_map</code> settings. (Default) |
| <code>1</code> | <code>cell_leakage_power</code> will <i>not</i> be re-calculated. (Behavior of release 3.1p1 and earlier.) |
| <code>2</code> | <code>cell_leakage_power</code> will be re-calculated <i>only</i> when <code>voltage_map=1</code> |

Note: Intended for backward compatibility only.

This variable must be set before the `write_library` command.

char_mos_term_cap_ski_mode

| | |
|------------------------------|--|
| <code>< 0 1 ></code> | Automates the MOS terminal capacitance calculation when using the SKI interface. Default: 1 |
| 0 | Follow the setting of the <code>char_mos_term_cap</code> variable. This setting is used for backward compatibility to 13.1 ISR4 and prior releases. |
| 1 | When the SKI interface is enabled (see <code>ski_enable</code>), and <code>char_mos_term_cap</code> is set to -1, then override the setting of <code>char_mos_term_cap</code> with a value of 2. Otherwise, use the setting of <code>char_mos_term_cap</code> . |

This variable must be set before the `char_library` command.

conditional_expression_accuracy

| | |
|----------------------------|--|
| <code><0 1></code> | Merges state dependent arcs into a single timing group (see <code>conditional_expression</code>). Use this variable to improve the accuracy before merging the individual paths together. Default: 1 |
| 0 | Uses coarse transistor resistance estimate so that different PVTs match. This setting is provided for backward compatibility to LIBERATE 15.1.2 and prior releases. |
| 1 | Uses fine transistor resistance estimate for less merging and better <code>merge</code> and <code>separate</code> state matching (see <code>conditional_expression</code>). |

This variable must be set before the `char_library` command.

constraint_search_time_reltol_mode

| | |
|----------------------------------|--|
| <code>< 0 1 2 ></code> | Affects when bisection will stop. For backward compatibility only. Default: 2 |
|----------------------------------|--|

| | |
|---|--|
| 0 | For backward compatibility to pre-3.2 versions. May result in more search iterations. (Not recommended.) |
| 1 | For backward compatibility to pre-13.1.3 version. May result in more search iterations. (Not recommended.) |
| 2 | The search tolerance follows <code>constraint_search_time_abstol</code> and <code>constraint_search_time_reltol</code> . (Default) |

For pre-3.2 versions of Liberate, if the constraint is less than 1ps, then bisection will stop only if the search window is < 1ps or `constraint_search_time_abstol`, whichever is smaller.

For version prior to Liberate 13.1.3, this occurs only when `constraint_bisection_mode=2`. For versions post Liberate 13.1.3, this does not occur.

This variable must be set before the `char_library` command.

default_non_unate_rcvr_cap_adjust

| | |
|-----------|--|
| < 0 1 > | Sets the receiver cap groups to follow the input pin direction. Default: 1 |
| 0 | The receiver cap groups will follow the <u>output</u> pin direction. (Backward compatibility setting.) |
| 1 | The receiver cap groups will follow the <u>input</u> pin direction for non-unate arcs. (Default) |

This variable must be set before the `char_library` command.

default_power_subtract_hidden_mode

| | |
|---------------|---|
| < 0 1 2 > | Controls <code>subtract_hidden_power</code> algorithm. Default: 2 |
| 0 | Use the algorithm from 3.2. The default <code>internal_power</code> is chosen and then the <code>hidden_power</code> is subtracted. |

- | | |
|---|---|
| 1 | Use the algorithm from 12.1 ISR1. The <code>hidden_power</code> is subtracted and then the <code>default internal_power</code> is selected. |
| 2 | Use the algorithm as of 12.1 ISR4. This is a fix to mode 1 where hidden power is sometimes not subtracted. (Default) |

This variable is used to control the algorithm used when `subtract_hidden_power` is enabled.

This variable must be set before the `write_library` command.

default_timing_tristate_enable

- | | |
|------------------------------|--|
| <code>< 0 1 ></code> | Selects the default group timing type for <code>three_state_enable</code> arcs. Default: 1 (default arc timing_type uses <code>three_state_enable</code>) |
| 0 | The default arc timing_type for <code>three_state_enable</code> arcs will be set to 'combinational'. This setting is provided for backward compatibility to release LIBERATE 13.1 ISR4 and prior releases. |
| 1 | The arc timing attribute uses <code>three_state_enable</code> (Default). |

This variable is used to control `timing_type` attribute for default `three_state_enable` timing groups.

This variable must be set prior to the `char_library` command.

def_arc_delay_metric_mode

| | |
|-----------|---|
| < 0 1 > | For backward compatibility with <code>define_arc -metric</code> . Default: 1 |
| 0 | Revert to release 12.1 and prior behavior where under certain conditions, the Liberate internal algorithms would override the user-specified <code>define_arc -metric</code> . |
| 1 | (Default and recommended.) |

Note: The `define_arc -metric` option should always control the constraint measurement metric.

This variable must be set prior to the `char_library` command.

disable_current_measure_effort

| | |
|-----------|--|
| < 0 1 > | Instructs the tool to interpolate the actual crossing time of output current crossing the current degradation threshold. Default: 1 |
|-----------|--|

The `three_state_disable` delay measurement interpolates between the time steps reported by the circuit simulation. Set this to 0 to revert to release 2.5 and prior behavior, where Liberate reports the time point when output current drops below the current degradation threshold.

This variable must be set before the `char_library` command.

driver_waveform_lib_mode

| | |
|-----------|---|
| < 0 1 > | Affects the way driver waveforms are stored in the library in the normalized_driver_waveform. Pertains to the falling waveform portion only. Use this variable for backward compatibility. Default: 1 |
| 0 | Format for normalized_driver_waveform used with version 3.2p3 or earlier. |
| 1 | Standard format for normalized_driver_waveform. (Default and recommended.) |

This variable must be set before the `write_library` command.

ecsm_p_invert_gnd_current

| | |
|-----------|--|
| < 0 1 > | Inverts current values for <code>ecsm_current_waveform</code> groups. Default: 1 (Invert current) |
| 0 | Do not invert currents. |
| 1 | Inverts the current values for compatibility with EPS. (Default) |

When reporting current values, Liberate usually follows the convention where positive currents flow into the cell and negative current flow out of the cell. In this fashion, supplies are normally positive and grounds are normally negative.

Cadence's power tools prefer to have ground currents reported as positive values. This variable controls whether or not to invert the current on grounds, which are reported in index_1 of `ecsm_current_waveform` groups.

This variable must be used before the `char_library` command.

extsim_model_include_multi_vector_mode

| | |
|-----------|--|
| < 0 1 > | For backward compatibility. Uses 3.1 (and prior) algorithm Default: 1 |
| 0 | Backward compatible to release 3.1. |
| 1 | (Default) |

Version 3.1p1 corrects an issue handling constraint-related vectors in the EMI flow when `extsim_flatten_netlist` is set to 0.

`extsim_save_driver`

| | |
|------------------------------|---|
| <code>< 0 1 ></code> | Saves SPICE decks for failing active driver simulations. Default: 1 |
| 0 | Specifies not to save any active driver simulation decks. This option is available for backward compatibility. |
| 1 | Save the simulation decks used for the final driver waveforms. This option works in combination with the <code>extsim_deck_dir</code> , <code>extsim_save_passed</code> , and <code>extsim_save_failed</code> variables. If the saving of decks is not enabled using these variables, no decks are saved. |

This variable is used to enable the saving of simulation decks used by the `set_driver_cell11` command to characterize the active driver output waveform simulation decks.

This variable must be set before the `char_library` command.

`floating_node_consistency_check`

| | |
|------------------------------|--|
| <code>< 0 1 ></code> | This variable enables floating node consistency checks introduced in 3.1p2. Default: 1 |
| 0 | Backward compatible to release 3.1p2 . Disables floating node consistency checks. |
| 1 | Enable the floating node consistency checks. (Default and recommended.) |

This variable should only be disabled to reproduce the inconsistent floating node handling behavior of Liberate versions 3.1p2 and earlier for regression purposes. Note that this behavior has known issues and is not recommended for general use.

This variable must be set before the `char_library` command.

ignore_dummy_fets

< 0 | 1 | 2 | 3 > Specifies method for dealing with Pode devices. Default: 2

- | | |
|---|--|
| 0 | Liberate Inside-View considers the dummy devices to have an impact on the functionality of the circuit. Previously, the gates to these devices were viewed as CCC inputs. When the gates were floating, they would generate two vectors, one for each value of 0/1 for each floating gate. When many of these devices are present in a cell, it results in an exponential increase in the number of vectors to evaluate or simulate for no real purpose. This can significantly increase runtime as the values of the nodes associated with these devices is considered during pre-processing and vector generation and differ only in how initial conditions (.ic) are set for the dummy devices with floating gates. This setting is provided for backward compatibility to 13.1 and prior releases. |
| 1 | Ignore functional impact. Set each floating gate to its disabled state. Multiple floating gates on the same simwire may be assigned different initial conditions (.ic). |
| 2 | Ignore functional impact. Set each floating gate to its disabled state. If multiple parasitic devices with floating gates are on the same simwire, a single .ic is assigned to the wire. (Default and Recommended) |
| 3 | Ignore functional impact. Do not assign any .ic to floating gates. SPICE simulators that are not capable of handling dummy devices natively may have problems with DC-convergence if this setting is used and is therefore not recommended. |

Use this variable to determine how Inside View handles non-functional or "dummy" parasitic transistors such as PODE devices. These are devices added for parasitic reasons and that have "pode" in the model name of the device.

This variable must be set before the `char_library` command.

non_seq_probe_mode

| | |
|-----------|--|
| < 0 1 > | Controls the probe selection mode for <i>nonseq_setup/hold</i> . Default: 1 |
| 0 | Compatibility with releases prior to version 2.2. |
| 1 | Compatibility with release 2.2 and later. (Default) |

Specifies the non_seq probing order. When set to 1, Liberate will try to honor constraint_output_pin.

This variable must be set before the `char_library` command.

power_multi_vector_mode

| | |
|---------------|---|
| < 0 1 2 > | Corrects issue of choosing wrong vector. Default: 2 |
| 0 | Backward compatible to release 3.1p1. |
| 1 | Backward compatible to release 13.1 ISR1. |
| 2 | Corrects issue. (Default) |

Related to an issue where Liberate could choose a wrong vector when an arc contained multiple vectors and the `voltage_map > 0`. This could produce power results that may not be worst case.

power_subtract_leakage_tran_mode

| | |
|-----------|--|
| < 0 1 > | Specifies whether Liberate should check individual simulation duration for multiple internal vectors, or use one reference. Default: 1 (Check individual vectors) |
| 0 | Use the same simulation duration for all internal vectors. (For backward compatibility only.) |
| 1 | Use individual simulation duration values for each vector. (Default and recommended.) |

Backward compatibility option. Prior to version 3.2p3, Liberate uses the same simulation duration for leakage subtraction for all internal vectors. This can produce inconsistent power results. Set this to 1 to use individual simulation duration values for each vector.

This variable must be set before the `char_library` command.

reset_negative_power_mode

| | |
|-----------|--|
| < 0 1 > | Controls the behavior for fixing negative power values. Default: 1 (Follow <code>reset_negative_power</code>) |
| 0 | Do not reset negative power values for bulk rails. (For backward compatibility only) |
| 1 | Follow <code>reset_negative_power</code> for all supplies. (Default and recommended) |

When set, Liberate follows `reset_negative_power` for bulk nodes. The recommended setting is 1. This is because it allows `reset_negative_power` to be applied to all supplies.

This variable must be set before the `char_library` command.

set_pin_slew_threshold_mode

| | |
|-----------|--|
| < 0 1 > | Controls handling of slew threshold derating for power. Default: 1 |
| 0 | Scale the <code>index_*</code> and value in the pin group specified. (Only for backward compatibility to release 13.1 ISR2) |
| 1 | If the pin specified in the <code>set_pin_threshold</code> command is the <code>arc</code> -pin, then scale only the <code>rise/fall_transition</code> value. When it is the <code>-related_pin</code> , then scale only the slew index, which is usually <code>index_1</code> . (Default) |

Note: This variable enables correct processing of the `set_pin_threshold` command settings.

This variable must be set before the `char_library` command.

switch_cell_infer_unateness_from_lDb

| | |
|-----------|--|
| < 0 1 > | Controls determination of unateness for switch cells. Default: 1 |
| 0 | Enables behavior where unateness of switch cells was always negative. <i>Not recommended.</i> |
| 1 | Unateness of switch cells is correctly determined and written to the library. (Default and recommended.) |

For backward compatibility when writing out a library from an older ldb (prior to v3.2). Only relevant to switch cells.

This variable must be set before the `write_library` command.

tristate_pin_cap_use_arc

| | |
|---------------|---|
| < 0 1 2 > | Default: 2 |
| 0 | For backwards compatibility. |
| 1 | New method only. |
| 2 | New method with fall back to original method if required. (Default) |

A change has been made in the way input capacitance is computed for tristate outputs and bidi pins. Previously, separate simulations were created to compute rise and fall capacitance. The state of side inputs for the simulations was determined from the `three_state` attribute and otherwise set arbitrarily. This could in some cases result in a measurement while the pin was enabled as an output which would give a very large value.

The new behavior is to not generate the extra simulations if input capacitance is already available from hidden or delay measurements. If such measurements are not available, as for tristate outputs, then any tristate enable measurements are used to determine a state to measure in which the output is hi-Z. Only if no tristate enable measurement is specified is the original construction used.

user_data_keep_simple_attr_quotes

| | |
|------------------------------|---|
| <code>< 0 1 ></code> | Specifies whether the quoted string value that defines the attribute name in the user data should be wrapped within quotes. Default: 1 |
| 0 | Old behavior for backward compatibility. Liberate reads in the value of an attribute from the user_data file and removes the quotes in the internal database. |
| 1 | If the value of an attribute in the user_data file has quotes, the attribute in the new library will also have quotes. |

If `user_data_quote_attr` includes the attribute name and/or the `user_data_quote_simple_attr` is set to 1, the attribute value is enclosed within quotes. This happens even if the value of the attribute has no quotes in the user_data file.

This variable must be set before the `write_library` command.

write_logic_function_async_mode

| | |
|------------------------------|---|
| <code>< 0 1 ></code> | Controls whether an arc with a <i>timing_type</i> of <i>preset</i> and <i>clear</i> should be changed to a <i>timing_type</i> of <i>combinational</i> based on the value to which the <code>write_logic_function</code> variable is set. Default: 1 |
| 0 | When the <code>write_logic_function</code> variable is set to 0, the arcs with a <i>timing_type</i> of <i>preset</i> and <i>clear</i> will be changed to a <i>timing_type</i> of <i>combinational</i> . Use this setting for backward compatibility to LIBERATE13.1 and prior releases. |
| 1 | The <i>timing_type</i> of <i>preset</i> and <i>clear</i> will not be affected by the value set for the <code>write_logic_function</code> variable. |

This variable must be set before the `write_library` command.

voltage_map_ldb_char_mode

| | |
|-----------|---|
| < 0 1 > | Corrects erroneous supply name in library. Default: 1 |
| 0 | Backward compatible to version 3.2p3 and earlier. |
| 1 | Corrects issue where default supply name was erroneously written to library when voltage_map=2. (Default) |

Applicable to circuits that do *not* use VDD, while running `read_library` / `write_library` flow.

This variable must be set before the `write_library` command.

Legacy Debug Variables

These variables may or may not work. This is just a list of variables that were used for development or debug purposes. **Use at your own risk!**

ccs_retry_info

`< 0 | 1 >` Enable printing of CCS retry criteria details. Default: 0 (don't print)

Setting this variable to a 1 will turn on printing of CCS retry criteria information into the log file. The information printed will include the cell name, pin, related pin, WHEN condition, transition and criteria. This option has no effect if `ccs_retry_mode=0`.

Example:

```
*Warning* (char_library) Arc cell=IIND4D2LVT pin=ZN related_pin=A2 when=fall_transition have the following issues:
```

```
Failed ccs_voltage_tail_tol criteria
```

This variable must be set before the `char_library` command.

ccs_retry_mode

`< 0 | 1 | 2 | 3 >` Enables CCS simulation retry methodologies. Default: 0 (Disabled)

- | | |
|---|--|
| 0 | Disabled. |
| 1 | Check <code>ccs_retry_multi_peak_tol</code> criteria and <code>set_ccs_retry_thresholds</code> against <code>ccs_abs_tol</code> and <code>ccs_rel_tol</code> values. |
| 2 | Check criteria from option 1 and <code>ccs_retry_voltage_tail_tol</code> . |
| 3 | Check <code>ccs_retry_voltage_tail_tol</code> criteria only. |

In certain cases the SPICE simulation options specified through `alspice_option`, `extsim_option`, or `extsim_ccs_option` may not have sufficient accuracy or can result in trapezoidal oscillation in the current waveform. Certain versions of third-party downstream

tools may poorly handle an `output_current` waveform with this oscillation, resulting in correlation errors.

Liberate can automatically detect these oscillations and re-simulate with higher accuracy SPICE settings. This variable controls the methodology and criteria used for detecting failures.

Note that enabling this methodology will have an impact on runtime and will likely result in many re-simulations. If used, only option 3 is advised. Prior to enabling this functionality, it is recommended to use the following options and check for CCS correlation:

- `ccs_init_voltage_comp_thresh 1.1`
- `ccs_voltage_tail_tol 0.951`
- `ccs_voltage_tail_trim_tol 0.999`

Depending on the version or settings of your versions of Library Compiler and STA tools, setting `ccs_voltage_tail_tol` to 0.981 might achieve better results.

See also:

- `ccs_retry_info`
- `ccs_retry_multi_peak_tol`
- `extsim_ccs_retry_option`
- `extsim_ccs_retry_tran_append`
- `set_ccs_retry_thresholds`

The default and recommended setting is 0.

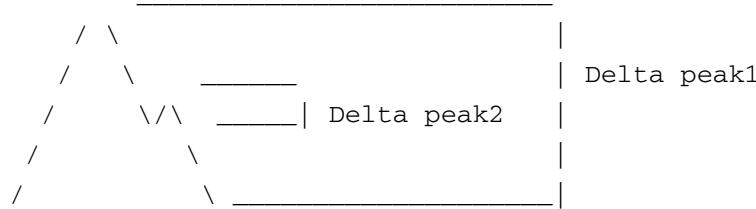
This variable must be set before the `char_library` command.

`ccs_retry_multi_peak_tol`

`<value>` Threshold for CCS waveform multi-peak failure criteria. Default: 0.05 (5% of peak)

If `ccs_retry_mode` is enabled, then Liberate will evaluate the CCS current waveform for multiple peaks. If multiple peaks are detected, then Liberate find the height of each peak. If the ratio between peaks is less than this variable, then this peak is not flagged for failing criteria.

ccs_retry_multi_peak_tol : 2% tolerance for multi-peak. If $(\text{delta peak2}) / (\text{delta peak1}) < 2\%$, this is not a failure:



The default and recommended setting is 0.05

This variable must be set before the `char_library` command.

ccs_retry_voltage_tail_tol

`<value>`

The stopping point as a ratio of supply of the CCS waveform for CCS retry criteria. Default: 0.981 (follow `ccs_voltage_tail_tol`)

For CCS timing waveform data, if `ccs_retry_mode` 2 or 3 are enabled then the tail of the integrated $v(t)$ obtained from the sampled $i(t)$ is checked against `ccs_retry_voltage_tail_tol` * `supply_swing`. Failing checks result in simulation retries with higher accuracy SPICE options. Default: 0.981 (follow `ccs_voltage_tail_tol`; output must swing to within 0.019% of supply).

This variable must be set before the `char_library` command.

Example:

```
set_var ccs_retry_voltage_tail_tol 0.951
```

extsim_ccs_retry_option

`<"options">`

Options to be used for CCS timing re-characterization with external SPICE. Default: set to `extsim_ccs_option`

This parameter specifies the list of options to be used by the external SPICE simulator when characterizing CCS timing. This command only applies when the `ccs` argument is used with `char_library` and when `ccs_retry_mode>0`.

This option set will supersede any settings in `extsim_ccs_option`. If used, this variable should contain options that will yield higher accuracy and control trapezoidal oscillation

present in a current waveform in a SPICE simulation. The option string is passed as an .option line in the SPICE decks Liberate creates for characterization.

See also `ccs_retry_mode` and `extsim_ccs_retry_tran_append`.

This variable must be set before the `char_library` command.

Example:

```
# Set SPICE options for CCS retry simulations
set_var extsim_ccs_retry_option "runlvl=6 accurate method=BDF"
```

`extsim_ccs_retry_tran_append`

`<"options">` Additional options to append to `.tran`. Default: `""` (none)

This parameter is used to add extra options to the `.tran` statement during `ccs_retry` simulations. If Spectre is used, it is recommended to set this option.

See also `ccs_retry_mode` and `extsim_ccs_retry_option`.

This variable must be set before the `char_library` command.

Example:

```
# Set conservative mode for Spectre for CCS retry simulations
set_var extsim_ccs_retry_tran_append "errpreset=conservative"
```

Virtuoso Liberate Reference Manual
Deprecated and Legacy Commands and Variables

Variables to Use When Qualifying and Migrating Between Versions

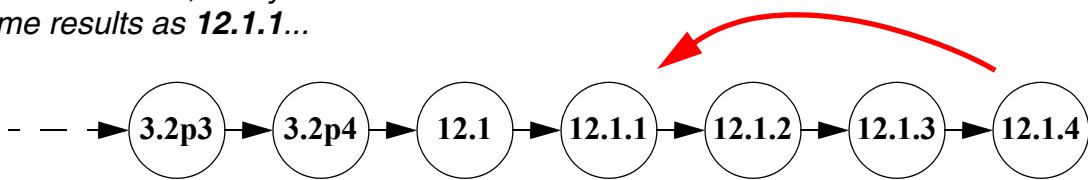
These variables allow you to run your current version of Liberate and get the same results as an earlier version.

The variables are presented as a Tcl script (`set_var` commands), and are organized to show which ones are needed to step back to a particular version. To use, include in your script all the `set_var`'s that go back to the version you need. (See example.)

Note: Default settings are generally changed to comply with updated specifications, requirements from downstream tools, or to correct significant issues. These backwards-compatible settings are provided to make migration easier, but we strongly recommend using the new defaults after updating to the new version.

Example:

You have **12.1.4**, and you want the same results as **12.1.1**...



Use these `set_var`'s in your script.

```
# 12.1.4 to 12.1.3
set_var constraint_bisection_mode 1
set_var constraint_search_bound_estimation_mode 1
...
# 12.1.3 to 12.1.2
set_var extsim_model_include_mode 1
...
# 12.1.2 to 12.1.1
set_var ccsp_segmentation_effort 0
set_var init_constraint_period_binning_mode 0
...
# 12.1.1 to 12.1
set_var ccs_segmentation_effort 0x1
set_var ccsn_compatibility_multi_corners 0
...
```

Virtuoso Liberate Reference Manual

Variables to Use When Qualifying and Migrating Between Versions

```
# 12.1.4 to 12.1.3
set_var constraint_bisection_mode 1
set_var constraint_search_bound_estimation_mode 1
set_var def_arc_constraint_use_arc_when true
set_var default_power_subtract_hidden_mode 1
set_var variety_netlist_mode 0

# 12.1.3 to 12.1.2.3
set_var extsim_model_include_mode 1
set_var mx_include_pull_driver false

# 12.1.2.3 to 12.1.2.2
# 12.1.2.2 to 12.1.2.1
# 12.1.2.1 to 12.1.2
# 12.1.2 to 12.1.1
set_var ccsp_segmentation_effort 0
set_var init_constraint_period_binning_mode 0
set_var variation_random_tran_thresh 0.5

# 12.1.1 to 12.1
set_var ccs_segmentation_effort 0x1
set_var ccsn_compatibility_multi_corners 0
set_var ccsn_dc_estimate_mode 0
set_var mx_greybox_constraint_method 0
set_var ski_elbr_mode 0

# 12.1 to 3.2p4_lcs
set_var bundle_log_filename ""
set_var ccs_cap_hidden_pin 1
set_var ecm_cap_mode 0
set_var mx_leakage_check_time false
set_var mx_mpw_allow_same_probe_on_both_rise_and_fall_clock_tree true
set_var syscall_mode 0

# 3.2p4_lcs to 3.2p3
set_var aocv_derate_fit 1
set_var aocv_derate_mode defaultAVG
set_var ccsn_allow_multiple_input_switching 0
set_var ccsn_io_handle_tristate 1
set_var delay_glitch_detection_thresh 0.001
set_var read_library_expand_bundles false
```

Virtuoso Liberate Reference Manual

Variables to Use When Qualifying and Migrating Between Versions

```
set_var spice_param_eval_mode 1
set_var syscall_mode 2

# 3.2p3 to 3.2p2
# 3.2p2 to 3.2p1_lcs
set_var mx_postprocess_clock_probes true

# 3.2p1_lcs to 3.2
set_var mx_fastsim_reuse false
set_var mx_leakage_check_window 1
set_var mx_margin_report 0
set_var mx_power_assign " clock "
set_var mx_verbose false

# 3.2 to 3.1p4
set_var constraint_output_load "min"
set_var mx_rcdb_use_fastsim_deck false
set_var use_altos_default_group true

# 3.1p4 to 3.1p3
set_var constraint_max_risefall false

# 3.1p3 to 3.1p2
set_var driver_cell_acc_mode 0
set_var leakage_fix_ccsp true
set_var mx_hold_comb " none "
set_var mx_seq_probing "state output internal"
set_var mx_setup_comb " none "

# 3.1p2 to 3.1p1
set_var mx_whitebox_monitor_memcore false
set_var tryAsFloat 0

# 3.1p1 to 3.1
set_var aocv_nominal_swap_mode 0
set_var ccs_cap_use_input_transition_tristate 0
set_var mx_dynamic_include_full_core false
set_var mx_full_rail_tol 1e-2
set_var rc_compatibility_mode_3_0 true
```

Virtuoso Liberate Reference Manual

Variables to Use When Qualifying and Migrating Between Versions

```
# 3.1 to 3.0p3
set_var als spice_diode false
set_var ccsn_arc_channel_check 0
set_var mx_fullsim_measurement false
set_var mx_mpw_mode -1
set_var tcl_new_source false
set_var variation_ecsm_cap_input_pin false

# 3.0p3 to 3.0p2
# 3.0p2 to 3.0p1
# 3.0p1 to 3.0
set_var switch_cell_powerdown_function 1

# 3.0 to 2.5p2a
set_var char_mos_term_cap false
set_var combo_period 10e-9
set_var constraintCombinatorial 1
set_var extsim_sanitize_param_name false
set_var init_num_period 0
set_var library_copyright 0
set_var library_revision 0
set_var lic_queue_timeout -1
set_var miller_cap_factor 0.0
set_var mx_allow_blobs true
set_var mx_dpartition_inactive_tie "all"
set_var mx_false_probe_keep " none "
set_var mx_ring_max_xtr_cnt 1000000
set_var mx_ring_model_fold true
set_var psp_model_enable true
set_var switch_cell_powerdown_function 0

# 2.5p2a to 2.5p2a
# 2.5p2a to 2.5p1
set_var ccs_segmentation_effort 1

# 2.5p1 to 2.5
# 2.5 to 2.4p4
set_var ccs_init_voltage_comp_thresh -1
set_var ccs_segmentation_effort 0
set_var conditional_leakage false
set_var mx_char_virtual_as_rail " none "
```

```
# 2.4p4 to 2.4p3
# 2.4p3 to 2.4p2
# 2.4p2 to 2.4p1
# 2.4p1 to 2.4p1
# 2.4p1 to 2.4
# 2.4 to 2.3p2
set_var ccs_base_curve_share_mode 0
set_var ccsp_cross_zero_compact true
set_var ccsp_rel_tol 0.05
set_var ccsp_tail_tol 0.01
set_var compact_ccs_va_digits 10000
set_var compact_ccs_va_reltol 0
set_var mx_seq_probing_max_in 10

# 2.3p2 to 2.3p1
# 2.3p1 to 2.3
set_var ccsn_pin_stage_merge_mode 1

# 2.3 to 2.2p2
set_var ccs_base_curve_points 15
set_var ccsn_pin_stage_merge_mode 0
set_var mx_hold_comb 0
set_var mx_hold_seq 1
set_var mx_seq_probing_check_function true
set_var mx_setup_comb 0
set_var mx_setup_seq 1

# 2.2p2 to 2.2p1
set_var ccsp_current_using_sum_abs_reltol false
set_var opt_max_iter 200
set_var opt_strategy 1

# 2.2p1 to 2.2
set_var accurate_ccs_variation 0
set_var ccs_rel_tol 0.02
set_var mismatch_threshold2 1.0
set_var non_seq_pin_swap 1
set_var opt_max_iter 1000
```

Virtuoso Liberate Reference Manual

Variables to Use When Qualifying and Migrating Between Versions

2.2 to 2.1p1

```
set_var mx_hold_comb true
set_var mx_setup_comb true
```

2.1p1 to 2.1

```
set_var ccsn_pin_stage_lshift 0
set_var constraint_search_time_reltol 0.1
set_var ibis_interpolate 1
```

2.1 to 2.0p3

```
set_var ibis_iv_step 0.15
set_var mx_whitebox_active_coupling_threshold 1e-13
set_var mx_whitebox_ring_in_depth 2
set_var mx_whitebox_ring_out_depth 2
```

2.0p3 to 2.0p2

```
set_var default_method 0
set_var mx_whitebox_active_sidebranch true
set_var server_timeout 1209600
```

2.0p2 to 2.0p1

```
set_var mpw_input_threshold 0.5
```

2.0p1 to 2.0

2.0 to 1.4p3

1.4p3 to 1.4p2

1.4p2 to 1.4p1

```
set_var mismatch_count1 -1
set_var res_tol_ratio 0.02
set_var unidirectional_tristate false
```

1.4p1 to 1.4

1.4 to 1.3p3

1.3p3 to 1.3p2

```
set_var binning_detail 2
set_var ccsn_allow_static 1
```

1.3p2 to 1.3p1.2

1.3p1.2 to 1.3p1.1

1.3p1.1 to 1.3p1

1.3p1 to 1.3

Virtuoso Liberate Reference Manual

Variables to Use When Qualifying and Migrating Between Versions

```
set_var max_bdd_size 10000

# 1.3 to 1.2p3
set_var constraint_clock_gater false

# 1.2p3 to 1.2p2
set_var mismatch_count1 5
set_var mismatch_threshold1 0.5

# 1.2p2 to 1.2p1
set_var def_arc_drive_side_bidi false
set_var extsim_check true
set_var non_linear_variation 0
set_var predriver_waveform_npts 18
set_var spectrespp true
set_var vector_limit 16

# 1.2p1 to 1.2
set_var ccs_min_pts 5
set_var floating_node_check false
set_var predriver_waveform_npts 0

# 1.2 to 1.1p9
set_var floating_node_check true
set_var immunity_search_voltage_abstol 2e-3
set_var predriver_waveform_npts 18
set_var rc_parviews false

# 1.1p9 to 1.1p8
# 1.1p8 to 1.1p7
set_var predriver_waveform_npts 0

# 1.1p7 to 1.1p6
set_var floating_node_check false

# 1.1p6 to 1.1p5
set_var floating_node_check true

# 1.1p5 to 1.1p4
set_var arc_partition 10
```

Virtuoso Liberate Reference Manual
Variables to Use When Qualifying and Migrating Between Versions

Glossary

Glossary of common industry terms, and other specialized terms used in this manual.

| | |
|-------------|--|
| AOCV | Advanced On-Chip Variation |
| BIST | Built-in Self Test |
| bsub | Batch Submission (part of LSF) |
| | |
| CCB | Channel Connected Blocks |
| CCC | Channel Connected Component (region) |
| CCR | Channel Connected Regions |
| CCS | Composite Current Source (Synopsys format) |
| CCSN | Composite Current Source, Noise |
| CCSP | Composite Current Source, Power |
| CCST | Composite Current Source, Timing |
| CSM | Current Source Models |
| | |
| DCOP | Desktop Communication Protocol |
| DFM | Design for Manufacturing |
| DSPF | Detailed Standard Parasitic Format |
| ECSM | Effective Current Source Model (Cadence format) |
| EIA | Electronics Industries Alliance (Standards body) |
| EMI | External_sim Model Include / Electro-magnetic Interference |
| IBIS | Input/output Buffer Information Specification |
| JMS | Job Management System |

Virtuoso Liberate Reference Manual

Glossary

| | |
|--------------|---|
| LSF | Load Sharing Facility (job scheduler) |
| MLD | Measurement Description Language (Spectre) |
| NLDM | Non-Linear Delay Model |
| | |
| OCV | On-Chip Variation |
| OMC | Open Modeling Coalition |
| PCR | Program Change Request |
| PVT | Process, Voltage, Temperature (Corner analysis) |
| PWL | Piece-Wise Linear |
| qsub | Queue Submission |
| | |
| SDF | Standard Delay Format |
| SGE | Sun Grid Engine (job scheduler) |
| SI | Signal Integrity |
| SKI | Spectre Kernal Interface |
| SPEF | Standard Parasitic Extraction Format |
| SSTA | Statistical Static Timing Analysis |
| STA | Static Timing Analysis |
| TCAM | Ternary Content-Addressable Memory |
| VCS | Verilog Compiled-code Simulator |
| VHDL | VHSIC Hardware Description Language |
| VHSIC | Very High-Speed (Scale?) Integrated Circuit |
| VITAL | VHDL Initiative Towards ASIC Libraries |