Libero® SoC v2021.1

Tcl Command Reference Guide for SmartFusion®2, IGLOO®2, and RTG4™

Introduction

Tcl, the Tool Command Language, pronounced `tickle`, is an easy-to-learn scripting language that is compatible with Libero® SoC software. You can run scripts from either the Windows or Linux command line or store and run a series of commands in a `*.tcl` batch file.

Libero SoC provides additional capabilities and built-in Tcl Commands:

- Running Tcl scripts from the command line.
- Exporting Tcl scripts.
- Extended run lib
- Tcl Commands, as specified in this document

Table of Contents

Introduction....1

1. Introduction to Tcl Scripting....11
   - 1.1. Tcl Commands and Supported Families....11
   - 1.2. Tcl Command Documentation Conventions....11
   - 1.3. Basic Syntax....12
   - 1.4. Types of Tcl Commands....14
   - 1.5. Variables....14
   - 1.6. Command Substitution....15
   - 1.7. Quotes and Braces....15
   - 1.8. Filenames....16
   - 1.9. Lists and Arrays....16
   - 1.10. Control Structures....17
   - 1.11. Print Statement and Return Values....18
   - 1.12. Running Tcl Scripts from the Command Line....18
   - 1.13. Exporting Tcl Scripts....19
   - 1.14. extended_run_lib....20
   - 1.15. Sample Tcl Script - Project Manager....22
   - 1.16. How to Derive Required Part Information from A "Part Number"....23

2. Project Manager Tcl Commands....24
   - 2.1. add_file_to_library....24
   - 2.2. add_library....25
   - 2.3. add_modelsim_path....25
   - 2.4. add_profile....26
   - 2.5. associate_stimulus....28
   - 2.6. build_design_hierarchy....29
   - 2.7. change_link....29
   - 2.8. change_vault_location....30
   - 2.9. check_fdc_constraints....31
   - 2.10. check_hdl....31
   - 2.11. check_ndc_constraints....32
   - 2.12. check_pdc_constraints....33
   - 2.13. check_sdc_constraints....33
   - 2.14. close_design....34
   - 2.15. close_project....35
   - 2.16. configure_core....36
   - 2.17. configure_tool....37
   - 2.18. create_and_configure_core....38
   - 2.19. create_links....40
   - 2.20. create_set....42
   - 2.21. create_smartdesign....43
   - 2.22. delete_component....44
   - 2.23. download_core....45
   - 2.24. download_latest_cores....46
   - 2.25. edit_profile....46

2.26. export_as_link....47
2.27. export_ba_files....48
2.28. export_bitstream_file....49
2.29. export_bsdl_file....56
2.30. export_component_to_tcl....57
2.31. export_design_summary....58
2.32. export_firmware....59
2.33. export_fp_pdc....60
2.34. export_ibis_file....61
2.35. export_io_pdc....62
2.36. export_job_data....63
2.37. export_netlist_file....64
2.38. export_pin_reports....65
2.39. export_profiles....66
2.40. export_prog_job....67
2.41. export_script....70
2.42. export_sdc_file....71
2.43. generate_component....72
2.44. generate_sdc_constraint_coverage....73
2.45. get_libero_release....74
2.46. get_libero_version....74
2.47. get_tool_options....75
2.48. get_tool_state....78
2.49. import_component....80
2.50. import_component_data....80
2.51. import_files....82
2.52. loopback_test....85
2.53. new_project....86
2.54. open_project....90
2.55. open_smartdesign....91
2.56. organize_constraints....92
2.57. organize_sources....93
2.58. organize_tool_files....95
2.59. prbs_test....96
2.60. project_settings....97
2.61. publish_block....100
2.62. refresh....102
2.63. remove_core....102
2.64. remove_library....103
2.65. remove_profile....104
2.66. rename_file....105
2.67. rename_library....106
2.68. run_tool....106
2.69. save_log....109
2.70. save_project....110
2.71. save_project_as....111
2.72. save_smartdesign....112
2.73. select_libero_design_device....113

2.74. select_profile 113
2.75. set_actel_lib_options 114
2.76. set_as_target 115
2.77. set_device 116
2.78. set_live_probe 118
2.79. set_modelsim_options 119
2.80. set_option 122
2.81. set_root 123
2.82. set_user_lib_options 124
2.83. unlink_files 125
2.84. unset_as_target 126
2.85. use_source_file 127

3. SmartDesign Tcl Commands 128
3.1. sd_add_pins_to_group 128
3.2. sd_clear_pin_attributes 128
3.3. sd_configure_core_instance 129
3.4. sd_connect_instance_pins_to_ports 129
3.5. sd_connect_net_to_pins 130
3.6. sd_connect_pins_to_constant 131
3.7. sd_connect_pin_to_port 131
3.8. sd_connect_pins 132
3.9. sd_create_bif_net 133
3.10. sd_create_bif_port 133
3.11. sd_create_bus_net 135
3.12. sd_create_bus_port 135
3.13. sd_create_pin_group 136
3.14. sd_create_pin_slices 137
3.15. sd_create_scalar_net 137
3.16. sd_create_scalar_port 138
3.17. sd_delete_instances 138
3.18. sd_delete_nets 139
3.19. sd_delete_pin_group 139
3.20. sd_delete_pin_slices 140
3.21. sd_delete_ports 140
3.22. sd_disconnect_instance 141
3.23. sd_disconnect_pins 141
3.24. sd_duplicate_instance 142
3.25. sd_hide_bif_pins 142
3.26. sd_instantiate_component 143
3.27. sd_instantiate_core 144
3.28. sd_instantiate_hdl_core 144
3.29. sd_instantiate_hdl_module 145
3.30. sd_instantiate_macro 145
3.31. sd_invert_pins 146
3.32. sd_mark_pins_unused 146
3.33. sd_remove_pins_from_group 147
3.34. sd_rename_instance 147

3.35. sd_rename_net ... 148
3.36. sd_rename_pin_group ... 149
3.37. sd_rename_port ... 149
3.38. sd_save_core_instance_config ... 150
3.39. sd_show_bif_pins ... 150
3.40. sd_update_instance ... 151

4. HDL Tcl Commands ... 152
4.1. create_hdl_core ... 152
4.2. remove_hdl_core ... 153
4.3. hdl_core_add_bif ... 154
4.4. hdl_core_assign_bif_signal ... 155
4.5. hdl_core_delete_parameters ... 156
4.6. hdl_core_extract_ports_and_parameters ... 157
4.7. hdl_core_remove_bif ... 159
4.8. hdl_core_rename_bif ... 160
4.9. hdl_core_unassign_bif_signal ... 161

5. Command Tools ... 163
5.1. CONFIGURE_ACTIONS_PROCEDURES ... 163
5.2. CONFIGURE_CHAIN ... 163
5.3. CONFIGURE_PROG_OPTIONS ... 164
5.4. CONFIGURE_PROG_OPTIONS_RTG4 (RTG4 only) ... 165
5.5. FLASH_FREEZE ... 167
5.6. EXPORTNETLIST ... 167
5.7. EXPORTSDF ... 168
5.8. GENERATEPROGRAMMINGDATA ... 168
5.9. GENERATEPROGRAMMINGFILE ... 168
5.10. INIT_LOCK ... 168
5.11. PROGRAMDEVICE ... 169
5.12. PLACEROUTE ... 170
5.13. PROGRAM_OPTIONS (SmartFusion2 and IGLOO2) ... 173
5.14. PROGRAM_RECOVERY ... 174
5.15. PROGRAMMER_INFO ... 175
5.16. SIM_POSTLAYOUT ... 176
5.17. SPM ... 177
5.18. SYNTHESIZE ... 181
5.19. USER_PROG_DATA (SmartFusion2, IGLOO2) ... 183
5.20. VERIFYPOWER ... 184
5.21. VERIFYTIMING ... 185

6. MSS Tcl Commands ... 186
6.1. mss_configure_envm ... 186
6.2. mss_configure_instance ... 186
6.3. mss_disable_instance ... 187
6.4. mss_enable_instance ... 187
6.5. mss_save_instance_config ... 188

7. SmartTime Tcl Commands ... 189

7.1. all_inputs....189
7.2. all_outputs....190
7.3. all_registers....190
7.4. check_constraints....192
7.5. clone_scenario....192
7.6. create_clock....193
7.7. create_generated_clock....195
7.8. create_scenario....198
7.9. create_set....199
7.10. expand_path....201
7.11. get_cells....203
7.12. get_clocks....204
7.13. get_current_scenario....205
7.14. get_nets....206
7.15. get_pins....207
7.16. get_ports....208
7.17. list_clock_groups....209
7.18. list_clock_latencies....210
7.19. list_clock_uncertainties....211
7.20. list_clocks....211
7.21. list_disable_timings....212
7.22. list_false_paths....213
7.23. list_generated_clocks....214
7.24. list_input_delays....214
7.25. list_max_delays....215
7.26. list_min_delays....216
7.27. list_multicycle_paths....217
7.28. list_objects....217
7.29. list_output_delays....218
7.30. list_paths....219
7.31. list_scenario....221
7.32. read_sdc....222
7.33. remove_all_constraints....223
7.34. remove_clock....223
7.35. remove_clock_groups....225
7.36. remove_clock_latency....226
7.37. remove_clock_uncertainty....227
7.38. remove_disable_timing....229
7.39. remove_false_path....230
7.40. remove_generated_clock....232
7.41. remove_input_delay....233
7.42. remove_max_delay....234
7.43. remove_min_delay....235
7.44. remove_multicycle_path....236
7.45. remove_output_delay....238
7.46. remove_scenario....239
7.47. remove_set....240
7.48. rename_scenario....241

7.49. report....241
7.50. save....245
7.51. set_clock_groups....246
7.52. set_clock_latency....248
7.53. set_clock_to_output....249
7.54. set_clock_uncertainty....250
7.55. set_current_scenario....253
7.56. set_disable_timing....254
7.57. set_external_check....255
7.58. set_false_path....256
7.59. set_input_delay....258
7.60. set_max_delay....260
7.61. set_min_delay....262
7.62. set_multicycle_path....264
7.63. set_options....266
7.64. set_output_delay....269
7.65. write_sdc....272

8. SmartPower Tcl Commands....273
8.1. smartpower_add_new_scenario....273
8.2. smartpower_add_pin_in_domain....273
8.3. smartpower_battery_settings....274
8.4. smartpower_change_clock_statistics....274
8.5. smartpower_change_setofpin_statistics....275
8.6. smartpower_commit....276
8.7. smartpower_compute_vectorless....276
8.8. smartpower_create_domain....276
8.9. smartpower_edit_scenario....277
8.10. smartpower_export_mpe_report....277
8.11. smartpower_import_vcd....278
8.12. smartpower_init_do....279
8.13. smartpower_init_set_clocks_options....281
8.14. smartpower_init_set_combinational_options....282
8.15. smartpower_init_set_enables_options....282
8.16. smartpower_init_set_primaryinputs_options....283
8.17. smartpower_init_set_registers_options....283
8.18. smartpower_init_setofpins_values....284
8.19. smartpower_remove_all_annotations....284
8.20. smartpower_remove_file....285
8.21. smartpower_remove_scenario....285
8.22. smartpower_report_power....286
8.23. smartpower_set_mode_for_pdpr....292
8.24. smartpower_set_operating_condition....293
8.25. smartpower_set_operating_conditions....293
8.26. smartpower_set_process....294
8.27. smartpower_set_temperature_opcond....294
8.28. smartpower_set_thermalmode....295
8.29. smartpower_set_voltage_opcond....295

8.30. smartpower_temperature_opcond_set_design_wide....296
8.31. smartpower_temperature_opcond_set_mode_specific....297
8.32. smartpower_voltage_opcond_set_design_wide....298
8.33. smartpower_voltage_opcond_set_mode_specific....299

9. FlashPro Express Tcl Commands....301
9.1. close_project....301
9.2. configure_flashpro3_prg....301
9.3. configure_flashpro4_prg....303
9.4. configure_flashpro5_prg....304
9.5. configure_flashpro6_prg....305
9.6. create_job_project....306
9.7. dump_tcl_support....307
9.8. enable_prg....308
9.9. enable_prg_type....309
9.10. export_script....310
9.11. open_project....311
9.12. ping_prg....312
9.13. refresh_prg_list....313
9.14. remove_prg....313
9.15. run_selected_actions....314
9.16. save_log....316
9.17. save_project....316
9.18. scan_chain_prg....317
9.19. self_test_prg....318
9.20. set_prg_name....319
9.21. set_programming_action....320
9.22. set_programming_file....321
9.23. set_programming_interface....322
9.24. set_spi_flash_action....323
9.25. set_spi_flash_file....324

10. SmartDebug Tcl Commands....326
10.1. add_probe_insertion_point....326
10.2. add_to_probe_group....326
10.3. construct_chain_automatically....326
10.4. create_probe_group....327
10.5. delete_active_probe....327
10.6. enable_device....327
10.7. event_counter....328
10.8. export_smart_debug_data....328
10.9. fhb_control....330
10.10. frequency_monitor....332
10.11. get_programmer_info....332
10.12. load_active_probe_list....332
10.13. loopback_mode....332
10.14. move_to_probe_group....333
10.15. optimize_dfe....333

10.16. pcie_config_space 333
10.17. pcie_ltssm_status 334
10.18. plot_eye 334
10.19. program_probe_insertion 335
10.20. read_active_probe 335
10.21. read_lsram 335
10.22. read_usram 336
10.23. remove_from_probe_group 337
10.24. remove_probe_insertion_point 337
10.25. run_selected_actions 338
10.26. save_active_probe_list 338
10.27. scan_chain_prg 338
10.28. scan_ecc_memories 338
10.29. select_active_probe 339
10.30. set_live_probe 339
10.31. set_debug_programmer 340
10.32. set_programming_action 340
10.33. set_programming_file 341
10.34. smartbert_test 341
10.35. static_pattern_transmit 342
10.36. ungroup 343
10.37. unset_live_probe 343
10.38. uprom_read_memory 344
10.39. write_active_probe 344
10.40. write_lsram 345
10.41. write_usram 346
10.42. xcvr_read_register 347
10.43. xcvr_write_register 349

11. Configure JTAG Chain Tcl Commands 352
11.1. add_actel_device 352
11.2. add_non_actel_device 352
11.3. add_non_actel_device_to_database 353
11.4. construct_chain_automatically 353
11.5. copy_device 353
11.6. cut_device 354
11.7. enable_device 354
11.8. paste_device 355
11.9. remove_device 355
11.10. remove_non_actel_device_from_database 355
11.11. select_libero_design_device 356
11.12. set_bsdl_file 356
11.13. set_device_ir 357
11.14. set_device_name 357
11.15. set_device_order 357
11.16. set_device_tck 358
11.17. set_device_type 358
11.18. set_programming_action 359

11.19. set_programming_file....359

12. System Builder Tcl Commands....360
12.1. is_synthesis_enabled....360
12.2. sb_configure_page....360
12.3. sb_add_core....361
12.4. sb_configure_core....361
12.5. sb_configure_core_count....362
12.6. sb_move_core....363
12.7. sb_enable_core....364
12.8. sb_disable_core....364
12.9. sb_enable_peripheral....364
12.10. sb_disable_peripheral....365
12.11. sb_configure_peripheral....365
12.12. sb_set_fic_direct_mode....366
12.13. sb_configure_envm....366
12.14. open_sb_component....367
12.15. generate_sb_component....367
12.16. close_sb_component....368

13. Revision History....369

14. Microchip FPGA Technical Support....370
14.1. Customer Service....370
14.2. Customer Technical Support....370
14.3. Website....370
14.4. Outside the U.S....370

The Microchip Website....371

Product Change Notification Service....371

Customer Support....371

Microchip Devices Code Protection Feature....371

Legal Notice....372

Trademarks....372

Quality Management System....373

Worldwide Sales and Service....374

1. Introduction to Tcl Scripting

This section provides a quick overview of the main features of Tcl.

- 1.3 Basic Syntax
- 1.4 Types of Tcl Commands
- 1.5 Variables
- 1.6 Command Substitution
- 1.7 Quotes and Braces
- 1.8 Filenames
- 1.9 Lists and Arrays
- 1.10 Control Structures
- 1.11 Print Statement and Return Values

For complete information on Tcl scripting, refer to one of the books available on this subject. You can also find information about Tcl at web sites such as http://www.tcl.tk.

1.1 Tcl Commands and Supported Families

When we specify a family name, we refer to the device family and all its derivatives, unless otherwise specified. See Supported Families in the Tcl command help topics for the families supported for a specific Tcl command.

1.2 Tcl Command Documentation Conventions

The following table shows the typographical conventions used for the Tcl command syntax.

Syntax Notation	Description
`command -argument`	Commands and arguments appear in `Courier New` typeface.
`variable`	Variables appear in `Courier New` typeface. You must substitute an appropriate value for the variable.
[-argument value] [variable]+	Optional arguments begin and end with a square bracket with one exception: if the square bracket is followed by a plus sign (+), then users must specify at least one argument. The plus sign (+) indicates that items within the square brackets can be repeated. Do not enter the plus sign character.

Note: All Tcl commands are case sensitive. However, their arguments are not.

1.2.1 Examples

Syntax for the `get_clocks` command followed by a sample command:

```
get_clocks variable
```

For example, `get_clocks clk1`.

Syntax for the `backannotate` command followed by a sample command:

```
backannotate -name file_name -format format_type -language language -dir directory_name [-
netlist] [-pin]
```

For example, `backannotate -dir {..\design} -name "fanouttest_ba.sdf" -format "SDF" -language "VERILOG" -netlist`.

1.2.2 Wildcard Characters

You can use the following wildcard characters in names used in Tcl commands:

Wildcard	What it Does
\	Interprets the next character literally.
?	Matches any single character.
*	Matches any string.
[]	Matches any single character among those listed between brackets (that is, [A-Z] matches any single character in the A-to-Z range).

Note: The matching function requires that you add a slash (\) before each slash in the port, instance, or net name when using wildcards in a PDC command. For example, if you have an instance named “A/B12” in the netlist, and you enter that name as “A\\/B*” in a PDC command, you will not be able to find it. In this case, you must specify the name as A\\\\/B*.

1.2.3 Special Characters [ ], { }, and \

Sometimes square brackets ([ ]) are part of the command syntax. In these cases, you must either enclose the open and closed square brackets characters with curly brackets ({ }) or precede the open and closed square brackets ([ ]) characters with a backslash (\). If you do not, you will get an error message.

For example:

```
pin_assign -port {LFSR_OUT[0]} -pin 15
or
pin_assign -port LFSR_OUT\[0\] -pin 180
```

Note: Tcl commands are case sensitive. However, their arguments are not.

1.2.4 Entering Arguments on Separate Lines

To enter an argument on a separate line, you must enter a backslash (\) character at the end of the preceding line of the command as shown in the following example:

```
backannotate -dir \
{..\design} -name "fanouttest_ba.sdf" -format "SDF" -language "VERILOG" \
-netlist
```

See Also

- 1. Introduction to Tcl Scripting
- 1.3 Basic Syntax

1.3 Basic Syntax

Tcl scripts contain one or more commands separated by either new lines or semicolons. A Tcl command consists of the name of the command followed by one or more arguments. The format of a Tcl command is:

```
command arg1 ... argN
```

The command in the following example computes the sum of 2 plus 2 and returns the result, 4.

```
expr 2 + 2
```

The `expr` command handles its arguments as an arithmetic expression, computing and returning the result as a string. All Tcl commands return results. If a command has no result to return, it returns an empty string.

To continue a command on another line, enter a backslash (\) character at the end of the line. For example, the following Tcl command appears on two lines:

```
import -format "edif" -netlist_naming "Generic" -edif_flavor "GENERIC" {prepi.edn}
```

Comments must be preceded by a hash character (#). The comment delimiter (#) must be the first character on a line or the first character following a semicolon, which also indicates the start of a new line. To create a multi-line comment, you must put a hash character (#) at the beginning of each line.

Note: Be sure that the previous line does not end with a continuation character (\). Otherwise, the comment line following it will be ignored.

1.3.1 Special Characters

Square brackets (`[ ]`) are special characters in Tcl. To use square brackets in names such as port names, you must either enclose the entire port name in curly braces, for example, `pin_assign -port {LFSR_OUT[15]} -iostd lvttl -slew High`, or lead the square brackets with a slash (\) character as shown in the following example:

```
pin_assign -port LFSR_OUT\[15\] -iostd lvttl -slew High
```

1.3.2 Sample Tcl Script

```
#Create a new project and set up a new design
new_project -location {D:/2Work/my_pf_proj} -name {my_pf_proj} -project_description {}\
-block_mode 0 -standalone_peripheral_initialization 0 -use_enhanced_constraint_flow 1\
-hdl {VERILOG} -family {PolarFire} -die {MPF300TS_ES} -package {FCG1152} -speed {-1} \
-die_voltage {1.0} -part_range {EXT} -adv_options {IO_DEFT_STD:LVCMOS 1.8V}\
-adv_options {RESTRICTPROBEPINS:1} -adv_options {RESTRICTSPIPINS:0}\
-adv_options {SYSTEM_CONTROLLER_SUSPEND_MODE:1} -adv_options {TEMPR:EXT} \
-adv_options {VCCI_1.2_VOLTR:EXT} -adv_options {VCCI_1.5_VOLTR:EXT} \
-adv_options {VCCI_1.8_VOLTR:EXT} -adv_options {VCCI_2.5_VOLTR:EXT} \
-adv_options {VCCI_3.3_VOLTR:EXT} -adv_options {VOLTR:EXT}
#Import HDL source file
import_files -convert_EDN_to_HDL 0 -hdl_source {C:/test/prep1.v}
#Import HDL stimulus file
import_files -convert_EDN_to_HDL 0 -stimulus {C:/test/prep1tb.v}
#set the top level design name
set_root -module {prep1::work}
#Associate SDC constraint file to Place and Route tool
organize_tool_files -tool {PLACEROUTE} -file {D:/2Work/my_pf_proj/constraint/user.sdc} \
-module {prep1::work} -input_type {constraint}
#Associate SDC constraint file to Verify Timing tool
organize_tool_files -tool {VERIFYTIMING} -file {D:/2Work/my_pf_proj/constraint/user.sdc}\
-module {prep1::work} -input_type {constraint}
#Run synthesize
run_tool -name {SYNTHESIZE}
#Configure Place and Route tool
configure_tool -name {PLACEROUTE} -params {DELAY_ANALYSIS:MAX} -params {EFFORT_LEVEL:false}\
-params {INCRPLACEANDROUTE:false} -params {MULTI_PASS_CRITERIA:VIOLATIONS}\
-params {MULTI_PASS_LAYOUT:false} -params {NUM_MULTI_PASSES:5} -params {PDPR:false}\
-params {RANDOM_SEED:0} -params {REPAIR_MIN_DELAY:false} -params {SLACK_CRITERIA:WORST_SLACK}
\
-params {SPECIFIC_CLOCK:} -params {START_SEED_INDEX:1} -params {STOP_ON_FIRST_PASS:false}\
-params {TDPR:true}
#Run Place and Route
run_tool -name {PLACEROUTE}
#Configure Timing Report Generation
configure_tool -name {VERIFYTIMING} -run_tool -name {PLACEROUTE}params
{CONSTRAINTS_COVERAGE:1}\
-params {FORMAT:XML} -params {MAX_TIMING_FAST_HV_LT:0} -params {MAX_TIMING_SLOW_LV_HT:1}\
-params {MAX_TIMING_SLOW_LV_LT:0} -params {MAX_TIMING_VIOLATIONS_FAST_HV_LT:0} \
-params {MAX_TIMING_VIOLATIONS_SLOW_LV_HT:1} -params {MAX_TIMING_VIOLATIONS_SLOW_LV_LT:0}\
-params {MIN_TIMING_FAST_HV_LT:1} -params {MIN_TIMING_SLOW_LV_HT:0} -params
{MIN_TIMING_SLOW_LV_LT:0} -params {MIN_TIMING_VIOLATIONS_FAST_HV_LT:1} -params
{MIN_TIMING_VIOLATIONS_SLOW_LV_HT:0} \
-params {MIN_TIMING_VIOLATIONS_SLOW_LV_LT:0}
#Run Verify Timing tool
run_tool -name {VERIFYTIMING}
#Run Power Verification tool
run_tool -name {VERIFYPOWER} #Export bitstream
export_bitstream_file -file_name {prep1} \
```

```
-export_dir {D:\2Work\my_pf_proj\designer\prep1\export} -format {STP} -master_file 0 \
-master_file_components {} -encrypted_uek1_file 0 -encrypted_uek1_file_components {} \
-encrypted_uek2_file 0 -encrypted_uek2_file_components {} \
-trusted_facility_file 1 -trusted_facility_file_components {FABRIC}
```

1.4 Types of Tcl Commands

This section describes the following types of Tcl commands:

- 1.4.1 Built-in Commands
- 1.4.2 Procedures Created with the proc Command

1.4.1 Built-in Commands

Built-in commands are provided by the Tcl interpreter. They are available in all Tcl applications. Here are some examples of built-in Tcl commands:

- Tcl provides several commands for manipulating file names, reading and writing file attributes, copying files, deleting files, creating directories, and so on.
- exec - run an external program. Its return value is the output (on stdout) from the program, for example:

```
set tmp [ exec myprog ] puts stdout $tmp
```

- You can easily create collections of values (lists) and manipulate them in a variety of ways.
- You can create arrays - structured values consisting of name-value pairs with arbitrary string values for the names and values.
- You can manipulate the time and date variables.
- You can write scripts that can wait for certain events to occur, such as an elapsed time or the availability of input data on a network socket.

1.4.2 Procedures Created with the proc Command

You use the `proc` command to declare a procedure. You can then use the name of the procedure as a Tcl command.

The following sample script consists of a single command named `proc`. The `proc` command takes three arguments:

- The name of a procedure (`myproc`)
- A list of argument names (`arg1 arg2`)
- The body of the procedure, which is a Tcl script

```
proc myproc { arg1 arg2 } {
# procedure body
}
myproc a b
```

1.5 Variables

With Tcl scripting, you can store a value in a variable for later use. You use the set command to assign variables. For example, the following set command creates a variable named x and sets its initial value to 10.

```
set x 10
```

A variable can be a letter, a digit, an underscore, or any combination of letters, digits, and underscore characters. All variable values are stored as strings.

In the Tcl language, you do not declare variables or their types. Any variable can hold any value. Use the dollar sign (`$`) to obtain the value of a variable, for example:

```
set a 1
set b $a
set cmd expr
set x 11
$cmd $x*$x
```

The dollar sign $ tells Tcl to handle the letters and digits following it as a variable name and to substitute the variable name with its value.

Global Variables
Variables can be declared global in scope using the Tcl global command. All procedures, including the declaration can access and modify global variables, for example:

```
global myvar
```

1.6 Command Substitution

By using square brackets ([]), you can substitute the result of one command as an argument to a subsequent command, as shown in the following example:

```
set a 12
set b [expr $a*4]
```

Tcl handles everything between square brackets as a nested Tcl command. Tcl evaluates the nested command and substitutes its result in place of the bracketed text. In the example above, the argument that appears in square brackets in the second set command is equal to 48 (that is, 12* 4 = 48).

Conceptually,

```
set b [expr $a * 4]
```

expands to

```
set b [expr 12 * 4 ]
```

and then to

```
set b 48
```

1.7 Quotes and Braces

The distinction between braces ({ }) and quotes (" ") is significant when the list contains references to variables. When references are enclosed in quotes, they are substituted with values. However, when references are enclosed in braces, they are not substituted with values.

The following table lists an example code.

With Braces	With Double Quotes
set b 2	set b 2
set t { 1 $b 3 }	set t " 1 $b 3 "
set s { [expr $b + $b] }	set s " [expr $b + $b] "
puts stdout $t	puts stdout $t
puts stdout $s	puts stdout $s

The above example code will generate the following output:

```
1 $b 3 vs. 1 2 3
[ expr $b + $b ] 4
```

1.8 Filenames

In Tcl syntax, filenames should be enclosed in braces { } to avoid backslash substitution and white space separation. Backslashes are used to separate folder names in Windows-based filenames. The problem is that sequences of "\n" or "\t" are interpreted specially. Using the braces disables this special interpretation and specifies that the Tcl interpreter handle the enclosed string literally. Alternatively, double-backslash "\\n" and "\\t" would work as well as forward slash directory separators "/n" and "/t". For example, to specify a file on your Windows PC at `c:\newfiles\thisfile.adb`, use one of the following:

```
{C:\newfiles\thisfile.adb}
C:\\newfiles\\thisfile.adb
"C:\\newfiles\\thisfile.adb"
C:/newfiles/thisfile.adb
"C:/newfiles/thisfile.adb"
```

If there is white space in the filename path, you must use either the braces or double-quotes. For example:

```
C:\program data\thisfile.adb
```

should be referenced in Tcl script as

```
{C:\program data\thisfile.adb} or "C:\\program data\\thisfile.adb"
```

If you are using variables, you cannot use braces { } because, by default, the braces turn off all special interpretation, including the dollar sign character. Instead, use either double-backslashes or forward slashes with double quotes. For example:

```
"$design_name.adb"
```

Note: To use a name with special characters such as square brackets [ ], you must put the entire name between curly braces { } or put a slash character \ immediately before each square bracket.

The following example shows a port name enclosed with curly braces:

```
pin_assign -port {LFSR_OUT[15]} -iostd lvttl -slew High
```

The next example shows each square bracket preceded by a slash:

```
pin_assign -port LFSR_OUT\[15\] -iostd lvttl -slew High
```

1.9 Lists and Arrays

A list is a way to group data and handle the group as a single entity. To define a list, use curly braces { } and double quotes " ". For example, the following set command {1 2 3 }, when followed by the list command, creates a list stored in the variable "a". This list will contain the items "1," "2," and "3".

```
set a { 1 2 3 }
```

Here is another example:

```
set e 2
set f 3
set a [ list b c d [ expr $e + $f ] ] puts $a
```

displays (or outputs):

```
b c d 5
```

Tcl supports many other list-related commands such as `lindex`, `linsert`, `llength`, `lrange`, and `lappend`.

1.9.1 Arrays

An array is another way to group data. Arrays are collections of items stored in variables. Each item has a unique address that you use to access it. You do not need to declare them nor specify their size.

Array elements are handled in the same way as other Tcl variables. You create them with the set command, and you can use the dollar sign ($) for their values.

```
set myarray(0) "Zero" set myarray(1) "One" set myarray(2) "Two"
for {set i 0} {$i < 3} {incr i 1} {
```

Output:

```
Zero One Two
```

In the example above, an array called `myarray` is created by the set statement that assigns a value to its first element. The for-loop statement prints out the value stored in each element of the array.

1.9.2 Special Arguments (Command-Line Parameters)

You can determine the name of the Tcl script file while executing the Tcl script by referring to the `$argv0` variable.

```
puts “Executing file $argv0”
```

To access other arguments from the command line, you can use the `lindex` command and the `argv` variable: To read the Tcl file name:

```
lindex $argv 0
```

To read the first passed argument:

```
lindex $argv 1
```

For example:

```
puts "Script name is $argv0" ; # accessing the scriptname puts "first argument is [lindex
$argv 0]"
puts "second argument is [lindex $argv 1]" puts "third argument is [lindex $argv 2]" puts
"number of argument is [llength $argv]" set des_name [lindex $argv 0]
puts "Design name is $des_name"
```

1.10 Control Structures

Tcl control structures are commands that change the flow of execution through a script. These control structures include commands for conditional execution (if-then-elseif-else) and looping (while, for, catch).

An "if" statement only executes the body of the statement (enclosed between curly braces) if the Boolean condition is found to be true.

1.10.1 if/else Statements

```
if { “$name” == “paul” } then {
…
# body if name is paul
} elseif { $code == 0 } then {
…
# body if name is not paul and if value of variable code is zero
} else {
…
# body if above conditions is not true
}
```

1.10.2 for Loop Statement

A "for" statement will repeatedly execute the body of the code as long as the index is within a specified limit.

```
for { set i 0 } { $i < 5 } { incr i } {
…
# body here
}
```

1.10.3 while Loop Statement

A "while" statement will repeatedly execute the body of the code (enclosed between the curly braces) as long as the Boolean condition is found to be true.

```
while { $p > 0 } {
…
}
```

1.10.4 catch Statement

A "catch" statement suspends normal error handling on the enclosed Tcl command. If a variable name is also used, then the return value of the enclosed Tcl command is stored in the variable.

```
catch { open "$inputFile" r } myresult
```

1.11 Print Statement and Return Values

1.11.1 Print Statement

Use the puts command to write a string to an output channel. Predefined output channels are “stdout” and “stderr.” If you do not specify a channel, then puts display text to the stdout channel.

Note: The STDIN Tcl command is not supported by Microchip SoC tools.

For example:

```
set a [ myprog arg1 arg2 ]
```

puts "the answer from myprog was $a (this text is on stdout)" puts stdout “this text also is on stdout”.

1.11.2 Return Values

The return code of a Tcl command is a string. You can use a return value as an argument to another function by enclosing the command with square brackets [ ]. For example:

```
set a [ prog arg1 arg2 ] exec $a
```

The Tcl command “exec” will run an external program. The return value of “exec” is the output (on stdout) from the program. For example:

```
set tmp [ exec myprog ] puts stdout $tmp
```

1.12 Running Tcl Scripts from the Command Line

You can run Tcl scripts from your Windows or Linux command line as well as pass arguments to scripts from the command line.

Note: Tcl commands in this section are not case-sensitive.

To execute a Tcl script file in the Libero SoC Project Manager software from a shell command line:
At the prompt, type the path to the Microchip SoC software installation location followed by the word "SCRIPT" and a colon, and then the name of the script file as follows:

- For Linux: `<location of Libero SoC software installation>/bin/libero script:<filename>.tcl`. For example, to run the Tcl script file "myscript.tcl", type:

```
C:\libero\bin\libero script:myscript.tcl
```

- For Windows: `<location of Libero SoC software installation>\Designer\bin\libero.exe script:<filename>.tcl`. Where `<location of Microchip SoC software>` is the root directory in which you installed the Microchip SoC software, and `<filename>` is the name, including a relative or absolute path, of the Tcl script file to execute. For example, to run the Tcl script file "myscript.tcl", type:

```
C:/Microchip/Libero/Designer/bin/libero.exe script:myscript.tcl
```

To pass arguments from the command line to your Tcl script file:
At the prompt, type the path to the Microchip SoC software installation location followed by the SCRIPT argument:

- For Linux: <location of Microchip SoC software>\bin\libero “SCRIPT_ARGS:<filename arg1 arg2 ...>”. For example:

```
C:\libero\bin\libero SCRIPT:myscript.tcl “SCRIPT_ARGS:one two three”
```

- For Windows: <location of Microchip SoC software>/Designer/bin/libero.exe “SCRIPT_ARGS:<filename arg1 arg2 ...>”. Where <location of Microchip SoC software> is the root directory in which you installed the Microchip SoC software, and “SCRIPT_ARGS:<filename arg1 arg2 ...>” is the name, including a relative or absolute path, of the Tcl script file and arguments you are passing to the script file. For example:

```
C:/Microchip/Libero/Designer/bin/libero.exe SCRIPT:myscript.tcl “SCRIPT_ARGS:one two three”
```

To obtain the output from the log file:
At the prompt, type the path to the Microchip SoC software installation location followed by the SCRIPT, SCRIPT_ARGS and LOGFILE arguments.

```
<location of Microchip SoC software> SCRIPT:<filename>.tcl “SCRIPT_ARGS:a b c”
LOGFILE:<output.log>
```

Where:

- `location of Microchip SoC software` is the root directory in which you installed the Microchip SoC software.
- `SCRIPT:<filename>.tcl` is the name, including a relative or absolute path, of the Tcl script file.
- `SCRIPT_ARGS` are the arguments you are passing to the script file.
- `output.log` is the name of the log file.

For example:

```
C:\libero\designer\bin\libero SCRIPT:testTclparam.tcl “SCRIPT_ARGS:a b c”
LOGFILE:testTclparam.log
```

1.13 Exporting Tcl Scripts

You can write out a Tcl script file that contains the commands executed in the current session. You can then use this exported Tcl script to re-execute the same commands interactively or in batch. You can also use this exported script to become more familiar with Tcl syntax.

You can export Tcl scripts from the Project Manager.

To export a Tcl session script from the Project Manager:

1. From the **File** menu, choose **Export Script File**. The **Export Script** dialog box appears.
2. Click **OK**. The **Script Export Options** dialog box appears:
3. Check the **Include Commands from Current Design [Project] Only** checkbox. This option applies only if you opened more than one design or project in your current session. If so, and you do not check this box, Project Manager exports all commands from your current session.
4. Select the radio button for the appropriate filename formatting. To export filenames relative to the current working directory, select **Relative filenames (default)** formatting. To export filenames that include a fully specified path, select **Qualified filenames (full path; including directory name)** formatting.
5. Choose **Relative filenames** if you do not intend to move the Tcl script from the saved location, or **Qualified filenames** if you plan to move the Tcl script to another directory or machine.
6. Click **OK**. Project Manager saves the Tcl script with the specified filename.

   Notes:
   - When exporting Tcl scripts, Project Manager always encloses filenames in curly braces to ensure portability.
   - Libero SoC software does not write out any Tcl variables or flow-control statements to the exported Tcl file, even if you had executed the design commands using your own Tcl script. The exported Tcl file only contains the tool commands and their accompanying arguments.

1.14 extended_run_lib

Note: This is not a Tcl command; it is a shell script that can be run from the command line.

The `extended_run_lib` Tcl script enables you to run the multiple pass layout in batch mode from a command line.

```
$ACTEL_SW_DIR/bin/libero script:$ACTEL_SW_DIR/scripts/extended_run_lib.tcl
logfile:extended_run.log "script_args:-root path/designer/module_name [-n numPasses] [-
starting_seed_index numIndex] [-compare_criteria value] [-c clockName] [-analysis value] [-
slack_criteria value] [-stop_on_success] [-timing_driven|-standard] [-power_driven value] [-
placer_high_effort value]"
```

Note: There is no option to save the design files from all the passes. Only the (Timing or Power) result reports from all the passes are saved.

Arguments

Parameter	Value	Description
-root	path/designer/module_name	The path to the root module located under the designer directory of the Libero project.
[-n]	numPasses	Sets the number of passes to run. The default number of passes is 5.
[-starting_seed_index]	numIndex	Indicates the specific index into the array of random seeds which is to be the starting point for the passes. Value may range from 1 to 100. If not specified, the default behavior is to continue from the last seed index that was used.

.....continued

<table>
<tr><th>Parameter</th><th>Value</th><th>Description</th></tr>
<tr><td>[-compare_criteria]</td><td>value</td><td>
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>frequency</td><td>Use clock frequency as criteria for comparing the results between passes. This option can be used in conjunction with the -c option (described below).</td></tr>
<tr><td>violations</td><td>Use timing violations as criteria for comparing the results between passes. This option can be used in conjunction with the -analysis, -slack_criteria and - stop_on_success options (described below).</td></tr>
<tr><td>power</td><td>Use total power as criteria for comparing the results between passes, where lowest total power is the goal.</td></tr>
</table>
</td></tr>
<tr><td>[-c]</td><td>clockName</td><td>Applies only when the clock frequency comparison criteria is used. Specifies the particular clock that is to be examined. If no clock is specified, then the slowest clock frequency in the design in a given pass is used. The clock name should match with one of the Clock Domains in the Summary section of the Timing report.</td></tr>
<tr><td>[-analysis]</td><td>value</td><td>Applies only when the timing violations comparison criteria is used. Specifies the type of timing violations (the slack) to examine. The following table shows the acceptable values for this argument:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>max</td><td>Examines timing violations (slack) obtained from maximum delay analysis. This is the default.</td></tr>
<tr><td>min</td><td>Examines timing violations (slack) obtained from minimum delay analysis.</td></tr>
</table>
</td></tr>
<tr><td>[-slack_criteria]</td><td>value</td><td>Applies only when the timing violations comparison criteria is used. Specifies how to evaluate the timing violations (slack). The type of timing violations (slack) is determined by the -analysis option. The following table shows the acceptable values for this argument:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>worst</td><td>Sets the timing violations criteria to Worst slack. For each pass obtains the most amount of negative slack (or least amount of positive slack if all constraints are met) from the timing violations report. The largest value out of all passes will determine the best pass. This is the default.</td></tr>
<tr><td>tns</td><td>Sets the timing violations criteria to Total Negative Slack (tns). For each pass it obtains the sum of negative slack values from the first 100 paths from the timing violations report. The largest value out of all passes determines the best pass. If no negative slacks exist for a pass, then the worst slack is used to evaluate that pass.</td></tr>
</table>
</td></tr>
<tr><td>[-stop_on_success]</td><td></td><td>Applies only when the timing violations comparison criteria is used. The type of timing violations (slack) is determined by the <code>-analysis</code> option. Stops running the remaining passes if all timing constraints have been met (when there are no negative slacks reported in the timing violations report).</td></tr>
</table>

.....continued

Parameter	Value	Description
[-timing_driven|-standard]		Sets layout mode to timing driven or standard (non-timing driven). The default is `-timing_driven` or the mode used in the previous layout command.
[-power_driven]	value	Enables or disables power-driven layout. The default is off or the mode used in the previous layout command. The following table shows the acceptable values for this argument: **Value** / **Description** off / Does not run power-driven layout. on / Enables power-driven layout.
[-placer_high_effort]	value	Sets placer effort level. The default is off or the mode used in the previous layout command. The following table shows the acceptable values for this argument: **Value** / **Description** off / Runs layout in regular effort. on / Activates high effort layout mode.

Returns
A non-zero value will be returned on error.

See Also
Place and Route - PolarFire®

1.15 Sample Tcl Script - Project Manager

The following Tcl commands create a new project and set your project options.

```
new_project -location {D:/2Work/my_pf_proj} -name {my_pf_proj} -project_description {}\
-block_mode 0 -standalone_peripheral_initialization 0 -use_enhanced_constraint_flow 1\
-hdl {VERILOG} -family {PolarFire} -die {MPF300TS_ES} -package {FCG1152} -speed {-1} \
-die_voltage {1.0} -part_range {EXT} -adv_options {IO_DEFT_STD:LVCMOS 1.8V}\
-adv_options {RESTRICTPROBEPINS:1} -adv_options {RESTRICTSPIPINS:0}\
-adv_options {SYSTEM_CONTROLLER_SUSPEND_MODE:1} -adv_options {TEMPR:EXT} \
-adv_options {VCCI_1.2_VOLTR:EXT} -adv_options {VCCI_1.5_VOLTR:EXT} \
-adv_options {VCCI_1.8_VOLTR:EXT} -adv_options {VCCI_2.5_VOLTR:EXT} \
-adv_options {VCCI_3.3_VOLTR:EXT} -adv_options {VOLTR:EXT}
#Import HDL source file
import_files -convert_EDN_to_HDL 0 -hdl_source {C:/test/prep1.v}
#Import HDL stimulus file
import_files -convert_EDN_to_HDL 0 -stimulus {C:/test/prep1tb.v}
#set the top level design name
set_root -module {prep1::work}
#Associate SDC constraint file to Place and Route tool
organize_tool_files -tool {PLACEROUTE} -file {D:/2Work/my_pf_proj/constraint/user.sdc} \
-module {prep1::work} -input_type {constraint}
#Associate SDC constraint file to Verify Timing tool
organize_tool_files -tool {VERIFYTIMING} -file {D:/2Work/my_pf_proj/constraint/user.sdc}\
-module {prep1::work} -input_type {constraint}
#Run synthesize
run_tool -name {SYNTHESIZE}
#Configure Place and Route tool
configure_tool -name {PLACEROUTE} -params {DELAY_ANALYSIS:MAX} -params {EFFORT_LEVEL:false}\
-params {INCRPLACEANDROUTE:false} -params {MULTI_PASS_CRITERIA:VIOLATIONS}\
-params {MULTI_PASS_LAYOUT:false} -params {NUM_MULTI_PASSES:5} -params {PDPR:false}\
```

```
-params {RANDOM_SEED:0} -params {REPAIR_MIN_DELAY:false} -params {SLACK_CRITERIA:WORST_SLACK}
\
-params {SPECIFIC_CLOCK:} -params {START_SEED_INDEX:1} -params {STOP_ON_FIRST_PASS:false}\
-params {TDPR:true}
#Run Place and Route
run_tool -name {PLACEROUTE}
#Configure Timing Report Generation
configure_tool -name {VERIFYTIMING} -run_tool -name {PLACEROUTE}params
{CONSTRAINTS_COVERAGE:1}\
-params {FORMAT:XML} -params {MAX_TIMING_FAST_HV_LT:0} -params {MAX_TIMING_SLOW_LV_HT:1} \
-params {MAX_TIMING_SLOW_LV_LT:0} -params {MAX_TIMING_VIOLATIONS_FAST_HV_LT:0} \
-params {MAX_TIMING_VIOLATIONS_SLOW_LV_HT:1} -params {MAX_TIMING_VIOLATIONS_SLOW_LV_LT:0}\
-params {MIN_TIMING_FAST_HV_LT:1} -params {MIN_TIMING_SLOW_LV_HT:0} -params
{MIN_TIMING_SLOW_LV_LT:0} -params {MIN_TIMING_VIOLATIONS_FAST_HV_LT:1} -params
{MIN_TIMING_VIOLATIONS_SLOW_LV_HT:0} \
-params {MIN_TIMING_VIOLATIONS_SLOW_LV_LT:0}
#Run Verify Timing tool
run_tool -name {VERIFYTIMING}
#Run Power Verification tool
run_tool -name {VERIFYPOWER}
#Export bitstream
export_bitstream_file -file_name {prep1} \
-export_dir {D:\2Work\my_pf_proj\designer\prep1\export} -format {STP} -master_file 0 \
-master_file_components {} -encrypted_uek1_file 0 -encrypted_uek1_file_components {} \
-encrypted_uek2_file 0 -encrypted_uek2_file_components {} \
-trusted_facility_file 1 -trusted_facility_file_components {FABRIC}
```

1.16 How to Derive Required Part Information from A "Part Number"

In order to use Tcl Commands such as `set_device` or a new design; certain part information items must be specified. Many of these items can be derived from the "Part Number" you have chosen. For example, suppose the Part Number is: **MPF300XT-1FCG784I**

Table 1-1. Part Information for MPF300XT-1FCG784I

Part Information	Description	Example
`-family <family name>`	The `<family name>` usually known	`-family {PolarFire}`
`-die <die name>`	Derive this information from the Part Number, the characters before the "-"	**MPF300XT**-1FCG784I `-die {MPF300XT}`
`-speed <speed grade>`	If there is a digit immediately after the "-", -<digit> will be the <speed grade> value (preceded by a "-"). **Note:** If there is no digit, the default speed grade is STD.	MPF300XT-**1**FCG784 `-speed {-1}`
`-package <package name>`	The next sequence of letters, followed by a sequence of digits will constitute the package type and "size". **Note:** If there is a trailing letter after the <digits>; this letter is **not** part of the <package name>; but is rather part of the <part range>	For PolarFire, this combination will simply constitute the <package name>. MPF300XT- 1**FCG784**I `-package {FCG784}`
`-part_range <part range>`	The last letter (if any) will indicate the `<part_range>`. • I: IND • E: EXT • M: MIL • <none>: COM	MPF300XT- 1FCG784**I** `-part_range {IND}`

2. Project Manager Tcl Commands

2.1 add_file_to_library

Description
This Tcl command adds a file to a library in your project.

```
add_file_to_library \
-library name \
-file name
```

Arguments

Parameter	Type	Description
library	string	Name of the library where you wish to add your file.
file	string	Specifies the new name of the file you wish to add (must be a full pathname).

Error Codes

Error Code	Description
None	File is not in the project

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4™	v12.4+
SmartFusion®2	v12.4+
IGLOO®2	v12.4+

Example
Add a file named "foo.vhd" from the ./project/hdl directory to the library 'my_lib'.

```
add_file_to_library -library my_lib -file ./project/hdl/foo.vhd
```

See Also
- 2.2 add_library
- 2.64 remove_library
- 2.67 rename_library

2.2 add_library

Description

This Tcl command adds a VHDL library to your project. To add a library, right-click the design module name in the Design Hierarchy select and Add VHDL Library from the context menu.

```
add_library -library name
```

Arguments

Parameter	Type	Description
library	string	Specifies the name of your new library.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

Create a new library with 'my_lib' name in your project.

```
add_library -library my_lib
```

See Also

- 2.64 remove_library
- 2.67 rename_library

2.3 add_modelsim_path

Description

This Tcl command adds a ModelSim simulation library to your project.

```
add_modelsim_path -lib library_name  [-path library_path] [-remove " "]
```

Arguments

Parameter	Type	Description
lib	string	Name of the library you want to add.
path	string	Path to library that you want to add. Enables you to change the mapping for your simulation library (both Verilog and VHDL). Type the pathname or click the Browse button to navigate to your library directory.
remove	string	Name of library you want to remove (if any).

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Add the ModelSim library 'msim_update2' located in the c:\modelsim\libraries directory and remove the library 'msim_update1'.

```
add_modelsim_path -lib msim_update2 [-path c:\modelsim\libraries] [-remove msim_update1]
```

2.4 add_profile

Description
This Tcl command enables you to add new profile in your project and specified a Software IDE tool in your Tools profile. This command sets the same values as the Add or Edit Profile dialog box. The newly added profile becomes the active tool profile for the specified type of tool. You must provide a unique name.

```
add_profile -name profilename \
-type value \
-tool profiletool \
-location tool_location \
[-args tool_parameters ] \
[-batch value ] \
[-license license_information ] \
[-32bit value ]
```

Arguments

Parameter	Type	Description
name	string	Specifies the unique name of your new profile.
type	string	Specifies your profile type, where value is one of the following: • synthesis - new profile for a synthesis tool. The default synthesis tool included with Libero SoC is Synplify Pro ME. • simulation - new profile for a simulation tool. The default simulator tool included with Libero SoC is ModelSim AE. • stimulus - new profile for a stimulus tool. Default not specified. Stimulus tool included with Libero SoC is WFL. • identifydebugger - new identify debugger tool profile. The default identify debugger tool included with Libero SoC is Identify Debugger.
tool	string	Name of the tool you are adding to the profile. It is mandatory.
location	string	Full pathname to the location of the tool you are adding to the profile. It is mandatory.
args	list of strings	Tool parameters (if any).

.....continued		
Parameter	**Type**	**Description**
batch	boolean	Runs the tool in batch mode (if TRUE). Possible values are: • TRUE | 1 - runs the profile in batch mode. • FALE | 0 - Does not run the profile in batch mode.
license	string	License information.
32bit	boolean	Valid values are: 1, 0.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Unable to add profile: user Profile with same name already exists.
None	Required parameter 'tool' is missing.
None	Required parameter 'location' is missing.
None	Invalid argument value: 'flashpro' (expecting synthesis, simulation, stimulus, coreconfig, identifydebugger or sw_ide).

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

Create a new FlashPro tool profile called 'myflashpro' linked to a FlashPro installation in my c:\programs\actel\flashpro\bin directory:

```
add_profile -name myflashpro -type flashpro -tool flashpro.exe -location
{c:\programs\actel\flashpro\bin\flashpro.exe} -batch FALSE
```

Create a new Synthesis tool profile called 'synpol' linked to a Synplify Pro ME installation in my /sqatest/bin directory:

```
add_profile -name synpol -type synthesis -tool "Synplify Pro ME" -location "/sqatest9/bin/
synplify_pro" -batch FALSE
```

See Also

- 2.25 edit_profile
- 2.65 remove_profile
- 2.74 select_profile
- 2.39 export_profiles

2.5 associate_stimulus

Description
This Tcl command associates a stimulus file in your project. Before running simulation, you must associate a test bench. If you attempt to run simulation without an associated test bench, the Libero SoC Project Manager asks you to associate a test bench or open ModelSim without a test bench.

```
associate_stimulus [-file {absolute path and name of the file}] [-mode value ] -module value
```

Arguments

Parameter	Type	Description
file	string	Specifies the absolute path and name of the file to which you want to associate your stimulus files.
mode	string	Specifies whether you are creating a new stimulus association, adding or removing. Possible values are: • new - creates a new stimulus file association. • add - adds a stimulus file to an existing association. • remove - removes a stimulus file association.
module	string	Sets the module, where value is the name of the module. This is mandatory.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	'file' is not in the project.
None	Required parameter 'module' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example associates a new stimulus file 'stim.vhd' for stimulus.

```
associate_stimulus -file stim.vhd -mode new -module stimulus
```

See Also
- 2.57 organize_sources

2.6 build_design_hierarchy

Description
This Tcl command rebuilds the Design/Stimulus Hierarchy. Any change to the design sources/stimuli invalidates the design hierarchy/stimulus hierarchy. You can build the Design Hierarchy and Simulation Hierarchy by clicking the Build Hierarchy button.

```
build_design_hierarchy
```

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command rebuilds the Design/Stimulus Hierarchy.

```
build_design_hierarchy
```

2.7 change_link

Description
This Tcl command changes the source of a linked file in your project.

```
change_link -file filename -path new_source_path
```

Arguments

Parameter	Type	Description
file	string	Specifies absolute or relative path and name of the linked file you want to change.
path	string	Location of the file (absolute or relative path) you want to link to.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

1. Change the link to a file 'sim1.vhd' in your project and link it to the `c:\Microchip\link_source\simulation_test.vhd` file.

```
change_link -file sim1.vhd -path {c:\Microchip\link_source\simulation_test.vhd}
```

2. Change link for the file 'test.v' from 'E:\Share\test.v' to 'E:\Share\test\srcs\test.v' using environment variable 'MSCC_ROOT_1' that has the root directory path 'E:\Share'.

```
change_link -file {${MSCC_ROOT_1}/test.v} -path {${MSCC_ROOT_1}/test/srcs/test.v}
```

See Also

- 2.19 create_links
- 2.83 unlink_files

2.8 change_vault_location

Description

This Tcl command changes the location of the vault. Equivalent to clicking the Project menu, and choosing Vault/ Repositories Setting and selecting new vault location, by default is "/usr/local/Microchip/common"/. The vault location is common to all Microchip software:

- Project Manager
- Firmware Catalog

Changing your vault location here updates the vault location for all the Microchip tools you use on your machine.

Note: This command overrides the vault location for all projects.

```
change_vault_location -location location
```

Arguments

Parameter	Type	Description
location	string	Specifies the new vault location. Value must be a file path. It is mandatory.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Parameter 'location' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command changes the old location of vault into new location.

```
change_vault_location -location {../vault}
```

2.9 check_fdc_constraints

Description
This Tcl command checks FDC (Synplify Netlist Constraint) constraints files associated with the Synthesis tool. FDC constraints are used to optimize the HDL design using Synopsys SynplifyPro Synthesis engine and have the *.fdc extension.

```
check_fdc_constraints -tool {synthesis}
```

Arguments

Parameter	Type	Description
tool	string	The valid value is synthesis only.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command checks FDC constraints files associated with the Synthesis tool.

```
check_fdc_constraints -tool {synthesis}
```

See Also
- 2.11 check_ndc_constraints

2.10 check_hdl

Description
This Tcl command checks the HDL in the specified file for validity.

```
check_hdl -file { absolute path and name of the HDL file }
```

Arguments

Parameter	Type	Description
file	string	Specified absolute path and name of the HDL file you want to check. It is mandatory.

Error Codes

Error Code	Description
None	HDL file is not in the project.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command checks HDL on the file hdl1.vhd.

```
check_hdl -file {/project/hdl/hdl1.vhd}
```

2.11 check_ndc_constraints

Description
This Tcl command checks NDC (Compile Netlist Constraint) constraints files associated with the Synthesis tool. NDC constraints are used to optimize the post-synthesis netlist with the Libero SoC Compile engine and have the *.ndc file extension.

```
check_ndc_constraints -tool {synthesis}
```

Arguments

Parameter	Type	Description
tool	string	The valid value is synthesis.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following Tcl command checks NDC constraints files associated with the Synthesis tool.

```
check_ndc_constraints -tool {synthesis}
```

See Also
- 2.9 check_fdc_constraints

2.12 check_pdc_constraints

Description
This Tcl command checks Physical Design Constraints (PDC) constraint files associated with the Libero Place and Route tool. PDC Tcl is divided between I/O attribute and pin information from all floorplanning and timing constraints.

- The I/O Attributes tab allows you to manage I/O attributes/constraints for your design's Inputs, Outputs and Inouts. All I/O constraint files (PDC) have the *.pdc file extension and are placed in the `<project_location>/constraint/io` folder.
- The Floor Planner tab allows you to manage floorplanning constraints. Floorplanning constraints files have the *.pdc file extension and are placed in the `<project_location>\constraint\fp` folder.

```
check_pdc_constraints -tool {designer}
```

Arguments

Parameter	Type	Description
tool	string	The valid value is designer.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command checks PDC constraints files associated with the Libero Place and Route (designer) tool.

```
check_pdc_constraints -tool {designer}
```

2.13 check_sdc_constraints

Description
This Tcl command checks Synopsys Design Constraints (SDC) constraint files associated with the Libero tools: designer, synthesis or timing.

Note: This command cannot be run until Compile has been run.

```
check_sdc_constraints -tool {tool_name}
```

Arguments

Parameter	Type	Description
tool	string	Valid values are: synthesis, designer and timing.

Error Codes

Error Code	Description
None	Invalid argument value: '' (expecting synthesis, designer, physynth, timing or compilenetlist).

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
This command checks the SDC constraint files associated with Timing Verification.

```
check_sdc_constraints -tool {timing}
```

This command checks the SDC constraint files associated with Place and Route.

```
check_sdc_constraints -tool {designer}
```

This command checks the SDC constraint files associated with Synthesis.

```
check_sdc_constraints -tool {synthesis}
```

2.14 close_design

Description
This Tcl command closes the current design and brings Designer to a fresh state to work on a new design. This is equivalent to selecting the **Close** command from the **File** menu.

```
close_design -component smartdesign_component_name
```

Arguments

Parameter	Type	Description
component	string	Specifies the name of the SmartDesign component to be closed. It is mandatory.

Error Codes

Error Code	Description
None	Parameter 'component' has illegal value.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example checks is the "Top" design component successfully closed or no.

```
if {[ catch {close_design -component {Top}} ]} {
Puts "Failed to close design"
# Handle Failure
} else {
puts "Design closed successfully"
# Proceed with processing a new design
}
```

See Also

- 2.21 create_smartdesign
- 2.55 open_smartdesign
- 2.72 save_smartdesign

2.15 close_project

Description
This Tcl command closes the current project in Libero SoC. This command is equivalent to selecting **Close Project** from the **File** menu.

```
close_project [-save value]
```

Arguments

Parameter	Type	Description
value	boolean	Saves the current project in Libero SoC before closing project. • TRUE, true or 1 - saves your project in Libero SoC before closing project. Default is true. • FALSE, false or 0 - closes your project without saving.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

```
close_project
```

See Also

- 2.54 open_project

2.16 configure_core

Description

This Tcl command modifies the configuration of an existing core component in the SmartDesign. This command works for core components created for different types of cores namely, Sg cores, System Builder cores and Direct cores. In the Libero SoC, choose `View > Windows > Catalog`. The Catalog displays a list of available cores, busses and macros. Double-click a core to open the core generator and configure it and add it to your design.

Limitations: The command does not work for SmartFusion2 and IGLOO2 System Builder components, SmartFusion2 MSS component, RTG4 PCIE_SERDES_IF_INIT(RTG4 High Speed Serial Interface 1 - EPCS and XAUI - with Initialization), NPSS_SERDES_IF_INIT(RTG4 High Speed Serial Interface 2 - EPCS and XAUI - with Initialization), and RTG4FDDRC_INIT(RTG4 DDR Memory Controller with initialization) core components.

```
configure_core -component_name component_name -params core_parameters
```

Arguments

Parameter	Type	Description
component_name	string	Specifies the name of the component to be configured. It is mandatory.
params	string	Specifies the parameters needed to configure the core component. It is mandatory. It can either take single parameter or multiple parameters at a time This command will fail if none of the core parameters are specified.

Error Codes

Error Code	Description
None	Required parameter 'component_name' is missing.
None	Unable to create core. A Component with that name already exists.
None	Parameter 'p' is not defined. Valid command formatting is 'configure_core -component_name "component_name" [-params "[params]+"].
None	Cannot find Spirit core configuration file for vendor:Actel library:Simulation name:<core_name> version:1.0.1.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following commands modifies the configuration of "Core_UART" and "PF_CCC_C0" core components - sets cores parameters values in the SmartDesign.

```
configure_core -component_name {PF_CCC_C0} -params "GL1_0_IS_USED:false" \
"GL0_0_IS_USED:true” “GL0_0_OUT_FREQ:200”}
```

```
configure_core -component_name {Core_UART} -params {"BAUD_VAL_FRCTN_EN:false" \
"RX_FIFO:0" "RX_LEGACY_MODE:0" "TX_FIFO:1" "USE_SOFT_FIFO:1"}
```

See Also
- 2.18 create_and_configure_core
- 2.63 remove_core
- 2.23 download_core
- 2.24 download_latest_cores

2.17 configure_tool

Description
This Tcl command is a general-purpose command that is used to set the parameters for any tool called by Libero for the families. The command requires the name of the tool and one or more parameters in the `tool_parameter:value` format. These parameters are separated and passed to the tool to set up its run.

```
configure_tool -name {<tool_name>} -params {<parameter>:<value>}
```

Arguments

Parameter	Type	Description
tool_name	string	Specifies the name of tool for which you wish to know the configured tool options. It is mandatory. Each tool_name has its own set of parameters. • COMPILE • CONFIGURE_ACTION_PROCEDURES • CONFIGURE_PROG_OPTIONS • CONFIGURE_PROG_OPTIONS_RTG4 • SYNTHESIZE • PLACEROUTE • VERIFYTIMING • VERIFYPOWER • GENERATEPROGRAMMINGDATA • GENERATEPROGRAMMINGFILE • PROGRAMDEVICE • PROGRAM_OPTIONS • PROGRAMMER_INFO • SPM • FLASH_FREEZE • PROGRAM_RECOVERY • USER_PROG_DATA • INIT_LOCK

.....continued		
Parameter	**Type**	**Description**
params	string	Specifies the tool options/parameters for the value you want to configure. List of parameters and values. There may be multiple `-params` arguments (see example below). This is mandatory.

Error Codes

Error Code	Description
None	Required parameter 'params' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example sets the COMPILE command options DISPLAY_FANOUT_LIMIT to 10 and MERGE_SDC to true. There are alternative ways to write these commands to fit your coding style, as shown in the following examples.

```
configure_tool \
-name {COMPILE} -params {DISPLAY_FANOUT_LIMIT:10} \
-params {MERGE_SDC:true}
```

See Also
- 2.47 get_tool_options

2.18 create_and_configure_core

Description
This Tcl command creates a configured core component for a core selected from the Libero Catalog. This command works for core components created for different types of cores namely, Sg cores, System Builder cores and Direct cores. In the Libero SoC, choose *View > Windows > Catalog*. The Catalog displays a list of available cores, bus and macros. From the Catalog, you can create a component from the list of available cores, add a processor or peripheral, add a bus interface to your SmartDesign component, instantiate simulation cores or add a macro (Arithmetic, Basic Block, etc.) to your SmartDesign component.

Limitations: The command does not work for SmartFusion2 and IGLOO2 System Builder components, SmartFusion2 MSS component, and RTG4 PCIE_SERDES_IF_INIT(RTG4 High Speed Serial Interface 1 - EPCS and XAUI - with Initialization), NPSS_SERDES_IF_INIT(RTG4 High Speed Serial Interface 2 - EPCS and XAUI - with Initialization) and RTG4FDDRC_INIT(RTG4 DDR Memory Controller with initialization) core components.

Note: For DirectCore and Solutions cores, refer to the core handbook or the core user guide for a list of valid parameters and values.

```
create_and_configure_core \
core_vlnv Vendor:Library:Name:Version \
-component_name component_name \
-params core_parameters
```

Arguments

Parameter	Type	Description
core_vlnv	string	Specifies the version identifier of the core being configured. It is mandatory.
component_name	string	Specifies the name of the configured core component. It is mandatory.
params	string	Specifies the parameters needed to configure the core component. It is mandatory. It can either take single parameter or multiple parameters at a time This command will fail if none of the core parameters are specified.

Error Codes

Error Code	Description
None	Required parameter 'core_vlnv' is missing.
None	Required parameter 'component_name' is missing.
None	Unable to create core. A Component with that name already exists.
None	Cannot find Spirit core configuration file for vendor:Actel library:Simulation name:<core_name> version:1.0.1.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

The following command configured SgCore(PF_CCC_C3) core - sets core parameters values, specifies the version identifier of the SgCore(PF_CCC_C3) core.

```
create_and_configure_core -core_vlnv {Actel:SgCore:PF_CCC:1.0.115} \
-component_name {PF_CCC_C3} \
-params {"PLL_IN_FREQ_0:25" \
"GL0_0_IS_USED:true" \
"GL0_0_OUT_FREQ:150" \
"GL0_1_IS_USED:true" \
"GL0_1_OUT_FREQ:50"}
```

See Also

- 2.16 configure_core
- 2.63 remove_core
- 2.23 download_core
- 2.24 download_latest_cores

2.19 create_links

Description
This Tcl command creates a link (or links) to a file/files in your project. Specify absolute or relative path and name of the file you want to link. The same file you cannot link to different libraries.

```
create_links [-convert_EDN_to_HDL "TRUE | FALSE"] \
[-hdl_source_folder "Source folder"]* \
[-library "library"] \
[-hdl_source "file"]* \
[-stimulus "file"]* \
[-edif "file"]* \
[-sdc "file"]* \
[-ndc "file"]* \
[-fp_pdc "file"]* \
[-io_pdc "file"]* \
[-net_fdc "file"]* \
[-verilog_netlist "file"]* \
[-vcd "file"]* \
[-dcf "file"]* \
[-pin "file"]* \
[-crt "file"]* \
[-gcf "file"]*
```

Arguments

Parameter	Type	Description
convert_EDN_to_HDL	boolean	Use the `-convert_EDN_to_HDL` parameter to convert the EDIF file to HDL and then import the converted HDL file. Valid values: TRUE, 1, true, FALSE, 0 or false. If the `-edif` option is not specified or the `-convert_EDN_to_HDL` is used with another option, EDIF to HDL conversion will fail.
hdl_source_folder	string	Name of the HDL folder you want to link. For unlink folder you must unlink files form folder one by one.
library	string	Specifies the name of the library where you want to link file. The same file you cannot link different libraries.
hdl_source	string	Name of the HDL file you want to link.
stimulus	string	Name of the stimulus file you want to link.
edif	string	Name of the EDIF Netlist file you want to link. It used with `convert_EDN_to_HDL` option.
sdc	string	Name of the SDC file you want to link.
ndc	string	Name of the NDC (Compile Netlist Constraint) file you want to link.
fp_pdc	string	Name of the Floor Planner PDC file you want to link.
io_pdc	string	Name of the IO PDC file you want to link.
net_fdc	string	Name of the FDC (Synplify Netlist Constraint) file you want to link.
vcd	string	Name of the VCD file you want to link.
verilog_netlist	string	Name of the VM (Synthesized Verilog Netlist) file you want to link.

.....continued		
Parameter	**Type**	**Description**
pin	string	Name of the PIN file you want to link. (Not supported for PolarFire, SmartFusion2, IGLOO2, and RTG4 families).
dcf	string	Name of the DCF(Timing Constraint Files) file you want to link. (Not supported for PolarFire, SmartFusion2, IGLOO2, and RTG4 families).
gcf	string	Name of the GCF (ProASIC® Constraint Files) file you want to link. (Not supported for PolarFire, SmartFusion2, IGLOO2, and RTG4 families).
crt	string	Name of the CRT (Criticality Files) file you want to link. (Not supported for PolarFire, SmartFusion2, IGLOO2, and RTG4 families).

Error Codes

Error Code	Description
None	Error: Parameter 'vhd' is not defined. Valid command formatting is 'create_links [-convert_EDN_to_HDL "TRUE | FALSE"] \ [-hdl_source_folder "Source folder"]* [-library "library"] \ [-ndc "file"]* [-crt "file"]* \ [-fp_pdc "file"]* [-hdl_source "file"]* \ [-stimulus "file"]* [-io_pdc "file"]* \ [-pin "file"]* [-gcf "file"]* \ [-sdc "file"]* [-net_fdc "file"]* \ [-verilog_netlist "file"]* \ [-dcf "file"]* [-vcd "file"]*
None	Cannot import the 'Timing Constraint Files'; your selected family does not support DCF constraints.
None	Cannot import the 'ProASIC Constraint Files'; your selected family does not support GCF constraints.
None	Cannot import the 'Criticality Files'; your selected family does not support CRT constraints.
None	Cannot import the 'Pin Files'; your selected family does not support PIN constraints.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

1. Create a link to the file hdl1.v.

```
create_links -hdl_source hdl1.v
```

2. Link files to the project located at "E:\Share\abc.edn" and "E:\Share\test.v" using Environment variable "MSCC_ROOT_1" that has the root directory path "E:\Share".

```
create_links \
-convert_EDN_to_HDL 0 \
-library {work} \
-edif {${MSCC_ROOT_1}/abc.edn} \
-hdl_source {${MSCC_ROOT_1}/test.v}
```

See Also

- 2.7 change_link

• 2.83 unlink_files

2.20 create_set

Description

This Tcl command creates a set of paths to be analyzed. Use the arguments to specify which paths to include. To create a set that is a subset of a clock domain, specify it with the `-clock` and `-type` arguments. To create a set that is a subset of an inter-clock domain set, specify it with the `-source_clock` and `-sink_clock` arguments. To create a set that is a subset (filter) of an existing named set, specify the set to be filtered with the `-parent_set` argument.

```
create_set\ -name <name>\ -parent_set <name>\ -type <set_type>\ -clock <clock name> \ -
source_clock <clock name>\ -sink_clock <clock name>\ -in_to_out\ -source <port/pin pattern> \
-sink <port/pin pattern>
```

Arguments

Parameter	Type	Description
name	string	Specifies a unique name for the newly created path set.
parent_set	string	Specifies the name of the set to filter from.
clock	string	Specifies that the set is to be a subset of the given clock domain. This argument is valid only if you also specify the `-type` argument.
type	string	Specifies the predefined set type on which to base the new path set. You can only use this argument with the `-clock` argument, not by itself. • reg_to_reg - paths between registers in the design. • async_to_reg - paths from asynchronous pins to registers. • reg_to_async - paths from registers to asynchronous pins. • external_recovery - the set of paths from inputs to asynchronous pins. • external_removal - the set of paths from inputs to asynchronous pins. • external_setup - paths from input ports to registers. • external_hold - paths from input ports to registers. • clock_to_out - paths from registers to output ports.
in_to_out	None	Specifies that the set is based on the "Input to Output" set, which includes paths that start at input ports and end at output ports.
source_clock	string	Specifies that the set will be a subset of an inter-clock domain set with the given source clock. You can only use this option with the `-sink_clock` argument.
sink_clock	string	Specifies that the set will be a subset of an inter-clock domain set with the given sink clock. You can only use this option with the `-source_clock` argument.
source	string	Specifies a filter on the source pins of the parent set. If you do not specify a parent set, this option filters all pins in the current design.

.....continued		
Parameter	**Type**	**Description**
sink	string	Specifies a filter on the sink pins of the parent set. If you do not specify a parent set, this option filters all pins in the current design.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example creates set with "my_user_set" name. Filters all C* ports and D* pins in the current design.

```
create_set -name { my_user_set } -source { C* } -sink { D* }
```

The following example creates set with "my_other_user_set" name that is a subset (filter) of an existing "my_user_set" set.

```
create_set -name { my_other_user_set } -parent_set { my_user_set } -source { CL* }
```

The following example creates set with "another_set" name which is the subset of an inter-clock domain set with the given source clock.

```
create_set -name { another_set } -source_clock { EXTERN_CLOCK } \
-sink_clock { MY_GEN_CLOCK }
```

2.21 create_smartdesign

Description
This Tcl command creates a SmartDesign to your project. To create design, click the Create SmartDesign tool name from Design Flow.

```
create_smartdesign -sd_name smartdesign_component_name
```

Arguments

Parameter	Type	Description
sd_name	string	Specifies the name of the SmartDesign component to be created. It is mandatory.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Creates new design component with "top" name.

```
create_smartdesign -sd_name {top}
```

2.22 delete_component

Description
This Tcl command deletes a component from the Design Hierarchy.

```
delete_component -component_name component_name
```

Arguments

Parameter	Type	Description
component_name	string	Specifies the name of the component to be deleted. It is mandatory.

Error Codes

Error Code	Description
None	Required parameter 'component_name' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command deletes a component with component name from the Design Hierarchy.

```
delete_component -component_name {component}
```

The following command deletes a component with shifter name from the Design Hierarchy.

```
delete_component -component_name {shifter}
```

See Also

- 2.43 generate_component

- 2.49 import_component

2.23 download_core

Description
This Tcl command downloads a core and adds it to your repository. The Catalog enables you to download cores from a web repository into a Vault. A Vault is a local directory (either local to your machine or on the local network) that contains cores downloaded from one or more repositories. A repository is a location on the web that contains cores that can be included in your design. The Catalog displays all the cores in your Vault.

You may want to import a core from a file when:

- You do not have access to the internet and cannot download the core.
- A core is not complete and has not been posted to the web (you have an evaluation core).

```
download_core -vlnv "Vendor:Library:Name:Version" [-location "location"]
```

Arguments

Parameter	Type	Description
vlnv	string	Vendor, library, name and version of the core you want to download. It is mandatory.
location	string	Location of the repository where you wish to add the core. It is optional.

Error Codes

Error Code	Description
None	Required parameter 'vlnv' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example downloads the System Builder (PF_DDR4) core to the repository www.microsemi.com/repositories/SgCore.

```
download_core -vlnv {Actel:SystemBuilder:PF_DDR4:1.0.102} \
        -location {www.microsemi.com/repositories/SgCore}
```

See Also

- 2.16 configure_core
- 2.18 create_and_configure_core
- 2.24 download_latest_cores

2.24 download_latest_cores

Description
This Tcl command is used to download the latest cores into the vault. A project does not need to be open to run this command. This command takes no arguments. The Catalog Options dialog box enables you to customize your Catalog. `Display only the latest version of a core` is checked by default. This option, if checked, shows the latest versions of cores that are not in the Vault, and also filters out any duplicate cores that have the same Vendor, Library, and Name, with an earlier version number.

Note: If there are no cores to be downloaded, you will see the following message:

```
Info:All the latest cores are present in the vault.
```

```
download_latest_cores
```

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command downloads the latest cores into the vault.

```
download_latest_cores
```

See Also
- 2.16 configure_core
- 2.23 download_core
- 2.63 remove_core

2.25 edit_profile

Description
This Tcl command enables you to edit profile in your project. You can edit profile from *Project > Tool Profiles* or right-click tool name and select **Edit Profile**.

```
edit_profile -name profilename -type value \
-tool {profile tool} \
-location {profile location} \
[-args parameters ] \
[-batch value ] \
[-new_name name ]
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of your new profile.

.....continued

Parameter	Type	Description
type	string	Specifies your profile type, where value is one of the following: • synthesis - new profile for a synthesis tool. • simulation - new profile for a simulation tool. • stimulus - new profile for a stimulus tool. • identifydebugger - new identify debugger tool profile.
tool	string	Name of the tool you are adding to the profile.
location	string	Full pathname to the location of the tool you are adding to the profile.
args	list of strings	Profile tool parameters (if any).
batch	string	Runs the tool in batch mode (if TRUE). Possible values are: • TRUE - runs the profile in batch mode. • FALSE - does not run the profile in batch mode.
new_name	string	Specifies new name of profile.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
To edit a FlashPro tool profile called 'mySynplify' linked to a new SynplifyPro installation in my c:\programs\actel\SynplifyPro\bin directory, change the name to updated_synplifypro.

```
edit_profile -name {mySynplify} -type synthesis -tool {Synplify.exe} -location \
c:\programs\actel\SynplifyPro\bin\synplify_pro -batch FALSE -new_name updated_synplifypro
```

See Also
- 2.4 add_profile
- 2.65 remove_profile
- 2.39 export_profiles

2.26 export_as_link

Description
This Tcl command exports a file to another directory and links to the file.

```
export_as_link -file {absoulte or relative path and name of the file} -path link_path
```

Arguments

Parameter	Type	Description
file	string	Specifies absolute or relative path and name of the file you want to export as a link.
path	string	Path of the link.

Error Codes

Error Code	Description
None	'c.v' must not be part of a component and must be local to the project.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Export the file hdl1.vhd as a link to `c:\Microchip\link_source`.

```
export_as_link -file hdl1.vhd -path {c:\Microchip\link_source}
```

2.27 export_ba_files

Description
Tcl command to export the backannotated files. The backannotated files are:

- <design_name>_fast_hv_lt_ba.v
- <design_name>_slow_lv_ht_ba.v
- <design_name>_slow_lv_lt_ba.v (Verilog backannotated netlist) or <design_name>_fast_hv_lt_ba.vhd
- <design_name>_slow_lv_ht_ba.vhd
- <design_name>_slow_lv_lt_ba.vhd (VHDL backannotated netlist)
- <design_name>_fast_hv_lt_ba.sdf
- <design_name>_slow_lv_ht_ba.sdf
- <design_name>_slow_lv_lt_ba.sdf (Standard Delay Format) timing file.

These files are passed to the default simulator for postlayout simulation. Before exporting, you need to run 'Place and Route'.

```
export_ba_files
-export_dir {absolute path to folder location} \
-export_file_name {name of file} \
-vhdl {value} \
-min_delay {value}
```

Arguments

Parameter	Type	Description
export_dir	string	Specifies the path where you wish to export the backannotated files.
export_file_name	string	File name to generate the files. If not specified, it takes <design_name> as the default. If specified it takes <design_name_file_name>.
vhdl	integer	Generates the <design_name>_ba.v and <design_name>_ba.sdf when set to 0 and <design_name>_ba.vhd and <design_name>_ba.sdf when set to 1. Default is 0.
min_delay	integer	Set to 1 to export enhanced min delays to include your best-case timing results in your Back Annotated file. Default is 0.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example creates the `E:\designer\export\sd1` directory where the following backannotated files are generated and exported.

- test_fast_hv_lt_ba.sdf
- test_fast_hv_lt_ba.v
- test_slow_lv_ht_ba.sdf
- test_slow_lv_ht_ba.v
- test_slow_lv_lt_ba.sdf
- test_slow_lv_lt_ba.v

```
export_ba_files \
-export_dir {E:\designs\export\sd1} \
-export_file_name {test} \
-vhdl 0 \
-min_delay 1
```

2.28 export_bitstream_file

Description
This Tcl command configures the parameters for the bitstream to be exported from Libero.

Note: RTG4 and PolarFire devices do not support the security, SPI directory, or serialization options that SmartFusion2 and IGLOO2 devices support.

- The following is the syntax for the Export Bitstream File Tcl command for SmartFusion2, IGLOO2, and RTG4.

Note: A Tcl script file exported from Libero will include all command options. You can modify options you need and remove options you do not need.

```
export_bitstream_file [-file_name "file_name"] \
[-export_dir "export_dir"] \
[-format "PPD | DAT | HEX | STP | CHAIN_STP | SPI | SVF" ] \
[-for_ihp "TRUE | FALSE" ] \
[-master_file "TRUE | FALSE" ] \
[-master_file_components "SECURITY | FABRIC | ENVM" ] \
[-encrypted_uek1_file "TRUE | FALSE" ] \
[-encrypted_uek1_file_components "FABRIC | ENVM" ] \
[-encrypted_uek2_file "TRUE | FALSE" ] \
[-encrypted_uek2_file_components "FABRIC | ENVM" ] \
[-encrypted_uek3_file "TRUE | FALSE" ] \
[-encrypted_uek3_file_components "FABRIC | ENVM" ] \
[-trusted_facility_file "TRUE | FALSE" ] \
[-trusted_facility_file_components 'FABRIC | ENVM" ] \
[-add_golden_image "TRUE | FALSE" ] \
[-golden_image_address "golden_image_address" ] \
[-golden_image_design_version "golden_image_design_version" ] \
[-add_update_image "TRUE | FALSE" ] \
[-update_image_address "update_image_address" ] \
[-update_image_design_version "update_image_design_version" ] \
[-serialization_stapl_type "serialization_stapl_type" ] \
[-serialization_target_solution "serialization_target_solution" ] \
[-script "script" ] \
[-force_rtg4_otp "TRUE | FALSE" ] \
[-master_include_plaintext_passkey "TRUE | FALSE" ] \
[-uek1_include_plaintext_passkey "TRUE | FALSE" ] \
[-uek2_include_plaintext_passkey "TRUE | FALSE" ] \
[-uek3_include_plaintext_passkey "TRUE | FALSE" ]
```

- The following is the syntax for the Export Bitstream File Tcl command for PolarFire.

```
export_bitstream_file [-file_name "file_name"] \
[-export_dir "export_dir"] \
[-format "PPD | DAT | HEX | STP | SPI" ] \
[-for_ihp "TRUE | FALSE" ] \
[-limit_SVF_file_size "TRUE | FALSE" ] \
[-limit_SVF_file_by_max_filesize_or_vectors "limit_SVF_file_by_max_filesize_or_vectors" ]
\
[-svf_max_filesize "svf_max_filesize" ] \
[-svf_max_vectors "svf_max_vectors" ] \
[-master_file "TRUE | FALSE" ] \
[-master_file_components "SECURITY | FABRIC | SNVM | ENVM | FABRIC_SNVM" ] \
[-encrypted_uek1_file "TRUE | FALSE" ] \
[-encrypted_uek1_file_components "FABRIC | SNVM | ENVM | FABRIC_SNVM" ] \
[-encrypted_uek2_file "TRUE | FALSE"] \
[-encrypted_uek2_file_components "FABRIC | SNVM | ENVM | FABRIC_SNVM" ] \
[-trusted_facility_file "TRUE | FALSE"] \
[-trusted_facility_file_components "FABRIC | SNVM | ENVM | FABRIC_SNVM" ] \
[-trusted_facility_keep_fabric_operational "TRUE | FALSE"] \
[-trusted_facility_skip_startup_seq "TRUE | FALSE"] \
[-zeroization_likenew_action "TRUE | FALSE" ] \
[-zeroization_unrecoverable_action "TRUE | FALSE" ] \
[-master_backlevel_bypass "TRUE | FALSE" ] \
[-uek1_backlevel_bypass "TRUE | FALSE" ] \
[-uek1_keep_fabric_operational "TRUE | FALSE"] \
[-uek1_skip_startup_seq "TRUE | FALSE"] \
[-uek2_backlevel_bypass "TRUE | FALSE" ] \
[-uek2_keep_fabric_operational "TRUE | FALSE"] \
[-uek2_skip_startup_seq "TRUE | FALSE"] \
[-master_include_plaintext_passkey "TRUE | FALSE" ] \
[-uek1_include_plaintext_passkey "TRUE | FALSE" ] \
[-uek2_include_plaintext_passkey "TRUE | FALSE" ] \
[-script "script" ]
```

Arguments

Parameter	Type	Description
file	string	The name of the file. File name must start with design name. If omitted, design name will be used.
export_dir	string	Location (absolute path) where the bitstream file will be exported. If omitted, <designer>/<design_name>/export folder will be used.
format	string	Specifies the bitstream file formats to be exported. Space is used as a delimiter.The value can be any one of PPD, STP, CHAIN_STP, DAT, SPI, HEX, SVF. If omitted, PPD and DAT files will be exported. **Notes:** 1. Export CHAIN_STP, SVF and SPI files are not supported in RTG4. 2. Export CHAIN_STP and SVF are not supported in PolarFire.
for_ihp	boolean	Specifies to export the bitstream files for Microchip In House Programming (IHP). Valid values are: TRUE, true, 1, FALSE, false, 0. Default is 0.
limit_SVF_file_size	boolean	Specify limit on the SVF file size. Valid values are: TRUE, true, 1, FALSE, false, 0.
limit_SVF_file_by_max_filesize_or_vectors	boolean	Specify limit on the SVF file size or vectors number. Valid values are: TRUE, true, 1, FALSE, false, 0.
svf_max_filesize	integer	Specify svf file maximum size. It is equal to or greater than 0 KB.
svf_max_vectors	integer	Specify maximum number of vectors in file. It must be equal to or greater than 0.
script	string	Absolute path of script file. This is an optional parameter.
force_rtg4_otp	boolean	Enforces the use of One-time programming (OTP). It is optional. Valid values are: TRUE, true, 1, FALSE, false, 0. Default is 0.

Security-Related Options

The following table lists the Security-related options.
Note: One of the trusted_facility file or master_file or encrypted_uek1_file or encrypted_uek2_file or encrypted_uek3_file must be set to “1”. 1 indicates that this particular file type will be exported; 0 indicates that it will not be exported. For example, if `trusted_facility_file` is set to 1, all other file types must be set to 0.

If `trusted_facility_file` is set to 0, a combination of master_file and uek1_file, uek2_file and uek3_file can be set to 1. In this case, master_file must be set to 1.

Export the Bitstream file as you may require the design components saved in the exported bitstream file.

Parameter	Type	Description
trusted_facility_file	boolean	Specifies the bitstream file to be exported. • 1 - indicates that this particular file type will be exported. • 0 - indicates that it will not be exported.

.....continued

Parameter	Type	Description
trusted_facility_file_components	string	Specifies the components of the design that will be saved to the bitstream file. Default is FABRIC. The value can be: • PolarFire - one of or any combination of FABRIC, SNVM, ENVM or FABRIC_SNVM. • SmartFusion2, IGLOO2, and RTG4 - one of or any combination of FABRIC, ENVM.

Zeroization Options:

The following table lists the Zeroization options.

Parameter	Type	Description
zeroization_likenew_action	boolean	Specifies that all the data will be erased and the device can be reprogrammed immediately.
zeroization_unrecoverable_action	boolean	Specifies that all the data will be erased and the device cannot be reprogrammed and it must be scrapped.

Custom Security Options

The following table lists the Custom security options.

Parameter	Type	Description
master_file	boolean	Specifies the bitstream files to be exported. Depends on the selected security. **Note:** If `-master_file` is 1, SECURITY must be selected.
master_file_components	string	Specifies the components in the design that will be saved to the bitstream file. • PolarFire - SECURITY, FABRIC, SNVM, FABRIC_SNVM, ENVM • SmartFusion2, IGLOO2, and RTG4 - SECURITY, FABRIC, ENVM **Notes:** 1. The SECURITY option is available in `-bitstream_file_components` only when file type is MASTER in `-bitstream_file_type`. 2. SNVM should be programmed with FABRIC. 3. Security only programming must be performed only on erased or new devices. If performed on device with fabric programmed, the fabric will be disabled after performing security only programming. You must reprogram the fabric to re-enable it.
encrypted_uek1_file	boolean	Specifies the bitstream file to be exported. Default is 0. Valid values are: • 1 - indicates that this particular file type will be exported. • 0 - indicates that it will not be exported.

.....continued

Parameter	Type	Description
encrypted_uek1_file_components	string	Specifies the components of the design that will be saved to uek1 bitstream. The value can be any one or both of FABRIC and ENVM. • PolarFire - FABRIC, ENVM, FABRIC_SNVM, SNVM • SmartFusion2, IGLOO2, and RTG4 - FABRIC, ENVM **Note:** SNVM should be programmed with FABRIC.
encrypted_uek2_file	boolean	Specifies the bitstream file to be exported. Default is 0. Valid values are: • 1 - indicates that this particular file type will be exported. • 0 - indicates that it will not be exported.
encrypted_uek2_file_components	string	Specifies the components of the design that will be saved to uek2 bitstream. • PolarFire - FABRIC, ENVM, FABRIC_SNVM, SNVM. • SmartFusion2, IGLOO2 and RTG4 - FABRIC, ENVM. **Note:** SNVM should be programmed with FABRIC.
encrypted_uek3_file	boolean	Specifies the bitstream file to be exported. Valid values are: • 1 - indicates that this particular file type will be exported. • 0 - indicates that it will not be exported.
encrypted_uek3_file_components	string	Specifies the components of the design that will be saved to uek3 bitstream. The value can be any one or both of FABRIC and ENVM.
master_include_plaintext_passkey	boolean	Specifies that the master file includes plaintext passkey. This argument is optional.
uek1_include_plaintext_passkey	boolean	Specifies that uek1 includes plaintext passkey. This argument is optional.
uek2_include_plaintext_passkey	boolean	Specifies that uek2 includes plaintext passkey. This argument is optional.
uek3_include_plaintext_passkey	boolean	Specifies that uek3 includes plaintext passkey. This argument is optional.

Bypass Back Level Protection Options

The following table lists the Bypass Back Level Protection options. These options are only supported by the SPI bitstream files. Export the Bitstream file as you may require the design components saved in the exported bitstream file.

Parameter	Type	Description
master_backlevel_bypass	boolean	Specifies the Bypass Back Level protection for Golden/Recovery bitstream if back level protection is enabled in _master file.
uek1_backlevel_bypass	boolean	Specifies the Bypass Back Level Protection for Golden/Recovery bitstream if back level protection is enabled in _uek1 file.
uek2_backlevel_bypass	boolean	Specifies the Bypass Back Level Protection for Golden/Recovery bitstream if back level protection is enabled in _uek2 file.

SPI-Related Options

The following table lists the SPI-related options. These are optional.

Parameter	Type	Description
add_golden_image	boolean	• 1 - To enable golden SPI image in SPI direct. • 0 - To disable golden SPI image in SPI direct.
golden_image_address	string	32-bit hexadecimal address for golden image.
golden_image_design_version	string	Decimal value for golden image design version.
add_update_image	boolean	• 1 - To enable golden update SPI image. • 0 - To disable golden update SPI image.
update_image_address	string	Hexadecimal value for update image address.
update_image_design_version	string	Decimal value for update image design version.

Serialization Options

The following table lists the serialization options. These are optional.

Parameter	Type	Description
serialization_stapl_type	string	Serialization stapl file type either SINGLE or MULTIPLE. Default is SINGLE.
serialization_target_solution	string	Target programming hardware – Flashpro_3_4_5 or generic_STAPL_player. Default is Flashpro_3_4_5.

Advanced Options

The following table lists the advanced options. These options are available for PolarFireSoC device family only.

Table 2-1.

Parameter	Type	Description
trusted_facility_keep_fabric_operational	boolean	Specifies to keep fabric in operational state during programming if file is programmed at a trusted facility. Valid values are: TRUE, true, 1, FALSE, false, 0.
trusted_facility_skip_startup_seq	boolean	Specifies to skip device startup sequence after programming if file is programmed at a trusted facility and `trusted_facility_keep_fabric_operational` is TRUE. Valid values are: TRUE, true, 1, FALSE, false, 0.
uek1_keep_fabric_operational	boolean	Specifies to keep fabric in operational state during programming if file is encrypted using UEK1 custom security. Valid values are: TRUE, true, 1, FALSE, false, 0.
uek1_skip_startup_seq	boolean	Specifies to skip device startup sequence after programming if file is programmed using UEK1 custom security and `uek1_keep_fabric_operational` is TRUE. Valid values are: TRUE, true, 1, FALSE, false, 0.

.....continued		
Parameter	**Type**	**Description**
uek2_ keep_fabric_operational	boolean	Specifies to keep fabric in operational state during programming if file is programmed using UEK2 custom security. Valid values are: TRUE, true, 1, FALSE, false, 0.
uek2_ skip_startup_seq	boolean	Specifies to skip device startup sequence after programming if file is programmed using UEK2 custom security and `uek2_keep_fabric_operational` is TRUE. Valid values are: TRUE, true, 1, FALSE, false, 0.

Error Codes

Error Code	Description
None	Warning: SPI Directory options are not supported for RTG4 devices and will be ignored.
None	Incorrect Update SPI Image address format. Address must be 32-bit HEX number.
None	Golden SPI Image parameters are required to export SPI directory.
None	Export SPI files is not supported. Export SVF files is not supported.
None	You have not configured custom security options. You can only export bitstream files to program at trusted facility.

Supported Families

	Supported Families	Supported Libero SoC Versions
PolarFire	yes	v12.4+
RTG4	yes	v12.4+
SmartFusion2	yes	v12.4+
IGLOO2	yes	v12.4+

Example
Export bitstream file for design with default security:

```
export_bitstream_file \
-trusted_facility_file 1 \
-trusted_facility_file_components {FABRIC}
```

Export bitstream file for design with custom security options.

Export bitstreams to master, uek1 and uek2 encrypted files. Master file to include security, fabric components and Export Pass Key in Plaintext, uek1 and uek2 encrypted files to include FABRIC with Like new Zeroization option enabled.

```
export_bitstream_file\
-file_name {fftousram_new} \
-export_dir
{X:\10_docs_review\pf2.2_sp1\Programming_sars\99412\clkint_fftousram_ac_latch_launch\des
igner\fftousram_new\export} \
-format {PPD DAT STP HEX} \
```

```
-for_ihp 1 \
-master_file 1 \
-master_file_components {SECURITY FABRIC} \
-encrypted_uek1_file 1 \
-encrypted_uek1_file_components {FABRIC} \
-encrypted_uek2_file 1 \
-encrypted_uek2_file_components {FABRIC} \
-trusted_facility_file 0 \
-trusted_facility_file_components {} \
-zeroization_likenew_action 1 \
-zeroization_unrecoverable_action 0 \
-master_backlevel_bypass 0 \
-uek1_backlevel_bypass 0 -uek2_backlevel_bypass 0 \
-master_include_plaintext_passkey 1 \
-uek1_include_plaintext_passkey 0 \
-uek2_include_plaintext_passkey 0
```

The following example intented for SmartFusion2 and IGLOO2 families, exports SPI directory for programming recovery:

```
export_bitstream_file \
-add_golden_image 1 \
-golden_image_address {1111} \
-golden_image_design_version {1} \
-add_update_image 1 \
-update_image_address {1211} \
-update_image_design_version {1}
```

The following example exports bitstream file for design with MSS/serialization clients. This example failed in case of PolarFire family.

```
export_bitstream_file \
-file_name {mss1} \
-format {STP} \
-trusted_facility_file 1 \
-trusted_facility_file_components {FABRIC} \
-serialization_stapl_type {SINGLE} \
-serialization_target_solution {FLASHPRO_3_4_5}
```

2.29 export_bsdl_file

Description
This Tcl command exports the BSDL to a specified file. The BSDL file provides a standard file format for electronics testing using JTAG. It describes the boundary scan device package, pin description and boundary scan cell of the input and output pins. BSDL models are available as downloads for many Microchip SoC devices. The exported file has a *.bsd file name extension.

```
export_bsdl_file -file {absolute or relative path and name of BSDL file}
```

Arguments

Parameter	Type	Description
file	string	Specifies the absolute or relative path and name of the *.bsd file. If the specified file path is missing, the file is created in the <project_name>/designer/<design_name> directory.

Error Codes

Error Code	Description
None	Parameter 'file' has illegal value.

.....continued	
Error Code	**Description**
None	Required parameter 'file' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command exports the BSDL to a sd1.bsd file.

```
export_bsdl_file -file {E:/designs/export/sd1.bsd}
```

2.30 export_component_to_tcl

Description
This Tcl command exports the Tcl command for the selected component. The components can be SmartDesign components, configured cores and HDL+ cores.

```
export_component_to_tcl \
-component component_name \
[-library library_name ] \
[-package package_name ] \
file file_path
```

Arguments

Parameter	Type	Description
component	string	Specifies the name of the component for which the Tcl command is exported. It is mandatory.
library	string	Specifies the name of the library the component belongs to. It is optional.
package	string	Specifies the name of the package the HDL+core belongs to. It is optional.
file	string	Specifies the path where you wish to export the Tcl file. It is mandatory.

Error Codes

Error Code	Description
None	Please specify a file path for the 'file' parameter.
None	Required parameter 'component' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command exports the Tcl command for "work" library "pattern_gen_checker" component.

```
export_component_to_tcl \
-component {pattern_gen_checker} \
-library {work} -package {} \
-file {./pattern_gen_checker.tcl}
```

2.31 export_design_summary

Description
This Tcl command exports an HTML file containing information about your root SmartDesign in your project. The HTML report provides information on:

- Generated Files
- I/Os
- Hardware Instances
- Firmware
- Memory Map

```
export_design_summary -file {absolute or relative path and name of HTML file}
```

Arguments

Parameter	Type	Description
file	string	Specifies the absolute or relative path and name of file where you wish to export the HTML file. It is mandatory.

Error Codes

Error Code	Description
None	Required parameter 'file' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example exports an HTML to specified file which contains information about your root SmartDesign.

```
export_design_summary -file {D:/Designs/test/sd1.html}
```

2.32 export_firmware

Description
This Tcl command exports design firmware configuration data, which consists of:

- Component configuration for MSS/HPMS, FDDR and SERDES blocks instantiating your design.
- Compatible firmware drivers for your peripherals

It also creates a workspace and project specific to the IDE tool of your choice (SoftConsole, Keil or IAR).

To open your exported firmware projects, you must invoke the third-party development tool (SoftConsole, Keil or IAR) outside Libero SoC.

If you make any changes to your design, you must re-export firmware.

```
export_firmware \
-export_dir {absolute or relative path} \
-create_project {0|1} \
-software_ide {SoftConsole | Keil | IAR EWARM}
```

Arguments

Parameter	Type	Description
export_dir	string	Specifies absolute or relative path and name of folder for the exported firmware. Default exported firmware created<project_name>/firmware direcotry.
create_project	boolean	Generates the workspace and project for the specified IDE tool. Default is 0. Valid values are: TRUE, 1, true, FALSE, 0 or false.
software_ide	string	Specifies one of three IDE tool name: SoftConsole | IAR EWARM | Keil. If you use `-create_project` parameter and `-software_ide` paramter at the same time, Libero exports the workspace and project for that Software IDE tool to the `export_path {SoftConsole|Keil|IAR}` folder. Default is IAR EWARM.

Error Codes

Error Code	Description
None	Required parameter 'export_dir' is missing.
None	Parameter 'export_dir' has illegal value.

Supported Families

	Supported Families	Supported Libero SoC Versions
PolarFire	no	v12.4+
RTG4	no	v12.4+
SmartFusion2	yes	v12.4+

.....continued		
	Supported Families	**Supported Libero SoC Versions**
IGLOO2	no	v12.4+

Example
The following command export design firmware configuration data, generates the workspace and project for the SoftConsole IDE tool.

```
export_firmware \
-export_dir {D:\Designs\software_drivers} \
-create_project {1} \
-software_ide {SoftConsole}
```

2.33 export_fp_pdc

Description
This Tcl command exports the Floorplanning Physical Design Constraint (*.pdc) File. You can export the Floorplan PDC file from Constraint Manager > I/O Attributes or Constraint Manager Floor Planner or from File menu. Constraints can be exported to PDC file for reference, but must be manually added to an existing PDC or imported via the Constraints Editor for the changes to affect the final paced and routed design. The exported file has *.pdc file name extension. Before exporting, you need to run 'Place and Route'.

```
export_fp_pdc \
-file { absolute or relative path and name of *.pdc file } \
-mode { PDC_PLACE | PDC_FULL_PLACEMENT }
```

Arguments

Parameter	Type	Description
file	string	Specifies absolute or relative path and name of the *.pdc file. It is mandatory.
mode	string	Choose the type of information that you want to export. Use `PDC_PLACE` to export user's floorplanning constraints, for example, fixed logic and regions. Use `PDC_FULL_PLACEMENT` to export information about all of the physical design constraints (I/O constraints, I/O Banks, routing constraints, region constraints, global and local clocks). This is an optional parameter. Default is `PDC_PLACE`.

Error Codes

Error Code	Description
None	Required parameter 'file' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
IGLOO2	v12.4+

Example
The following example exports information about all of the physical designs constraints (I/O constraints, I/O Banks, routing constraints, region constraints, global and local clocks).

```
export_fp_pdc \
-file {E:/designs/export/sd1_fp.pdc} \
-mode {PDC_FULL_PLACEMENT}
```

See Also
- 2.35 export_io_pdc

2.34 export_ibis_file

Description
This Tcl command exports the IBIS (Input/Output Buffer Information Specification) model report. The IBIS model report provides an industry-standard file format for recording parameters like driver output impedance, rise/fall time, and input loading, which may then be used by software applications such as Signal Integrity tools or IBIS simulators. The exported file has a *.ibs(name <root>.ibs) file name extension.

```
export_ibis_file -file {absolute or relative path and name of *.ibs file}
```

Arguments

Parameter	Type	Description
file	string	Specifies the absolute or relative path and name of IBIS file to export. If the file path is not specified, the file is created in `<project_name>/designer/<design_name>` directory.

Error Codes

Error Code	Description
None	Parameter 'file' has illegal value.
None	Required parameter 'file' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command exports the IBIS (Input/Output Buffer Information Specification) model report.

```
export_ibis_file -file {E:/designs/export/sd1.ibs}
```

2.35 export_io_pdc

Description
This Tcl command exports the I/O Physical Design Constraint (*.pdc) File. You can export the I/O PDC file from Constraint Manager > I/O Attributes or Constraint Manager Floor Planner or from File menu. Constraints can be exported to PDC file for reference, but must be manually added to an existing PDC or imported via the Constraints Editor for the changes to affect the final paced and routed design. The exported file has *.pdc file name extension. Before exporting, you need to run 'Place and Route'.

```
export_io_pdc -file { absolute or relative path and name of *.pdc file }
```

Arguments

Parameter	Type	Description
file	string	Specifies absolute or relative path and name of the *.pdc file. It is mandatory. There may be multiple -file arguments (see example below).

Error Codes

Error Code	Description
None	Required parameter 'file' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example exports information about all of the physical design constraints (I/O constraints, I/O Banks, routing constraints, region constraints, global and local clocks). Created both sd_io1.pdc and sd_io2.pdc files.

```
export_io_pdc -file {./sd_io1.pdc} \
-file {sd_io2.pdc}
```

See Also
- 2.33 export_fp_pdc

2.36 export_job_data

Description
This Tcl command configures the parameters for the Job Manager Data Container file (JDC) to be exported from Libero and used by Job Manager. The exported file has *.jdc file name extension. All parameters are optional. Default values are used if parameters are omitted.

```
export_job_data -file_name name \
-export_dir {absolute or relative path of the exported file} \
-components "SECURITY | FABRIC | ENVM" \
-include_spi_flash value
```

Arguments

Parameter	Type	Description
file_name	string	Name of the file that will be saved. If omitted, it will be the design name.
export_dir	string	Specified absolute or relative path where the file will be saved. If omitted, the file will be saved in the `Libero projects/designer/<design_name>/export` directory with *.jdc file extension.
components	string	Specifies the components of the design that will be saved to the file. The value can be any one or a combination of `SECURITY`, `FABRIC`, `SNVM` and `ENVM` if they are available in the design. If the parameter is omitted, all available components of the design will be saved. **Note:** The SECURITY component must be selected if user security is initialized for the current Libero design.
include_spi_flash	boolean	Use the Configure Design Initialization Data and Memories tool to configure this option. Valid value is 0 and 1. Default is 0.

Error Codes

Error Code	Description
None	PolarFire: No bitstream components or SPI Flash data are selected to include in exported programming data file.
None	PolarFire: SPI Flash Memory is not configured. Use the Configure Design Initialization Data and Memories tool to configure it.
None	IGLOO2, PolarFire: The eNVM component is not available in the current design.
None	SmartFusion2: The eNVM component is not supported in the current design.
None	PolarFire: Invalid argument value: expecting SECURITY, FABRIC, SNVM, ENVM or FABRIC_SNVM.
None	SmartFusion: components: Invalid argument value: 'FABRIC_SNVM' (expecting SECURITY, FABRIC or ENVM)
None	SmartFusion2, IGLOO2: Parameter 'include_spi_flash' is not defined. Valid command formatting is 'export_job_data [-file_name "file_name"] \ [-export_dir "export_dir"] \ [-components "[SECURITY | FABRIC | ENVM]+"]
None	SmartFusion2, IGLOO2: There are no bitstream components to include in exported programming data file.

.....continued	
Error Code	**Description**
None	IGLOO2: Invalid argument value: 'FM' (expecting SECURITY, FABRIC or ENVM).

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example exports Job Manager Data Container file (sd1.jdc) with FABRIC component.

```
export_job_data \
-file_name {sd1} \
-export_dir {D:\sd_prj\test3T\designer\sd1\export} \
-components {FABRIC}
```

2.37 export_netlist_file

Description
This Tcl command exports the netlist after the compile state has completed. The netlist can be either Verilog or VHDL. Microchip recommends exporting the netlist after the compile state has successfully completed.

```
export_netlist_file \
                    -file { absolute or relative path and filename for netlist } \
                    -vhdl { value }
```

Arguments

Parameter	Type	Description
file	string	Specifies the absolute or relative path and name of netlist file. If specified relative path of netlist file created in <project_name>/designer/<design_name> directory.
vhdl	boolean	Generates the netlist in VHDL (when set to 1) or Verilog (when set to 0). Default is 0 (Verilog netlist).

Error Codes

Error Code	Description
None	Required parameter 'file' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command exports the netlist (Verilog) after the compile state has completed.

```
export_netlist_files \
-file {E:/designs/export/sd1/sd1.v} \
-vhdl 0
```

2.38 export_pin_reports

Description
This Tcl command configures and exports a pin report file to a specified folder/directory location. The pin report lists the pins in your device sorted according to your preference: sort by Port Name or Sorted by Package Pin Name. The pin report generates two files:

- <design>_pinrpt_name.rpt - pin report sorted by name.
- <design>_pinrpt_number.rpt - pin report sorted by pin number.

Export Pin Report generates a Bank Report by default; the filename is `<design>-bankrpt.rpt`. Export Pin Report also generates an I/O Register Combining Report listing the I/Os which have been combined into a Register for getting timing performance. You must select at least one report.

```
export_pin_reports \
-export_dir {absolute path to folder location} \
-pin_report_by_name {value} \
-pin_report_by_pkg_pin {value} \
-bank_report {value} \
-io_report {value}
```

Arguments

Parameter	Type	Description
export_dir	string	Specifies the folder, disk location where you want to save pin report. It is mandatory.
pin_report_by_name	integer	Set to 1 to have the pin report sorted by pin name. By default, this box is checked.
pin_report_by_pkg_pin	integer	Set to 1 to have pin report sorted by package pin number, 0 to not sort by package pin number. By default, this box is checked.
bank_report	integer	Set to 1 to generate the I/O bank report, 0 to not generate the report. By default, this box is checked.
io_report	integer	Set to 1 to generate the I/O report, 0 to not generate the report. By default, this box is checked.

Error Codes

Error Code	Description
None	Required parameter 'export_dir' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command exports pin report sorted by pin name and not sorted by package pin number, generated I/O and I/O bank reports.

```
export_pin_reports \
-export_dir {E:/designs/export} \
-pin_report_by_name {1} \
-pin_report_by_pkg_pin {0} \
-bank_report {1} \
-io_report {1}
```

2.39 export_profiles

Description
This Tcl command exports your tool profiles. Performs the same action as the **Export Profiles** dialog box.

```
export_profile -file { absolute path and name of exported file } [-export value ]
```

Arguments

Parameter	Type	Description
file	string	Specifies the absolute path and name to the exported profile file.
export	string	Specifies your profile export options. The following table shows the acceptable values for this argument: • predefined - exports only predefined profiles • user - exports only user profiles • all - exports all profiles

Error Codes

Error Code	Description
None	Parameter 'file' : the file '/workspace/zarine_workspace/prj_mng/profiles_command/z_exp.tcl' has got an invalid extension. Valid extension is 'ini'

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command exports all profiles to the file 'all_profiles'.

```
export_profiles -file all_profiles [-export all]
```

See Also
- 2.4 add_profile
- 2.25 edit_profile
- 2.65 remove_profile
- 2.74 select_profile

2.40 export_prog_job

Description
This Tcl command configures the parameters for the FlashPro Express programming job to be exported.

Note: The Programming Mode (JTAG/SPI-Slave) setting from the Programming Connectivity and Interface tool will be exported in the job file.
RTG4 devices do not support the security options supported by SmartFusion2 and IGLOO2 devices.

The syntax for the export programming job Tcl command for SmartFusion2 and IGLOO2 is shown below:

```
export_prog_job \
-job_file_name {file} \
-export_dir {absolute or relative path} \
-bitstream_file_type {TRUSTED_FACILITY | MASTER | UEK1 | UEK2} \
-bitstream_file_components {SECURITY | FABRIC | ENVM} \
-include_plaintext_passkey {0 | 1} \
-design_bitstream_format {PPD | STP} \
-prog_optional_procedures \
{action1 | procedure1 | procedure2 ; action2 | procedure1 | procedure2 | procedure3;}
```

The syntax for the export programming job Tcl command for RTG4 is below:

```
export_prog_job \
-job_file_name {file} \
-export_dir {dir} \
-force_rtg4_otp {0 | 1} \
-design_bitstream_format {PPD | STP}
```

The syntax for the export programming job Tcl command for PolarFire is below:

```
export_prog_job \
-job_file_name {file} \
-export_dir {dir} \
-bitstream_file_type {TRUSTED_FACILITY | MASTER | UEK1 | UEK2} \
-bitstream_file_components {SECURITY | ENVM | FABRIC | SNVM | FABRIC_SNVM} \
-zeroization_likenew_action {0 | 1} \
-zeroization_unrecoverable_action {0 | 1} \
-program_design {0 | 1} \
```

```
-program_spi_flash {0 | 1} \
-include_plaintext_passkey {0 | 1} \
-design_bitstream_format {PPD | STP} \
-prog_optional_procedures \
{action1|procedure1|procedure2;action2|procedure1|procedure2|procedure3;} \
-skip_recommended_procedures \
{action1 | procedure1 | procedure2 ; action2 | procedure1 | procedure2 | procedure3;}
```

Arguments

Parameter	Type	Description
job_file_name	string	The name of the exported file. Name must start with design name. If omitted, design name will be used.
export_dir	string	Location where the job file will be saved; any folder can be specified (absolute or release path). The default folder is the Libero designer/<design_name>/export folder.
force_rtg4_otp	boolean	Enforces the use of one-time programming (OTP). This argument is optional. Valid values are: TRUE, 1, true, FALSE, 0 or false. The default value is 0.
bitstream_file_type	string	Bitstream file to be included in the programming job. Only one bitstream file can be included in a programming job. Possible values are: TRUSTED_FACILITY, MASTER, UEK1, UEK2 or UEK3. If omitted, type will be TRUSTED_FACILITY.
bitstream_file_components	list of components	The list of components to be included in the programming job. Components should be delimited by space. `bitstream_file_components` can be any one of SECURITY, ENVM, FABRIC SNVM (only PolarFire) or FABRIC_SNVM (only PolarFire) any combination of them. **Notes:** 1. The SECURITY option is available in `-bitstream_file_components` only when file type is MASTER in `-bitstream_file_type`. 2. SNVM must always be programmed with FABRIC. 3. Security-only programming must be performed only on erased or new devices. If performed on a device with fabric programmed, the fabric will be disabled after performing security-only programming. You must reprogram the fabric to re-enable it.
include_plaintext_passkey	boolean	Includes plaintext passkey. Valid values are: TRUE, 1, true, FALSE, 0 or false. This argument is optional. Default is 0.
design_bitstream_format	string	Specifies the bitstream file formats to be exported. Space is used as a delimiter. The value can be any one of PPD, STP. If omitted, the bitstream file will be in PPD format. • PPD will allow for improved programming times with FlashPro6 programmer. • Use STAPL for static algorithm and data from release to release.
prog_optional_procedures	string	Specifies optional procedures to program. Format: action1 | procedure1 | procedure2; action2 | procedure1 | procedure2 | procedure3; Available actions and procedures depend on the selected bitstream components. Mandatory procedures are always included. See the section Configure Actions and Procedures for supported actions and procedures.

.....continued

Parameter	Type	Description
zeroization_likenew_action	boolean	Specifies that all data will be erased and the device can be reprogrammed immediately. Valid values are TRUE, 1, true, FALSE, 0 or false. Default is 0.
zeroization_unrecoverable_action	boolean	Specifies that all data will be erased. The device cannot be reprogrammed and it must be scrapped. Valid values are TRUE, 1, true, FALSE, 0 or false. Default is 0.
program_design	boolean	Specifies to program the design. This argument is optional. Valid values are TRUE, 1, true, FALSE, 0 or false. Default is 1.
program_spi_flash	boolean	Specifies to program SPI Flash. Configure it before using. Use the Configure Design Initialization Data and Memories tool to configure it. This argument is optional. Valid values are TRUE, 1, true, FALSE, 0 or false. Default is 0.
skip_recommended_procedures	string	Specifies recommended procedures to skip. Format: action1 | procedure1 | procedure2; action2 | procedure1 | procedure2 | procedure3; See the section Configure Actions and Procedures for supported actions and procedures.

Return Type	Description
integer	Returns 0 on success, 1 on failure.

Error Codes

Error Code	Description
None	bitstream_file_type: Invalid argument value: 'A' (expecting MASTER, UEK1, UEK2, UEK3 or TRUSTED_FACILITY).
None	bitstream_file_components: Invalid argument value: 'A' (expecting SECURITY, FABRIC, SNVM, ENVM or FABRIC_SNVM).
None	Custom security component is available for Master bitstream file only.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
SmartFusion2/IGLOO2

The following example exported FlashPro Express programming job in X:\12.0_Release\g4_fftousram\designer\top\export\top file. Included MASTER Bitstream file with PPD format, SECURITY and FABRIC components, plaintext passkey and specified PROGRAM, DO_VERIFY; optional procedures to program:

```
export_prog_job \
-job_file_name {top} \
-export_dir {X:\12.0_Release\g4_fftousram\designer\top\export} \
```

```
-bitstream_file_type {MASTER} \
-bitstream_file_components {SECURITY FABRIC} \
-include_plaintext_passkey 1 \
-design_bitstream_format {PPD} \
-prog_optional_procedures {PROGRAM | DO_VERIFY;}
```

RTG4

The following example exported FlashPro Express programming job in X:\12.0_Release\rtg4_ff_usram\designer\top\export\top file with PPD format and also enforces the use of one-time programming (OTP):

```
export_prog_job \
-job_file_name {top} \
-export_dir {X:\12.0_Release\rtg4_ff_usram\designer\top\export} \
-force_rtg4_otp 1 \
-design_bitstream_format {PPD}
```

PolarFire

The following example exported FlashPro Express programming job in X:\12.0_Release\pf_fftousram_ac_latch_launch\designer\fftousra\fftousram file. Included MASTER Bitstream file with PPD format, SECURITY, FABRIC and SNVM components, plaintext passkey and specified PROGRAM, DO_VERIFY; optional procedures to program, programs the design and device is not reprogrammed:

```
export_prog_job \
-job_file_name {fftousram} \
-export_dir {X:\12.0_Release\pf_fftousram_ac_latch_launch\designer\fftousram\export} \
-bitstream_file_type {MASTER} \
-bitstream_file_components {SECURITY FABRIC SNVM} \
-zeroization_likenew_action 0 \
-zeroization_unrecoverable_action 0 \
-program_design 1 \
-program_spi_flash 0 \
-include_plaintext_passkey 0 \
-design_bitstream_format {PPD} \
-prog_optional_procedures {PROGRAM | DO_VERIFY;} \
-skip_recommended_procedures {VERIFY_DIGEST | DO_ENABLE_FABRIC;}
```

2.41 export_script

Description

This Tcl command explicitly exports the Tcl command equivalents of the current Libero session. With this command you can re-execute the same commands interactively or in batch.

You must supply a file name with the `-file` parameter and the `-relative_path` parameter to specify whether an absolute or relative path is used in the exported script file.

```
export_script \
-file { absolute or relative path to constraint file } \
-relative_path <value>
```

Arguments

Parameter	Type	Description
file	string	Specifies the absolute or relative path and name to the exported script file.
relative_path	boolean	Sets your option to use a relative or absolute path in the exported script; use 1 for relative path, 0 for absolute.

Error Codes

Error Code	Description
None	Parameter 'file' has illegal value.
None	Required parameter 'relative_path' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command exports the Tcl command equivalents of the current Libero session.

```
export_script -file {./exported.tcl} -relative_path 1
```

2.42 export_sdc_file

Description
This Tcl command exports the file for timing constraints. The exported file has a `*.sdc` file name extension.

```
export_sdc_file -file { absolute path and name of *.sdc file }
```

Arguments

Parameter	Type	Description
file	string	Specifies the SDC file to export.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command exports the SDC file for timing constraints to sd1.sdc.

```
export_sdc_file -file {E:/designs/export/sd1.sdc}
```

2.43 generate_component

Description
This Tcl command generates a SmartDesign or a core component VHDL code. After generating component, the VHDL file is placed in the `<project_folder>/component/work/<component_name>` folder.

```
generate_component \
-component_name component_name \
[-recursive 0|1 ]
```

Arguments

Parameter	Type	Description
component_name	string	Specifies the name of the SmartDesign component or the core component to be generated. It is mandatory.
recursive	integer	Specifies if a SmartDesign component needs to be generated recursively. It is optional. It is '0' by default and generates only the specified component. If set to '1', all the dependent components which are in ungenerated state will be generated along with the SmartDesign component. It is recommended to generate all components individually.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command generates SmartDesign "sd2" only.

```
generate_component -component_name {sd2}
```

The following command generates SmartDesign "TOP" and all its dependent components which are in ungenerated state.

```
generate_component -component_name {TOP} -recursive 1
```

See Also
- 2.22 delete_component
- 2.49 import_component
- 2.30 export_component_to_tcl

2.44 generate_sdc_constraint_coverage

Description
This Tcl command generates the constraint coverage report. The constraint coverage report contains information about the coverage of the paths from associated SDC constraints in the design. Two constraints coverage reports can be generated, one for Place and Route and one for Timing Verification.

To run this command, there is no need to run Place-and-Route first, but the design must be in the post-synthesis state. The generated constraint coverage reports (*.xml) are listed in the Reports tab and are physically located in the `<prj_folder>/designer/<module>/*constraints_coverage.xml` file.

Note: This command cannot be run until Compile has been run. Constraint Coverage Reports can be generated only after synthesis. A warning message appears if the design is not in the post-synthesis state when this button is clicked.

```
generate_sdc_constraint_coverage -tool {PLACEROUTE | VERIFYTIMING}
```

Arguments

Parameter	Type	Description
tool	string	Specifies whether the constraint coverage report is based on the SDC constraint file associated with Place and Route or associated with Timing Verification. Use this dialog box to configure the 'Verify Timing' tool to generate a timing constraint coverage report and detailed static timing analysis and violation reports based on different combinations of process speed, operating voltage, and temperature. This is mandatory.

Error Codes

Error Code	Description
None	Required parameter 'tool' is missing.

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
This command generates the SDC Constraint Coverage report for the SDC file associated with Place and Route.

```
generate_sdc_constraint_coverage -tool {PLACEROUTE}
```

This command generates the SDC Constraint Coverage report for the SDC file associated with Timing Verification.

```
generate_sdc_constraint_coverage -tool {VERIFYTIMING}
```

See Also
- Understanding Constraints Coverage Reports

2.45 get_libero_release

Description
This Tcl command returns the release number of the Libero SoC release. The value that is returned is the same as the release number that is displayed in the *Help > About Libero Window*.

```
get_libero_release
```

Arguments

Return Type	Description
string	Return the release number of the Libero SoC release.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example displays Libero release number with var1 variable.

```
get_libero_release
#save into a variable
set var1 [get_libero_release]
#display the variable
puts "Libero Release is $var1"
```

You will see output similar to the following.

```
Libero Version is 12.6.0
```

See Also
- 2.46 get_libero_version

2.46 get_libero_version

Description
This Tcl command returns the version number of the Libero SoC version. The value that is returned is the same as the version number that is displayed in the *Help > About Libero window*.

```
get_libero_version
```

Arguments

Return Type	Description
string	Returns the version number of the Libero SoC version.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example displays Libero version number with var2 variable.

```
get_libero_version
#save into a variable
set var2 [get_libero_version]
#display variable
puts "Libero Version is $var2"
```

You will see output similar to the following.

```
Libero Version is 12.6.0
```

See Also
- 2.45 get_libero_release

2.47 get_tool_options

Description
This Tcl command is used to get the configured options/parameters of a tool in the Libero Design Flow. It can be used to get the value of a single tool option or multiple tool options.

```
get_tool_options -name {tool_name} -params {parameter_names}
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of tool for which you wish to know the configured tool options. It is mandatory. • SYNTHESIZE • COMPILE • PLACEROUTE • VERIFYTIMING • VERIFYPOWER • EXPORTNETLIST • CONFIGURE_PROG_OPTIONS • SPM • PROGRAMDEVICE • GENERATEPROGRAMMINGFILE • CONFIGURE_ACTION_PROCEDURES • PROGRAM_SPI_FLASH_IMAGE • SPM_OTP • FLASH_FREEZE • INIT_LOCK • IO_PROGRAMMING_STATE • UPDATE_ENVM • EXPORTSDF
params	string	Specifies the tool options/parameters for which you want to know the configured value. It is optional. It can either take single parameter or multiple parameters at a time. If omitted, all configured options/parameters will returned.

Return Type	Description
options/parameters	Returns the configured options/parameters of a tool in the Libero Design Flow.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

The following table shows all the tools for which this command is applicable.

Tool Name (Tcl)	Tool Display Name	Supported Families
SYNTHESIZE	Synthesize	SmartFusion2, IGLOO2, RTG4, PolarFire
COMPILE	Compile	SmartFusion2, IGLOO2, RTG4, PolarFire
PLACEROUTE	Place and Route	SmartFusion2, IGLOO2, RTG4, PolarFire

.....continued

Tool Name (Tcl)	Tool Display Name	Supported Families
VERIFYTIMING	Verify Timing	SmartFusion2, IGLOO2, RTG4, PolarFire
VERIFYPOWER	Verify Power	SmartFusion2, IGLOO2, RTG4, PolarFire
EXPORTNETLIST	File > Export > Netlist	SmartFusion2, IGLOO2, RTG4, PolarFire
CONFIGURE_PROG_OPTIONS	Configure Programming Options	SmartFusion2, IGLOO2, RTG4, PolarFire
SPM	Configure Security	SmartFusion2, IGLOO2, PolarFire
PROGRAMDEVICE	Run PROGRAM Action	SmartFusion2, IGLOO2, RTG4, PolarFire
GENERATEPROGRAMMINGFILE	Generate Bitstream	SmartFusion2, IGLOO2, RTG4, PolarFire
CONFIGURE_ACTION_PROCEDURES	Configure Actions and Procedures	SmartFusion2, IGLOO2, RTG4, PolarFire
FLASH_FREEZE	Configure Flash*Freeze	SmartFusion2, IGLOO2
INIT_LOCK	Configure Register Lock Bits	SmartFusion2, IGLOO2, RTG4, PolarFire
IO_PROGRAMMING_STATE	Configure I/O States During JTAG Programming	SmartFusion2, IGLOO2, RTG4, PolarFire
UPDATE_ENVM	Update eNVM Memory Content	SmartFusion2, IGLOO2, RTG4, PolarFire
EXPORTSDF	Generate Back Annotated Files	SmartFusion2, IGLOO2, RTG4, PolarFire
PROGRAM_SPI_FLASH_IMAGE	Run PROGRAM SPI_IMAGE Action	PolarFire
SPM_OTP	Configure Permanent Locks for Production	PolarFire

Example

1. The following example gets the value of a synthesize tool single-RETIMING parameter.

```
puts [get_tool_options -name {SYNTHESIS} -params {RETIMING}]
Output: true
```

2. The following example gets the values of multiple parameter.

```
set p [get_tool_options -name {PLACEROUTE} \
-params {REPAIR_MIN_DELAYEFFORT_LEVEL IOREG_COMBINING}]
puts "$p"
Output:
REPAIR_MIN_DELAY true EFFORT_LEVEL true IOREG_COMBINING false
```

3. When no parameters are given, in this case the following command returns all configured parameters of a VERIFYTIMING Libero tool.

```
puts [get_tool_options -name {VERIFYTIMING}]
Output:
CONSTRAINTS_COVERAGE true FORMAT XML MAX_TIMING_FAST_HV_LT false
MAX_TIMING_MULTI_CORNER true MAX_TIMING_SLOW_LV_HT false MAX_TIMING_SLOW_LV_LT
false MAX_TIMING_VIOLATIONS_FAST_HV_LT false MAX_TIMING_VIOLATIONS_MULTI_CORNER
```

```
true MAX_TIMING_VIOLATIONS_SLOW_LV_HT false MAX_TIMING_VIOLATIONS_SLOW_LV_LT
false MIN_TIMING_FAST_HV_LT false MIN_TIMING_MULTI_CORNER true
MIN_TIMING_SLOW_LV_HT false MIN_TIMING_SLOW_LV_LT false
MIN_TIMING_VIOLATIONS_FAST_HV_LT false MIN_TIMING_VIOLATIONS_MULTI_CORNER true
MIN_TIMING_VIOLATIONS_SLOW_LV_HT false MI
```

2.48 get_tool_state

Description
This Tcl command is used to get the state of a tool in the Libero Design Flow. It can be run on all tools which have a tool state in the UI (such as green check mark/error/out of date/has not run yet/tool run with warnings etc). The output of this Tcl command is equivalent to the tooltip message seen on the tool in the Libero Design Flow window in the UI.

```
get_tool_state -name {tool_name}
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of tool for which you wish to get the tool state. It is mandatory. • SYNTHESIZE • COMPILE • PLACEROUTE • VERIFYTIMING • VERIFYPOWER • PROGRAMDEVICE • GENERATE_MEMORY_MAP • GENERATEPROGRAMMINGFILE • EXPORTPROGRAMMINGFILE • EXPORTPROGRAMMINGJOB • EXPORTJOBDATA • EXPORTNETLIST • EXPORTSMARTDEBUGDATA • PUBLISHBLOCK • EXPORTSDF • GENERATEPROGRAMMINGDATA

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

The following table shows the possible tool states. Note that not all tool states mentioned apply to every tool.

Tool State	Description
Tool successfully run	When the execution is successful.

.....continued

Tool State	Description
Device/Project settings have changed	When the Device/Project settings that affect the tool are modified.
Tool configuration has changed	When a tool's configuration options are changed.
Tool failed	When a tool execution fails.
Tool has not run yet	When a tool has not run yet.
Tool inputs are out of date	When a parent tool state is cleaned or when a design source is modified or something in another tool is modified that the current tool is dependent on.
Tool executed	When a tool ran successfully.
Timing constraints have not been met	When the Verify Timing tool ran successfully but the design has timing violations.
Timing constraints have been met	When the Verify Timing tool ran successfully and there are no timing violations for the design.

The following table shows all the tools for which this command is applicable.

Tool Name (Tcl)	Tool Display Name	Supported Families
SYNTHESIZE	Synthesize	SmartFusion2, IGLOO2, RTG4, PolarFire
COMPILE	Compile	SmartFusion2, IGLOO2, RTG4, PolarFire
PLACEROUTE	Place and Route	SmartFusion2, IGLOO2, RTG4, PolarFire
VERIFYTIMING	Verify Timing	SmartFusion2, IGLOO2, RTG4, PolarFire
VERIFYPOWER	Verify Power	SmartFusion2, IGLOO2, RTG4, PolarFire
PROGRAMDEVICE	Run PROGRAM Action	SmartFusion2, IGLOO2, RTG4, PolarFire
GENERATE_MEMORY_MAP	Generate Memory Map	SmartFusion2, IGLOO2, RTG4
GENERATEPROGRAMMINGFILE	Generate Bitstream	SmartFusion2, IGLOO2, RTG4, PolarFire
EXPORTPROGRAMMINGFILE	Export Bitstream	SmartFusion2, IGLOO2, RTG4, PolarFire
EXPORTPROGRAMMINGJOB	Export FlashPro Express Job	SmartFusion2, IGLOO2, RTG4, PolarFire
EXPORTJOBDATA	Export Job Manager Data	SmartFusion2, IGLOO2, PolarFire
EXPORTNETLIST	File > Export > Netlist...	SmartFusion2, IGLOO2, RTG4, PolarFire
EXPORTSMARTDEBUGDATA	Export SmartDebug Data	SmartFusion2, IGLOO2, RTG4, PolarFire
PUBLISHBLOCK	Publish Block	SmartFusion2, IGLOO2, RTG4, PolarFire
EXPORTSDF	Generate Back Annotated Files	SmartFusion2, IGLOO2, RTG4, PolarFire
GENERATEPROGRAMMINGDATA	Generate FPGA Array Data	SmartFusion2, IGLOO2, RTG4, PolarFire

Example
The following example gets the state of synthesize tool in the Libero Design flow.

```
set state [get_tool_state -name {SYNTHESIZE}]
puts "$state"
Output: Tool successfully run
```

2.49 import_component

Description
This Tcl command imports a component *.cxf file into the Libero project. After import, the *.cxf file is placed in the `<project_folder>/component/work/<component_name>`folder.

Note: Only the *.cxf file format is supported for component import.

```
import_component -file <path_to_component.cxf>
```

Arguments

Parameter	Type	Description
file	string	The `-file` argument specifies the location of the component *.cxf file to import. Both absolute and relative paths are supported.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example imports a component `my_mult.cxf` file into the Libero project.

```
import_component -file {D:/test/my_design/my_mult.cxf}
```

See Also
- 2.50 import_component_data
- 2.22 delete_component
- 2.43 generate_component

2.50 import_component_data

Description
A Libero SoC general-purpose Tcl command to import component data into an existing Libero project. Component refers to MDDR, FDDR and SERDES peripherals in SmartFusion2 devices. Component Data refers to initialization/ configuration register values (*init_reg or *init.mem files) of those peripherals. Use this command if and when:

1. The synthesized netlist or HDL files in the existing Libero SoC project contains no component (MDDR, FDDR, and SERDES) information.
2. You want to add components (MDDR, FDDR or SERDES) into the existing design.

Note: The eNVM config file can have any name. Either *_init.reg (register configuration file) or *.mem files (memory files) can be used. The two cannot be used together in the same `import_component_data` command.

```
import_component_data
-module root # name of the top_level (root) \
```

```
-fddr file_path_and_name # has to be FDDR_init.reg or .mem \
-mddr file_path_and_name # has to be MDDR_init.reg or .mem \
-serdes0 file_path_and_name # has to be SERDESIF_0_init.reg or .mem \
-serdes1 file_path_and_name # has to be SERDESIF_1_init.reg or .mem \
-serdes2 file_path_and_name # has to be SERDESIF_2_init.reg or .mem \
-serdes3 file_path_and_name # has to be SERDESIF_3_init.reg or .mem \
-envm_cfg file_path_and_name # SmartFusion2, IGLOO2 only \
-uprom_cfg file_path_and_name # RTG4 only
```

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

The component name for IGLOO2 devices may have different file extension (*.mem or *.reg), depending on the Libero SoC release version used to generate the components.

The following is an example of importing design components created with a Libero SoC pre-v11.4 release into an IGLOO2 project.

```
import_component_data \
-module <root> \
-fddr <file_path>/FDDR_init.mem \
-mddr <file_path>/MDDR_init.mem \
-serdes0 <file_path>/SERDESIF_0_init.mem \
-serdes1 <file_path>/SERDESIF_1_init.mem \
-serdes2 <file_path>/SERDESIF_2_init.mem \
-serdes3 <file_path>/SERDESIF_3_init.mem \
-envm_cfg <user_cfg_file_path>
```

The following is an example of importing design components created with Libero SoC v11.4 or subsequent releases into an IGLOO2 project. Note the *.reg file extension.

```
import_component_data \
-module <root> \
-fddr <file_path>/FDDR_init.reg \
-mddr <file_path>/MDDR_init.reg \
-serdes0 <file_path>/SERDESIF_0_init.reg \
-serdes1 <file_path>/SERDESIF_1_init.reg \
-serdes2 <file_path>/SERDESIF_2_init.reg \
-serdes3 <file_path>/SERDESIF_3_init.reg \
-envm_cfg <user_cfg_file_path>
```

The following is an example of importing design components created with a Libero SoC pre-v11.4 release into a SmartFusion2 project.

```
import_component_data \
-module <root> \
-fddr <file_path>/FDDR_init.reg \
-mddr <file_path>/MDDR_init.reg \
-serdes0 <file_path>/SERDESIF_0_init.reg \
-serdes1 <file_path>/SERDESIF_1_init.reg \
-serdes2 <file_path>/SERDESIF_2_init.reg \
-serdes3 <file_path>/SERDESIF_3_init.reg \
-envm_cfg <user_cfg_file_path>
```

The following is an example of importing design components created with Libero SoC v11.4 or a subsequent release into a SmartFusion2 project.

```
import_component_data \
-module <root> \
-envm_cfg <user_cfg_file_path>
```

2.51 import_files

Description
This Tcl command enables you to import design source files and constraint files.

For importing constraint files, import_files has retired the `-pdc` parameter for SmartFusion2, IGLOO2, PolarFire and RTG4. It has been replaced with two new parameters to match the new design flow. Physical Design Constraints (PDC) Tcl must now be divided between I/O attribute and pin information from all floorplanning and timing constraints. These commands must now reside in and be imported as separate files. The new parameters specify the type of *.pdc file being imported.

Use of the `-pdc` parameter with SmartFusion2 or IGLOO2 or RTG4 families will cause an error. The path to the file can be absolute or relative but must be enclosed in curly braces { }.

Use the `-convert_EDN_to_HDL` parameter to convert the EDIF file to HDL and then import the converted HDL file.

Note: The EDIF File is not imported.

```
import_files \
-schematic {file} \
-symbol {file} \
-smartgen_core {file} \
-ccp {file} \
-stimulus {file} \
-hdl_source {file} \
-io_pdc {file} \
-fp_pdc {file} \
-edif {file} \
-sdc {file} \
-crt {file} \
-pin {file} \
-dcf {file} \
-pdc {file} \
-gcf {file} \
-vcd {file} \
-saif {file} \
-simulation {file} \
-profiles {file} \
-cxf {file} \
-templates {file} \
-ccz {file} \
-modelsim_ini {file} \
-library {file} \
-convert_EDN_to_HDL {true | false}
```

Arguments

Parameter	Type	Description
schematic	string	Specifies the schematics you wish to import into your IDE project. Type parameter must be repeated for each file.
symbol	string	Specifies the symbols you wish to import into your IDE project. Type parameter must be repeated for each file.
smartgen_core	string	Specifies the SmartGen Cores you wish to import into your project. Type parameter must be repeated for each file.

.....continued

Parameter	Type	Description
ccp	string	Specifies the ARM or Cortex-M1 cores you wish to import into your project. Type parameter must be repeated for each file.
stimulus	string	Specifies HDL stimulus files you wish to import into your project. Type parameter must be repeated for each file.
hdl_source_folder	string	Name of the HDL folder you want to import into your project.
hdl_source	string	Specifies the HDL source files you wish to import into your project. Type parameter must be repeated for each file.
io_pdc	string	PolarFire only - Specifies the PDC file that contains the I/O attribute and pin information.
fp_pdc	string	PolarFire only - Specifies the PDC file that contains the timing and placement information.
edif	string	Specifies the EDIF files you wish to import into your project. Type parameter must be repeated for each file. This is a mandatory option if you want to convert EDIF to HDL with the `-convert_EDN_to_HDL` option. This option is not supported in PolarFire.
sdc	string	Specifies the SDC constraint files you wish to import into your project. Type parameter must be repeated for each file.
crt	string	Specifies the CRT constraint files you wish to import into your project. Type parameter must be repeated for each file.This option is not supported for PolarFire, SmartFusion2, IGLOO2, and RTG4 families.
pin	string	Specifies the PIN constraint files you wish to import into your project. Type parameter must be repeated for each file. Not supported for PolarFire, SmartFusion2, IGLOO2, and RTG4 families.
dcf	string	Specifies the DCF constraint files you wish to import into your project. Type parameter must be repeated for each file. Not supported for PolarFire, SmartFusion2, IGLOO2, and RTG4 families.
gcf	string	Specifies the GCF constraint files you wish to import into your project. Type parameter must be repeated for each file. Not supported for PolarFire, SmartFusion2, IGLOO2, and RTG4 families.
vcd	string	Specifies the VCD constraint files you wish to import into your project. Type parameter must be repeated for each file.
saif	string	Specifies the SAIF constraint files you wish to import into your project. Type parameter must be repeated for each file.
simulation	string	Specifies the simulation files you wish to import into your Libero SoC project. Type parameter must be repeated for each file.
profiles	string	Specifies the profile files you wish to import into your Libero SoC project. Type parameter must be repeated for each file.
cxf	string	Specifies the CXF (Component) file you wish to import into your Libero SoC project. Type parameter must be repeated for each file.

.....continued		
Parameter	**Type**	**Description**
templates	string	Specifies the template file you wish to import into your project.
ccz	string	Specifies the IP Control core file you wish to import into your project.
modelsim_ini	string	Specifies the ModelSIM INI file that you wish to import into your project.
library	string	Specifies the library file that you wish to import into your project. If a library file is not available it will be created and added to the library.
convert_EDN_to_HDL	boolean	The `-edif` option is mandatory. If the `-edif` option is not specified or the `-convert_EDN_to_HDL` is used with another option, EDIF to HDL conversion will fail.

Error Codes

Error Code	Description
None	Unable to find the file 'a.v'.
None	Cannot import the 'ProASIC Constraint Files'; your selected family does not support GCF constraints.
None	Cannot import the 'Criticality Files'; your selected family does not support CRT constraints.
None	Cannot import the 'Timing Constraint Files'; your selected family does not support DCF constraints.
None	Cannot import the 'Pin Files'; your selected family does not support PIN constraints.
None	Error: Parameter is not defined. Valid command formatting is 'import_files [-convert_EDN_to_HDL "TRUE | FALSE"] \ [-hdl_source_folder "Source folder"]* \ [-library "library"] \ [-cxz "file"]* \ [-cxf "file"]* \ [-ccp "file"]* \ [-crt "file"]* \ [-hdl_source "file"]* \ [-stimulus "file"]* \ [-templates "file"]* \ [-modelsim_ini "file"]* \ [-fp_pdc "file"]* \ [-io_pdc "file"]* \ [-sdc "file"]* \ [-ndc "file"]* \ [-net_fdc "file"]* \ [-icf "file"]* \ [-ccz "file"]* \ [-cpz "file"]* \ [-pin "file"]* \ [-gcf "file"]* \ [-saif "file"]* \ [-schematic "file"]* \ [-simulation "file"]* \ [-smartgen_core "file"]* \ [-symbol "file"]* \ [-verilog_netlist "file"]* \ [-dcf "file"]* \ [-profiles "file"]* \ [-vcd "file"]*

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The command below imports the HDL source files file1.vhd and file2.vhd.

```
import_files -hdl_source file1.vhd -hdl_source file2.vhd
```

2.52 loopback_test

Description
This Tcl command used to start and stop the loopback tests. Loopback data stream patterns are generated and checked by the internal SERDES block. These are used to self-test signal integrity of the device. You can switch the device through predefined tests.

```
loopback_test [-deviceName device_name ] -start -serdes num -lane num -type LoopbackType
```

```
loopback_test [-deviceName device_name ] -stop -serdes num -lane num
```

Arguments

Parameter	Type	Description
deviceName	string	Parameter is optional if only one device is available in the current configuration or set for debug (see the SmartDebug User Guide for details).
start	None	Starts the loopback test.
stop	None	Stops the loopback test.
serdes	integer	Serdes block number. Must be between 0 and 4 and varies between dies.
lane	integer	Serdes lane number. Must be between 0 and 4
type	string	Specifies the loopback test type. Must be meso (PCS Far End PMA RX to TX Loopback), plesio and parallel

Supported Families

	Supported Families	Supported Libero SoC Versions
PolarFire	yes	v12.4+
RTG4	yes	v12.4+
SmartFusion2	yes	v12.4+
IGLOO2	yes	v12.4+

Example

```
loopback_test -start -serdes 1 -lane 1 -type meso
loopback_test -start -serdes 0 -lane 0 -type plesio
loopback_test -start -serdes 1 -lane 2 -type parallel
loopback_test -stop -serdes 1 -lane 2
```

See Also

- 2.59 prbs_test

2.53 new_project

Description

This Tcl command creates a new project in Libero SoC. If you do not specify a location, Libero SoC saves the new project in your current working directory.

```
new_project -name project_name \
-location project_location \
-family family_name \
-project_description brief text description of project \
-die device_die \
-package package_name \
-hdl HDL_type \
-speed speed_grade \
-die_voltage value \
-part_range value \
-block_mode {1 | 0} \
-ondemand_build_dh {1 | 0} \
-adv_options value \
-use_relative_path {1 | 0} \
-linked_files_root_dir_env root_dir_env \
-standalone_peripheral_initialization {1 | 0} \
-instantiate_in_smartdesign {1 | 0} \
-use_enhanced_constraint_flow {1 | 0}
```

Arguments

Parameter	Type	Description
name	string	The name of the project. This is used as the base name for most of the files generated from Libero SoC.
location	string	The location of the project. Must not be an existing directory.
project_description	string	A brief text description of the design in your project.
family	string	The Microchip SoC device family for your targeted design.
die	string	Sets device die for your targeted design.
package	string	Sets device package for your targeted design.
hdl	string	Sets the HDL type for your new project. Valid values are: • VHDL - sets your new projects HDL type to VHDL. • VERILOG - sets your new projects to Verilog.
speed	string	Sets the speed grade for your project. Possible values depend on your device, die and package. See your device datasheet for details.
die_voltage	floating point	Sets the die voltage for your project. Possible values depend on your device. See your device datasheet for details.
part_range	string	Sets your default temperature range for your project • PolarFire: EXT (Extended), IND (Industrial), MIL (Military) • SmartFusion2: COM (Commercial), IND, TGrade2 (Automotive), MIL • IGLOO2: COM, IND, TGrade1, TGrade2, MIL • RTG4: MIL
ondemand_build_dh	boolean	Enter "1" to enable or "0" (default) to disable On Demand Build Design Hierarchy.

.....continued

Parameter	Type	Description
block_mode	boolean	Enter "1" to enable or "0" (default) to disable design block creation.
instantiate_in_smartdesign	boolean	Enter "1" to enable or "0" (default) to disable Instantiate SystemBuilder/MSS components in a Smart Design. When set to "1", a System Builder or MSS component is auto-instantiated in a SmartDesign component upon creation. The default is 1.
use_relative_path	boolean	Enter "1" to use relative path or "0" (default) to use absolute path setting for the linked files in the project.
linked_files_root_dir_env	boolean	The System Environment variable that has valid root directory path. All the linked files in the project will be referenced relative to the path set in the Environment variable. The value in this argument is used only if the relative path is set in -use_relative_path argument.
adv_options	string	Sets your advanced options, such as temperature and voltage settings. For more information see table below.

The following are advanced options for temperature and voltage settings.

Value	Description
IO_DEFT_STD:LVTTL	Sets your I/O default value to LVTTL. This value defines the default I/O technology to be used for any I/Os that the user does not explicitly set a technology for in the I/O Editor. It could be any of: • LVTTL • LVCMOS 3.3V • LVCMOS 2.5V • LVCMOS 1.8V • LVCMOS 1.5V • LVCMOS 1.2V
DSW_VCCA_VOLTAGE_RAMP_RATE	(SmartFusion2 and IGLOO2 only) This value defines the Maximum VDD and VPP power supply ramp rate. Power-up management circuitry is designed into every SmartFusion2 and IGLOO2 SoC FPGA. These circuits ensure easy transition from the powered-off state to powered-up state of the device. The SmartFusion2, IGLOO2 system controller is responsible for systematic power-on reset whenever the device is powered on or reset. All the I/Os are held in a high-impedance state by the system controller until all power supplies are at their required levels and the system controller has completed the reset sequence. The power-on reset circuitry in SmartFusion2 and IGLOO2 devices requires the VDD and VPP supplies to ramp monotonically from 0 V to the minimum recommended operating voltage within a predefined time. There is no sequencing requirement on VDD and VPP. Four ramp rate options are available during design generation: • 50 us • 1 ms • 10 ms • 100 ms Each selection represents the maximum ramp rate to apply to VDD and VPP.

.....continued

Value	Description
PLL_SUPPLY	(SmartFusion2, IGLOO2 only) This value sets the voltage for the power supply you plan to connect to all the PLLs in your design, such as MDDR, FDDR, SERDES, and FCCC. Two values are available: • 2.5 • 3.3
RESTRICTPROBEPINS	This value reserves your pins for probing if you intend to debug using SmartDebug. Two values are available: • 1 (Probe pins are reserved) • 0 (No probe pins are reserved)
RESTRICTSPIPINS:1	(RTG4 only) Sets to 1 to reserve pins for SPI functionality in Programming. This reserved SPI pin option is displayed in the Compile Report when the compile process completes.
SYSTEM_CONTROLLER_SUSPEND_MODE	(SmartFusion2, IGLOO2 only) Enables SmartFusion2 and IGLOO2 designers to suspend operation of the System Controller. Enabling this bit instructs the System Controller to place itself in a reset state once the device is powered up. This effectively suspends all system services from being performed. For a list of system services, refer to the SmartFusion2 or IGLOO2 System Controller User Guide for your device on the Microchip website. Two values are available: • 1 (System Controller Suspend Mode is enabled) • 0 (System Controller Suspend Mode is disabled

The following are options for Analysis Operating Conditions so that Timing and Power analysis can be performed at different operating conditions.

Value	Description
TEMPR	Sets your default temperature range for operating condition analysis. • COM (Commercial) • MIL (Military) • IND (Industrial)
VCCI_1.2_VOLTR	Sets the Default I/O Voltage Range for 1.2V, which could be • COM (Commercial) • MIL (Military) • IND (Industrial) • Custom These settings are propagated to Verify Timing, Verify Power and Backannotated Netlist to perform Timing/Power Analysis.
VCCI_1.5_VOLTR	Sets the Default I/O Voltage Range for 1.5V, which could be • COM (Commercial) • MIL (Military) • IND (Industrial) • Custom These settings are propagated to Verify Timing, Verify Power and Backannotated Netlist to perform Timing/Power Analysis.

.....continued	
Value	**Description**
VCCI_1.8_VOLTR	Sets the Default I/O Voltage Range for 1.8V, which could be • COM (Commercial) • MIL (Military) • IND (Industrial) • Custom These settings are propagated to Verify Timing, Verify Power and Backannotated Netlist to perform Timing/Power Analysis.
VCCI_2.5_VOLTR	Sets the Default I/O Voltage Range for 2.5V, which could be • COM (Commercial) • MIL (Military) • IND (Industrial) • Custom These settings are propagated to Verify Timing, Verify Power and Backannotated Netlist to perform Timing/Power Analysis.
VCCI_3.3_VOLTR	Sets the Default I/O Voltage Range for 3.3V, which could be • COM (Commercial) • MIL (Military) • IND (Industrial) • Custom These settings are propagated to Verify Timing, Verify Power and Backannotated Netlist to perform Timing/Power Analysis.
VOLTR	Sets the core voltage range for operating condition analysis. These settings are propagated to Verify Timing, Verify Power and Backannotated Netlist to perform Timing/Power Analysis. Can be one of the following: • COM (Commercial) • MIL (Military) • IND (Industrial)

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

1. Creates a new project in the `./designs/mydesign` directory, with the HDL type Verilog for the SmartFusion2 family.

```
new_project -location {./designs/mydesign} \
-name {mydesign} \
-use_enhanced_constraint_flow 1 \
-use_relative_path 1 -linked_files_root_dir_env {MSCC_ROOT_1} \
-standalone_peripheral_initialization 1 -hdl {VERILOG} -family {SmartFusion2} \
-die {M2S150TS} -package {FCS536} -speed {-1} -die_voltage {1.2} \
-part_range {COM} -adv_options {DSW_VCCA_VOLTAGE_RAMP_RATE:100_MS} \
```

```
-adv_options {IO_DEFT_STD:LVCMOS 2.5V} \
-adv_options {PLL_SUPPLY:PLL_SUPPLY_25} \
-adv_options {RESTRICTPROBEPINS:1} \
-adv_options {SYSTEM_CONTROLLER_SUSPEND_MODE:0} \
-adv_options {TEMPR:IND} -adv_options {VCCI_1.2_VOLTR:IND} \
-adv_options {VCCI_1.5_VOLTR:IND} -adv_options {VCCI_1.8_VOLTR:IND} \
-adv_options {VCCI_2.5_VOLTR:IND} -adv_options {VCCI_3.3_VOLTR:IND} \
-adv_options {VOLTR:IND}
```

2. Creates a new project in the `D:/2Work/my_pf_proj` directory, with the HDL type Verilog for the PolarFire. Set up a new design and run Libero tools.

```
new_project -location {D:/2Work/my_pf_proj} -name {my_pf_proj} \
-project_description {} - block_mode 0 -standalone_peripheral_initialization 0 \
-use_enhanced_constraint_flow 1 -use_relative_path 1 \
-linked_files_root_dir_env {MSCC_ROOT_1} -hdl {VERILOG} -family {PolarFire} \
-die {MPF300TS_ES} -package {FCG1152} -speed {-1} - die_voltage {1.0} \
-part_range {EXT} -adv_options {IO_DEFT_STD:LVCMOS 1.8V} -adv_options
{RESTRICTPROBEPINS:1} \
-adv_options {RESTRICTSPIPINS:0} -adv_options {SYSTEM_CONTROLLER_SUSPEND_MODE:1} \
-adv_options {TEMPR:EXT} -adv_options {VCCI_1.2_VOLTR:EXT} -adv_options
{VCCI_1.5_VOLTR:EXT} \
-adv_options {VCCI_1.8_VOLTR:EXT} -adv_options {VCCI_2.5_VOLTR:EXT} \
-adv_options {VCCI_3.3_VOLTR:EXT} -adv_options {VOLTR:EXT}
#Import HDL source file
import_files -convert_EDN_to_HDL 0 -hdl_source {C:/test/prep1.v}

#Import HDL stimulus file
import_files -convert_EDN_to_HDL 0 -stimulus {C:/test/prep1tb.v}
#set the top level design name
set_root -module {prep1::work}

#Associate SDC constraint file to Place and Route tool
organize_tool_files -tool {PLACEROUTE} -file {D:/2Work/my_pf_proj/constraint/user.sdc} \
-module {prep1::work} -input_type {constraint}

#Associate SDC constraint file to Verify Timing tool
organize_tool_files -tool {VERIFYTIMING} -file {D:/2Work/my_pf_proj/constraint/user.sdc}
\
-module {prep1::work} -input_type {constraint}
#Run synthesize
run_tool -name {SYNTHESIZE}
#Configure Place and Route tool
configure_tool -name {PLACEROUTE} -params {DELAY_ANALYSIS:MAX} -params
{EFFORT_LEVEL:false} \
-params {INCRPLACEANDROUTE:false} -params {MULTI_PASS_CRITERIA:VIOLATIONS} \
-params {MULTI_PASS_LAYOUT:false} -params {NUM_MULTI_PASSES:5} -params {PDPR:false} \
-params {RANDOM_SEED:0} -params {REPAIR_MIN_DELAY:false} -params
{SLACK_CRITERIA:WORST_SLACK} \
-params {SPECIFIC_CLOCK:} -params {START_SEED_INDEX:1} -params
{STOP_ON_FIRST_PASS:false} \
-params {TDPR:true}
```

2.54 open_project

Description

This Tcl command opens an existing Libero SoC project. You can create backup of your original project before opening.

```
open_project -file project_name -do_backup_on_convert value -backup_file backup_filename
```

Arguments

Parameter	Type	Description
file	string	Must include the complete path to the PRJ file. If you do not provide the full path, Libero SoC infers that you want to open the project from your current working directory.
do_backup_on_convert	string	Sets the option to backup your files if you open a project created in a previous version of Libero SoC. • TRUE - creates a backup of your original project before opening. • FALSE - opens your project without creating a backup.
backup_file	string	Sets the name of your backup file (if you choose to do_backup_on_convert).

Error Codes

Error Code	Description
None	Unable to find project file.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command opens project.prjx project from test directory without creating a backup file.

```
open_project -file {c:/netlists/test/project.prjx}
```

See Also
- 2.15 close_project
- 2.53 new_project
- 2.70 save_project

2.55 open_smartdesign

Description
This Tcl command opens a SmartDesign. You must either open or create a SmartDesign before using any of the SmartDesign specific commands "sd_*".

Note: This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

```
open_smartdesign \
-sd_name smartdesign_component_name
```

Arguments

Parameter	Type	Description
sd_name	string	Specifies the name of an existing SmartDesign component. It is mandatory. You can specify the name of SmartDesign as follow: {<design_name>} or {<design_name>::<module_name>}.

Error Codes

Error Code	Description
None	Parameter 'sd_name' has illegal value.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Open "top" SmartDesign from your project.

```
open_smartdesign -sd_name {top}
```

See Also

- 2.21 create_smartdesign

2.56 organize_constraints

Description
This Tcl command organizes the constraint files in your project. The Organize Constraint Files dialog box enables you to set the constraint file and order in the Libero SoC.

```
-organize_constraints \
[-file name ] \
[-mode value ] \
-designer_view name \
-module value \
-tool value
```

Arguments

Parameter	Type	Description
file	string	Specifies the name of the file to which you want to associate your stimulus files.

.....continued		
Parameter	**Type**	**Description**
mode	string	Specifies whether you are creating a new stimulus association, adding or removing; possible values are: • new - creates a new stimulus file association. • add - adds a stimulus file to an existing association. • remove - removes a stimulus file association.
designer_view	string	Sets the name of the Designer View in which you wish to add the constraint file, where name is the name of the view (such as impl1).
module	string	Sets the module, where value is the name of the module.
tool	string	Identifies the intended use for the file, possible values are: • synthesis - file to be used for synthesis. • designer - file to be used in Designer. • phsynth - file to be used in physical synthesis.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The example adds the constraint file delta.vhd in the Designer View impl2 for the Designer tool.

```
-organize_constraints -file delta.vhd -mode new -designer_view impl2 -module constraint_tool
designer
```

See Also
- 2.58 organize_tool_files

2.57 organize_sources

Description
This Tcl command enables you to set the source file order in the Libero SoC.

To specify the file order:
1. In the Design Flow window, right-click Synthesize or Simulation tool and choose *Organize Input Files > Organize Source Files*. The **Organize Source Files** dialog box appears.
2. Click the Use list of files organized by User radio button to Add/Remove source files for the selected tool.
3. Use the Up and Down arrows to change the order of the Associated Source files.

```
-organize_sources \
        [-file name ] \
        [-mode value ] \
        -module value \
```

```
        -tool value \
        [-use_default value ]
```

Arguments

Parameter	Type	Description
file	string	Specifies the name of the file to which you want to associate your source files. It is optional. Default is empty.
mode	string	Specifies whether you are creating a new source files association, adding, or removing for the selected tool; possible values are: • new - creates a new source file association. • add - adds a source file to an existing association. • remove - removes a source file association.
module	string	Sets the module, where value is the name of the module. You can specify as {<module::work>} or {<module>} <module>. This is mandatory.
tool	string	Identifies the intended use for the file, possible values are: • synthesis - file to be used for synthesis. • simulation - file to be used in simulation.
use_default	string	Uses the default values for synthesis or simulation; possible values are: • TRUE, true, 1 - uses default values for synthesis or simulation. This is the default value. • FALSE, false, 0 - uses user-defined values for synthesis or simulation.

Error Codes

Error Code	Description
None	'file' is not in the project.
None	Required parameter 'module' is missing.
None	mode: Invalid argument value: 'a' (expecting new, add or remove).
None	tool: Invalid argument value: 'designer' (expecting synthesis or simulation).

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The example organizes a new stimulus file 'stim.vhd' using default settings.

```
-organize_sources -file stim.vhd -mode new -module stimulus -tool synthesis -use_default TRUE
```

See Also

- 2.58 organize_tool_files

2.58 organize_tool_files

Description

This Tcl command is used to specify specific files to be passed to and used by a Libero tool. If you do not want to pass file by Libero tools then specify `-file` option value as empty: `-file {}`.

```
organize_tool_files \
-tool {tool_name} \
-file {absolute path to specific file} \
-module {<design_name>::work} \
-input_type {value}
```

Arguments

Parameter	Type	Description
tool	string	Specifies the name of the tool files you want to organize. Valid values are:`SYNTHESIZE | COMPILE | PLACEROUTE | SIM_PRESYNTH | SIM_POSTSYNTH | SIM_POSTLAYOUT | VERIFYTIMING`.
file	string	Specifies the absolute path to the specific file; there may be multiple -file arguments (see example below). It is mandatory. You can specify file as: `-file {filename}` or `-file "filename"`.
module	string	Module definition, format is `<design_name>::work` or `<design_name>`. It is optional. Default is `<root_design_name>::work`.
input_type	string	Specifies type of input file. Possible values are:`constraint | source | simulation | stimulus | unknown`. It is mandatory.

Error Codes

Error Code	Description
None	'user.sdc' is not in the project.
None	Required parameter 'input_type' is missing.
None	input_type: Invalid argument value: '' (expecting source, constraint, simulation, stimulus or unknown).

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command organizes the test_derived.sdc and user.sdc files of SDC (Synopsys Design Constraint) file type for the tool VERIFYTIMING for the sd1::work design.

```
organize_tool_files \
-tool {VERIFYTIMING} \
-file {D:/Designs/my_proj/constraints/test_derived.sdc} \
-file {D:/Designs/my_proj/constraints/user.sdc} \
-module {sd1::work} \
-input_type {constraint}
```

2.59 prbs_test

Description
This Tcl command used in PRBS test to start, stop, reset the error counter and read the error counter value. PRBS data stream patterns are generated and checked by the internal SERDES block. These are used to self-test signal integrity of the device. You can switch the device through several predefined patterns.

```
prbs_test [-deviceName device_name ] -start -serdes num -lane num [-near] -pattern PatternType
prbs_test [-deviceName device_name ] -stop -serdes num -lane num
prbs_test [-deviceName device_name ] -reset_counter -serdes num -lane num
prbs_test [-deviceName device_name ] -read_counter -serdes num -lane num
```

Arguments

Parameter	Type	Description
deviceName	string	Parameter is optional if only one device is available in the current configuration or set for debug (see the SmartDebug User Guide for details).
start	None	Starts the prbs test.
stop	None	Stops the prbs test.
reset_counter	None	Resets the prbs error count value to 0.
read_counter	None	Reads and prints the error count value.
serdes	integer	Serdes block number. Must be between 0 and 4 and varies between dies.
lane	integer	Serdes lane number. Must be between 0 and 4.
near	None	Corresponds to near-end (on-die) option for prbs test. Not specifying implies off-die.
pattern	string	The pattern sequence to use for PRBS test. It can be one of the following: `prbs7, prbs11, prbs23, or prbs31`.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

```
prbs_test -start -serdes 1 -lane 0 -near -pattern prbs11
prbs_test -start -serdes 2 -lane 2 -pattern custom -value all_zeros
prbs_test -start -serdes 0 -lane 1 -near -pattern user -value 0x0123456789ABCDEF0123
```

2.60 project_settings

Description

This Tcl command modifies project flow settings for your Libero SoC project. The Project Settings dialog box enables you to modify your Device, HDL, and Design Flow settings and your Simulation Options. In Libero SoC, from the Project menu, click Project Settings.

```
project_settings \
[-hdl "VHDL | VERILOG"] \
[-verilog_mode "SYSTEM_VERILOG | VERILOG_2K"] \
[-vhdl_mode "VHDL_2008 | VHDL_93"] \
[-auto_update_modelsim_ini "TRUE | FALSE"] \
[-auto_update_viewdraw_ini "TRUE | FALSE"] \
[-standalone_peripheral_initialization "TRUE | FALSE"] \
[-ondemand_build_dh "TRUE | FALSE"] \
[-auto_generate_synth_hdl "TRUE | FALSE"] \
[-auto_run_drc "TRUE | FALSE"] \
[-auto_generate_viewdraw_hdl "TRUE | FALSE"] \
[-auto_file_detection "TRUE | FALSE"] \
[-enable_set_mitigation "TRUE | FALSE"] \
[-enable_design_separation "TRUE" | "FALSE"] \
[-display_fanout_limit "display_fanout_limit"] \
[-block_mode "TRUE | FALSE"] \
[-abort_flow_on_sdc_errors "TRUE | FALSE"] \
[-abort_flow_on_pdc_errors "TRUE | FALSE"] \
[-sim_flow_mode "TRUE | FALSE"] \
[-auto_generate_physynth_hdl "TRUE | FALSE"] \
[-instantiate_in_smartdesign "TRUE | FALSE"] \
[-enable_viewdraw "TRUE | FALSE"] \
[-vm_netlist_flow "TRUE | FALSE"]
[-system_verilog_mfcu "TRUE | FALSE"]
```

Arguments

Parameter	Type	Description
hdl	string	Sets your project HDL type. Valid values are: VHDL or VERILOG.
verilog_mode	string	Sets the Verilog standard to Verilog-2001 or System Verilog. Valid values are: VERILOG_2K or SYSTEM_VERILOG. The default value is SYSTEM_VERILOG.
vhdl_mode	string	Sets the VHDL standard. Valid values are: VHDL-2008 or VHDL-1993. The default value is VHDL-2008.
auto_update_modelsim_ini	boolean	Sets your auto-update modelsim.ini file option. Valid values are True, true, 1, FALSE, false or 0: • TRUE, true, 1 - updates the modelsim.ini file automatically. The default value is 1. • FALSE, false, 0 - modelsim.ini file does not update automatically.

.....continued

Parameter	Type	Description
auto_update_viewdraw_ini	boolean	Sets your auto-update viewdraw.ini file option. Valid values are True, true, 1, FALSE, false or 0: • TRUE, true, 1 - updates the viewdraw.ini file automatically. The default value is 1. • FALSE, false, 0 - viewdraw.ini file does not update automatically.
block_mode	boolean	Enables you to create and publish design blocks (*.cxz files) in Libero SoC. Puts the Project Manager in Block mode, Design blocks are low-level components that may have completed the place-and-route step and met the timing and power requirements. These low-level design blocks can then be imported into a Libero SoC project and re-used as components in a higher level design. By default, this box is unchecked. • TRUE, true, 1 - Enable to create and publish design blocks file. • FALSE, false, 0 - Disables to create and publish design blocks file.
auto_generate_synth_hdl	boolean	Auto-generates your HDL file after synthesis. Valid values are: • TRUE, true, 1 - enables auto-generation of your HDL file after synthesis. • FALSE, false, 0 - disables auto-generation of your HDL file after synthesis. The default value is 0.
auto_run_drc	boolean	Automatically runs the design rule check immediately after synthesis. Valid values are: • TRUE, true, 1 - enables auto-runs the design rule check immediately after synthesis. • FALSE, false, 0 - disables auto-runs the design rule check immediately after synthesis. the default value is 0.
auto_generate_viewdraw_hdl	boolean	Automatically generates your HDL netlist after Save and Check in ViewDraw. Valid values are: • TRUE, true, 1 - enables auto-generates your HDL netlist. The default value is 1. • FALSE, false, 0 - disables auto-generates your HDL netlist.
auto_file_detection	boolean	Automatically detects when new files have been added to the Libero SoC project folder. Valid values are: • TRUE, true, 1 - automatically detects new added files in Libero SoC project. The default value is 1. • FALSE, false, 0 - automatically does not detects new added files in Libero SoC project.
standalone_peripheral_initialization	boolean	Use Standalone Initialization for MDDR/FDDR/SERDES Peripherals – Check this box if you want to create your own peripheral initialization logic in SmartDesign for each of your design peripherals (MDDR/FDDR/SERDES). When checked, System Builder does not build the peripherals initialization logic for you. Standalone initialization is useful if you want to make the initialization logic of each peripheral separate from and independent of each other. By default, this box is checked. Valid values are: TRUE, true, 1 , FALSE, false or 0.

.....continued

Parameter	Type	Description
ondemand_build_dh	integer	Enable/disable On Demand Build Design Hierarchy. Valid values are: TRUE, true, 1, FALSE, false or 0. The default value is 1.
enable_set_mitigation	boolean	Enable Single Event Transient mitigation - Controls the mitigation of Single Event Transient (SET) in the FPGA fabric. When this box is checked, SET filters are turned on globally (including URAM, LSRAM, MACC, I/O FF, Regular FF, DDR_IN, DDR_OUT) to help mitigate radiation-induced transients. By default, this box is unchecked. Valid values are: TRUE, true, 1, FALSE, false or 0.
enable_design_separation	integer	Set it to “1” if your design is for security and safety critical applications and you want to make your design’s individual subsystems (design blocks) separate and independent (in terms of physical layout and programming) to meet your design separation requirements. When set to “1”, Libero generates a parameter file (MSVT.param) that details design blocks present in the design and the number of signals entering and leaving a design block. Microchip provides a separate tool, known as Microchip Separation Verification Tool (MSVT), which checks the final design place and route result against the MSVT.param file and determines whether the design separation meets your requirements. The default value is 1. This option available for SmartFusion2 and IGLOO2.
display_fanout_limit	integer	Use this option to set the limit of high fanout nets to be displayed; the default value is 10. This means the top 10 nets with the highest fanout will appear in the root_compile_netlist.log file.
abort_flow_on_pdc_errors	boolean	Abort flow if errors are found in Physical Design Constraints(PDC). Valid values are: TRUE, true, 1, FALSE, false or 0. By default, this box is checked.
abort_flow_on_sdc_errors	boolean	Abort flow if errors are found in Timing Constraints (SDC). Valid values are: TRUE, true, 1, FALSE, false or 0. By default, this box is checked.
auto_generate_physynth_hdl	boolean	Auto-generates your HDL file after physical synthesis. Valid values are: TRUE, true, FALSE, false or 0. The default value is 1.
instantiate_in_smartdesign	boolean	Instantiate System Builder/MSS component in a SmartDesign on creation - Uncheck this box if you are using this project to create System Builder or MSS components and do not plan on using them in a SmartDesign based design. This is especially useful for design flows where the System Builder or MSS components are stitched in a design using HDL. Valid values are: TRUE, true, 1, FALSE, false or 0.
enable_viewdraw	boolean	Enable/disable to create a schematic source file in Libero SoC. Valid values are: TRUE, true, 1, FALSE, false or 0.
vm_netlist_flow	boolean	Sets Synthesis gate level netlist format. By default, this box is checked. Valid values are: TRUE, true, 1, FALSE, false or 0. • TRUE, true, 1 - sets Verilog netlist format. • FALSE, false, 0 - sets EDIF netlist format.

.....continued		
Parameter	**Type**	**Description**
sim_flow_mode	boolean	Instantiate simulation cores in your SmartDesign Testbench. Simulation cores are basic cores that are useful for stimulus, such as driving clocks, resets, and pulses. Valid values are: TRUE, true, 1, FALSE, false or 0.
system_verilog_mfcu	boolean	Enable/disable the System Verilog Multi-File Compilation Unit (MFCU). Valid values are: TRUE, true, 1, FALSE, false or 0. **Note:** This option is only applicable when -verilog_mode is “SYSTEM_VERILOG”.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Example of SmartFusion2, IGLOO2:

The following example sets your project to VHDL, disables the auto-update of the ModelSim INI or ViewDraw INI files, enables the automatically generation of HDL after synthesis, enables auto-detection for files, sets the display of high fanout nets to the top 12 high fanout nets, enables SET filters to mitigate radiation-induced transients, and enables design separation methodology for the design.

```
project_settings -hdl "VHDL" \
-auto_update_modelsim_ini "FALSE" \
-auto_update_viewdraw_ini "FALSE"\
-block_mode "FALSE" -auto_generate_synth_hdl "TRUE" \
-auto_file_detection "TRUE" \
-display_fanout_limit {12} \
-enable_set_mitigation {1}
```

2.61 publish_block

Description
Tcl command publishes a block with the conditions related to place and route. This is the project_setting command. To enable Block Creation for a new project:

1. Select **New Project** from the **Libero SoC Project** menu.
2. Check the **Enable Block Creation** checkbox.
3. Select the **Enhanced Constraint Flow** for the new project. In an existing project, from the **Project** menu, select *Project Settings > Design Flow* and check the **Enable Block Creation** checkbox.
4. After Block Creation is enabled, **Publish Block** appears in the **Design Flow** window.
5. Expand **Publish Design**, right-click **Publish Block** and select **Export** from the context-menu that appears. By default, this option is not unchecked.

```
publish_block \
-file {absolute or relative path} \
[-publish_placement value] \
[-publish_routing value] \
```

```
[-publish_region value] \
[-vhdl value ]
```

Arguments

Parameter	Type	Description
file	string	Specifies the location (absolute or relative) to publish the block. Default is /designer/<designer_name>/<designer_name>.cxz.
publish_placement	boolean	Valid values are: TRUE, 1, true, FALSE, 0 or false • FALSE, false, 0 - no placement or routing will be published and preserved. Only the netlist is preserved. • TRUE, true, 1 - publishes placement if all the macros in your design are placed or assigned to a region. Default this box is checked.
publish_routing	boolean	Valid values are: TRUE, 1, true, FALSE, 0 or false • FALSE, false, 0 - routing will not be published and added to the block. This block will be completely rerouted completely in the top design. • TRUE, true, 1 - publish routing to be part of the block. `publish_placement` must be 1 for this option to take effect. All the macros should be placed or assigned to a region. Default this box is checked.
publish_region	boolean	Valid values are: TRUE, 1, true, FALSE, 0 or false • FALSE, false, 0 - region constraints are not added to the block published. • TRUE, true, 1 - Region constraints will be published and preserved. This is not recommended and should be done only if the user wants to keep the regions in the top design. Example: the user wants to see an empty region in the top design. In general, the regions used to control placement should not be part of the block. Default this box is checked.
vhdl	boolean	Valid values are: TRUE, 1, true, FALSE, 0 or false • FALSE, false, 0 - generates a Verilog netlist to be used for synthesis and simulation. Default value. • TRUE, true, 1 - generates a VHDL file format.

Error Codes

Error Code	Description
None	Required parameter 'file' is missing.
None	Required parameter 'publish_placement' is missing.
None	Required parameter 'publish_routing' is missing.
None	Required parameter 'publish_region' is missing.

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
IGLOO2	v12.4+

Example
The following example exports Publish Block file in the `\test_block\designer\top\top.cxz` file, generates a Verilog netlist. Publishes placement and region if all the macros in your design are placed or assigned to a region:

```
publish_block \
-file {D:\test_block\designer\top\top.cxz} \
-publish_placement 1 \
-publish_routing 1 \
-publish_region 1 \
-vhdl 0
```

2.62 refresh

Description
This Tcl command refreshes your project, rebuilds the Design Hierarchy, updates the view and checks for updated links and files. This command is equivalent to selecting **Refresh Design Hierarchy (F5)** from the **View** menu.

```
refresh
```

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command refreshes project, rebuilds the Design Hierarchy, updates the view and checks for updated links and files.

```
refresh
```

2.63 remove_core

Description
This Tcl command removes a core from your project.

```
remove_core -vlnv "Vendor:Library:Name:Version"
```

Arguments

Parameter	Type	Description
vlnv	string	Specifies the version identifier of the core being configured. It is mandatory.

Error Codes

Error Code	Description
None	The core is not in the vault.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example removes Simulation RESET_GENR from project.

```
remove_core -vlnv {Actel:Simulation:RESET_GEN:1.0.1}
```

See Also

- 2.16 configure_core
- 2.23 download_core
- 2.18 create_and_configure_core

2.64 remove_library

Description
This Tcl command removes a VHDL library from your project. To remove library, right-click the design module name in the **Design Hierarchy** and select **Remove VHDL Library** from the context menu.

```
remove_library -library name
```

Arguments

Parameter	Type	Description
library	string	Specifies the name of the library you wish to remove.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Remove a library with the "my_lib" name from your project.

```
remove_library -library my_lib
```

See Also
- 2.2 add_library
- 2.67 rename_library

2.65 remove_profile

Description
This Tcl command deletes a tool profile.

```
remove_profile -name profilename
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of the profile you wish to delete.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command deletes the profile 'custom1'.

```
remove_profile -name custom1
```

See Also
- 2.4 add_profile
- 2.74 select_profile
- 2.25 edit_profile
- 2.39 export_profiles

2.66 rename_file

Description
This Tcl command copies/renames a file specified by the `-file` parameter to a different name specified by the `-target` parameter. Creates a new file in the <project_name> directory.

```
rename_file -file { absolute path and the name of file } -target { new_filename }
```

Arguments

Parameter	Type	Description
file	string	Specifies the absolute path and original name of the file.
target	string	Specifies the just new name of the file. If specified constraint file, then created new file in constraint directory, otherwise new file created in the <project_name> directory, not renamed.

Error Codes

Error Code	Description
None	Required parameter 'file' is missing.
None	Parameter 'file' has illegal value.
None	Required parameter 'target' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command copies the constraint a.sdc file specified by the `-file` parameter to a b.sdc specified by `-target` parameter in the "c:/user/" directory.

```
rename_file -file {c:/user/a.sdc} -target {b.sdc}
```

The following command copies the a.v verilog file specified by the `-file` parameter to a b.v specified by `-target` parameter in the project directory not in hdl.

```
rename_file -file { /libero_prj/hdl/a.v } -target {b.v}
```

2.67 rename_library

Description
This Tcl command renames a VHDL library from your project. To renames library, right-click the design module name in the Design Hierarchy select Rename VHDL Library from the context menu.

```
rename_library -library name -name name
```

Arguments

Parameter	Type	Description
library	string	Identifies the current name of the library that you wish to rename.
name	string	Specifies the new name of the library.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Rename a library from 'my_lib' to 'test_lib1'.

```
rename_library -library my_lib -name test_lib1
```

See Also
- 2.2 add_library
- 2.64 remove_library

2.68 run_tool

Description
This Tcl command starts the specified tool. For tools that support command files, an optional command file can be supplied through the `-script` parameter.

Note: Where possible, the value of `tool_name` corresponds to the name of the tool in Libero SoC.
Invoking some tools will cause Libero SoC to automatically run some upstream tools in the design flow. For example, invoking Place and Route will invoke Synthesis (if not already run) before it runs Place and Route.

```
run_tool -name {tool_name} -script {absolute or relative path to script file}
```

Arguments

Parameter	Type	Description
name	string	Specified tool name. The following is a list of supported tool names. • SYNTHESIZE • COMPILE • SIM_PRESYNTH • SIM_POSTSYNTH • SIM_POSTLAYOUT • PLACEROUTE • VERIFYTIMING • VERIFYPOWER • GENERATEPROGRAMMINGFILE • GENERATE_MEMORY_MAP • PROGRAMDEVICE • CONFIGURE_CHAIN • SMARTDEBUG • SSNANALYZER • UPDATE_ENVM • UPDATE_UPROM • EXPORTNETLIST • EXPORTSDF • GENERATEPROGRAMMINGDATA • GENERATEDEBUGDATA • GENERATE_SPI_FLASH_IMAGE • PROGRAM_SPI_FLASH_IMAGE
script	string	Specify absolute or relative path of the script file. This is an optional parameter.

Supported tool_names
The following table lists the supported tool names for `run_tool -name {tool_name}`.

tool_name	Parameter	Description
SYNTHESIZE	-script {`script_file`}	Runs synthesis on your design.
COMPILE	N/A	Runs Compile with default or configured settings.
SIM_PRESYNTH	N/A	Runs pre-synthesis simulation with your default simulation tool
SIM_POSTSYNTH	N/A	Runs post-synthesis simulation with your default simulation tool.
SIM_POSTLAYOUT	N/A	Runs the post layout simulation on the simulation tool.
PLACEROUTE	N/A	Runs Place and Route tool with default or configured settings.

.....continued

tool_name	Parameter	Description
VERIFYTIMING	-script {script_file}	Runs timing analysis with default settings/ configured settings in script_file.
VERIFYPOWER	-script {script_file}	Runs power analysis with default settings/ configured settings in script_file.
GENERATEPROGRAMMINGFILE	N/A	Generates the bitstream used for programming within Libero.
GENERATE_MEMORY_MAP (SmartFusion2, IGLOO2 and RTG4 only)	N/A	Exports an XML file in <prj_folder> component/ work/<design> /<design>_DataSheet.xml. The file contains information about your root SmartDesign in your project.
PROGRAMDEVICE	N/A	Programs your device with configured parameters.
CONFIGURE_CHAIN	-script {script_file}	Takes a script that contains FlashPro-specific Tcl commands and passes them to FlashPro Express for execution.
SMARTDEBUG	-script {script_file}	Takes a script that contains SmartDebug-specific Tcl commands and passes them to SmartDebug for execution.
SSNANALYZER	-script {script_file}	Takes a script that contains Simultaneous Switching Noise (SSN)-specific Tcl commands and passes them to the SSN tool for execution. Simultaneous Switching Noise (SSN) is a Libero SoC tool that analyzes and generates a Nosie Margin report for I/Os after layout. Depends on Die ane Package.
UPDATE_ENVM (SmartFusion2 and IGLOO2 only)	-script {update_config_file}	Takes a script file that updates the client(s) in the ENVM. In the script file, the client(s) to be updated may be a serialization client or a data storage client or a mix of serialization clients and data storage clients.
UPDATE_UPROM (RTG4 only)	-script {update_config_file}	Takes a script file that updates the data storage client(s) in RTG4 UPROMS.
EXPORTNETLIST	N/A	This command exports a .v/.vhd netlist file to the active synthesis implementation folder.
EXPORTSDF	N/A	This command exports the back annotated files to the designer/impl1 folder.
GENERATEPROGRAMMINGDATA	N/A	Generates the files needed by generating programming bitstream files.
GENERATEDEBUGDATA (PolarFire only)	N/A	Generates the files needed by SmartDebug during device debug.
GENERATE_SPI_FLASH_IMAGE (PolarFire only)	N/A	Generates SPI Flash Image file used for programming SPI FLASH Image within Libero.
PROGRAM_SPI_FLASH_IMAGE (PolarFire only)	N/A	Programs SPI Flash Image with configured parameters.

Error Codes

Error Code	Description
None	You must specify the Tcl script to run with the CONFIGURE_CHAIN tool.
None	You must specify the Tcl script to run with the SMARTDEBUG tool.
None	PROGRAMDEVICE: No programmer is connected.
None	SPI Flash Memory is not configured. Use the Configure Design Initialization Data and Memories tool to configure it.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

```
run_tool \
-name {COMPILE}
run_tool \
-name {SYNTHESIZE} -script {./control_synopsys.tcl}
# control _synopsys.tcl contains the synthesis-specific Tcl commands
run_tool \
-name {VERIFYTIMING} \
-script {./SmartTime.tcl}
# Script file contains SmartTime-specific Tcl commands
run_tool \
-name {VERIFYPOWER} \
-script {./SmartPower.tcl}
# Script file contains SmartPower-specific Tcl commands
run_tool \
-name {SMARTDEBUG}
-script {./sd_test.tcl}
# Script file contains SmartDebug-specific Tcl commands
```

2.69 save_log

Description
This Tcl command saves your Libero SoC log file.

```
save_log -file { absolute or relative path of log file }
```

Arguments

Parameter	Type	Description
file	string	Specifies the log file name. It must be the absolute or relative path.

Error Codes

Error Code	Description
None	Required parameter 'file' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example saves the log of Libero SoC with the name 'my_logfile.log'.

```
save_log -file {my_logfile.log}
```

2.70 save_project

Description
This Tcl command saves the current project in Libero SoC. Equivalent to clicking the File menu, and choosing Save Project.

```
save_project
```

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Saves the project in your current working directory.

```
save_project
```

See Also
- 2.53 new_project
- 2.15 close_project

2.71 save_project_as

Description
This Tcl command saves the current project in Libero SoC with a different name and in a specified directory. You must specify a location using the `-location` parameter.

```
save_project_as \
-name project_name \
-location project_location \
-files value \
-replace_links value
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of your new project.
location	string	Must include the complete path of the PRJ file. If you do not provide the full path, Libero SoC infers that you want to save the project to your current working directory. This is a required parameter.
files	string	Specifies the files you want to copy into your new project. • all - copies all your files into your new project. • project - copies only your Libero SoC project files into your new project. • sources - copies only the source files into your new project.
replace_links	boolean	Specifies whether or not you want to update your file links in your new project. • true | 0 - replaces (updates) the file links in your project during your save. • false | 1 - saves your project without updating the file links.

Error Codes

Error Code	Description
None	Invalid argument value: 'none' (expecting all, project or sources).
None	Invalid argument value: 'source' (expecting all, project or sources).

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Saves your current Libero SoC project as mydesign.prjx in the c:/netlists/testprj/mydesign directory.

```
save_project_as -location {c:/netlists/testprj/mydesign} -name mydesign.prjx
```

See Also
- 2.53 new_project
- 2.54 open_project
- 2.70 save_project

2.72 save_smartdesign

Description
This Tcl command saves all the changes made in a SmartDesign component.

```
save_smartdesign -sd_name smartdesign_component_name
```

Arguments

Parameter	Type	Description
sd_name	string	Specifies the name of the SmartDesign component to be saved. It is mandatory.

Error Codes

Error Code	Description
None	Parameter 'component' has illegal value.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example saves "top" SmartDesign.

```
save_smartdesign -sd_name {top}
```

See Also
- 2.55 open_smartdesign

2.73 select_libero_design_device

Description
This command selects the Libero design device for the Programming Connectivity and Interface tool within Libero. This command is needed when the tool cannot automatically resolve the Libero design device when there are two or more identical devices that match the Libero design device in the configured JTAG chain.

Note: This Tcl command is typically used in a Tcl command script file that is passed to the Libero run_tool command.

```
run_tool -name {CONFIGURE_CHAIN} -script {flashPro_cmd.tcl}
```

```
select_libero_design_device -name { device_name }
```

Arguments

Parameter	Type	Description
name	string	Specifies a user-assigned unique device name in the JTAG chain.

Supported Families

	Supported Families	Supported Libero SoC Versions
PolarFire	yes	v12.4+
RTG4	yes	v12.4+
SmartFusion2	yes	v12.4+
IGLOO2	yes	v12.4+

Example
This command selects the Libero design device with {M2S050TS (2)} name and {my_design_device} name for the Programming Connectivity and Interface tool within Libero:

```
select_libero_design_device -name {M2S050TS (2)}
```

```
select_libero_design_device -name {my_design_device}
```

2.74 select_profile

Description
This Tcl command selects an existing profile to use in your project.

```
select_profile -name profilename
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of the profile you wish to use.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command selects an existing profile 'custom1'.

```
select_profile -name custom1
```

See Also
- 2.4 add_profile
- 2.25 edit_profile
- 2.65 remove_profile
- 2.39 export_profiles

2.75 set_actel_lib_options

Description
This Tcl command enables you to choose the default library for your device, or to specify your own library. Enter the full pathname of your own library to use it for simulation.

```
set_actel_lib_options -use_default_sim_path value -sim_path {path}
```

Arguments

Parameter	Type	Description
use_default_sim_path	boolean	Enables/Disables you to choose the default library for your device. This is mandatory. Valid values are: • TRUE, true, 1 - Uses the default simulation library. • FALSE, false, 0 - Disables the default simulation library; enables you to specify a different simulation library with the `-sim_path {full pathname}` option.
sim_path	string	Specifies the full pathname to your simulation library. It is optional.

Error Codes

Error Code	Description
None	Required parameter 'use_default_sim_path' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example specifies "c:\sim_lib\test." as simulation library.

```
set_actel_lib_options -use_default_sim_path FALSE -sim_path {c:\sim_lib\test}
```

See Also
- 2.79 set_modelsim_options

2.76 set_as_target

Description
This Tcl command sets a SDC, PDC file as the target file to receive and store new constraints. To set *.sdc *.pdc files as target, click the Manage Constraints tool name from Design Flow.

```
set_as_target -type { constraint_file_type } -file { constraint_file_path }
```

Arguments

Parameter	Type	Description
type	string	Specifies the file type: sdc, io_pdc or fp_pdc.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
This command sets the SDC file project_folder /constraints/user.sdc as the target to receive and store new SDC commands.

```
set_as_target -type {sdc} -file {./constraint/user.sdc}
```

This command sets the PDC file project_folder /constraints/user.pdc as the target to receive and store new PDC commands.

```
set_as_target -type {pdc} -file {./constraint/user.pdc}
```

See Also

- 2.84 unset_as_target

2.77 set_device

Description

This Tcl command sets your device family, die, and package in the Project Manager. This command generates set_device.log file that contains information of device details. The file is saved in the `<project_name>/tooldata` folder.

Note: The changes you made may invalidate your components and/or design flow; you may have to regenerate your components and rerun your design flow.

```
set_device [-family family ] [-die die ] [-die_voltage value ] [-part_range part_range ] [-
package package ] [-speed speed_grade ] [-adv_options value ]
```

Arguments

Parameter	Type	Description
family	string	Sets device family. If you have already set device name, you cannot change the device family name for the current project, you can change the rest of the arguments.
die	string	Sets device die. It is optional.
die_voltage	floating point	Sets the die voltage for your project. Possible values depend on your device. See your device datasheet for details.
part_range	string	Sets your default temperature range for your project • PolarFire: EXT, IND, MIL • SmartFusion2: COM, IND, TGrade 2, MIL • IGLOO2: COM, IND, TGrade 1, TGrade 2, MIL • RTG4: MIL
package	string	Sets device package. It is optional.
speed	string	Sets device speed grade. Valid value is: -1 and STD. It is optional.

.....continued		
Parameter	**Type**	**Description**
adv_options	string	Sets your advanced options, such as temperature and voltage settings. • IO_DEFT_STD:LVTTL - sets your I/O default value to LVTTL. • TEMPR:COM - sets your default temperature range; can be COM (Commercial), MIL (Military) or IND (industrial). • VCCI_1.5_VOLTR:COM - sets VCCI to 1.5 and voltage range to Commercial. • VCCI_1.8_VOLTR:COM - sets VCCI to 1.8 and voltage range to Commercial. • VCCI_2.5_VOLTR:COM - sets VCCI to 2.5 and voltage range to Commercial. • VCCI_3.3_VOLTR:COM - sets VCCI to 3.3 and voltage range to Commercial. • VOLTR:COM - sets your voltage range; can be COM (Commercial), MIL (Military) or IND (industrial). • RESTRICTPROBEPINS:1 - (For SmartFusion2, IGLOO2 and RTG4 only) Sets to 1 to reserve your pins for probing if you intend to debug using SmartDebug.

You can pass custom specific temperature ranges depending on the selected part range.

The value parameter needs minimum, typical and maximum temperatures values to be added. The following table denotes the minimum, typical and maximum temperatures that are preset by the tool. You can set any value for the temperatures present within the range.

Part Range	Minimum	Typical	Maximum
EXT	0	25	100
COM (Available only for SmartFusion2 and IGLOO2 devices)	0	25	85
IND	- 40	25	100
MIL	- 55	25	125

Error Codes

Error Code	Description
None	Unable to select speed 'ALL'. The speed value must be one of the following: '-1 STD'.
None	Invalid value "TGrade2" for PART_RANGE variable; expected one of COM IND.
None	Invalid value "MIL" for VCCI_1.2_VOLTR variable; expected one of COM IND.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command sets SmartFusion2 family with M25005 die, 484 FBGA package, {-1} speed, IND part range, temperature and voltage settings to your device.

```
set_device -family {SmartFusion2} \
-die {M2S005} \
-package {484 FBGA} \
-speed {-1} \
-die_voltage {1.2} \
-part_range {IND} \
-adv_options {IO_DEFT_STD:LVCMOS 2.5V} \
-adv_options {RESERVEMIGRATIONPINS:1} \
-adv_options {VCCI_2.5_VOLTR:COM} \
-adv_options {VCCI_3.3_VOLTR:COM} \
-adv_options {VOLTR:IND}
```

The following commands sets RTG4 family with RTG4150_ES die, 1657_CG package, {STD} speed, MIL part range, and custom temperature range set at a minimum = 0, typical = 25, and maximum = 70.

```
set_device -family {RTG4} -die {RTG4150_ES} -package {1657_CG} -speed {STD} -die_voltage
{1.2} -part_range {MIL} -adv_options  {TEMPR:0 25 70}
```

The following commands sets PolarFire family with MPF200T die, FCG784 package, {-1} speed, EXT part range, and custom temperature range set at a minimum = 0, typical = 25, and maximum = 70.

```
set_device -family {PolarFire} -die {MPF200T} -package {FCG784} -speed {-1} -part_range {EXT}
-adv_options  {TEMPR:0 25 70}
```

2.78 set_live_probe

Description
This Tcl command set_live_probe channels A and/or B to the specified probe point(s). At least one probe point must be specified. Only exact probe name is allowed (i.e. no search pattern that may return multiple points).

```
set_live_probe [-deviceName device_name ] [-probeA probe_name ] [-probeB probe_name ]
```

Arguments

Parameter	Type	Description
deviceName	string	Parameter is optional if only one device is available in the current configuration or set for debug (see SmartDebug user guide for details).
probeA	string	Specifies target probe point for the probe channel A.
probeB	string	Specifies target probe point for the probe channel B.

Supported Families

	Supported Families	Supported Libero SoC Versions
PolarFire	yes	v12.4+
RTG4	yes	v12.4+
SmartFusion2	yes	v12.4+
IGLOO2	yes	v12.4+

Exceptions

- The array must be programmed and active
- Active probe read or write operation will affect current settings of Live probe since they use same probe circuitry inside the device
- Setting only one Live probe channel affects the other one, so if both channels need to be set, they must be set from the same call to set_live_probe
- Security locks may disable this function
- In order to be available for Live probe, ProbeA and ProbeB I/O's must be reserved for Live probe respectively

Example

Sets the Live probe channel A to the probe point A12 on device sf2:

```
set_live_probe [-deviceName sf2] [-probeA A12]
```

Sets the Live probe channel A to the probe point A12 on device MPF300TS_ES:

```
set_live_probe [-deviceName MPF300TS_ES] [-probeA A12]
```

2.79 set_modelsim_options

Description

This Tcl command sets your ModelSim simulation options. You can set change how Libero SoC handles Do files in simulation, import your own Do files, set simulation run time, and change the DUT name used in your simulation. You can sets options from the *Project Settings > Simulation* menu. Default values are used if parameters are omitted.

```
set_modelsim_options \
[-use_automatic_do_file "TRUE | FALSE"] \
[-user_do_file {path}] \
[-sim_runtime {value}] \
[-tb_module_name {value}] \
[-tb_top_level_name {value}] \
[-include_do_file "TRUE | FALSE" \
[-included_do_file {value}] \
[-type {value}] \
[-resolution {value}] \
[-add_vsim_options {value}] \
[-display_dut_wave "TRUE | FALSE"] \
[-log_all_signals "TRUE | FALSE"] \
[-do_file_args {value}] \
[-dump_vcd "TRUE | FALSE"] \
[-vcd_file "VCD file name"] \
[-sdf_corner "sdf_corner"] \
[-verilog {value}] \
[-VHDL {value}] \
[-disable_pulse_filtering "TRUE | FALSE"] \
[-timeunit {value}] \
[-timeunit_base {value}] \
[-precision {value}] \
[-precision_base {value}]
```

Arguments

Parameter	Type	Description
use_automatic_do_file	boolean	Automatically create a DO file that will enable you to simulate your design. Valid values are: • TRUE, true, 1 - uses the default automatic.do file in your project. Default this box checked. • FALSE, false, 0 - uses a different *.do file; use the other simulation options to specify it.
user_do_file	string	Specifies the location of your user-defined *.do file.
sim_runtime	number and unit of time	Sets your simulation runtime. It is optional. Value is the number and unit of time, such as {1000 ns}.
tb_module_name	string	Specifies your test bench module name, where value is the name. Default value is "test bench".
tb_top_level_name	string	Sets the top-level instance name in the test bench, where value is the name. Default is <top>_0.
include_do_file	boolean	Enables you to include DO file. Valid values are: • TRUE, true, 1 - Includes the *.do file. • FALSE, false, 0 - Does not include the *.do file.
included_do_file	string	Specifies the path of the included *.do file, where value is the name of the file. Including a DO file enables you to customize the set of signal waveforms that will be displayed in ModelSim. Specify this argument with `-include_do_file` argument. Default is `work.do`.
type	string	Resolution type; possible values are: • min - minimum. Default value. • typ - typical. • max - maximum.
resolution	unit of time	Sets your resolution value. Value is the number and unit of time, such as {1ps}. The default is family-specific, but you can customize it to fit your needs.
add_vsim_options	string	Adds more Vsim options, where value specifies the option(s).
display_dut_wave	boolean	Enables ModelSim to display signals for the tested design; possible values are: • FALSE, false, 0 - displays the signal for the top_level_test bench. • TRUE, true, 1 - enables ModelSim to display the signals for the tested design.
log_all_signals	boolean	Saves and logs all signals during simulation. • TRUE, true, 1 - logs all signals. • FALSE, false, 0 - does not log all signals. Default this box checked.
do_file_args	list of strings	Specifies *.do file command parameters. Default is empty.

.....continued

Parameter	Type	Description
dump_vcd	boolean	Dumps the VCD file when simulation is complete; possible values are: • TRUE, true, 1 - dumps the VCD file • FALSE, false, 0 - does not dump the VCD file. Default this box checked.
vcd_file	string	Specifies the name of the dumped VCD file, where value is the name of the file. Default is "power.vcd".
sdf_corner {parameter}	string	Sets the corner on which the post layout simulation will be done. • slow_lv_ht - slow process, low voltage and high temperature operating conditions. Default value. • slow_lv_lt - slow process, low voltage and low temperature operating conditions. • fast_hv_lt - fast process, high voltage and low temperature operating conditions.
verilog	integer	HDL Testbench file type can be either Verilog or VHDL; possible values are: 1 or 0. Default is 0.
VHDL	integer	HDL Testbench file type can be either Verilog or VHDL; possible values are: 1 or 0. Default is 0.
disable_pulse_filtering	boolean	Specifies to enable/disable pulse filtering during SDF based simulations. • TRUE, true, 1 - enable pulse filtering. • FALSE, false, 0 - disable pulse filtering. Default value.
timeunit	integer	TimeScale time unit value. Default is 1.
timeunit_base	unit of time	TimeScale precision base value. The default setting is ns; possible values are: • s - second • ms - milisecond • us - microsecond • ns - nanosecond • ps - picosecond • fs - femtosecond
precision	integer	TimeScale precision value. Default is 100.
precision_base	unit of time	TimeScale precision base value. The default setting is ps; possible values are: • s - second • ms - milisecond • us - microsecond • ns - nanosecond • ps - picosecond • fs - femtosecond

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Sets ModelSim options to use the automatic *.do file, sets simulation runtime to 1000ns, sets the testbench module name to "testbench", sets the testbench top level to top_0, sets simulation type to "max", resolution to 1ps, adds no vsim options, does not log signals, adds no additional DO file arguments, dumps the VCD file with a name power.vcd.

```
set_modelsim_options -use_automatic_do_file 1 -sim_runtime {1000ns} \
-tb_module_name {testbench} -tb_top_level_name {top_0} -include_do_file 0 \
-type {max} -resolution {1ps} -add_vsim_options {} -display_dut_wave 0 \
-log_all_signals 0 -do_file_args {} - dump_vcd 0 -vcd_file {power.vcd}
```

See Also
- 2.75 set_actel_lib_options

2.80 set_option

Description
This Tcl command sets your synthesis and FPGA Hardware Breakpoint Auto Instantiation options on a module. Default values are used if parameters are omitted.

```
set_option [-synth "TRUE | FALSE" ] [-fhb "TRUE | FALSE" ] [-physynth "TRUE | FALSE" ] [-
module " module_name "] [-report_preserve_cdc "TRUE | FALSE" ] [-min_cdc_sync_flops " value
"]
```

Arguments

Parameter	Type	Description
synth	boolean	Enables/Disables Libero synthesis tool for root design in your project. Default is 1. Possible values are: • TRUE | true | 1 - Enables synthesis tool for your design. Default value. • FALSE | false | 0 - Disables synthesis tool for your design.
fhb	boolean	• TRUE | true | 1 - Enables FPGA Hardware Breakpoint Auto Instantiation. Default value. • FALSE | false | 0 - Disable FPGA Hardware Breakpoint Auto Instantiation.
physynth	boolean	Valid values are: TRUE, true, 1, FALSE, false or 0.
module	string	Identifies the module on which you will run synthesis. Default is {<root_design_name>::work}.
report_preserve_cdc	boolean	• TRUE | true | 1 - Enables the CDC reporting. Default value. • FALSE | false | 0 - Disables the CDC reporting for your design.

.....continued		
Parameter	**Type**	**Description**
min_cdc_sync_flops	string	Enter the minimum number of synchronizer registers to be detected and reported in CDC report. Default value is 2. It is illegal to set value less than 2 and warning will be generated by Libero SoC if value is set less than 2. If the number of synchronizer registers are less than the minimum value set, then those paths will be tagged as Unsafe CDC paths.

Error Codes

Error Code	Description
None	Project Manager does not support PLACE, you cannot activate it in the flow.
None	Invalid argument value: expecting TRUE, 1, true, FALSE, 0 or false.
None	Parameter 'module' has illegal value.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command sets synthesis option on the module test1.vhd.

```
set_option -synth TRUE -module module_name
```

2.81 set_root

Description
This Tcl command sets the module you specify as the root. Project has one top module that is the root of the design hierarchy for the implementation.

```
set_root module_name
```

Arguments

Parameter	Type	Description
set_root	string	Specifies the name the module you want to set as root.

Error Codes

Error Code	Description
None	Cannot find module name.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Set the module mux8 as root of your design hierarchy.

```
set_root mux8
```

2.82 set_user_lib_options

Description
This Tcl command sets your user library options during simulation. If you do not use a custom library these options are not available.

```
set_user_lib_options \
-name {value} \
-path {path} \
-option {value}
```

Arguments

Parameter	Type	Description
name	string	Sets the name of your user library. This is mandatory.
path	string	Sets the path name of your user library. This is mandatory.
option	string	Sets your default compile options on your user library. This is mandatory; possible values are: • do_not_compile - user library is not compiled. • refresh - user library is refreshed. • compile - user library is compiled. • recompile - user library is recompiled. • refresh_and_compile - user library is refreshed and compiled.

Error Codes

Error Code	Description
None	Required parameter 'path' is missing.
None	Required parameter 'name' is missing.
None	Required parameter 'option' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The example below sets the name for the user library to "test1", the path to `c:/msemi_des_files/libraries/test1`, and the compile option to "do not compile".

```
set_user_lib_options -name {test1} -path {c:/msemi_des_files/libraries/test1} \
-option {do_not_compile}
```

2.83 unlink_files

Description
This Tcl command removes a link to a file in your project and it is not removed link to a folder. You can unlink a file form a folder, and to unlink the whole folder you have to unlink all files one by one from the link folder.

```
unlink_files -file { absolute or related path and name of the linked file } [-from_disk value]
```

Arguments

Parameter	Type	Description
file	string	Absolute or relative path and name of the linked (remote) file you want to unlink. There may be multiple -file arguments (see example below).
from_disk	boolean	Removes file from disk. Valid values are: TRUE, FALSE.

Error Codes

Error Code	Description
None	Parameter 'file' is missing or has invalid value.
None	Required parameter 'file' is missing.
None	Unable to find the file.
None	Error: Valid command formatting is 'unlink_files [-file "file"]+ [-from_disk "TRUE | FALSE"].

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

1. Unlink the file file1.vhd from my project.

```
unlink_files -file {E:\Share\file1.vhd}
```

2. Unlink files “E:\Share\abc.v” and “E:\Share\abc.sdc" from the project using Environment variable “MSCC_ROOT_1” that has the root directory path “E:\Share”.

```
unlink_files -file {${MSCC_ROOT_1}/abc.v} -file {${MSCC_ROOT_1}/abc.sdc}
```

See Also

- 2.19 create_links
- 2.7 change_link

2.84 unset_as_target

Description

This Tcl command unsets a target file in the Constraints view to not to store constraints. To unset *.sdc *.pdc files as target, click the Manage Constraints tool name from Design Flow.

```
unset_as_target -file { absolute path to file }
```

Arguments

Parameter	Type	Description
file	string	Specifies the absolute path to file to be unset as a target.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

This command unsets the SDC file <project_folder>/constraint/user.sdc as target to not store SDC command.

```
unset_as_target -file {constraint/user.sdc}
```

See Also

- 2.76 set_as_target

2.85 use_source_file

Description
This Tcl command defines a module for your project.

```
use_source_file \
-file {full pathname} \
-module value
```

Arguments

Parameter	Type	Description
file	string	Specifies the Verilog or VHDL file. Value is the name of the file you wish use (including the full pathname). This is mandatory.
module	string	Specifies the module in which you want to use the file. This is mandatory.

Error Codes

Error Code	Description
None	Required parameter 'file' is missing.
None	Required parameter 'module' is missing.
None	'file1.vhd' does not define module 'top'.
None	'/prj/hdl/file1.v' is not in the project.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
Specify file1.vhd in the `./project/hdl` directory, in the module named top.

```
use_source_file -file "./project/hdl/file1.vhd" -module "top"
```

3. SmartDesign Tcl Commands

The SmartDesign Tcl commands can be used to create a design in the SmartDesign. You must either create or open a SmartDesign before you can use any of the SmartDesign commands - sd_* .

All SmartDesign Tcl commands are supported by the SmartFusion2, IGLOO2, and RTG4 families.

3.1 sd_add_pins_to_group

```
sd_add_pins_to_group \
-sd_name smartdesign_component_name \
-instance_name instance_name \
-group_name group_name \
-pin_names pin_names
```

Tcl command; adds one or more pins to a pin group on an instance in a SmartDesign component.

3.1.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-instance_name instance_name`

Specifies the name of the instance on which the pin group is present. It is mandatory.

`-group_name group_name`

Specifies the name of the group to add the pins to. It is mandatory.

`-pin_names pin_names`

Specifies the list of instance pins to be added to the pin group. It is mandatory.

3.1.2 Examples

```
sd_add_pins_to_group -sd_name {TOP} -instance_name
{COREAXI4INTERCONNECT_C0_0} -group_name {Group} -pin_names {ARESETN ACLK}
```

3.1.2.1 See Also

Tcl Command Documentation Conventions

3.2 sd_clear_pin_attributes

```
sd_clear_pin_attributes \
-sd_name smartdesign_component_name \
-pin_names port_or_pin_names
```

Tcl command; clears all attributes on one or more pins/ports in a SmartDesign. Pin attributes include pin inversion, mark as unused and constant value settings.

3.2.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-pin_names port_or_pin_names`

Specifies the name of the port/pin for which all the attributes must be cleared. It is mandatory.

3.2.2 Examples

```
sd_clear_pin_attributes -sd_name {sd1} -pin_names {RAM1K18_0:A_DOUT_CLK}
sd_clear_pin_attributes -sd_name {top} -pin_names {CARRY_OUT}
```

3.2.3 Notes

This command will not work on multiple pins/ports in this release. Support for multiple pins/ports will be provided in the next Libero release. This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.2.3.1 See Also

Tcl Command Documentation Conventions

3.3 sd_configure_core_instance

```
sd_configure_core_instance \
-sd_name smartdesign_component_name \
-instance_name core_instance_name \
-params core_parameters \ [-validate_rules 0|1]
```

Tcl command; configures the parameters of a core instance (Direct Instantiation) in a SmartDesign component. This command is typically used after instantiating a core from the catalog directly into a SmartDesign component (Direct Instantiation) without first creating a component for the core (using sd_instantiate_core). This command can configure multiple core parameters at a time.

3.3.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-instance_name instance_name`

Specifies the name of the core instance in the SmartDesign which needs to be configured. It is mandatory.

`-params core_parameters`

Specifies the parameters that need to be configured for the core instance. It is mandatory.

`-validate_rules 0|1`

Validates the rules of the updated configuration. It is optional.

3.3.2 Examples

```
sd_configure_core_instance -sd_name {SD1} -instance_name {COREFIFO_0} - params
{"SYNC:0" "param2:value2" "param3:value3"} -validate_rules 0
```

3.3.2.1 See Also

Tcl Command Documentation Conventions

3.4 sd_connect_instance_pins_to_ports

```
sd_connect_instance_pins_to_ports \
-sd_name smartdesign_component_name \
-instance_name instance_name
```

Tcl command; connects all pins of an instance to new SmartDesign top level ports.

3.4.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-instance_name` *`instance_name`*

Specifies the instance name for which all the pins must be connected to top level ports. It is mandatory. The instance pins are connected to new top level ports created with the same instance pin names. If a top level port with the same name already exists, then the tool automatically creates a new port with name

<port_name>_<index> (index is an automatically generated integer starting at 0 such that the port name is unique in the SmartDesign).

3.4.2 Examples

```
sd_connect_instance_pins_to_ports -sd_name {top} -instance_name
{CORESPI_C0_0}
sd_connect_instance_pins_to_ports -sd_name {top} -instance_name
{ddr_out_0}
```

3.4.3 Notes

This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.4.3.1 See Also

Tcl Command Documentation Conventions

3.5 sd_connect_net_to_pins

Tcl command; connects a list of SmartDesign top level ports and/or instance pins to a net.

```
sd_connect_net_to_pins \
-sd_name smartdesign_component_name \
-net_name net_name \-pin_names port_or_pin_names
```

3.5.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-net_name` *`net_name`*

Specifies the name of the net to be connected to pins/ports in the SmartDesign component. It is mandatory.

`-pin_names` *`port_or_pin_names`*

Specifies the name of the ports/pins to be connected to the net in the SmartDesign. It is mandatory. The command will fail if:

- The ports/pins do not exist.
- The ports/pins and the net being connected are of different range/size.
- There is more than one port/pin driving the net.

3.5.2 Examples

```
sd_connect_net_to_pins -sd_name {shifter} -net_name {ready_net} -pin_names {"READY"}
sd_connect_net_to_pins -sd_name {top} -net_name {clk_net} -pin_names {CLK
RAM64x12_0:R_CLK RAM64x12_0:W_CLK}
```

3.5.3 Notes

This command is not required to build a SmartDesign component. It is not exported when you select Libero **Project** - `**Export Script File**' or '**Export Component Description(Tcl)**' on a SmartDesign component. This command is typically used in conjunction with 'sd_create_*_net' command to connect two or more ports/pins to a net.

3.5.3.1 See Also

Tcl Command Documentation Conventions

3.6 sd_connect_pins_to_constant

```
sd_connect_pins_to_constant \
-sd_name smartdesign_component_name \
-pin_names port_or_pin_names \
-value constant_value
```

Tcl command; connects SmartDesign top level output ports or input instance pins to constant values.

3.6.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-pin_names` *`port_or_pin_names`*

Specifies the names of the top level output ports or the instance level input pins to be tied to constant values. It is mandatory. Bus pins/ports and pin/port slices can also be tied to constant values. This command will fail if the specified port/pin does not exist. The command will also fail if the assigned object is a port of direction IN/INOUT or a pin of direction OUT/INOUT.

`-value` *`constant_value`*

Specifies the constant value to be assigned to the port/pin. It is mandatory. The acceptable values to this argument are GND/VCC/hexadecimal numbers.

3.6.2 Examples

```
sd_connect_pins_to_constant -sd_name {top} -pin_name {bypass} -value
{GND}
sd_connect_pins_to_constant -sd_name {top} -pin_name {sle_0:en} -value
{VCC}
sd_connect_pins_to_constant -sd_name {top} -pin_name {ram64x12_0:w_data}
-value {0x7f}
```

3.6.3 Notes

This command will not work on multiple pins/ports in this release. Support for multiple pins/ports will be provided in the next Libero release.

3.6.3.1 See Also

Tcl Command Documentation Conventions

3.7 sd_connect_pin_to_port

```
sd_connect_pin_to_port \
-sd_name smartdesign_component_name \
```

Tcl command; connects a SmartDesign instance pin to a new top level port. This command is equivalent to the 'Promote to Top Level` GUI action on an instance pin.

```
-pin_name pin_name \
[-port_name port_name]
```

3.7.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-pin_name pin_name`

Specifies the name of the instance level pin that needs to be connected to a top level port. It is mandatory.

`-port_name port_name`

Specifies the name of the new top level port that the instance pin will be connected to. It is optional. If the port name is not specified, the new port takes the name of the instance pin. If the port name as defined by these rules already exists, the tool automatically creates a new port with name <port_name>_<index> (index is an automatically generated integer starting at 0 such that the port name is unique in the SmartDesign).

3.7.2 Examples

```
sd_connect_pin_to_port -sd_name {top} -pin_name {DFN1_0:D} sd_connect_pin_to_port -
sd_name {top} -pin_name {DFN1_0:Q} -port_name
{Q_OUT}
```

3.7.3 Notes

This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.7.3.1 See Also

Tcl Command Documentation Conventions

3.8 sd_connect_pins

```
sd_connect_pins \
-sd_name smartdesign_component_name \
-pin_names port_or_pin_or_slice_names_
```

Tcl command; connects a list of SmartDesign top level ports and/or instance pins together.

3.8.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-pin_names port_or_pin_or_slice_names`

Specifies the port names, pin names and/or slice names to be connected together. It is mandatory. This command will fail if the ports, pins or slices do not exist. This command will also fail if the ports, pins and/or slices are not of the same size/range.

3.8.2 Examples

```
sd_connect_pins -sd_name {top} -pin_names {CLK MACC_PA_0:CLK DFN1_0:CLK}
sd_connect_pins -sd_name {top} -pin_names {MACC_PA_0:A RAM1K20_0:A_DIN[17:0]}
```

3.8.2.1 See Also

Tcl Command Documentation Conventions

3.9 sd_create_bif_net

```
sd_create_bif_net \
-sd_name smartdesign_component_name \
-net_name net_name
```

Tcl command; creates a bus interface (BIF) net in a SmartDesign component. Any net created must be connected to two or more ports/pins using the command “sd_connect_net_to_pins”.

3.9.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-net_name net_name`

Specifies the name of the net to be added in the SmartDesign component. It is mandatory. The command will fail if there is an already existing net with the same name.

3.9.2 Examples

```
sd_create_bif_net -sd_name {TOP} -net_name {bifnet1}
```

Note: This new bif net is visible in the UI only when it is connected to two or more ports/pins using the command “sd_connect_net_to_pins” as shown below.

```
sd_connect_net_to_pins -sd_name {TOP} -net_name {bifnet1} -pin_names {"AHBmmaster0"
"CoreAHBLite_C0_0:AHBmmaster0"}
```

3.9.3 Notes

This command is not required to build a SmartDesign component. It is not exported when you select Libero **Project** - `**Export Script File**’ or **‘Export Component Description(Tcl)**’ on a SmartDesign component. This command is used to manually create a Tcl script and specify a new name to the net that connects two or more ports/pins.

3.9.3.1 See Also

Tcl Command Documentation Conventions

3.10 sd_create_bif_port

Tcl command; creates a SmartDesign Bus Interface port of a given type. This command is used to create top level Bus Interface ports in a SmartDesign component to connect to the instance level Bus Interface ports of the same type.

```
sd_create_bif_port \
-sd_name smartdesign_component_name \
-port_name port_name \
-port_bif_vlnv vendor:library:name:version \
-port_bif_role port_bif_role \
-port_bif_mapping [bif_port_name:port_name]+
```

To use this command, it is recommended to first use the GUI to instantiate the core component or the HDL module with Bus Interface port to be promoted in the SmartDesign. Then use the UI action "Promote to Top Level" on the Bus Interface port of interest and export the Tcl script for the SmartDesign component by selecting " Export Component Description(Tcl)" on the right-click menu of the SmartDesign component in the Design Hierarchy. You can then use the Tcl command ‘sd_create_bif_port’ from the exported Tcl script (note to change the SmartDesign name in the command) to create a bus interface port anywhere in a regular Libero script. Note that there can be different Bus Interface types and roles defined by the arguments –port_bif_vlnv and – port_bif_role.

3.10.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-port_name` *`port_name`*

Specifies the name of the Bus Interface port to be added in the SmartDesign. It is mandatory.

`-port_bif_vlnv {vendor:library:name:version}`

Specifies the version identifier of the Bus Interface port to be added in the SmartDesign. It is mandatory.

`-port_bif_role {port_bif_role}`

Specifies the role of the Bus Interface port to be added in the SmartDesign. Role values depend on the type of Bus Interface (VLNV) that is being defined for the port. The figure below shows the roles for different Bus Interface ports supported by Libero.

Name	Vendor	Library	Role
AHB	AMBA	AMBA2	master
AHB	AMBA	AMBA2	slave
AHB	AMBA	AMBA2	mirroredMaster
AHB	AMBA	AMBA2	mirroredSlave
APB	AMBA	AMBA2	master
APB	AMBA	AMBA2	slave
APB	AMBA	AMBA2	mirroredMaster
APB	AMBA	AMBA2	mirroredSlave
AXI	AMBA	AMBA3	master
AXI	AMBA	AMBA3	slave
AXI	AMBA	AMBA3	mirroredMaster
AXI	AMBA	AMBA3	mirroredSlave
AXI	AMBA	AMBA3	system
AXI4	AMBA	AMBA4	master
AXI4	AMBA	AMBA4	slave
AXI4	AMBA	AMBA4	mirroredMaster
AXI4	AMBA	AMBA4	mirroredSlave
DDR3	Actel	busdef.memory	master
DDR3	Actel	busdef.memory	slave
PF_APB_LINK	Actel	busdef.link	master
PF_APB_LINK	Actel	busdef.link	slave
PF_CDR_CLK	Actel	busdef.clock	master
PF_CDR_CLK	Actel	busdef.clock	slave
PF_DRI	Actel	busdef.dri	master
PF_DRI	Actel	busdef.dri	slave
PF_DRI	Actel	busdef.dri	mirroredMaster
PF_DRI	Actel	busdef.dri	mirroredSlave
PF_TXPLL_XCVR_CLK	Actel	busdef.clock	master
PF_TXPLL_XCVR_CLK	Actel	busdef.clock	slave

`-port_bif_mapping {[bif_port_name:port_name ]+}`

Specifies the mapping between the bus interface formal names and the SmartDesign ports mapped onto that bus interface port. It is mandatory.

3.10.2 Examples

```
sd_create_bif_port -sd_name {sd1} -port_name {BIF_1} -port_bif_vlnv
{AMBA:AMBA2:APB:r0p0} -port_bif_role {slave} -port_bif_mapping {\ "PADDR:PADDR" \
"PSELx:pselx" \ "PENABLE:PENABLE" \
```

```
"PWRITE:PWRITE" \ "PRDATA:PRDATA" \ "PWDATA:PWDATA" \ "PREADY:PREADY" \
"PSLVERR:PSLVERR" }
```

3.10.2.1 See Also

Tcl Command Documentation Conventions

3.11 sd_create_bus_net

```
sd_create_bus_net \
-sd_name smartdesign_component_name \
-net_name net_name \
-net_range [left_index_range:right_index_range]
```

Tcl command; creates a bus net of a given range in a SmartDesign component. Any net created must be connected to two or more ports/pins using the command “sd_connect_net_to_pins”.

3.11.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-net_name net_name`

Specifies the name of the net to be added in the SmartDesign component. It is mandatory.

`-net_range [left_index_range:right_index_range]`

Specifies the range of the net added to the SmartDesign component. The range is defined by its left and right range indices. It is mandatory.

3.11.2 Examples

```
sd_create_bus_net -sd_name {top} -net_name {ab1} -net_range {[5:0]]}
```

Note: This new net is visible in the UI only when it is connected to two or more ports/pins using the command “sd_connect_net_to_pins” as shown below.

```
sd_connect_net_to_pins -sd_name {top} -net_name {ab1} -pin_names {a RAM64x12_0:R_ADDR}
```

3.11.3 Notes

This command is not required to build a SmartDesign component. It is not exported when you select Libero **Project** - `**Export Script File**’ or ‘**Export Component Description(Tcl)**’ on a SmartDesign component. This command is used to manually create a Tcl script and specify a new name to the net that connects two or more ports/pins.

3.11.3.1 See Also

Tcl Command Documentation Conventions

3.12 sd_create_bus_port

```
sd_create_bus_port \
-sd_name smartdesign_component_name \
-port_name port_name \-port_direction IN|OUT|INOUT \
-port_range {[left_range_index:right_range_index]}
```

Tcl command; creates a bus port of a given range in a SmartDesign component.

3.12.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-port_name port_name`

Specifies the name of the bus port added to be SmartDesign component. It is mandatory.

`-port_direction IN|OUT|INOUT`

Specifies the direction of the bus port added to the SmartDesign component. It is mandatory.

`-port_range {[left_range_index:right_range_index]}`

Specifies the range of the bus port added to the SmartDesign component. The range is defined by the left and right indices. It is mandatory. The range must be specified inside the square brackets.

3.12.2 Examples

```
sd_create_bus_port -sd_name {top} -port_name {test_port13} -port_direction {OUT} -
port_range {[9:36]}
```

```
sd_create_bus_port -sd_name {top} -port_name {test_port4} -port_direction {IN} -
port_range {[31:0]}
```

3.12.2.1 See Also

Tcl Command Documentation Conventions

3.13 sd_create_pin_group

Tcl command; creates a group of pins in a SmartDesign component. A pin group is only used to manage the complexity of the SmartDesign canvas. There is no actual netlist functionality related to pin group commands. Pin groups cannot be created for top level ports.

```
sd_create_pin_group \
-sd_name smartdesign_component_name \
-instance_name instance_name \
[-group_name group_name] \
[-pin_names pin_to_be_added_to_the_group]
```

3.13.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-instance_name instance_name`

Specifies the name of the instance on which the pin group is added. It is mandatory.

`-group_name group_name`

Specifies the name of the pin group. It is optional. If the group name is not specified, the default name will be 'Group'. If the name 'Group' is already taken, then the group name will be 'Group_<index>' (index is auto-incremented).

`-pin_names pins_to_be_added_to_the_group`

Specifies the list of instance pins to be added to the pin group. It is optional.

3.13.2 Examples

```
sd_create_pin_group -sd_name {TOP} -instance_name
{COREAXI4INTERCONNECT_C0_0} -group_name {MyGroup} -pin_names {ACLK ARESETN}
```

3.13.2.1 See Also

Tcl Command Documentation Conventions

3.14 sd_create_pin_slices

```
sd_create_pin_slices \
-sd_name smartdesign_component_name \
-pin_name port_or_pin_name \
-pin_slices port_or_pin_slices
```

Tcl command; creates slices for a SmartDesign top level bus port or an instance level bus pin.

3.14.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-pin_name port_or_pin_name`

Specifies the name of the bus port or bus pin to be sliced. It is mandatory. This command will fail if the port/pin is scalar or if the bus port/pin does not exist.

`-pin_slices port_or_pin_slices`

Specifies the port/pin slices as a list of bus ranges which must be contained within the port/pin bus range. It is mandatory. This command will fail if the sliced object is top level OUT/INOUT port and the slice ranges overlap. This command will also fail if the sliced object is an instance level IN/INOUT pin and the slice ranges overlap.

3.14.2 Examples

```
sd_create_pin_slices -sd_name {sub} -pin_name {Rdata} -pin_slices {[4:3] [2:0]} # top level port slicing
```

```
sd_create_pin_slices -sd_name {sub} -pin_name {DDR_memory_arbiter_C0_0:VIDEO_RDATA_4_O}
-pin_slices {[3:3] [2:0]} # instance level pin slicing
```

3.14.2.1 See Also

Tcl Command Documentation Conventions

3.15 sd_create_scalar_net

```
sd_create_scalar_net \
-sd_name smartdesign_component_name \
-net_name net_name
```

Tcl command; creates a scalar net in a SmartDesign component. Any net created must be connected to two or more ports/pins using the command “sd_connect_net_to_pins”.

3.15.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-net_name net_name`

Specifies the name of the net added to the SmartDesign component. It is mandatory.

3.15.2 Examples

```
sd_create_scalar_net -sd_name {top} -net_name {clk_net}
```

Note: This new net is visible in the UI only when it is connected to two or more ports/pins using the command “sd_connect_net_to_pins” as shown below

```
sd_connect_net_to_pins -sd_name {top} -net_name {clk_net} -pin_names {CLK
RAM64x12_0:R_CLK RAM64x12_0:W_CLK}
```

3.15.3 Notes

This command is not required to build a SmartDesign component. It is not exported when you select Libero **Project** - `**Export Script File**' or '**Export Component Description(Tcl)**' on a SmartDesign component. This command is used to manually create a Tcl script and specify a new name to the net that connects two or more ports/pins.

3.15.3.1 See Also

Tcl Command Documentation Conventions

3.16 sd_create_scalar_port

```
sd_create_scalar_port \
-sd_name smartdesign_component_name \
-port_name port_name \
-port_direction IN|OUT|INOUT
```

Tcl command; creates a scalar port in a SmartDesign component.

3.16.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-port_name` *`port_name`*

Specifies the name of the port added to the SmartDesign component. It is mandatory.

`-port_direction` *`IN|OUT|INOUT`*

Specifies the direction of the port added to the SmartDesign component. It is mandatory.

3.16.2 Examples

```
sd_create_scalar_port -sd_name {main} -port_name {po2} -port_direction {INOUT}
```

3.16.2.1 See Also

Tcl Command Documentation Conventions

3.17 sd_delete_instances

```
sd_delete_instances \
-sd_name smartdesign_component_name \
-instance_names instance_names
```

Tcl command; deletes one or more instances from a SmartDesign component.

3.17.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-instance_names` *`instance_names`*

Specifies the instance names to be deleted. It is mandatory.

3.17.2 Examples

```
sd_delete_instances -sd_name {top} -instance_names {RAM64X12_0}
```

```
sd_delete_instances -sd_name {SUB} -instance_names {coreahblite_c0_0
coreriscv_axi4_c0_0 pf_ccc_c0_0}
```

3.17.3 Notes

This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.17.3.1 See Also

Tcl Command Documentation Conventions

3.18 sd_delete_nets

```
sd_delete_nets \
-sd_name smartdesign_component_name \
-net_names net_names
```

Tcl command; deletes one or more nets from the SmartDesign component.

3.18.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-net_names net_names`

Specifies the net names to be deleted. It is mandatory.

3.18.2 Examples

```
sd_delete_nets -sd_name {topp} -net_names {B_REN_0}
```

3.18.3 Notes

This command will not delete multiple nets in this release. Support for deleting multiple nets will be provided in the next Libero release. This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.18.3.1 See Also

Tcl Command Documentation Conventions

3.19 sd_delete_pin_group

```
sd_delete_pin_group \
-sd_name smartdesign_component_name \
-instance_name instance_name \
-group_name group_name
```

Tcl command; deletes a pin group from an instance in a SmartDesign component.

3.19.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-group_name group_name`

Specifies the name of the pin group to be deleted. It is mandatory.

`-instance_name instance_name`

Specifies the name of the instance from which the group pin needs to be deleted. It is mandatory.

3.19.2 Examples

```
sd_delete_pin_group -sd_name {TOP} -instance_name
{COREAXI4INTERCONNECT_C0_0} -group_name {Group}
```

3.19.2.1 See Also

Tcl Command Documentation Conventions

3.20 sd_delete_pin_slices

```
sd_create_pin_slices \
-sd_name smartdesign_component_name \
-pin_name port_or_pin_name \
-pin_slices port_or_pin_slices
```

Tcl command; deletes SmartDesign top level port slices or instance pin slices.

3.20.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-pin_name` *`port_or_pin_name`*

Specifies the name of the bus port or bus pin for which the slices must be deleted. It is mandatory.

`-pin_slices` *`port_or_pin_slices`*

Specifies the ranges of the port and/or pin slices to be deleted. It is mandatory.

3.20.2 Examples

```
sd_delete_pin_slices -sd_name {top} -pin_name {MACC_pa_0:p} -pin_slices
{[21] [13] [28]} # deletes instance pin slices
sd_delete_pin_slices -sd_name {top} -pin_name {A} -pin_slices {[17:16] [15:1] [0]} #
deletes top level port slices
```

3.20.3 Notes

This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.20.3.1 See Also

Tcl Command Documentation Conventions

3.21 sd_delete_ports

```
sd_delete_ports \
-sd_name smartdesign_component_name \
-port_names port_names
```

Tcl command; deletes one or more ports from the SmartDesign component.

3.21.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

-port_names *port_names*

Specifies the names of the ports to be deleted. It is mandatory.

3.21.2 Examples

sd_delete_ports -sd_name {sd1} -port_names {REF_CLK_0}

3.21.3 Notes

This command will not work on multiple ports in this release. Support for multiple ports will be provided in the next Libero release. This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.21.3.1 See Also

Tcl Command Documentation Conventions

3.22 sd_disconnect_instance

sd_disconnect_instance \

-sd_name *smartdesign_component_name* \

-instance_name *instance_name*

Tcl command; clears all the connections on an instance in a SmartDesign component.

3.22.1 Arguments

-sd_name *smartdesign_component_name*

Specifies the name of the SmartDesign component. It is mandatory.

-instance_name *instance_name*

Specifies the name of the instance for which all the connections must be cleared. It is mandatory.

3.22.2 Examples

sd_disconnect_instance -sd_name {sd1} -instance_name {RAM1K18_1}

3.22.3 Notes

This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.22.3.1 See Also

Tcl Command Documentation Conventions

3.23 sd_disconnect_pins

sd_disconnect_pins \

-sd_name *smartdesign_component_name* \

-pin_names *port_or_pin_or_slice_names*

Tcl command; disconnects a list of SmartDesign top level ports and/or instance pins from the net they are connected to.

3.23.1 Arguments

-sd_name *smartdesign_component_name*

Specifies the name of the SmartDesign component. It is mandatory.

`-pin_names port_or_pin_or_slice_names`

Specifies the port, pin and/or slice names to be disconnected. It is mandatory. This command will fail if the ports, pins and/or slices do not exist.

3.23.2 Examples

```
sd_disconnect_pins -sd_name {topp} -pin_names {B_ren RAM1K20_0:B_ADRR[12]}
```

```
sd_disconnect_pins -sd_name {SD1} -pin_names {AND2_0:B AND3_0:B AND3_0:A
PF_XCVR_ERM_C0_0:LANE0_RX_READY}
```

3.23.3 Notes

This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.23.3.1 See Also

Tcl Command Documentation Conventions

3.24 sd_duplicate_instance

Tcl command; creates a new instance in a SmartDesign with the same module/component as the original instance.

```
sd_duplicate_instance \
-sd_name smartdesign_component_name \
-instance_name instance_name \[-duplicate_instance_name duplicate_instance_name]
```

3.24.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-instance_name instance_name`

Specifies the name of the instance to be duplicated. It is mandatory.

`-duplicate_instance_name duplicate_instance_name`

Specifies the name of the duplicate instance. It is optional. If the duplicate_instance_name is not specified, it will be automatically generated as <instance_name><index> (index is an automatically generated integer starting at 0 such that the instance name is unique in the SmartDesign).

3.24.2 Examples

```
sd_duplicate_instance -sd_name {top} -instance_name {PF_CCC_C0_0}
```

```
sd_duplicate_instance -sd_name {top} -instance_name {SUB_0} - duplicate_instance_name
{T1}
```

3.24.3 Notes

This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.24.3.1 See Also

Tcl Command Documentation Conventions

3.25 sd_hide_bif_pins

```
sd_hide_bif_pins \
```

```
-sd_name smartdesign_component_name \
-bif_pin_name name_of_the_bif_pin_or_port \-pin_names pins_or_ports_to_be_exposed
```

Tcl command; hides one or more already exposed internal scalar or bus pins/ports of a Bus Interface pin/port.

3.25.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-bif_pin_name name_of_the_bif_pin_name`

Specifies the name of the Bus Interface pin for which the internal pins must be hidden. It is mandatory.

`-pin_name pins_to_be_exposed`

Specifies the bus interface internal pin/port names to be hidden. It is mandatory.

3.25.2 Examples

```
sd_hide_bif_pins -sd_name {sd1} -bif_pin_name {COREAXI4INTERCONNECT_C0_0:AXI4mmaster0}
- pin_names {COREAXI4INTERCONNECT_C0_0:MASTER0_AWADDR}
```

```
sd_hide_bif_pins -sd_name {SD1} -bif_pin_name {CLKS_FROM_TXPLL_0} -pin_names
{TX_PLL_LOCK_0}
```

3.25.3 Notes

This command will not hide multiple pins/ports in this release. Support to hide multiple pins/ports will be provided in the next Libero release. This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.25.3.1 See Also

Tcl Command Documentation Conventions

3.26 sd_instantiate_component

```
sd_instantiate_component \
-sd_name smartdesign_component_name \
-component_name component_module_name \ [-instance_name instance name]
```

Tcl command; instantiates a Libero SmartDesign component or a core component into another SmartDesign component.

3.26.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component in which other components will be instantiated. It is mandatory.

`-component_name component_module_name`

Specifies the name of the component being instantiated in the SmartDesign component. It is mandatory. The components include SmartDesign components, core components created for different types of cores from the catalog and blocks.

`-instance_name instance name`

Specifies the instance name of the Libero component being instantiated in the SmartDesign component. It is optional. By default, the instance name is <component_module_name>_<index> (index is an automatically generated integer starting at 0 such that the instance name is unique in the SmartDesign).

3.26.2 Examples

```
sd_instantiate_component -sd_name {sub} -component_name {sd1} - instance_name {sd1_0}
```

```
sd_instantiate_component -sd_name {top} -component_name {PF_CCC_C0}
```

3.26.2.1 See Also

Tcl Command Documentation Conventions

3.27 sd_instantiate_core

```
sd_instantiate_core \
-sd_name smartdesign_component_name \-core_vlnv vendor:library:name:verison \[-
instance_name
instance_name]
```

Tcl command; instantiates a core from the catalog directly into a SmartDesign component (Direct Instantiation) without first having to create a component for the core. The file-set related to the core is generated only when the SmartDesign in which the core is instantiated is generated. The GUI equivalent of this command is not currently supported in Libero. To instantiate a core in a SmartDesign component in the GUI, you have to first create a component for the core.

3.27.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-core_vlnv vendor:library:name:verison`

Specifies the version identifier of the core being instantiated in the SmartDesign component. It is mandatory.

`-instance_name instance_name`

Specifies the instance name of the core being instantiated in the SmartDesign. It is optional. By default, the instance name is <core_name>_<index> (index is an automatically generated integer starting at 0 such that the instance name is unique in the SmartDesign).

3.27.2 Examples

```
sd_instantiate_core -sd_name {top} -core_vlnv
{Actel:DirectCore:COREAXI4INTERCONNECT:2.5.100} -instance_name
{COREAXI4INTERCONNECT_C0_0}
```

3.27.2.1 See Also

Tcl Command Documentation Conventions

3.28 sd_instantiate_hdl_core

```
sd_instantiate_hdl_core \
-sd_name smartdesign_component_name \
-hdl_core_name hdl_core_module_name \ [-instance_name instance_name]
```

Tcl command; instantiates a HDL+ core in a SmartDesign component. HDL+ core definition must be created on a HDL module before using this command.

3.28.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-hdl_core_name hdl_core_module_name`

Specifies the name of the HDL+ core module being instantiated in the SmartDesign component. It is mandatory.

-instance_name *instance_name*

Specifies the instance name of the HDL+ core being instantiated in the SmartDesign. It is optional. By default, the instance name is <hdl_core_module_name>_<index> (index is an automatically generated integer starting at 0 such that the instance name is unique in the SmartDesign).

3.28.2 Examples

```
sd_instantiate_hdl_core -sd_name {top} -hdl_core_name {temp} -instance_name {temp3}
```

3.28.2.1 See Also

Tcl Command Documentation Conventions

3.29 sd_instantiate_hdl_module

```
sd_instantiate_hdl_module \
-sd_name smartdesign_component_name \-hdl_module_name hdl_module_name \-hdl_file
hdl_file \ [-instance_name instance_name]
```

Tcl command; instantiates a HDL module in a SmartDesign component. The HDL file in which the HDL module is defined must be imported/linked before running this command.

3.29.1 Arguments

-sd_name *smartdesign_component_name*

Specifies the name of the SmartDesign component. It is mandatory.

-hdl_module_name *hdl_module_name*

Specifies the name of the HDL module being instantiated in the SmartDesign component. It is mandatory.

-hdl_file *hdl_file*

Specifies the path of the HDL file in which the HDL module is defined. The HDL file path can be relative to project folder for imported files but the path has to be complete for linked files. It is mandatory.

-instance_name *instance_name*

Specifies the instance name of the HDL module. It is optional. By default, the instance name is

<hdl_module_name>_<index> (index is an automatically generated integer starting at 0 such that the instance name is unique in the SmartDesign).

3.29.2 Examples

```
sd_instantiate_hdl_module -sd_name {top} -hdl_module_name {and1} -hdl_file
{hdl\and1.v} sd_instantiate_hdl_module -sd_name {top} -hdl_module_name {and_ex} -
hdl_file
{hdl\and_ex.v} -instance_name {test_hdl_hdl_module_name_plus1_1}
```

3.29.2.1 See Also

Tcl Command Documentation Conventions

3.30 sd_instantiate_macro

Tcl command; instantiates a Microsemi primitive macro in a SmartDesign component.

```
sd_instantiate_macro \
-sd_name smartdesign_component_name \
-macro_name macro_module_name | [-instance_name instance_name]
```

3.30.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-macro_name` *`macro_module_name`*

Specifies the name of the macro being instantiated in the SmartDesign component. It is mandatory.

`-instance_name` *`instance_name`*

Specifies the instance name of the macro. It is optional. By default, the instance name is <macro name>_<index> (index is an automatically generated integer starting at 0 such that the instance name is unique in the SmartDesign).

3.30.2 Examples

```
sd_instantiate_macro -sd_name {TOP} -macro_name {MX2} -instance_name {MX2_0}
```

sd_instantiate_macro -sd_name {TOP} -macro_name {MACC_PA}

3.30.2.1 See Also

Tcl Command Documentation Conventions

3.31 sd_invert_pins

```
sd_invert_pins \
-sd_name smartdesign_component_name \
-pin_names port_or_pin_names
```

Tcl command; inverts one or more top level ports or instance level pins in a SmartDesign.

3.31.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-pin_names` *`port_or_pin_names`*

Specifies the port or pin names to be inverted. It is mandatory. This parameter can take multiple values. This command will fail if the port/pin does not exist.

3.31.2 Examples

```
sd_invert_pins -sd_name {main} -pin_names {A}
sd_invert_pins -sd_name {main} -pin_names {MX2_1:S MX2_1:Y A B}
```

3.31.2.1 See Also

Tcl Command Documentation Conventions

3.32 sd_mark_pins_unused

Tcl command; marks one or more SmartDesign instance level output pins as unused. When an output pin is marked as unused, no Design Rule Check (DRC) warning will be printed for floating output pins while generating the SmartDesign.

```
sd_mark_pins_unused \
-sd_name smartdesign_component_name \
-pin_names port_or_pin_names
```

3.32.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-pin_names port_or_pin_names`

Specifies the names of the instance pins to be marked as unused. It is mandatory.

3.32.2 Examples

```
sd_mark_pins_unused -sd_name {top} -pin_names {PF_CCC_C0_0:PLL_LOCK_0}
```

3.32.3 Notes

This command will not work on multiple pins in this release. Support for multiple pins will be provided in the next Libero release.

3.32.3.1 See Also

Tcl Command Documentation Conventions

3.33 sd_remove_pins_from_group

```
sd_remove_pins_from_group \
-sd_name smartdesign_component_name \
-instance_name instance_name \
-group_name group_name \
-pin_names pin_names
```

Tcl command; removes one or more pins from a pin group on an instance in a SmartDesign.

3.33.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-instance_name instance_name`

Specifies the name of the instance on which the pin group is present. It is mandatory.

`-group_name group_name`

Specifies the name of the pin group from which pins need to be removed. It is mandatory.

`-pin_names pin_names`

Specifies the list of pin names to be removed from the pin group. It is mandatory.

3.33.2 Examples

```
sd_remove_pins_from_group -sd_name {TOP} -instance_name
{COREAXI4INTERCONNECT_C0_0} -group_name {Group} -pin_names {ARESETN ACLK}
```

3.33.2.1 See Also

Tcl Command Documentation Conventions

3.34 sd_rename_instance

```
sd_rename_instance \
-sd_name component_name \
-current_instance_name instance_name \
-new_instance_name new_instance_name
```

Tcl command; renames an instance in a SmartDesign component. This command can be used to rename any type of instances (instances of other SmartDesigns components, core components, HDL modules, HDL+ cores and Microsemi macros) in a SmartDesign.

3.34.1 Arguments

`-sd_name` *`component_name`*

Specifies the name of the SmartDesign component in which the instance name has to be renamed. It is mandatory.

`-current_instance_name` *`instance_name`*

Specifies the name of the instance to be renamed. It is mandatory.

`-new_instance_name` *`new_instance_name`*

Specifies the new instance name. It is mandatory.

3.34.2 Examples

```
sd_rename_instance -sd_name {top} -current_instance_name {DFN1_0} - new_instance_name
{DFN1_new}
```

3.34.3 Notes

This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.34.3.1 See Also

Tcl Command Documentation Conventions

3.35 sd_rename_net

```
sd_rename_net \
-sd_name smartdesign_component_name \
-current_net_name current_net_name \-new_net_name_new_net_name
```

Tcl command; renames a net in a SmartDesign component.

3.35.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-current_net_name` *`current_net_name`*

Specifies the name of the net to be renamed in the SmartDesign. It is mandatory.

`-new_net_name` *`new_net_name`*

Specifies the new name of the net in the SmartDesign. It is mandatory.

3.35.2 Examples

```
sd_rename_net -sd_name {top} -current_net_name {clk_net} -new_net_name {clk_rclk_wclk}
sd_rename_net -sd_name {PCIe_EP_Demo} -current_net_name {USER_RESETN} -new_net_name
{reset_input}
```

3.35.3 Notes

This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.35.3.1 See Also

Tcl Command Documentation Conventions

3.36 sd_rename_pin_group

```
sd_rename_pin_group \
-sd_name smartdesign_component_name \
-instance_name instance_name \
-current_group_name current_pin_group_name \
-new_pin_group_name new_pin_group_name
```

Tcl command; renames a pin group on an instance in a SmartDesign component.

3.36.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-instance_name instance_name`

Specifies the name of the instance on which the pin group is present. It is mandatory.

`-current_group_name current_pin_group_name`

Specifies the name of the pin group to be renamed. It is mandatory.

`-new_group_name new_group_name`

Specifies the new name of the pin group. It is mandatory.

3.36.2 Examples

```
sd_rename_pin_group -sd_name {TOP} -instance_name
{COREAXI4INTERCONNECT_C0_0} -current_group_name {Group} -new_group_name
{MyNewGroup}
```

3.36.2.1 See Also

Tcl Command Documentation Conventions

3.37 sd_rename_port

```
sd_rename_port \
-sd_name smartdesign_component_name \
-current_port_name port_name \
-new_port_name new_port_name
```

Tcl command; renames a SmartDesign port.

3.37.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-current_port_name port_name`

Specifies the name of the port to be renamed in the SmartDesign component. It is mandatory. Note that only port names can be renamed, and not port types (scalar ports cannot be renamed as bus ports and vice versa).

`-new_port_name new_port_name`

Specifies the new name of the specified port. It is mandatory.

3.37.2 Examples

```
sd_rename_port -sd_name {top} -library {work} -current_port_name {c1} -new_port_name
{c2}
```

3.37.3 Notes

This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.37.3.1 See Also

Tcl Command Documentation Conventions

3.38 sd_save_core_instance_config

```
sd_save_core_instance_config \
-sd_name smartdesign_component_name \
-instance_name core_instance_name
```

Tcl command; this command is used to save the core instance configuration specified using one or more 'sd_configure_core_instance' commands. This command is typically used after configuring a core instance in a SmartDesign, to save that core instance's configuration.

3.38.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-instance_name` *`instance_name`*

Specifies the name of the core instance in the SmartDesign for which the configuration must be saved. It is mandatory.

3.38.2 Examples

```
sd_save_core_instance_config -sd_name {SD1} -instance_name {COREFIFO_0}
```

3.38.2.1 See Also

Tcl Command Documentation Conventions

3.39 sd_show_bif_pins

```
sd_show_bif_pins \
-sd_name smartdesign_component_name \
-bif_pin_name name_of_the_bif_pin_or_port \
-pin_names pins_or_ports_to_be_exposed
```

Tcl command; exposes one or more internal scalar or bus pins/ports of a Bus Interface pin/port. A Bus Interface pin/port is usually a group of normal scalar or bus pins/ports grouped together and used to connect instances that have similar interfaces. The internal pins/ports underneath the Bus Interface pin/port may have to be exposed in some cases to connect to some logic in the design.

3.39.1 Arguments

`-sd_name` *`smartdesign_component_name`*

Specifies the name of the SmartDesign component. It is mandatory.

`-bif_pin_name` *`name_of_the_bif_pin_or_port`*

Specifies the name of the Bus Interface pin/port for which the internal pins/ports need to be exposed. It is mandatory.

`-pin_names pins_or_ports_to_be_exposed`

Specifies the names of the Bus Interface internal pins/ports to be exposed. It is mandatory.

3.39.2 Examples

```
sd_show_bif_pins -sd_name {TOP} -bif_pin_name {COREAXI4INTERCONNECT_C0_0:AXI4mmaster0}
- pin_names {COREAXI4INTERCONNECT_C0_0:MASTER0_AWADDR}

sd_show_bif_pins -sd_name {SD1} -bif_pin_name {CLKS_FROM_TXPLL_0} -pin_names

{TX_PLL_LOCK_0}
```

3.39.3 Notes

This command will not expose multiple pins/ports in this release. Support to expose multiple scalar or bus pins/ports will be provided in the next Libero release.

3.39.3.1 See Also

Tcl Command Documentation Conventions

3.40 sd_update_instance

```
sd_update_instance \

-sd_name smartdesign_component_name \

-instance_name instance_name
```

Tcl command; updates an instance in a SmartDesign with its latest definition. This command is useful when the interface (port-list) of the component/module instantiated in a SmartDesign has changed. This command can be used to update any type of instance such as instances of other SmartDesign components, core components, HDL modules and HDL+ cores in a SmartDesign.

3.40.1 Arguments

`-sd_name smartdesign_component_name`

Specifies the name of the SmartDesign component. It is mandatory.

`-instance_name instance_name`

Specifies the name of the instance to be updated. It is mandatory.

3.40.2 Examples

```
sd_update_instance -sd_name {top} -instance_name {CORESMIP_C0_0}
```

3.40.3 Notes

This command is not required to build a SmartDesign component. This command maps to an interactive user action in the SmartDesign Canvas and will not be present in the exported SmartDesign component Tcl description.

3.40.3.1 See Also

Tcl Command Documentation Conventions

4. HDL Tcl Commands

4.1 create_hdl_core

Description
This Tcl command uses to create a core component from an HDL core. If specified, incorrect module name command will fail.

```
create_hdl_core -file "file name" -module "module name" \
[-library "library name"] [-package "package name"]
```

Arguments

Parameter	Type	Description
file	string	Specify the file absolute path of the module from which you create a core component. This is a mandatory argument.
module	string	Specify the module name for which you want to create a core component. This is a mandatory argument.
library	string	Specify the library name from which you want to create a HDL core. This is an optional argument.
package	string	Specify the package name from which you want to create a core component. This is an optional argument.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'file' is missing.
None	Parameter 'param_name' is not defined. Valid command formatting is'create_hdl_core -file "file name" -module "module name" [-library "library"] [-package "package"]'.
None	Required parameter 'module' is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example creates core component for "test_hdl_core" module from the "hdl_core.v" HDL:

```
create_hdl_core -file {./HDL_CORE_TEST/hdl/hdl_core.v} \
-module {test_hdl_core}
```

4.2 remove_hdl_core

Description
This Tcl command removes and HDL core component from the current project. The command will fail if the module name is not specified or is incorrect.

```
remove_hdl_core -hdl_core_name { hdl_core_name }
```

Arguments

Parameter	Type	Description
hdl_core_name	string	Specify the module name from which you want to delete a core component. This is a mandatory argument.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'hdl_core_name' is missing.
None	Parameter 'param_name' is not defined. Valid command formatting is ' remove_hdl_core -hdl_core_name "hdl core name"'.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example removes 'test_hdl_core' HDL core component:

```
remove_hdl_core -hdl_core_name {test_hdl_core}
```

4.3 hdl_core_add_bif

Description
This Tcl command adds a bus interface to an HDL core.

The command will fail if the module name or Bus Interface Definition are not specified or are incorrect.

```
hdl_core_add_bif \
-hdl_core_name { hdl_core_name } \
-bif_definition { Name:Vendor:Library:Role } \
-bif_name { bus_interface_name } \
[-signal_map { signal_map }]
```

Arguments

Parameter	Type	Description
hdl_core_name	string	Specify the HDL core name to which the bus interface needs to be added. This is a mandatory argument.
bif_definition	string	Specify the Bus Interface Definition Name, Vendor, Library and Bus Role of the core in the format {N:V:L:R}. This is a mandatory argument.
bif_name	string	Specify the bus interface port name being added to the HDL core. This is a mandatory argument.
signal_map	list of strings	This argument is used to specify the signal map of the bus interface. This is an optional argument.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'hdl_core_name' is missing.
None	The bus interface 'BIF_name' has already been defined.
None	Parameter 'signal_map' has illegal value.
None	Parameter 'param_name' is not defined. Valid command formatting is 'hdl_core_add_bif -hdl_core_name "hdl_core_name" -bif_definition "BIF definition" -bif_name "BIF name" [-signal_map "[signal map]+"]'

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
IGLOO2	v12.4+

Example
The following command adds 'BIF_1' bus interface to 'test_hdl_core' HDL core with the spceified bus interface definition.

```
hdl_core_add_bif -hdl_core_name {test_hdl_core} \
-bif_definition {AHB:AMBA:AMBA2:master} -bif_name {BIF_1}
```

See Also
- hdl_core_remove_bif
- hdl_core_rename_bif

4.4 hdl_core_assign_bif_signal

Description
This Tcl command maps a bus interface signal definition name to an HDL core module port name.

This Tcl command maps bus interface signal definition name to an HDL core module port name. The command will fail if the HDL core name is not specified or is incorrect.

```
hdl_core_assign_bif_signal
-hdl_core_name { hdl_core_name } \
-bif_name { bus_interface_name } \
-bif_signal_name { bif_signal_name } \
-core_signal_name { core_signal_name }
```

Arguments

Parameter	Type	Description
hdl_core_name	string	Specify the HDL core name to which the bus interface signal needs to be added. This is a mandatory argument.
bif_name	string	Specify the bus interface name for which you want to map a core signal. This is a mandatory argument.
bif_signal_name	string	Specify the bus interface signal name that you want to map with the core signal name. This is a mandatory argument.
core_signal_name	string	Specify the core signal name for which you want to map the bus interface signal name. This is a mandatory argument.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'hdl_core_name' is missing.

.....continued

Error Code	Description
None	Required parameter 'bif_name' is missing.
None	Parameter 'bus_interface' is missing or has invalid value.
None	Required parameter 'bif_signal_name' is missing.
None	Required parameter 'core_signal_name' is missing.
None	The bus interface 'BIF_name' has not been defined.
None	The bus interface signal 'SIGNAL NAME' has not been defined in the bus definition of the bus interface 'BIF name'.
None	Parameter 'param_name' is not defined. Valid command formatting is 'hdl_core_assign_bif_signal -hdl_core_name "hdl_core_name" -bif_name "BIF name" -bif_signal_name "BIf signal name" -core_signal_name "core signal name"'

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command adds 'HWRITE' bus interface signal to 'test_hdl_core' HDL core, maps with the 'myHRESULT' core signal name:

```
hdl_core_assign_bif_signal -hdl_core_name {test_hdl_core} \
-bif_name {BIF_1} -bif_signal_name {HWRITE} -core_signal_name {myHRESULT}
```

See Also

- hdl_core_unassign_bif_signal

4.5 hdl_core_delete_parameters

Description
This Tcl command deletes parameters from a HDL core definition. After this command usage It will be impossible to configure parameters of HDL core.

The command will fail if the module name or parameter list are not specified or are incorrect.

```
hdl_core_delete_parameters -hdl_core_name { module_name } \
-parameters { parameter_list }
```

Arguments

Parameter	Type	Description
hdl_core_name	string	Specify the HDL core name from which you want to delete parameters. This is a mandatory argument.
parameters	list of strings	Specify the list of parameters from a HDL core. This is typically done to remove parameters from the list of parameters that was automatically extracted using the "hdl_core_extract_ports_and_params" command. This is a mandatory argument.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'hdl_core_name' is missing.
None	Required parameter 'parameters' is missing.
None	Parameter 'param_name' is not defined. Valid command formatting is 'hdl_core_delete_parameters -hdl_core_name "hdl core name" -parameters "[parameter list]+"'

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example deletes "WIDTH" parameters from a "test_hdl_core" HDL core definition:

```
hdl_core_delete_parameters -hdl_core_name {test_hdl_core} \
-parameters {WIDTH}
```

See Also

- hdl_core_extract_ports_and_parameter

4.6 hdl_core_extract_ports_and_parameters

Description
This Tcl command automatically extracts ports and generic parameters from an HDL core module description.

The command will fail if the module name is not specified or is incorrect.

```
hdl_core_extract_ports_and_parameters \
-hdl_core_name { hdl_core_name }
```

Arguments

Parameter	Type	Description
hdl_core_name	string	Specifies the HDL core name from which you want to extract signal names and generic parameters. This is a mandatory argument.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'hdl_core_name' is missing.
None	Parameter 'param_name' is not defined. Valid command formatting is 'hdl_core_extract_ports_and_parameters -hdl_core_name "hdl_core_name"'

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example automatically extracts ports and parameters from an "test_hdl_core" HDL core module description:

```
hdl_core_extract_ports_and_params -hdl_core_name {test_hdl_core}
```

See Also

- hdl_core_delete_parameters

4.7 hdl_core_remove_bif

Description
This Tcl command removes existing bus interface from an HDL core. The command will fail if the module name is not specified or is incorrect.

```
hdl_core_remove_bif \
-hdl_core_name { hdl_core_name } \
-bif_name { bus_interface_name }
```

Arguments

Parameter	Type	Description
hdl_core_name	string	Specify the HDL core name from which the bus interface needs to be removed. This is a mandatory argument.
bif_name	string	Specify the bus interface name that needs to be removed from the HDL core. This is a mandatory argument.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'hdl_core_name' is missing.
None	Parameter 'bus_interface' is missing or has invalid value.
None	The bus interface 'BIF_name' has not been defined.
None	Parameter 'param_name' is not defined. Valid command formatting is 'hdl_core_remove_bif -hdl_core_name "hdl_core_name" -bif_name "BIF name"'

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command removes "BIF_1" bus interface from the "mod1" HDL core with the spceified bus interface name:

```
hdl_core_remove_bif -hdl_core_name {mod1} -bif_name {BIF_1}
```

See Also

- hdl_core_add_bif:q
- hdl_core_rename_bif

4.8 hdl_core_rename_bif

Description

This Tcl command renames existing bus interface port of a HDL core. The command will fail if the module name is not specified or is incorrect.

```
hdl_core_rename_bif
-hdl_core_name { hdl_core_name } \
-current_bif_name { current_bus_interface_name } \
-new_bif_name { new_bus_interface_name }
```

Arguments

Parameter	Type	Description
hdl_core_name	string	Specify the HDL core name for which the bus interface needs to be renamed. This is a mandatory argument.
current_bif_name	string	Specify the bus old bus interface name that needs to be renamed for the HDL core. This is a mandatory argument.
new_bif_name	string	Specify the new bus interface name that needs to be updated for the HDL core. This is a mandatory argument.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'hdl_core_name' is missing.
None	Parameter 'current_bus_interface' is missing or has invalid value.
None	Required parameter 'new_bif_name' is missing
None	The bus interface 'BIF_name' has not been defined.
None	You must specify at least one parameter among: 'name, connection_required, interface_rendering, description, export'.
None	Parameter 'param_name' is not defined. Valid command formatting is 'hdl_core_rename_bif -hdl_core_name "hdl core name" -current_bif_name "Current BusInterface name" -new_bif_name "New BusInterface name"'

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command renames the existing 'BIF_1' bus interface port name from the 'dffp' HDL core with the spceified new 'BIF_2' bus interface name:

```
hdl_core_rename_bif -hdl_core_name {test_hdl_plus} \
-current_bif_name {BIF_1} -new_bif_name {BIF_2}
```

See Also
- hdl_core_add_bif
- hdl_core_remove_bif

4.9 hdl_core_unassign_bif_signal

Description
This Tcl command unmaps an existing bus interface signal from a bus interface. The command will fail if the HDL core name is not specified or is incorrect.

```
hdl_core_unassign_bif_signal
-hdl_core_name { hdl_core_name } \
-bif_name { bus_interface_name } \
-bif_signal_name { bif_signal_name }
```

Arguments

Parameter	Type	Description
hdl_core_name	string	Specify the HDL core name from which the bus interface signal needs to be deleted. This is a mandatory argument.
bif_name	string	Specify the bus interface name for which you want to unassign a core signal. This is a mandatory argument.
bif_signal_name	string	Specify the bus interface signal name for which you want to unassign a core signal. This argument is mandatory.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'hdl_core_name' is missing.

.....continued

Error Code	Description
None	Required parameter 'bif_name' is missing.
None	The bus interface 'BIF_name' has not been defined.
None	Required parameter 'bif_signal_name' is missing.
None	Parameter 'bus_interface' is missing or has invalid value.
None	The bus interface signal 'SIGNAL NAME' has not been defined in the bus interface 'BIF name'.
None	Parameter 'param_name' is not defined. Valid command formatting is 'hdl_core_unassign_bif_signal-hdl_core_name "hdl_core_name" -bif_name "BIF name" -bif_signal_name "BIf signal name"'

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command unmaps/unassigns an existing 'PENABLE' bus interface signal from a 'test_hdl_plus' HDL core for 'BIF_2' bus interface:

```
hdl_core_unassign_bif_signal -hdl_core_name {test_hdl_plus} \
-bif_name {BIF_2} -bif_signal_name {PENABLE}
```

See Also

- hdl_core_assign_bif_signal

5. Command Tools

5.1 CONFIGURE_ACTIONS_PROCEDURES

CONFIGURE_ACTIONS_PROCEDURES is a command tool used in configure_tool. Configure_tool -name

```
configure_tool -name {CONFIGURE_ACTIONS_PROCEDURES}
-params \
{-prog_optional_procedures:action1|procedure1|procedure2;action2|procedure1|
procedure2|procedure3; \
-params \
{-skip_recommended_procedures:action1|procedure1|procedure2;action2|procedure1|
procedure2|procedure3;
```

{CONFIGURE_ACTIONS_PROCEDURES} is used to configure actions and procedures.

For more information about programming actions and supported procedures, see Configure Actions and Procedures.

5.1.1 Supported Families

SmartFusion2 IGLOO2

5.1.2 Example

```
configure_tool -name {CONFIGURE_ACTIONS_PROCEDURES}
-params \
{prog_optional_procedures:PROGRAM|DO_VERIFY;}\ params\
{skip_recommended_procedures:VERIFY_DIGEST|DO_ENABLE_FABRIC|DO_ENABLE_ENVM;}
```

5.1.3 Return

Returns 0 on success and 1 on failure.

5.2 CONFIGURE_CHAIN

CONFIGURE_CHAIN is a command tool used in run_tool. The command run_tool -name {CONFIGURE_CHAIN} takes a script file that contains specific Tcl commands and passes them to FlashPro Express for execution.

```
run_tool -name {CONFIGURE_CHAIN} -script {fpro_cmds.tcl}
```

fpro_cmds.tcl is a Tcl script that contains specific Tcl commands to configure JTAG chain. For details on JTAG chain programming Tcl commands, refer to the Tcl commands section in the Libero SoC Online Help.

Do not include any project-management commands such as open_project, save_project, or close_project in this *fpro_cmds.tcl* script file. The run_tool –name {CONFIGURE_CHAIN} command generates these project-management commands for you.

Note: For a new Libero project without a JTAG chain, executing this command causes Libero to first add the existing design device to the JTAG chain and then execute the commands from the script. If, for example, the script *fpro_cmds.tcl* contains commands to add four devices, executing the command run_tool –name

{CONFIGURE_CHAIN} -script {*fpro_cmds.tcl*} will create a JTAG chain of the Libero design device and the four devices. For existing Libero projects that already have a JTAG chain, the command is executed on the existing JTAG chain.

5.2.1 Supported Families

SmartFusion2 IGLOO2 RTG4

5.2.2 Example

```
run_tool -name {CONFIGURE_CHAIN} -script {d:/fpro_cmds.tcl}
```

#Example fpro_cmds.tcl command file for the –script parameter

```
add_actel_device \
-file {./sd_prj/sp_g3/designer/impl1/sd1.stp} \
-name {dev1}
enable_device -name {M2S050TS_5} -enable 0 add_non_actel_device \
-ir 2 \
-tck 1.00 \
-name {Non-Microsemi Device} add_non_actel_device \
-ir 2 \
-tck 1.00 \
-name {Non-Microsemi Device (2)} remove_device -name {Non-Microsemi Device}
set_device_to_highz -name {M2S050TS_5} -highz 1 add_actel_device \
-device {M2S050TS} \
-name {M2S050TS (3)}
select_libero_design_device -name {M2S050TS (3)}
```

5.2.3 Return

Returns 0 on success and 1 on failure.

5.3 CONFIGURE_PROG_OPTIONS

CONFIGURE_PROG_OPTIONS is a command tool used in configure_tool. Configure_tool -name

configure_tool -name {CONFIGURE_PROG_OPTIONS}

-params {design_version:<value>}

-params {silicon_signature:<value>}

-params {enable_auto_update:*true* | *false*}

-params {enable_prog_recovery:*true* | *false*}

-params {spi_clk_freq:<*value*>}

-params {spi_data_transfer_mode: <*value*>}

{CONFIGURE_PROG_OPTIONS} sets the programming options.

The following table lists the parameter names and values.

5.3.1 configure_tool –name {CONFIGURE_PROG_OPTIONS} parameter:value pair

Name	Value	Description
design_version	Integer {0 through 65535}	Sets the design version. It must be greater than the Back level version in SPM Update Policy.

Name	Value	Description
enable_auto_update	boolean {true | false}	Enables auto update when set to true and disables it when set to false

enable_prog_recovery	boolean {true | false}	Enables programming recovery when set to true and disables it when set to false
silicon_signature	Hex {<max length 8 Hex characters>}	32-bit (8 hex characters) silicon signature to be programmed into the device. This field can be read from the device using the JTAG USERCODE instruction.
spi_clock_freq	Sets SPI clock frequency from a list of possible values {1.00 | 2.08 |3.13|4.16|5.00| 6.25| 8.30|12.50|25.00}	
spi_data_transfer_mod e	{100 | 101|111|110}	SPI data transfer mode sets the values for SPS, SPO and SPH in the UI. SPS has a fixed value of 1 and cannot be changed. The user can change the value of only SPO and SPH to 0 or 1.

5.3.2 Supported Families

SmartFusion2 IGLOO2

5.3.3 Example

```
configure_tool -name {CONFIGURE_PROG_OPTIONS}\
-params {design_version:255}
-params {enable_auto_update:true}
-params {enable_prog_recovery:true}
-params {silicon_signature:abcdef}
-params {spi_clk_freq:25.00}
-params {spi_data_transfer_mode:100}
```

5.3.4 Return

Returns 0 on success and 1 on failure.

5.3.4.1 See Also

Configure Programming Options (SmartFusion2 and IGLOO2)

5.4 CONFIGURE_PROG_OPTIONS_RTG4 (RTG4 only)

CONFIGURE_PROG_OPTIONS_RTG4 is a command tool used in configure_tool. Configure_tool -name

```
configure_tool -name {CONFIGURE_PROG_OPTIONS_RTG4}
-params {design_version:1}
-params {silicon_signature: abcd}
-params {disable_digest_check:true}
-params {disable_fabric_erase_write_verify:true}
-params {disable jtag:true}
```

{CONFIGURE_PROG_OPTIONS_RTG4} sets the programming options for RTG4 devices.

-params {disable_probe_read_write:false}
-params {disable_spi:false}
-params {one_time_programmable:false}

```
-params {system_controller_supsend_mode:false}
```

The following table lists the parameter names and values.

5.4.1 configure_tool –name {CONFIGURE_PROG_OPTIONS_RTG4} parameter:value pair

Name	Value	Description
design_version	Integer {0 through 65535}	Sets the design version. It must be greater than the Back level version in SPM Update Policy.
silicon_signature	Hex {<max length 8 Hex characters>}	32-bit (8 hex characters) silicon signature to be programmed into the device. This field can be read from the device using the JTAG USERCODE instruction.
disable_digest_check	boolean {true | false}	
disable_fabric_erase_write_verify	boolean {true | false}	
disable_jtag	boolean {true | false}	
disable_probe_read_write	boolean {true | false}	
disable_spi	boolean {true | false}	
one_time_programmabl e	boolean {true | false}	
system_controller_suspsend_mode	boolean {true | false}	

5.4.2 Supported Families
RTG4

5.4.3 Example
```
configure_tool -name {CONFIGURE_PROG_OPTIONS_RTG4}\
-params {design_version:255}\
-params {silicon_signature: abcd}\
-params {disable_digest_check:true}\
-params {disable_fabric_erase_write_verify:true}\
-params {disable_jtag:true}\
-params {disable_probe_read_write:false}\
-params {disable_spi:false}\
-params {one_time_programmable:false}\
-params {system_controller_supsend_mode:false}
```

5.4.4 Return
Returns 0 on success and 1 on failure.

5.4.5 See Also
Configure Programming Options (RTG4 Only)

5.5 FLASH_FREEZE

FLASH_FREEZE is a command tool used in configure_tool. You use the configure_tool -name {FLASH_FREEZE} command to specify:

- The state of the uRAM and LSRAM when the FPGA fabric is in the Flash Freeze state.

```
configure_tool -name {FLASH_FREEZE}
-params {name:value}
-params {name:value}
```

- The MSS clock source when the FPGA fabric is in the Flash Freezestate.

5.5.1 configure_tool –name {FLASH_FREEZE} parameter:value pair

Params_name	<Params_value>	Description
FF_RAM_STATE	Enum {SUSPEND | SLEEP }	Specifies the uRAM and LSRAM state during Flash Freeze. Default is SUSPEND.
FF_MSS_CLOCK	Enum {RCOSC_1MHZ | RCOSC_50MHZ}	Specifies the MSS Clock Source during Flash Freeze. Default is RCOSC_1MHZ.

5.5.2 Supported Families

SmartFusion2, IGLOO2, RTG4

5.5.3 Example

```
configure_tool -name {FLASH_FREEZE}\
-params {FF_RAM_STATE:SUSPEND}\
-params {FF_MSS_CLOCK:RCOSC_1MHZ}
```

5.5.4 Return

Returns 0 on success and 1 on failure.

5.6 EXPORTNETLIST

EXPORTNETLIST is a command tool used in the run_tool command. This command exports a .v/.vhd netlist file to the active synthesis implementation folder.

```
run_tool -name {EXPORTNETLIST}
```

5.6.1 Supported Families

SmartFusion2 IGLOO2

RTG4

5.6.2 Example

```
run_tool -name {EXPORTNETLIST}
```

5.6.3 Return

Returns 0 on success and 1 on failure.

5.7 EXPORTSDF

EXPORTSDF is a command tool used in the run_tool command. This command exports the back annotated files to the designer/impl1 folder.

```
run_tool -name {EXPORTSDF}
```

5.7.1 Supported Families

SmartFusion2, IGLOO2, RTG4

5.7.2 Example

```
run_tool -name {EXPORTSDF}
```

5.7.3 Return

Returns 0 on success and 1 on failure.

5.8 GENERATEPROGRAMMINGDATA

GENERATEPROGRAMMINGDATA is the name of a command tool used in the run_tool command. The run_tool

-name {GENERATEPROGRAMMINGDATA} Tcl command generates the files needed for generating programming bitstream files.

```
run_tool -name {GENERATEPROGRAMMINGDATA}
```

This command takes no parameters.

5.8.1 Return

Returns 0 on success and 1 on failure.

5.8.2 Supported Families

SmartFusion2 IGLOO2 RTG4

5.9 GENERATEPROGRAMMINGFILE

GENERATEPROGRAMMINGFILE is a command tool used in the configure_tool command and the run_tool command. The configure_tool -name {GENERATEPROGRAMMINGFILE} Tcl command configures tool options. The run_tool -name {GENERATEPROGRAMMINGFILE} Tcl command generates the Bitstream used for programming.

```
run_tool -name {GENERATEPROGRAMMINGFILE}
```

This command takes no parameters.

5.9.1 Supported Families

SmartFusion2, IGLOO2, RTG4

5.9.2 Return

Returns 0 on success and 1 on failure.

5.10 INIT_LOCK

INIT_LOCK is a TCL equivalent command name for the Configure Register Lock Bits tool. It is passed as a parameter to the configure_tool command.

```
configure_tool -name {INIT_LOCK} -params {INIT_LOCK_FILE: \
<full _or_relative_path_to_config_lock_bit_file>}
```

The INIT_LOCK command imports a Lock Bit Configuration File (*.txt) and configures the Register Lock Bits of FDDR, MSS, and SERDES blocks for SmartFusion2 and IGLOO2 devices so that they cannot be overwritten by Fabric or MSS Masters that have write access to these registers. This command takes only one parameter, INIT_LOCK_FILE, which has the configuration file's full path or relative path as its value.

5.10.1 Supported Families

SmartFusion2 and IGLOO2

5.10.2 Example

```
configure_tool -name {INIT_LOCK}\
-params {INIT_LOCK_FILE:D:/designs/g4_fclk_mddr_clk/sb_init_config_lock_bits_src.txt}
# full path
configure_tool -name {INIT_LOCK}\
-params {INIT_LOCK_FILE: ../ sb_init_config_lock_bits_src.txt} # relative path from
project folder
```

5.10.3 Return

```
configure_tool -name { INIT_LOCK } -params {INIT_LOCK_FILE: \
<full _or_relative_path_to_config_lock_bit_file>}
```

Returns 0 on success and 1 on failure.

5.11 PROGRAMDEVICE

PROGRAMDEVICE is a command tool used in configure_tool and run_tool. Configure_tool allows you to configure the tool's parameters and values prior to executing the tool. Run_tool executes the tool with the configured parameters.

To program the design in Libero SoC, you must first configure the PROGRAMDEVICE tool with configure_tool command and then execute the PROGRAMDEVICE command with the run_tool command.

```
configure_tool -name {PROGRAMDEVICE}
-params {prog_action:params_value}
run_tool -name {PROGRAMDEVICE}
```

Use the commands to configure your programming action and the programming procedures associated with the program action.

5.11.1 configure_tool –name {PROGRAMDEVICE} parameter:value pair

Name	Value	Description

prog_action	String { PROGRAM | VERIFY | ERASE | DEVICE_INFO | READ_IDCODE | ENC_DATA_AUTHENTICATION | VERIFY_DIGEST}	PROGRAM – Programs all selected family features: FPGA Array, targeted eNVM clients and security settings. VERIFY – Verifies all selected family features: FPGA Array, targeted eNVM clients and security settings. ERASE – Erases the selected family features: FPGA Array and security settings. DEVICE_INFO – Displays the IDCODE, the design name, the checksum, and device security settings and programming environment information programmed into the device.READ_IDCODE – Reads the device ID code from the device. ENC_DATA_AUTHENTICATION - Encrypted bitstream authentication data. VERIFY_DIGEST – Calculates the digests for the components included in the bitstream and compares them against the programmed values

5.11.2 run_tool –name {PROGRAMDEVICE} Parameter:value pair

Name	Value	Description
None		

5.11.3 Supported Families

SmartFusion2 IGLOO2

5.11.4 Example

```
configure_tool \
-name {PROGRAMDEVICE} \
-params {prog_action:PROGRAM} \
configure_tool -name {PROGRAMDEVICE} -params {prog_action:DEVICE_INFO} run_tool -name {PROGRAMDEVICE} #Takes no parameters
```

5.11.5 Return

`configure_tool -name {PROGRAMDEVICE}` returns 0 on success and 1 on failure.

`run_tool -name {PROGRAMDEVICE}` returns 0 on success and 1 on failure.

5.12 PLACEROUTE

To place and route a design in Libero SoC, you must first configure the PLACEROUTE tool with the configure_tool command and then execute the PLACEROUTE tool with the run_tool command.

5.12.1 configure_tool

`configure_tool -name {PLACEROUTE} [-params {[name:value ]+}]+`

5.12.2 Parameters

Name	Value	Description

TDPR	true | false | 1 | 0	Set to true or 1 to enable Timing-Driven Place and Route. Default is 1.
PDPR	true | false | 1 | 0	Set to true or 1 to enable Power-Driven Place and Route. Default is false or 0.
IOREG_COMBINING	true | false | 1 | 0	Set to true or 1 to enable I/O Register Combining. Default is true or 1.
EFFORT_LEVEL	true | false | 1 | 0	Set to true or 1 to enable High Effort Layout to optimize design performance. Default is false or 0.
INCRPLACEANDROUTE	true | false | 1 | 0	Set to true or 1 to use previous placement data as the initial placement for the next run. Default is false or 0.
REPAIR_MIN_DELAY	true | false | 1 | 0	Set to 1 to enable Repair Minimum Delay violations for the router when TDPR option is set to true or 1. Default is false.
NUM_MULTI_PASSES	1-25	Specifies the number of passes to run. The default is 5. Maximum is 25.
START_SEED_INDEX	1-100	Indicates the random seed index which is the starting point for the passes. Its value should range from 1 to 100. If not specified, the default behavior is to continue from the last seed index which was used.
MULTI_PASS_LAYOUT	true | false | 1 | 0	Set to true or 1 to enable Multi-Pass Layout Mode for Place and Route. Default is false or 0.
MULTI_PASS_CRITERIA	SLOWEST_CLOCK SPECIFIC_CLOCK VIOLATIONS TOTAL_POWER	Specifies the criteria used to run multi-pass layout: • SLOWEST_CIOCK: Use the slowest clock frequency in the design in a given pass as the performance reference for the layout pass. • SPECIFIC_CLOCK: Use a specific clock frequency as the performance reference for all layout passes. • VIOLATIONS: Use the pass that best meets the slack or timing-violations constraints. This is the default. – TOTAL POWER: Specifies the best pass to be the one that has the lowest

Name	Value	Description
		total power (static + dynamic) out of all layout passes.
SPECIFIC_CLOCK	Clock_Name	Applies only when MULTI_PASS_CRITERIA is set to SPECIFIC_CLOCK. It specifies the name of the clock in the design used for Timing Violation Measurement.

DELAY_ANALYSIS	max | min	Used only when MULTI_PASS_CRITERIA is set to “VIOLATIONS”. Specifies the type of timing violations (slacks) to be examined. The default is 'max'. • max: Use timing violations (slacks) obtained from maximum delay analysis • min: Use timing violations (slacks) obtained from minimum delay analysis.
STOP_ON_FIRST_PASS	true | false | 1 | 0	Applies only when MULTI_PASS_CRITERIA is set to “VIOLATIONS”. It stops performing remaining passes if all timing constraints have been met (when there are no negative slacks reported in the timing violations report). Note: The type of timing violations (slacks) used is determined by the 'DELAY_ANALYSIS ' parameter.
SLACK_CRITERIA	WORST_SLACK TOTAL_NEGATIVE_SLA CK	Applies only when MULTI_PASS_CRITERIA is set to VIOLATIONS. Specifies how to evaluate the timing violations (slacks). The default is WORST_SLACK. • WORST_SLACK: The largest amount of negative slack (or least amount of positive slack if all constraints are met) for each pass is identified and then the largest value out of all passes will determine the best pass. This is the default. • TOTAL_NEGATIVE_SLACK: The sum of negative slacks from the first 100 paths for each pass in the Timing Violation report is identified. The largest value out of all passes will determine the best pass. If no negative slacks exist for a pass, then use the worst slack to evaluate that pass. Note: The type of timing violations (slacks) used is determined by the 'DELAY_ANALYSIS' parameter.
RGB_COUNT	1-18	Allows an entity to override the placer's RGB/ RCLK bandwidth constraint. This option is useful for Block Creation.

5.12.3 Return Value

Returns 0 on success and 1 on failure.

5.12.4 run_tool

run_tool -name {PLACEROUTE}

5.12.5 Parameters

None

5.12.6 Return Value

Returns 0 on success and 1 on failure.

5.12.7 Supported Families

SmartFusion2, IGLOO2, RTG4

5.12.8 Example

```
configure_tool -name {PLACEROUTE}\
-params {EFFORT_LEVEL:true}\
-params {INCRPLACEANDROUTE:false}\
-params {IOREG_COMBINING:false}\
-params {MULTI_PASS_CRITERIA:VIOLATIONS}\
-params {MULTI_PASS_LAYOUT:false}\
-params {NUM_MULTI_PASSES:5}\
-params {PDPR:false}\
-params {REPAIR_MIN_DELAY:true}\
-params {SLACK_CRITERIA:WORST_SLACK}\
-params {SPECIFIC_CLOCK:}\
-params {START_SEED_INDEX:1}\
-params {STOP_ON_FIRST_PASS:false}\
-params {TDPR:true}\
-params {USE_RAM_MATH_INTERFACE_LOGIC:false} run_tool –name{PLACEROUTE}
```

5.13 PROGRAM_OPTIONS (SmartFusion2 and IGLOO2)

```
configure_tool -name {PROGRAM_OPTIONS}
-params {name:value}
[-params {name:value}]
```

PROGRAM_OPTIONS is a command tool used in configure_tool. Configure_tool allows you to configure the tool's parameters and values. This Tcl command is the Tcl equivalent of the Configure Bitstream options in the Design Flow window.

The following table lists the parameter names and values.

5.13.1 configure_tool –name {PROGRAM_OPTIONS} parameter:value pair

Name	Value	Description
program_envm	Boolean {true | false | 1 | 0}	Include eNVM component in the programming bitstream ("true" only if eNVM available in the design).

Name	Value	Description
program_fabric	Boolean {true | false | 1 | 0}	Include fabric component in the programming bitstream.
program_mode	String {selected feature}	Only "selected_features" is supported.
program_security	Boolean {true | false | 1 | 0}	Include custom security component in the programming bitstream ("true" only if custom security was defined).

5.13.2 Supported Families

SmartFusion2, IGLOO2

5.13.3 Example

```
configure_tool -name {PROGRAM_OPTIONS}\
-params {program_envm:false}\
-params {program_fabric:true}\
-params {program_mode:selected_features}\
-params {program_security:true}
```

5.13.4 Return

Returns 0 on success and 1 on failure.

5.14 PROGRAM_RECOVERY

```
configure_tool -name {PROGRAM_RECOVERY}
-params {enable_auto_update:value}
-params {enable_prog_recovery:value}
-params {spi_clk_freq:value}
-params {spi_data_transfer_mode:value}
```

Configures the parameters in the Configure User Programming Data dialog box within Libero.

5.14.1 Arguments

`Enable_auto_update {true|false}`

Enables auto update when set to true and disables it when set to false.

`Enable_prog_recovery {true|false}`

Enables programming recovery when set to true and disables when set to false.

`Spi_clk_freq {1.00|2.08|3.13|4.16|5.00|6.25|8.30|12.50|25.00}`

SPI clock frequency can be set to one of the values specified above.

`Spi_data_transfer_mode {100}`

SPI data transfer mode will set the values for SPS, SPO and SPH in the UI. SPS has a fixed value of 1. The user can change the values of only SPO and SPH to 0 or 1.

5.14.2 Supported Families

SmartFusion2, IGLOO2

5.14.3 Exceptions

None

5.14.4 Example

The following example sets the auto update and programming recovery to be ON. SPI clock frequency is set to 25MHz. SPO and SPH are set to 0.

```
configure_tool -name {PROGRAM_RECOVERY}\
-params {enable_auto_update:true}\
-params {enable_prog_recovery:true}\
-params {spi_clk_freq:25.00}\
-params {spi_data_transfer_mode:100}
```

5.15 PROGRAMMER_INFO

PROGRAMMER_INFO is a command tool used in configure_tool. Configure_tool -name

```
configure_tool -name {PROGRAMMER_INFO}
-params [{name:value}]
```

{PROGRAMMER_INFO} sets the programmer settings, similar to the way FlashPro commands set the programmer settings. This command supports all five programmers: FlashPro3, FlashPro4, FlashPro5, and FlashPro6.

The following tables list the parameter names and values.

5.15.1 configure_tool –name {PROGRAMMER_INFO} Parameter:value(FlashPro6)

Name	Value	Description
flashpro6_force_freq	String {OFF | ON}	For FlashPro6 Programmer only.
flashpro6_ freq	Integer (Hertz)	For FlashPro6 Programmer only.

5.15.2 configure_tool –name {PROGRAMMER_INFO} Parameter:value(FlashPro5)

Name	Value	Description
flashpro5_clk_mode	String {free_running_clk | discrete_clocking}	For FlashPro5 Programmer only.
flashpro5_force_freq	String {OFF | ON}	For FlashPro5 Programmer only.
flashpro5_ freq	Integer (Hertz)	For FlashPro5 Programmer only.
flashpro5_vpump	String {ON | OFF}	For FlashPro5 Programmer only.

5.15.3 configure_tool –name {PROGRAMMER_INFO} Parameter:value(FlashPro4)

Name	Value	Description
flashpro4_clk_mode	String {free_running_clk | discrete_clocking}	For FlashPro4 Programmer only.

Name	Value	Description
flashpro4_force_freq	String {OFF | ON}	For FlashPro4 Programmer only.
flashpro4_ freq	Integer (Hertz)	For FlashPro4 Programmer only.
flashpro4_vpump	String {ON | OFF}	For FlashPro4 Programmer only.

5.15.4 configure_tool –name {PROGRAMMER_INFO} parameter:value (FlashPro3)

Name	Value	Description
flashpro3_clk_mode	String {free_running_clk | discrete_clocking}	For FlashPro3 Programmer only.
flashpro3_force_freq	String {OFF | ON}	For FlashPro3 Programmer only.
flashpro3_ freq	Integer (Hertz)	For FlashPro3 Programmer only.

flashpro3_vpump	String {ON | OFF}	For Flash For FlashPro3 Programmer only.

For a detailed description of the parameters and values, refer to Programmer Settings in the Libero Online Help.

5.15.5 Supported Families

SmartFusion2 IGLOO2 RTG4

5.15.6 Examples

For FlashPro3 programmer

```
configure_tool -name {PROGRAMMER_INFO}\
-params {flashpro3_clk_mode:free_running_clk}\
-params {flashpro3_force_freq:OFF}\
-params {flashpro3_freq:400000}\
-params {flashpro3_vpump:ON}
```

For FlashPro4 programmer

```
configure_tool -name {PROGRAMMER_INFO}\
-params {flashpro4_clk_mode:free_running_clk\}
-params {flashpro4_force_freq:OFF}\
-params {flashpro4_freq:400000}\
-params {flashpro4_vpump:ON}
```

For FlashPro5 programmer

```
configure_tool -name {PROGRAMMER_INFO}\
-params {flashpro5_clk_mode:free_running_clk}\
-params {flashpro5_force_freq:OFF}\
-params {flashpro5_freq:400000}\
-params {flashpro5_vpump:ON}
```

For FlashPro6 programmer

```
configure_tool -name {PROGRAMMER_INFO}\
-params {flashpro6_force_freq:OFF}\
-params {flashpro6_freq:400000}
```

5.15.7 Return

Returns 0 on success and 1 on failure.

5.16 SIM_POSTLAYOUT

SIM_POSTLAYOUT is a command tool used in the run_tool command. The run_tool -name {SIM_POSTLAYOUT} Tcl command runs the post layout simulation on the simulation tool.

```
run_tool -name {SIM_POSTLAYOUT}
```

5.16.1 Supported Families

SmartFusion2, IGLOO2, RTG4

5.16.2 Example

```
run_tool -name {SIM_POSTLAYOUT}
```

5.16.3 Return

5.17 SPM

Returns 0 on success and 1 on failure.

To configure security using Tcl, you must use the configure_tool Tcl command to pass the SPM configuration parameters.

```
configure_tool -name {SPM}
-params {name:value}
[-params {name:value}]+
```

5.17.1 configure_tool –name {SPM} parameter:value pair

Note: true | 1 will select the checkbox in the SPM UI

Name	Type	Value	Description
back_level_bypass	bool	false | true | 1 | 0	If true, will bypass the backlevel protection; Update Policy
back_level_protectio n	bool	false | true | 1 | 0	If true, will set back level protection; Update Policy
back_level_update_ version	Integer	0 - 65535	Set back level version; Update Policy
debug_cortex_m3	bool	false | true | 1 | 0	If true, will disable Cortex M3 debug. This lock bit is protected by DPK; Debug Policy; SmartFusion2 only

Name	Type	Value	Description
debug_digest_reque st	bool	false | true | 1 | 0	If true, disables design digest check request via JTAG and SPI. Use FlashLock/UPK1 to allow digest check; Debug Policy
debug_disable_jtag	bool	false | true | 1 | 0	If true, disables JTAG (1149.1) test instructions (HIGHZ, EXTEST, INTEST, CLAMP, SAMPLE, and PRELOAD). I/Os will be tri-stated when in JTAG programming mode. Use FlashLock/UPK1 to unlock; Debug Policy
debug_passkey	hex	64 hex characters	Value of DPK; Debug Policy
debug_ujtag_access	bool	false | true | 1 | 0	If true, disables access to UJTAG. Use FDPK to unlock; Debug Policy
disable_user_encryp tion_key_1	bool	false | true | 1 | 0	If true, disables UEK1; Key Mode Policy
disable_user_encryp tion_key_2	bool	false | true | 1 | 0	If true, disables UEK2; Key Mode Policy

disable_user_encryption_key_3	bool	false | true | 1 | 0	If true, disables UEK3 Note: UEK3 is only available for M2S060, M2GL060, M2S090, M2GL090, M2S150, and M2GL150 devices. All other devices will set this to false by default. See the SmartFusion2 SoC FPGA and IGLOO2 FPGA Security Best Practices User Guide for details. Key Mode Policy
factory_access	string	Flashlock | permanent | open	Sets Microsemi factory test mode access level Open: All Microsemi factory test mode access without FlashLock/UPK1 FlashLock (default): Microsemi factory test mode is disabled. FlashLock/UPK1 is required to unlock. Permanent: Permanently disable Microsemi factory test mode access
iap_isp_services	bool	false | true | 1 | 0	If true, disables access to IAP/ISP services; Update Policy
security_key_mode	string	Default | custom	Default: Bitstream encrypted with default key. No security lock bits are

Name	Type	Value	Description
			set. Custom: Custom security settings. Allows user encryption keys, security policy settings, and Microsemi factory test mode access level.
smartdebug_access	string	Full | none	Full: SmartDebug has full access to debug features None: Disable read/write access to SmartDebug architecture. DPK is required for read/write access. Debug Policy
update_auto_prog_lock	bool	false | true | 1 | 0	If true, disables Auto Programming; Update Policy
update_envm_protection	bool	Passkey | open	Open: Updates to eNVM are allowed using UEK1 or UEK2; FlashLock/UPK1 is NOT required for updates Passkey: eNVM updates are disabled. Use FlashLock/UPK1 to unlock Write/Verify/Read operations. Update Policy

update_fabric_protection	bool	Passkey | open	Open: Updates to Fabric are allowed using UEK1 or UEK2; FlashLock/UPK1 is NOT required for updates Passkey: Fabric updates are disabled. Use FlashLock/UPK1 to unlock Erase/Write/Verify operations. Update Policy
update_jtag_lock	bool	false | true | 1 | 0	If true, disables access to JTAG programming. Use FlashLock/UPK1 to unlock; Update Policy
update_spi_slave_lock	bool	false | true | 1 | 0	If true, disables access to SPI Slave. Use FlashLock/UPK1 to unlock; Update Policy
use_debug_policy	bool	false | true | 1 | 0	If true, Debug Policy will be used
use_key_mode_policy	bool	false | true | 1 | 0	If true, Key Mode Policy will be used
use_update_policy	bool	false | true | 1 | 0	If true, Update Policy will be used
use_user_key_set_1	bool	false | true | 1 | 0	If true, enables User Key Set 1

Name	Type	Value	Description
use_user_key_set_2	bool	false | true | 1 | 0	If true, enables User Key Set 2
use_user_key_set_3	bool	false | true | 1 | 0	If true, enables User Key Set 3 Note: User Key Set 3 is only available for M2S060, M2GL060, M2S090, M2GL090, M2S150, and M2GL150 devices.
user_encryption_key_1	hex	64 hex characters	Value of UEK1
user_encryption_key_2	hex	64 hex characters	Value of UEK2
user_encryption_key_3	hex	64 hex characters	Value of UEK3 Note: UEK3 is only available for M2S060, M2GL060, M2S090, M2GL090, M2S150, and M2GL150 devices. All other devices will set this to false by default. See the SmartFusion2 SoC FPGA and IGLOO2 FPGA Security Best Practices User Guide for details.
user_passkey_1	hex	64 hex characters	Value of Flashlock/UPK1
user_passkey_2	hex	64 hex characters	Value of UPK2

user_security_policy _protection	string	Flashlock|permane nt	Flashlock: User keys and Security policies will be protected from erase/write by FlashLock/UPK1 Permanent: Permanently protect UEK1, UEK2, Security Policies, and Microsemi factory test mode access level. NOTE: Once programmed, these settings cannot be changed.

5.17.2 Supported Families

SmartFusion2 IGLOO2

5.17.3 Example

```
configure_tool \
-name {SPM} \
-params {back_level_bypass:false} \
-params {back_level_protection:false} \
-params {back_level_update_version:} \
-params {debug_cortex_m3:false} \
-params {debug_digest_request:false} \
-params {debug_disable_jtag:false} \
-params {debug_passkey:} \
-params {debug_ujtag_access:false} \
-params {disable_user_encryption_key_1:false} \
-params {disable_user_encryption_key_2:false} \
-params {disable_user_encryption_key_3:false} \
-params {factory_access:flashlock} \
-params {iap_isp_services:true} \
-params {security_key_mode:custom} \
-params {smartdebug_access:full} \
-params {update_auto_prog_lock:true} \
-params {update_envm_protection:passkey} \
-params {update_fabric_protection:passkey} \
-params {update_jtag_lock:false} \
-params {update_spi_slave_lock:false} \
-params {use_debug_policy:false} \
-params {use_key_mode_policy:false} \
-params {use_update_policy:false} \
-params {use_user_key_set_1:true} \
-params {use_user_key_set_2:false} \
-params {use_user_key_set_3:false} \
-params
```

```
{user_encryption_key_1:9E108123949848EC7453336DFBBC0CAE60C8541C2AFA7010FA209F7396F3EA17}
\
-params {user_encryption_key_2:} \
-params {user_encryption_key_3:} \
-params
{user_passkey_1:B52EED23B1C4C5BAE1384791CE4F7A069D940A6933329A0A9CE5B24E21C13D39} \
-params {user_passkey_2:} \
-params {user_security_policy_protection:flashlock}
```

5.17.4 Return

Returns 0 on success and 1 on failure.

5.18 SYNTHESIZE

SYNTHESIZE is a command tool used in configure_tool and run_tool. Configure_tool is a general-purpose Tcl command that allows you to configure a tool's parameters and values prior to executing the tool. The run_tool Tcl command then executes the specified tool with the configured parameters.

```
[-params {name:value}]
run_tool -name {SYNTHESIZE}
```

To synthesize your design in Libero SoC, you first configure the synthesize tool with the configure_tool command and then execute the command with the run_tool command.

The following tables list the parameter names and values.

5.18.1 configure_tool –name {SYNTHESIZE} parameter:value pair

Name	Value	Description
CLOCK_ASYNC	Integer	Specifies the threshold value for asynchronous pin promotion to a global net. The default is 12.
CLOCK_GLOBAL	Integer	Specifies the threshold value for Clock pin promotion. The default is 2.
CLOCK_DATA	Integer value between 1000 and 200,000.	Specifies the threshold value for data pin promotion. The default is 5000.
RAM_OPTIMIZED_FOR_POWER	Boolean {true | false | 1 | 0}	Set to true or 1 to optimize RAM for Low Power; RAMS are inferred and configured to ensure the lowest power consumption. Set to false or 0 to optimize RAM for High Speed at the expense of more FPGA resources.
RETIMING	Boolean {true | false | 1 | 0}	Set to true or 1 to enable Retiming during synthesis. Set to false or 0 to disable Retiming during synthesis.
SYNPLIFY_OPTIONS	String	Specifies additional synthesis-specific options. Options specified by this parameter override the same options specified in the user Tcl file if there is a conflict.

SYNPLIFY_TCL_FILE	String	Specifies the absolute or relative path name to the user Tcl file containing synthesis-specific options.
BLOCK_MODE	Boolean {true | false | 1 | 0}	Set to true or 1 when you have blocks in your design and you want to enable the Block mode. Set it to false or 0 if you don't have blocks in your design. Default is false or 0.
BLOCK_PLACEMENT _CONFLICTS	String {ERROR|KEEP|LOCK|DISCAR D}	Instructs the COMPILE engine what to do when the software encounters a placement conflict. When set to: ERROR - Compile errors out if any instance from a Designer block becomes unplaced. This is the default. KEEP - If some instances get unplaced for any reason, the non-conflicting elements remaining are preserved but not locked. Therefore, the placer can move them into another location if necessary. LOCK - If some instances get unplaced for any reason, the non-conflicting elements remaining are preserved and locked. DISCARD – Discards any placement from the block, even if there are no conflicts.
BLOCK_ROUTING_C ONFLICTS	String {ERROR|KEEP|LOCK|DISCAR D}	Instructs the COMPILE engine what to do when the software encounters a routing conflict. When set to: ERROR - Compile errors out if any route in any preserved net from a Designer block is deleted. This is the default.

Name	Value	Description
		KEEP – If a route is removed from a net for any reason, the routing for the non-conflicting nets is kept unlocked. The router can re-route these nets. LOCK – If routing is removed from a net for any reason, the routing for the non- conflicting nets is kept as locked, and the router will not change them. DISCARD - Discards any routing from the block, even if there are no conflicts.
PA4_GB_COUNT	Integer	The number of available global nets is reported. Minimum for all dies is "0". Default and Maximum values are die-dependent: 005/010 die: Default = Max = 8 025/050/060/090/150 die: Default=Max=16 RT4G075/RT4G150: Default=24, Max=48. Note: For RTG4, default is 48.
PA4_GB_MAX_RCLK INT_INSERTION	Integer	Specifies the maximum number of global nets that could be demoted to row-globals. Default is 16, Min is 0 and Max is 50.
PA4_GB_MIN_GB_F ANOUT_TO_USE_RC LKINT	Integer	Specifies the Minimum fanout of global nets that could be demoted to row-globals. Default is 300. Min is 25 and Max is 5000.
LANGUAGE_SYSTE M_VLOG	Boolean {true | false}	Set to true if the Verilog files contain System Verilog constructs.

LANGUAGE_VERILOG_2001	Boolean {true | false}	Set to true if Verilog files contain Verilog 2001 constructs.

5.18.2 run_tool –name {SYNTHESIZE}

Supported Families

SmartFusion2 IGLOO2 RTG4

5.18.3 Example

```
configure_tool -name {SYNTHESIZE} -params {BLOCK_MODE:false}\
-params {BLOCK_PLACEMENT_CONFLICTS:ERROR} -params\
{BLOCK_ROUTING_CONFLICTS:ERROR} -params {CLOCK_ASYNC:12}\
-params {CLOCK_DATA:5010} -params {CLOCK_GLOBAL:2} -params\
-params {PA4_GB_MAX_RCLKINT_INSERTION:16} -params\
{PA4_GB_MIN_GB_FANOUT_TO_USE_RCLKINT:299} -params\
{RAM_OPTIMIZED_FOR_POWER:false} -params {RETIMING:false}
-params {SYNPLIFY_OPTIONS:
set_option -run_prop_extract 1;
set_option -maxfan 10000;
set_option -clock_globalthreshold 2;
set_option -async_globalthreshold 12;
set_option -globalthreshold 5000;
set_option -low_power_ram_decomp 0;}\
-params {SYNPLIFY_TCL_FILE:C:/Users/user1/Desktop/tclflow/synthesis/test.tcl} run_tool
-name {SYNTHESIZE} #Takes no parameters
```

5.18.4 Return

```
configure_tool -name {SYNTHESIZE}
```

Returns 0 on success and 1 on failure.

```
run_tool -name {SYNTHESIZE}
```

Returns 0 on success and 1 on failure.

5.19 USER_PROG_DATA (SmartFusion2, IGLOO2)

```
configure_tool -name {USER_PROG_DATA}
-params {name:value}
-params {name:value}
```

USER_PROG_DATA is a command tool used in configure_tool. Configure_tool -name {USER_PROG_DATA} sets the Design Version and Silicon Signature in your device.

The following table lists the parameter names and values.

5.19.1 configure_tool –name {USER_PROG_DATA} parameter:value pair

Name	Value	Description

design_version	Integer (0 through 65535)	Sets the design version. It must be greater than the Back level version in SPM Update Policy.
silicon_signature	Hex {<max length 8 Hex characters>}	32-bit (8 hex characters) silicon signature to be programmed into the device. This field can be read from the device using the JTAG USERCODE instruction.

5.19.2 Supported Families

SmartFusion2, IGLOO2

5.19.3 Example

```
configure_tool -name {USER_PROG_DATA}\
-params {design_version:255}\
-params {silicon_signature:abcdffff}
```

5.19.4 Return

Returns 0 on success and 1 on failure.

5.20 VERIFYPOWER

VERIFYPOWER is a command tool used in run_tool. The command run_tool passes a script file that contains power-specific Tcl commands to the VERIFYPOWER command and executes it.

```
run_tool -name {VERIFYPOWER} -script {power_analysis.tcl}
```

where

<power_analysis.tcl> is a script that contains power-specific Tcl commands. You can include power-specific Tcl commands to generate power reports. See the sample power_analysis Tcl Script below for details.

5.20.1 Return

Returns 0 on success and 1 on failure.

5.20.2 Supported Families

SmartFusion2 IGLOO2 RTG4

5.20.3 Example

```
run_tool -name {VERIFYPOWER} -script {<power_analysis.tcl>}
```

5.20.4 Sample power_analysis Tcl Script <power_analysis.tcl>

The following example changes SmartPower operating condition settings from the default to 40C junction temperature and 1.25V VDD.

It then creates a report called A4P5000_uSRAM_POWER_64X18_power_report.txt.

Change from pre-defined temperature and voltage mode (COM,IND,MIL) to SmartPower custom

```
smartpower_set_temperature_opcond -use "design" smartpower_set_voltage_opcond -voltage
"VDD" -use "design"
```

Set the custom temperature to 40C ambient temperature.

```
smartpower_temperature_opcond_set_design_wide -typical 40 -best 40 -worst 40
```

Set the custom voltage to 1.25V

```
smartpower_voltage_opcond_set_design_wide -voltage "VDD" -typical 1.25 -best 1.25 -
worst 1.25
```

5.21 VERIFYTIMING

VERIFYTIMING is a command tool used in run_tool. Run_tool passes a script file that contains timing- specific Tcl commands to the VERIFYTIMING command and executes it.

```
run_tool -name {VERIFYTIMING} -script {timing.tcl}
```

where

<timing.tcl> is a script that contains SmartTime-specific Tcl commands. You can include SmartTime- specific Tcl commands to create user path sets and to generate timing reports. See sample the Sample SmartTime Tcl Script below for details.

5.21.1 Supported Families

SmartFusion2 IGLOO2 RTG4

5.21.2 Example

```
run_tool -name {VERIFYTIMING} -script {<timing.tcl>}
```

5.21.3 Return

Returns 0 on success and 1 on failure.

Sample SmartTime Tcl Script <*timing.tcl*>

```
# Create user path set -from B_reg
create_set -name from_B_reg \
-source {B_reg*[*]:CLK} \
-sink {*}
# Create user set -from A, B, C
create_set -name from_in_ports \
-source {A B C} \
-sink {*}
# Generate Timing Reports report \
-type timing \
-analysis min \
-format text \
-max_paths 10 \
-print_paths yes \
-max_expanded_paths 10 \
-include_user_sets yes \
min_timing.rpt
# Export SDC
write_sdc -scenario {Primary} exported.sdc
#save the changes
save
```

6. MSS Tcl Commands

6.1 mss_configure_envm

This command is used to specify a `.cfg`file with all clients info in the ENVM core instance of the MSS component.

```
mss_configure_envm \
-component_name {component_name} \
-cfg_file {file_path}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the MSS component.
-cfg_file {file_path}	string	Mandatory. Path of the configuration file(.cfg) used to configure the envm.

Example

```
mss_configure_envm -component_name {test_sb_MSS} -cfg_file{./ENVM.cfg}
```

6.2 mss_configure_instance

Description
This command is used to configure the parameters of a core instance inside the MSS component.

```
mss_configure_instance \
-component_name {component_name} \
-instance_name {instance_name} \
-params{param:value list}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the MSS component.
-instance_name {instance_name}	string	Mandatory. Name of the core instance to be configured inside the MSS component.
-params{param:value list}	string	Mandatory. List of parameters and values to be configured for the selected core instance inside the MSS component.

Example

```
mss_configure_instance -component_name {test_sb_MSS} -instance_name {CC} -params {\
"CACHE_ENABLED:false" \
"CC_CACHE_REGION:128MB_0001" }
```

6.3 mss_disable_instance

Description
This command is used to disable a core instance inside the MSS component.

```
mss_disable_instance \
-component_name {component_name} \
-instance_name {instance_name}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the MSS component.
-instance_name {instance_name}	string	Mandatory. Name of the core instance to be disabled inside the MSS component.

Example

```
mss_disable_instance -component_name {test_sb_MSS} -instance_name {I2C_1}
```

6.4 mss_enable_instance

Description
This command is used to enable a core instance inside the MSS component.

```
mss_enable_instance \
-component_name {component_name} \
-instance_name {instance_name}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the MSS component.
-instance_name {instance_name}	string	Mandatory. Name of the core instance to be enabled inside the MSS component.

Example

```
mss_enable_instance -component_name {test_sb_MSS} -instance_name {MMUART_0}
```

6.5 mss_save_instance_config

Description
This command is used to save the configuration of a core instance inside the MSS component specified using the 'mss_configure_instance' command.

```
mss_save_instance_config \
-component_name {component_name} \
-instance_name {instance_name}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the MSS component.
-instance_name {instance_name}	string	Mandatory. Name of the core instance inside the MSS component.

Example

```
mss_save_instance_config -component_name {test_sb_MSS} -instance_name {CAN}
```

7. SmartTime Tcl Commands

7.1 all_inputs

Description
This Tcl command returns an object representing all input and inout pins in the current design. This command is usually used with a command which puts the same attributes on input ports. If you want only certain ports, use get_ports.

```
all_inputs
```

Arguments

Parameter	Type	Description
None	None	None

Return Type	Description
object	Returns an object representing all input and inout pins in the current design.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions
You can only use this command as part of a `-from`, `-to` argument in the following Tcl commands: `set_min_delay`, `set_max_delay`, `set_multicycle_path`, and `set_false_path`. It cannot be used with `-through` option.

Example
The following example sets a maximum delay by constraining all paths from all_inputs to ck1 clock with a delay less then 2 ns.

```
set_max_delay 2 -from [all_inputs] -to [get_clocks ck1]
```

Related Examples on GitHub
- all_inputs

7.2 all_outputs

Description
This Tcl command returns an object representing all output and inout pins in the current design. This command is usually used with a command which puts the same attributes on output ports. If you want only certain ports, use get_ports command.

```
all_outputs
```

Arguments

Parameter	Type	Description
None	None	None

Return Type	Description
object	Returns an object representing all output and inout pins in the current design.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions
You can only use this command as part of a `-from`, `-to` argument in the following Tcl commands: `set_min_delay`, `set_max_delay`, `set_multicycle_path`, and `set_false_path`. It cannot be used with `-through` option.

Example
The following example sets a maximum delay by constraining all paths from all_inputs to all_outputs with a delay less then 2 ns.

```
set_max_delay 2 -from [all_inputs] -to [all_outputs]
```

Related Examples on GitHub
- all_outputs

7.3 all_registers

Description
This Tcl command returns an object representing register pins or register cells (default) in the current scenario based on the given parameters. If you do not specify an option, this command returns an object representing registers cells.

```
all_registers [-clock clock_name ] [-async_pins] \
[-output_pins] [-data_pins] [-clock_pins]
```

Arguments

Parameter	Type	Description
clock	string	Specifies the name of the clock domain to which the registers belong. If no clock is specified, all registers in the design will be targeted.
async_pins	None	Lists all register pins that are async pins for the specified clock (or all registers asynchronous pins in the design).
output_pins	None	Lists all register pins that are output pins for the specified clock (or all registers output pins in the design).
data_pins	None	Lists all register pins that are data pins for the specified clock (or all registers data pins in the design).
clock_pins	None	Lists all register pins that are clock pins for the specified clock (or all registers clock pins in the design).

Return Type	Description
object	Returns an object representing register pins or cells in the current scenario based on the given parameters.

Error Codes

Error Code	Description
Error: SDC0021	Invalid max delay constraint: the -from value is incorrect.
Error: SDC0023	Invalid max delay constraint: the -to value is incorrect.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions

You can only use this command as part of a `-from`, `-to` argument in the following Tcl commands: `set_min_delay`, `set_max_delay`, `set_multicycle_path`, and `set_false_path`.

Example

The following example sets a maximum delay by constraining all paths from ff_m:CLK or ff_s2:CLK to ff_m:Q pin with a delay less than 2.000 ns.

```
set_max_delay 2.000 -from { ff_m:CLK ff_s2:CLK } \
-to [all_registers -clock_pins -clock {ff_m:Q}]
```

Related Examples on GitHub

- all_registers

7.4 check_constraints

Description
This Tcl command checks all timing constraints in the current scenario for validity. This command performs the same checks as when the constraint is entered through SDC or Tcl.

When a constraint file is checked, the Constraint Checker does the following:

- checks the syntax.
- compares the design objects (pins, cells, nets, ports) in the constraint file versus the design objects in the netlist (RTL or post-layout ADL netlist). Any discrepancy (e.g., constraints on a design object which does not exist in the netlist) are flagged as errors and reported in the *_sdc.log file.

```
check_constraints
```

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example checks timing constraints in the current scenario.

```
check_constraints
```

7.5 clone_scenario

Description
This Tcl command creates a timing scenario with the new_scenario_name, which includes a copy of all constraints in the original scenario. The new scenario is then added to the list of scenarios. You must provide a unique name (that is, it cannot already be used by another timing scenario).
Note: It is recommended to use the organize_tool_files command instead of clone_scenario.

```
clone_scenario original new_scenario_name
```

Arguments

Parameter	Type	Description
original	string	Specifies the name of the source timing scenario to clone (copy). The source must be a valid, existing timing scenario.
new_scenario_name	string	Specifies the name of the new scenario to be created.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example creates a new timing scenario with the name "my_new_scenario by duplicating an existing one (primary).

```
clone_scenario primary my_new_scenario
```

See Also
- 7.8 create_scenario
- 7.46 remove_scenario
- 7.48 rename_scenario

Related Examples on GitHub
- clone-scenario

7.6 create_clock

Description
This Tcl command creates a clock constraint on the specified sources in the current design, or a virtual clock if no source other than a name is specified. It also defines its period and waveform. The static timing analysis tool uses this information to propagate the waveform across the clock network to the clock pins of all sequential elements driven by this clock source.

The clock information is also used to compute the slacks in the specified clock domain that drive optimization tools such as place-and-route.

```
create_clock [ -name clock_name ] [-add] -period period_value \
[ -waveform edge_list ][ source_objects ]
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of the clock constraint. You must specify either a clock name or a source. If the `-name` option is not used, the clock name is specified as source name. The clock name is used to refer to the clock in other commands. You can specify name as: `-name {clk}` or `-name clk`.

.....continued

Parameter	Type	Description
add	None	Specifies that a new clock constraint is created at the same source port as the existing clock without overriding the existing constraint. The name of the new clock constraint with the `-add` option must be different than the existing clock constraint. Otherwise, it will override the existing constraint, even with the `-add` option. The `-name` option must be specified with the `-add` option.
period	real	Specifies the clock period in nanoseconds. The value you specify is the minimum time over which the clock waveform repeats. The `period_value` must be greater than zero.
waveform	real	Specifies the rise and fall times of the clock waveform in ns over a complete clock period. There must be exactly two transitions in the list, a rising transition followed by a falling transition. So in edge list, falling edge value must be less then rising edge value. You can define a clock starting with a falling edge by providing an edge list where fall time is less than rise time otherwise constraint will be disabled. If you do not specify the `-waveform` option, the tool creates a default waveform, with a rising edge at instant 0.0 ns and a falling edge at instant (period_value/2) ns.
source_objects	list of string	Specifies the source of the clock constraint. The source can be ports, pins, or nets in the design. If you specify a clock constraint on a pin that already has a clock, the new clock replaces the existing one. You must specify either a source or a clock name.

Return Type	Description
integer	Returns the ID of the clock constraint.

Error Codes

Error Code	Description
Error: SDC0001	Invalid clock constraint: clock source is incorrect.
Error: SDC0006	Invalid clock constraint: clock period is incorrect for the specified clock.
Error: SDC0007	Invalid clock constraint: waveform is incorrect.
Error: SDC0061	Invalid clock constraint: Missing or Illegal parameter/value.
Error: SDC0069	Invalid clock constraint: Need to specify clock name with `-add` option.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example creates two clocks, one on port CK1 with a period of 6, and the other on port CK2 with a period of 6, a rising edge at 0, and a falling edge at 3.

```
create_clock -name {my_user_clock} -period 6 CK1
```

```
create_clock -name {my_other_user_clock} -period 6 -waveform {0 3} {CK2}
```

The following example creates a clock on port CK3 with a period of 7, a rising edge at 2, and a falling edge at 4:

```
create_clock -period 7 -waveform {2 4} [get_ports {CK3}]
```

The following example creates a new clock constraint clk2, in addition to clk1, on the same source port clk1 without overriding it.

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports clk1]
```

```
create_clock -name clk2 -add -period 20 -waveform {0 10} [get_ports clk1]
```

The following example does not add a new clock constraint, even with the `-add` option, but overrides the existing clock constraint because of the same clock names.
Note: To add a new clock constraint in addition to the existing clock constraint on the same source port, the clock names must be different.

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports clk1]
```

```
create_clock -name clk1 -add -period 50 -waveform {0 25} [get_ports clk1]
```

Related Examples on GitHub

- create_clock

7.7 create_generated_clock

Description
This Tcl command creates a generated clock in the current design at a declared source by defining its frequency with respect to the frequency at the reference pin. The static timing analysis tool uses this information to compute and propagate its waveform across the clock network to the clock pins of all sequential elements driven by this source.

The generated clock information is also used to compute the slacks in the specified clock domain that drive optimization tools such as place-and-route.

```
create_generated_clock [-name clock_name ] [-add] [-master_clock clock_name ] -source
reference_pin
[- divide_by divide_factor ] [-multiply_by multiply_factor ] [-invert] source [-edges values ]
[-edge_shift values ]
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of the clock constraint. If the `-name` option is not used, the generated clock receives the same name specified in the source. The clock name is used to refer to the clock in other commands. You can specify `-name {my_gen_clk}` or `-name my_gen_clk`.

.....continued

Parameter	Type	Description
add	None	Specifies that the generated clock constraint is a new clock constraint in addition to the existing one at the same source. Use this option to capture the case where multiple generated clocks must be specified on the same source, because multiple clocks fan into the master pin. If you specify this option, you must also use the `-name` option. The name of the clock constraint should be different from the existing clock constraint. With this option, `-master_clock` option and `-name` options must be specified.
master_clock	string	Specifies the master clock used for the generated clock when multiple clocks fan into the master pin. This option must be used in conjunction with `-add` option of the generated clock. **Notes:** 1. The `master_clock` option is used only with the `-add` option for the generated clocks. 2. If there are multiple master clocks fanning into the same reference pin, the first generated clock specified will always use the first master clock as its source clock. 3. The subsequent generated clocks specified with the -add option can choose any of the master clocks as their source clock (including the first master clock specified).
source	string	Specifies the reference pin in the design from which the clock waveform is to be derived. You must specify `-source` reference pin.
divide_by	integer	Specifies the frequency division factor. This option cannot be used with `-egde` list. If `-egde` is specified, `divide_by` value defaults to one. For instance, if the `divide_factor` is equal to `2`, the generated clock period is twice the reference clock period. If you set `divide_by` value as `1.2` or `4/2` or `8a2` then it is being truncated as `1` or `4` or `8`, and no warning is reported.
multiply_by	integer	Specifies the frequency multiplication factor. This option cannot be used with `-egde` list. If `egde` is specified, `multiply_by` and `divide_by` values defaults to one. For instance, if the `multiply_factor` is equal to `2`, the generated clock period is half the reference clock period. If you set `multiply_by` value as `1.2` or `4/2` or `8a2` then it is being truncated as `1` or `4` or `8`, and no warning is reported.
invert	None	Specifies that the generated clock waveform is inverted with respect to the reference clock.
source	list of strings	Specifies the source of the clock constraint on internal pins of the design. If you specify a clock constraint on a pin that already has a clock, the new clock replaces the existing clock. Only one source is accepted. Wildcards are accepted as long as the resolution shows one pin. You must specify a source.
edges	list of integers	Specifies a list of positive integers that represents the edges from the source clock that are to form the edges of the generated clock. Three values must be specified to generate the clock. If you specify less than three, a tool tip indicates an error. This option cannot be used with `-divide_by/-multiply_by` factor.

.....continued

Parameter	Type	Description
edge_shift	list of floating point numbers	Specify a list of three floating point numbers that represents the amount of shift, in nanoseconds, that the specified edges are to undergo to yield the final generated clock waveform. These floating point values can be positive or negative. Positive value indicates a shift later in time, while negative indicates a shift earlier in time. With this option `-edges` option must be specified.

Return Type	Description
integer	Returns the ID of the generated clock constraint.

Error Codes

Error Code	Description
Error: SDC0004	Invalid generated clock constraint: name does not match any clock name or source.
Error: SDC0015	Invalid generated clock constraint: port list is incorrect.
Error: SDC0016	Invalid generated clock constraint: port list is empty.
Error: SDC0061	Invalid generated clock constraint: `-edges` argument is empty invoked from within command.
Error: SDC0062	Invalid generated clock constraint: `-edges`list size must be three.
Error: SDC0063	Invalid generated clock constraint:`-edges` list elements are not in increasing order.
Error: SDC0065	Invalid generated clock constraint: `-edges` cannot be used with `-multiply_by` or `-divide_by`.
Error: SDC0066	Invalid generated clock constraint: `-edge_shift` does not have accompanying `-edges`.
Error: SDC0069	Invalid clock constraint: Need to specify clock name with `-add` option.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

The following example creates a generated clock on pin `U1/reg1:Q` with a period twice as long as the period at the reference port CLK.

```
create_generated_clock -name {my_user_clock} -divide_by 2 -source \
[get_ports {CLK}] U1/reg1:Q
```

The following example creates a generated clock at the primary output of myPLL with a period 3⁄4 of the period at the reference pin clk.

```
create_generated_clock -divide_by 3 -multiply_by 4 -source clk [get_pins {myPLL:CLK1}]
```

The following example creates a new generated clock gen2 in addition to gen1 derived from same master clock as the existing generated clock, and the new constraint is added to pin `r1/CLK`.

```
create_generated_clock -name gen1 -multiply_by 1 -source [get_ports clk1] \
[get_pins r1/CLK]
create_generated_clock -name gen2 -add -master_clock clk1 -source [get_ports clk1] \
-multiply_by 2 [get_pins r1/CLK]
```

The following example does not create a new generated clock constraint in addition to the existing clock, but will override even with the `-add` option enabled, because the same names are used.

```
create_generated_clock -name gen2 -source [get_ports clk1] -multiply_by 3 \
[get_pins r1/CLK]
create_generated_clock -name gen2 -source [get_ports clk1] -multiply_by 4 \
-master_clock clk1 -add [get_pins r1/CLK]
```

See Also

- 7.6 create_clock
- 7.40 remove_generated_clock

Related Examples on GitHub

- create_generated_clock

7.8 create_scenario

Description

This Tcl command creates a new timing scenario with the specified name. You must provide a unique name (that is, it cannot already be used by another timing scenario).

A timing scenario is a set of timing constraints used with a design. Scenarios enable you to easily refine the set of timing constraints used for Timing-Driven Place-and-Route, so as to achieve timing closure more rapidly.

This command creates an empty timing scenario with the specified name and adds it to the list of scenarios.

Note: It is recommended to use the organize_tool_files command instead of this command.

```
create_scenario name
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of the new timing scenario. This is mandatory.

Error Codes

Error Code	Description
None	Required parameter _AtclParam0_ is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example creates a new timing scenario with the "scenario_A" name.

```
create_scenario scenario_A
```

Related Examples on GitHub
- create_scenario

See Also
- 7.5 clone_scenario
- 7.31 list_scenario
- 7.46 remove_scenario
- 7.48 rename_scenario

7.9 create_set

Description
This Tcl command creates a set of paths to be analyzed. Use the arguments to specify which paths to include. To create a set that is a subset of a clock domain, specify it with the `-clock` and `-type` arguments. To create a set that is a subset of an inter-clock domain set, specify it with the `-source_clock` and `-sink_clock` arguments. To create a set that is a subset (filter) of an existing named set, specify the set to be filtered with the `-parent_set` argument.

```
create_set\ -name <name>\ -parent_set <name>\ -type <set_type>\ -clock <clock name> \ -
source_clock <clock name>\ -sink_clock <clock name>\ -in_to_out\ -source <port/pin pattern> \
-sink <port/pin pattern>
```

Arguments

Parameter	Type	Description
name	string	Specifies a unique name for the newly created path set.
parent_set	string	Specifies the name of the set to filter from.
clock	string	Specifies that the set is to be a subset of the given clock domain. This argument is valid only if you also specify the `-type` argument.

.....continued

Parameter	Type	Description
type	string	Specifies the predefined set type on which to base the new path set. You can only use this argument with the `-clock` argument, not by itself. • reg_to_reg - paths between registers in the design. • async_to_reg - paths from asynchronous pins to registers. • reg_to_async - paths from registers to asynchronous pins. • external_recovery - the set of paths from inputs to asynchronous pins. • external_removal - the set of paths from inputs to asynchronous pins. • external_setup - paths from input ports to registers. • external_hold - paths from input ports to registers. • clock_to_out - paths from registers to output ports.
in_to_out	None	Specifies that the set is based on the "Input to Output" set, which includes paths that start at input ports and end at output ports.
source_clock	string	Specifies that the set will be a subset of an inter-clock domain set with the given source clock. You can only use this option with the `-sink_clock` argument.
sink_clock	string	Specifies that the set will be a subset of an inter-clock domain set with the given sink clock. You can only use this option with the `-source_clock` argument.
source	string	Specifies a filter on the source pins of the parent set. If you do not specify a parent set, this option filters all pins in the current design.
sink	string	Specifies a filter on the sink pins of the parent set. If you do not specify a parent set, this option filters all pins in the current design.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example creates set with "my_user_set" name. Filters all C* ports and D* pins in the current design.

```
create_set -name { my_user_set } -source { C* } -sink { D* }
```

The following example creates set with "my_other_user_set" name that is a subset (filter) of an existing "my_user_set" set.

```
create_set -name { my_other_user_set } -parent_set { my_user_set } -source { CL* }
```

The following example creates set with "another_set" name which is the subset of an inter-clock domain set with the given source clock.

```
create_set -name { another_set } -source_clock { EXTERN_CLOCK } \
-sink_clock { MY_GEN_CLOCK }
```

Related Examples on GitHub

- create_set

See Also

- 7.47 remove_set

7.10 expand_path

Description

This is SmartTime-specific Tcl command displays expanded path information (path details) for paths. The paths to be expanded are identified by the parameters required to display these paths with list_paths. For example, to expand the first path listed with list_paths `-clock {MYCLOCK} -type {register_to_register}`, use the command expand_path `-clock {MYCLOCK} -type {register_to_register}`. Path details contain the pin name, type, net name, cell name, operation, delay, total delay, and edge as well as the arrival time, required time, and slack. These details are the same as details available in the SmartTime Expanded Path window.

```
expand_path \
-index value \
-set name \
-clock clock_name \
-type set_type \
-analysis {max| min} \
-format {csv | text} \
-from_clock clock_name \
-to_clock clock_name
```

Arguments

Parameter	Type	Description
index value	list of integers	Specify the index of the path to be expanded in the list of paths and display them. The index starts at 1, and defaults to 1. If index value is less than 1, then it is considered as 1. List of specified indexes can be not sequential. Only the paths with indices lower than the max_paths option value will be expanded.
analysis {min|max}	string	Specify whether the timing analysis is done via max-delay (setup check) or min-delay (hold check). Valid values are - min/max or mindelay/maxdelay.
format {csv | text}	string	Specify the file format of the output. It can be either text - ASCII text format (default) or csv (comma separated values).
set	string	Displays a list of paths from the named set. You can either use the `-set` option to specify a user set by its name or use both `-clock` and `-type` to specify a set.
clock	string	Displays the set of paths belonging to the specified clock domain. You can either use this option along with `-type` to specify a set or use the `-set` option to specify the name of the set to display.

.....continued

Parameter	Type	Description
type	string	Specifies the type of paths in the clock domain to display in a list. You can only use this option with the `-clock` option. You can either use this option along with `-clock` to specify a set or use the `-set` option t specify a set name. • reg_to_reg - paths between registers in the design. • async_to_reg - path from asynchronous pins to registers. • reg_to_async - path from registers to asynchronous pins. • external_recovery - set of paths from input ports to asynchronous pins. • external_removal - set of paths from input ports to asynchronous pins. • external_setup - path from input ports to registers. • external_hold - path from input ports to registers. • clock_to_out - path from registers to output ports.
from_clock	string	Displays a list of timing paths for an inter-clock domain set belonging to the source clock specified. You can only use this option with the `-to_clock` option, not by itself.
to_clock	string	Displays a list of timing paths for an inter-clock domain set belonging to the sink clock specified. You can only use this option with the `-from_clock` option, not by itself.

Return Type	Description
string	Displays expanded path information (path details) for paths.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example displays first expanded path information (path details) for paths between registers in the design.

```
puts [expand_path -clock { myclock } -type { reg_to_reg }]
```

The following example displays expanded paths details with 1, 2 and 3 indexes from list of paths.

```
puts [expand_path -clock { myclock } -type { reg_to_reg } -index { 1 2 3 } -format text]
```

Related Examples on GitHub

- expand_path

See Also

- 7.30 list_paths

7.11 get_cells

Description

This Tcl command returns a collection of instance (cell) objects in the current design that match a specified search pattern. You can only use this command as part of a `-from`, `-to` argument in the following Tcl commands: `set_max delay`, `set_multicycle_path`, and `set_false_path`. Wildcards can be used to select multiple cells at once. If no objects match the criteria, the empty string is returned.

```
get_cells pattern
```

Arguments

Parameter	Type	Description
pattern	string	Specifies the pattern to match the instances to return. For example, `get_cells U18*` returns all instances starting with the characters `U18`, where `*` is a wild card character that represents any character string. This is mandatory.

Return Type	Description
object	Returns an object representing the cells (instances) that match those specified in the pattern argument.

Error Codes

Error Code	Description
None	Required parameter _AtclParam0_ is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

The following example sets maximum delay constraining all paths from reg* cells to out ports with a delay less than 2 ns.

```
set_max_delay 2 -from [get_cells {reg*}] -to [get_ports {out}]
```

Related Examples on GitHub

- get_cells

See Also

- 7.12 get_clocks
- 7.14 get_nets
- 7.15 get_pins
- 7.16 get_ports

7.12 get_clocks

Description

This Tcl command returns an object representing the clock(s) that match those specified in the current timing scenario. Wildcards can be used to select multiple clocks at once. If no objects match the criteria, the empty string is returned.

- If this command is used as a `-from` argument in either the set maximum (`set_max_delay`), or set minimum delay (`set_min_delay`), false path (`set_false_path`), and multicycle constraints (`set_multicycle_path`), the clock pins of all the registers related to this clock are used as path start points.
- If this command is used as a `-to` argument in either the set maximum (`set_max_delay`), or set minimum delay (`set_min_delay`), false path (`set_false_path`), and multicycle constraints (`set_multicycle_path`), the synchronous pins of all the registers related to this clock are used as path endpoints.

```
get_clocks pattern
```

Arguments

Parameter	Type	Description
pattern	string	Mandatory. Specifies the pattern to match to the SmartTime on which a clock constraint has been set.

Return Type	Description
object	Returns an object representing the clock(s) that match those specified in the pattern argument.

Error Codes

Error Code	Description
None	Required parameter _AtclParam0_ is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example sets maximum delay constraining all paths from datal port to ck1 clock with a delay less then 2 ns.

```
set_max_delay -from [get_ports datal] -to [get_clocks ck1]
```

Related Examples on GitHub
- get_clocks

See Also
- 7.6 create_clock
- 7.7 create_generated_clock

7.13 get_current_scenario

Description
This Tcl command returns the name of the current timing scenario.

```
get_current_scenario
```

Arguments

Parameter	Type	Description
None	None	None

Return Type	Description
string	Name of the current timing scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
With this command we get the name of the current timing scenario.

```
get_current_scenario
```

Related Examples on GitHub
- get_current_scenario

See Also
- 7.8 create_scenario

- 7.55 set_current_scenario
- 7.46 remove_scenario
- 7.48 rename_scenario

7.14 get_nets

Description
This Tcl command returns a collection of nets matching the pattern you specify. You can only use this command as source objects in create clock (`create_clock`) or create generated clock (`create_generated_clock`) constraints and as `-through` arguments in the set false path, set minimum delay, set maximum delay, and set multicycle path constraints. Wildcards can be used to select multiple nets at once. If no objects match the criteria, the empty string is returned.

```
get_nets pattern
```

Arguments

Parameter	Type	Description
pattern	string	Specifies the pattern to match the names of the nets to return. For example, `get_nets N_255*` returns all nets starting with the characters `N_255`, where `*` is a wildcard that represents any character string. This is mandatory.

Return Type	Description
object	Returns an object representing the nets that match those specified in the pattern argument.

Error Codes

Error Code	Description
None	Required parameter _AtclParam0_ is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example sets maximum delay constraining all paths from `RDATA1` port passing `-through net_chkp1 net_chkqi` nets.

```
set_max_delay 2 -from [get_ports RDATA1] -through [get_nets {net_chkp1 net_chkqi}]
```

The following example specifies all paths through the nets Tblk/rm/n* to be false.

```
set_false_path -through [get_nets {Tblk/rm/n*}]
```

The following example creates a clock on cknet net with a period of 2.5 ns.

```
create_clock -name  mainCLK -period 2.5 [get_nets {cknet}]
```

Related Examples on GitHub

- get_nets

See Also

- 7.6 create_clock
- 7.7 create_generated_clock
- 7.58 set_false_path
- 7.61 set_min_delay
- 7.60 set_max_delay
- 7.62 set_multicycle_path

7.15 get_pins

Description

This Tcl command returns an object representing the pin(s) that match those specified in the pattern argument. Wildcards can be used to select multiple pins at once. If no objects match the criteria, the empty string is returned.

```
get_pins pattern
```

Arguments

Parameter	Type	Description
pattern	string	Specifies the pattern to match the pins to return. For example, `get_pins clock_gen*` returns all pins starting with the characters `clock_gen`, where `*` is a wildcard that represents any character string. This is mandatory.

Return Type	Description
object	Returns an object representing the pin(s) that match those specified in the pattern argument.

Error Codes

Error Code	Description
None	Required parameter _AtclParam0_ is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example creates a clock on pin `clock_gen/reg2:Q` with a period of 10 ns.

```
create_clock -period 10 [get_pins clock_gen/reg2:Q]
```

Related Examples on GitHub
- get_pins

See Also
- 7.6 create_clock
- 7.7 create_generated_clock
- 7.58 set_false_path
- 7.61 set_min_delay
- 7.60 set_max_delay
- 7.62 set_multicycle_path

7.16 get_ports

Description
This Tcl command returns an object representing the port(s) that match those specified in the pattern argument. Wildcards can be used to select multiple ports at once. If no objects match the criteria, the empty string is returned.

```
get_ports pattern
```

Arguments

Parameter	Type	Description
pattern	string	Specifies the pattern to match the ports.

Return Type	Description
object	Returns an object representing the port(s) that match those specified in the pattern argument.

Error Codes

Error Code	Description
None	Required parameter _AtclParam0_ is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example creates a clock on port `CK1` with a period of 10 ns.

```
create_clock -period 10 [get_ports CK1]
```

Related Examples on GitHub
- get_ports

See Also
- 7.6 create_clock
- 7.7 create_generated_clock
- 7.59 set_input_delay
- 7.64 set_output_delay
- 7.58 set_false_path
- 7.61 set_min_delay
- 7.60 set_max_delay
- 7.62 set_multicycle_path

7.17 list_clock_groups

Description
This Tcl command returns the details for all the existing clock groups in the current timing constraint scenario.

```
list_clock_groups
```

Arguments

Parameter	Type	Description
None	None	None

Return Type	Description
string	Details about all of the clock groups constraints in the current timing constraint scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
With this command we get the details about all of the existing clock groups in the current timing constraint scenario.

```
puts [list_clock_groups]
```

Related Examples on GitHub
- list_clock_groups

See Also
- 7.51 set_clock_groups
- 7.35 remove_clock_groups

7.18 list_clock_latencies

Description
This Tcl command returns details about all of the clock latencies in the current timing constraint scenario.

```
list_clock_latencies
```

Arguments

Return Type	Description
string	Returns details about all of the clock latencies in the current timing constraint scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
With this command we get the details about all of the clock latencies in the current timing constraint scenario.

```
puts [list_clock_latencies]
```

Related Examples on GitHub
- list_clock_latencies

See Also

- 7.52 set_clock_latency
- 7.36 remove_clock_latency

7.19 list_clock_uncertainties

Description

This Tcl command returns details about all of the clock uncertainties in the current timing constraint scenario.

```
list_clock_uncertainties
```

Arguments

Parameter	Type	Description
None	None	None

Return Type	Description
string	Returns details about all of the clock uncertainties.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

With this command we get the details about all of the clock uncertainties in the current timing constraint scenario.

```
puts [list_clock_uncertainties]
```

Related Examples on GitHub

- list_clock_uncertainties

See Also

- 7.54 set_clock_uncertainty
- 7.37 remove_clock_uncertainty

7.20 list_clocks

Description

This Tcl command returns details about all of the clock constraints in the current timing constraint scenario.

```
list_clocks
```

Arguments

Parameter	Type	Description
None	None	None

Return Type	Description
string	Returns details about all of the clock constraints in the current timing constraint scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example displays the details about all of the clock constraints in the current timing constraint scenario.

```
puts [list_clocks]
```

Related Examples on GitHub
- list_clocks

See Also
- 7.6 create_clock
- 7.34 remove_clock

7.21 list_disable_timings

Description
This Tcl command returns the list of disable timing constraints for the current scenario.

```
list_disable_timings
```

Arguments

Return Type	Description
string	Returns list of disable timing constraints for the current timing constraint scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
With this command we get the disable timing constraints in the current timing constraint scenario.

```
puts [list_disable_timings]
```

Related Examples on GitHub
- list_disable_timings

See Also
- 7.56 set_disable_timing
- 7.38 remove_disable_timing

7.22 list_false_paths

Description
This Tcl command returns details about all of the false paths in the current timing constraint scenario.

```
list_false_paths
```

Arguments

Return Type	Description
string	Returns details about all of the false paths in the current timing constraint scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
With this command we get the details about all of the false paths in the current timing constraint scenario.

```
puts [list_false_paths]
```

Related Examples on GitHub
- list_false_paths

See Also

- 7.58 set_false_path
- 7.39 remove_false_path

7.23 list_generated_clocks

Description

This Tcl command returns details about all of the generated clock constraints in the current timing constraint scenario.

```
list_generated_clocks
```

Arguments

Return Type	Description
string	Returns details about all of the generated clock constraints in the current timing constraint scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

The following example displays the details about all of the generated clock constraints in the current timing constraint scenario.

```
puts [list_generated_clocks]
```

Related Examples on GitHub

- list_generated_clocks

See Also

- 7.7 create_generated_clock
- 7.40 remove_generated_clock

7.24 list_input_delays

Description

This Tcl command returns details about all of the input delay constraints in the current timing constraint scenario.

```
list_input_delays
```

Arguments

Return Type	Description
string	Details about all of the input delay constraints in the current timing constraint scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
With this command we get the details about all of the input delay constraints in the current timing constraint scenario.

```
puts [list_input_delays]
```

Related Examples on GitHub
- list_input_delays

See Also
- 7.41 remove_input_delay
- 7.59 set_input_delay

7.25 list_max_delays

Description
This Tcl command returns details about all of the maximum delay constraints in the current timing constraint scenario.

```
list_max_delays
```

Arguments

Return Type	Description
string	Details about all of the max delay constraints in the current timing constraint scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
With this command we get the details about all of the maximum delay constraints in the current timing constraint scenario.

```
puts [list_max_delays]
```

Related Examples on GitHub
- list_max_delays

See Also
- 7.42 remove_max_delay
- 7.60 set_max_delay

7.26 list_min_delays

Description
This Tcl command returns details about all of the minimum delay constraints in the current timing constraint scenario.

```
list_min_delays
```

Arguments

Return Type	Description
string	Details about all of the min delay constraints in the current timing constraint scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
With this command we get the details about all of the minimum delay constraints in the current timing constraint scenario.

```
puts [list_min_delays]
```

Related Examples on GitHub
- list_min_delays

See Also
- 7.43 remove_min_delay
- 7.61 set_min_delay

7.27 list_multicycle_paths

Description
This Tcl command returns details about all of the multicycle paths in the current timing constraint scenario.

```
list_multicycle_paths
```

Arguments

Return Type	Description
string	Returns details about all of the multicycle paths in the current timing constraint scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
With this command we get the details about all of the multicycle paths constraints in the current timing constraint scenario.

```
puts [list_multicycle_paths]
```

Related Examples on GitHub
- list_multicycle_paths

See Also
- 7.44 remove_multicycle_path
- 7.62 set_multicycle_path

7.28 list_objects

Description
This Tcl command returns a list of object matching the parameter. Objects can be nets, pins, ports, clocks or instances.

```
list_objects <object>
```

Arguments

Parameter	Type	Description
objects	string	Any timing constraint parameter (object can be nets, pins, ports, clocks or instances). This is mandatory.

Return Type	Description
list of objects	Returns a list of nets, pins, ports, clocks or instances.

Error Codes

Error Code	Description
None	Required parameter _AtclParam0_ is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example lists all the inputs in your design.

```
list_objects [all_inputs]
```

You can also use wildcards to filter your list, as in the following command.

```
list_objects [get_ports a*]
```

Related Examples on GitHub

- list_objects

7.29 list_output_delays

Description
This Tcl command returns details about all of the output delay constraints in the current timing constraint scenario.

```
list_output_delays
```

Arguments

Return Type	Description
string	Details about all of the output delay constraints in the current timing constraint scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
With this command we get the details about all of the output delay constraints in the current timing constraint scenario.

```
puts [list_output_delays]
```

Related Examples on GitHub

- list_output_delays

See Also

- 7.45 remove_output_delay
- 7.64 set_output_delay

7.30 list_paths

Description
This SmartTime-specific Tcl command returns a list of the n worst paths matching the arguments. The number of paths returned can be changed using the `set_options -limit_max_paths` <value> command.

```
list_paths \
-analysis <max | min> \
-format <csv | text> \
-set <name> \
-clock <clock name> \
-type <set_type> \
-from_clock <clock name> \
-to_clock <clock name> \
-in_to_out \
-from <port/pin pattern> \
-to <port/pin pattern>
```

Arguments

Parameter	Type	Description
analysis	string	Specifies whether the timing analysis is done for `max-delay` (setup check) or `min-delay` (hold check). Valid values are: max or min.
format	string	Specifies the list format. It can be either text (default) or csv (comma separated values). Text format is better for display and csv format is better for parsing.

.....continued

Parameter	Type	Description
set	string	Returns a list of paths from the named set. You can either use the `-set` option to specify a user set by its name or use both `-clock` and `-type` to specify a set.
clock	string	Returns a list of paths from the specified clock domain. This option requires the `-type` option. You cannot use wildcards when specifying a clock name.
type	string	Specifies the type of paths to be included. It can only be used along with `-clock`. Valid values are: • `reg_to_reg` -paths between registers in the design. • `async_to_reg` -paths from asynchronous pins to registers. • `reg_to_async` -paths from registers to asynchronous pins of registers. • `external_recovery` -paths from input ports to asynchronous pins of registers. • `external_removal` -paths from input ports to asynchronous pins of registers. • `external_setup` -paths from input ports to data pins of registers. • `external_hold` -paths from input ports to data pins of registers. • `clock_to_out` -paths from registers to output ports.
from_clock	string	Used along with `-to_clock` to get the list of paths of the inter-clock domain between the two clocks.
to_clock	string	Used along with `-from_clock` to get the list of paths of the inter-clock domain between the two clocks.
in_to_out	None	Used to get the list of path between input and output ports.
from	string	Filter the list of paths to those starting from ports or pins matching the pattern.
to	string	Filter the list of paths to those ending at ports or pins matching the pattern.

Return Type	Description
list of strings	Returns a list of the n worst paths matching the arguments.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command displays the list of register to register paths of clock domain clk1.

```
puts [ list_paths -clock clk1 -type reg_to_reg ]
```

Related Examples on GitHub
- list_paths

7.31 list_scenario

Description
This Tcl command returns a list of names of all of the available timing scenarios.

```
list_scenario name
```

Arguments

Return Type	Description
list of strings	List of all names of the available timing scenarios.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
With this command we get the list of available timing scenario names.

```
list_scenario
```

Related Examples on GitHub
- list_scenario

See Also
- 7.5 clone_scenario
- 7.55 set_current_scenario
- 7.13 get_current_scenario
- 7.46 remove_scenario
- 7.48 rename_scenario

7.32 read_sdc

Description
The read_sdc Tcl command evaluate an SDC file, adding all constraints to the specified scenario (or the current/ default one if none is specified). Existing constraints are removed if -add is not specified.

```
read_sdc \
-add \
-scenario scenario_name \
-netlist ( user | optimized ) \
-pin_separator ( : | / ) \
-ignore_errors file_name
```

Arguments

Parameter	Type	Description
add	None	Specifies that the constraints from the SDC file will be added on top of the existing ones, overriding them in case of a conflict. If not used, the existing constraints are removed before the SDC file is read.
scenario	string	Specifies the scenario to add the constraints to. The scenario is created if none exists with this name.
netlist	string	Specifies whether the SDC file contains object defined at the post-synthesis netlist (user) level or physical (optimized) netlist (used for timing analysis).
pin_separator	char	Specify the pin separator used in the SDC file. It can be either ':' or '/'.
ignore_errors	None	Optional. Specifies whether to avoid reporting errors for derived constraints targeting the logic that becomes invalid due to logic optimization. It is an optional argument. Some IPs may have extra logic present depending on other IPs used in the design but the synthesis tool will remove this logic if fewer IPs were used. In such cases, the implementation flow will halt without -ignore_errors flag. **Note:** It is not recommended to use this flag outside similar use cases.
file	string	Specify the SDC file name.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command removes all constraints from the current/default scenario and adds all constraints from design.sdc file to it.

```
read_sdc design.sdc
```

7.33 remove_all_constraints

Description
This Tcl command removes all timing constraints from analysis.

```
remove_all_constraints
```

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example removes all timing constraints from analysis.

```
remove_all_constraints
```

Related Examples on GitHub
- remove_all_constraints

See Also
- 7.4 check_constraints

7.34 remove_clock

Description
This Tcl command removes the specified clock constraint from the current timing scenario. If the specified name does not match a clock constraint in the current scenario, or if the specified ID does not refer to a clock constraint, this command fails.

Do not specify both the clock and port names and the constraint ID.

```
remove_clock -name clock_name | -id constraint_ID
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of the clock constraint to remove from the current scenario. You must specify either a clock name or an ID. **Note:** You must specify clock name as {CLK}, not [get_clocks {CLK}].
id	integer	Specifies the ID of the clock constraint to remove from the current scenario. You must specify either an ID or a clock name that exists in the current scenario.

Error Codes

Error Code	Description
None	Invalid clock name argument.
None	Only one argument is needed.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions

You cannot use wildcards when specifying a clock names.

Example

The following example removes the clock constraint named "my_user_clock".

```
remove_clock -name my_user_clock
```

The following example removes the clock constraint using its ID.

```
set clockId [create_clock -name my_user_clock -period 2]
```

```
remove_clock -id $clockId
```

Related Examples on GitHub

- remove_clock

See Also

- 7.6 create_clock
- 7.7 create_generated_clock

7.35 remove_clock_groups

Description
This Tcl command removes a clock group by specifying its name or its group ID. If the arguments do not match, or if the ID does not refer to a clock group, the command fails.
Note: The exclusive flag is not needed when removing a clock group by ID. These flags are mutually exclusive. Only one can be specified.

```
remove_clock_groups [-id constraint_ID | -name groupname ] \
[-physically_exclusive | -logically_exclusive | -asynchronous]
```

Arguments

Parameter	Type	Description
id	integer	Specifies the clock group by the ID. You must specify either a clock group ID or a clock group name that exists in the current scenario.
name	string	Specifies the clock group by name (to be always followed by the exclusive flag).
physically_exclusive	None	Specifies that the clock groups are physically exclusive with respect to each other.
logically_exclusive	None	Specifies that the clocks groups are logically exclusive with respect to each other.
asynchronous	None	Specifies that the clock groups are asynchronous with respect to each other.

Error Codes

Error Code	Description
None	Only one argument is needed.
None	Invalid clock Groups name argument.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions
You cannot use wildcards when specifying a clock groups name.

Example
The following commands removes clock groups with the "mygroup3" names and the clock groups with 12 IDs.

```
remove_clock_groups -name mygroup3 -physically_exclusive
```

```
remove_clock_groups id 12
```

Related Examples on GitHub
- remove_clock_groups

See Also
- 7.51 set_clock_groups
- 7.17 list_clock_groups

7.36 remove_clock_latency

Description
This Tcl command removes a clock source latency from the specified clock and from all edges of the clock. If the specified name does not match a generated clock constraint in the current scenario, or if the specified ID does not refer to a generated clock constraint, this command fails.

Do not specify both the clock and port names and the constraint ID.

```
remove_clock_latency -source clock_name | -id constraint_ID
```

Arguments

Parameter	Type	Description
source	string	Specifies either the clock name or source name of the clock constraint from which to remove the clock source latency. You must specify either a clock name or an ID.
id	integer	Specifies the ID of the clock constraint to remove from the current scenario. You must specify either an ID or a clock name that exists in the current scenario.

Error Codes

Error Code	Description
None	Only one argument is needed.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions
You cannot use wildcards when specifying the clock names.

Example
The following example removes the clock source latency from the specified clock.

```
remove_clock_latency -source [get_clocks {my_clock} ]
```

Related Examples on GitHub
- remove_clock_latency

See Also
- 7.52 set_clock_latency

7.37 remove_clock_uncertainty

Description
This Tcl command removes a clock-to-clock uncertainty from the current timing scenario by specifying either its exact arguments or its ID.

If the specified arguments do not match clocks with an uncertainty constraint in the current scenario, or if the specified ID does not refer to a clock-to-clock uncertainty constraint, this command fails. Do not specify both the exact arguments and the ID.

```
remove_clock_uncertainty -from | -rise_from | -fall_from from_clock_list -to | -rise_to | \
-fall_to to_clock_list -setup {value} -hold {value} | -id constraint_ID
```

Arguments

Parameter	Type	Description
from	list of strings	Specifies that the clock-to-clock uncertainty applies to both rising and falling edges of the source clock list. Only one of the `-from`, `-rise_from`, or `-fall_from` arguments can be specified for the constraint to be valid.
rise_from	list of strings	Specifies that the clock-to-clock uncertainty applies only to rising edges of the source clock list. Only one of the `-from`, `-rise_from`, or `-fall_from` arguments can be specified for the constraint to be valid.
fall_from	list of strings	Specifies that the clock-to-clock uncertainty applies only to falling edges of the source clock list. Only one of the `-from`, `-rise_from`, or `-fall_from` arguments can be specified for the constraint to be valid.
from_clock_list	list of strings	Specifies the list of clock names as the uncertainty source.
to	list of strings	Specifies that the clock-to-clock uncertainty applies to both rising and falling edges of the destination clock list. Only one of the `-to`, `-rise_to`, or `-fall_to` arguments can be specified for the constraint to be valid.

.....continued

Parameter	Type	Description
rise_to	list of strings	Specifies that the clock-to-clock uncertainty applies only to rising edges of the destination clock list. Only one of the `-to`, `-rise_to`, or `-fall_to` arguments can be specified for the constraint to be valid.
fall_to	list of strings	Specifies that the clock-to-clock uncertainty applies only to falling edges of the destination clock list. Only one of the `-to`, `-rise_to`, or `-fall_to` arguments can be specified for the constraint to be valid.
to_clock_list	list of strings	Specifies the list of clock names as the uncertainty destination.
setup	None	Specifies that the uncertainty applies only to setup checks. If none or both `-setup` and `-hold` are present, the uncertainty applies to both setup and hold checks.
hold	None	Specifies that the uncertainty applies only to hold checks. If none or both `-setup` and `-hold` are present, the uncertainty applies to both setup and hold checks.
id	integer	Specifies the ID of the clock constraint to remove from the current scenario. You must specify either the exact parameters to set the constraint or its constraint ID.

Error Codes

Error Code	Description
None	Only one argument is needed.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example removes uncertainties from Clk1 clock to Clk2 clock domains.

```
remove_clock_uncertainty -from [ get-clock {Clk1} ] -to [ get_clock {Clk2} ]
```

The following example removes uncertainties between Clk1, Clk2, Clk3 and Clk4 clock domains with specific edges.

```
remove_clock_uncertainty -from [ get_clocks {Clk1} ]  -fall_to [ get_clocks {Clk2 Clk3} ] -
setup
```

```
remove_clock_uncertainty 4.3 -fall_from [ get_clocks {Clk1 Clk2} ] -rise_to [ get_clocks {*} ]
```

```
remove_clock_uncertainty 0.1 -rise_from [ get_clocks {Clk1 Clk2} ] \
-fall_to [ get_clocks {Clk3 Clk4} ] -setup
```

```
remove_clock_uncertainty 5 -rise_from [ get_clocks {Clk1} ] -to [ get_clocks {*} ]
```

```
remove_clock_uncertainty -id $clockId
```

Related Examples on GitHub

- remove_clock_uncertainty

See Also

- 7.19 list_clock_uncertainties
- 7.54 set_clock_uncertainty

7.38 remove_disable_timing

Description

This Tcl command removes a disable timing constraint by specifying its arguments, or its ID. If the arguments do not match a disable timing constraint, or if the ID does not refer to a disable timing constraint, the command fails.

```
remove_disable_timing -from value -to value name -id constraint_ID
```

Arguments

Parameter	Type	Description
from	string	Specifies the starting port. The `-from` and `-to` arguments must either both be present or both ommitted for the constraint to be valid.
name	string	Specifies the cell(instance) name where the disable timing constraint will be removed. It is an error to supply both a cell name and a constraint ID, as they are mutually exclusive. No wildcards are allowed when specifying a clock name, either alone or in an accessor command1.
to	string	Specifies the ending port. The `-from` arguments must either both be present or both ommitted for the constraint to be valid.
id	string	Specifies the constraint name where the disable timing constraint will be removed. It is an error to supply both a cell name and a constraint ID, as they are mutually exclusive. No wildcards are allowed when specifying a clock name, either alone or in an accessor command1.

Error Codes

Error Code	Description
None	Required parameter _AtclParam0_ is missing.
None	Only one argument is needed.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions
You cannot use wildcards when specifying a clock name, either alone or in an accessor command such as get_pins or get_ports.

Example
The following example removes disable timing constraint between A and Y ports.

```
remove_disable_timing -from A -to Y -id new_constraint
```

Related Examples on GitHub

- remove_disable_timing

See Also

- 7.56 set_disable_timing
- 7.21 list_disable_timings

7.39 remove_false_path

Description
This Tcl command removes a false path constraint from the current timing scenario by specifying either its exact arguments or its ID. If the arguments do not match a false path constraint in the current scenario, or if the specified ID does not refer to a false path constraint, this command fails.
Note: Do not specify both false path arguments and the constraint ID.

```
remove_false_path [-from from_list] [-to to_list] [-through through_list] | -id constraint_ID
```

Arguments

Parameter	Type	Description
from	list of strings	Specifies a list of timing path starting points. A valid timing starting point is a clock, a primary input, an inout port, or a clock pin of a sequential cell.

.....continued		
Parameter	**Type**	**Description**
through	list of strings	Specifies a list of pins or nets through which the disabled paths must pass.
to	list of strings	Specifies a list of timing path ending points. A valid timing ending point is a clock, a primary output, an inout port, or a data pin of a sequential cell.
id	integer	Specifies the ID of the false path constraint to remove from the current scenario. You must specify either the exact false path arguments to remove or the constraint ID that refers to the false path constraint to remove.

Error Codes

Error Code	Description
None	Invalid arguments -from/-to/-through.
None	Only one argument is needed.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions
You cannot use wildcards when specifying a clock name, either alone or in an accessor command such as get_pins or get_ports.

Example
The following example specifies all false path to remove.

```
remove_false_path -through U0/U1:Y
```

The following example removes the false path constraint using its id.

```
set fpId [set_false_path -from [get_clocks c*] -through {topx/reg/*} \
-to [get_ports out15] ]
```

```
remove_false_path -id $fpId
```

Related Examples on GitHub
- remove_false_path

See Also
- 7.58 set_false_path

7.40 remove_generated_clock

Description
This Tcl command removes the specified generated clock constraint from the current scenario. If the specified name does not match a generated clock constraint in the current scenario, or if the specified ID does not refer to a generated clock constraint, this command fails.

Do not specify both the clock and port names and the constraint ID.

```
remove_generated_clock -name clock_name | -id constraint_ID
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of the generated clock constraint to remove from the current scenario. You must specify either a clock name or an ID.
id	integer	Specifies the ID of the generated clock constraint to remove from the current scenario. You must specify either an ID or a clock name that exists in the current scenario.

Error Codes

Error Code	Description
None	Invalid clock name argument.
None	Only one argument is needed.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions
You cannot use wildcards when specifying a clock names.

Example
The following example removes the generated clock constraint named "my_user_clock".

```
remove_generated_clock -name my_user_clock
```

Related Examples on GitHub

- remove_generated_clock

See Also

- 7.7 create_generated_clock

7.41 remove_input_delay

Description

This Tcl command removes an input delay by specifying both the clocks and port names or the ID of the input delay constraint to remove. If the clocks and port names do not match an input delay constraint in the current scenario, or if the specified ID does not refer to an input delay constraint, this command fails.

Do not specify both the clock and port names and the constraint ID.

```
remove_input_delay -clock clock_name port_pin_list | -id constraint_ID
```

Arguments

Parameter	Type	Description
clock	string	Specifies the clock name to which the specified input delay value is assigned. **Note:** you must specify clock name as {CLK}, not [get_clocks {CLK}].
port_pin_list	list of strings	Specifies the port names to which the specified input delay value is assigned.
id	integer	Specifies the ID of the clock with the input_delay value to remove from the current scenario. You must specify either both a clock name and list of port names or the input_delay constraint ID.

Error Codes

Error Code	Description
None	Parameter -clock has illegal value.
None	Invalid clock/port arguments.
None	Only one argument is needed.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions

You cannot use wildcards when specifying a clock or port names, either alone or in an accessor command.

Example
The following example removes the input delay from CLK1 on port data1.

```
remove_input_delay -clock [get_clocks CLK1] [get_ports data1]
```

Related Examples on GitHub
- remove_input_delay

See Also
- 7.59 set_input_delay

7.42 remove_max_delay

Description
This Tcl command removes a maximum delay constraint from the current timing scenario by specifying either its exact arguments or its ID. If the arguments do not match a maximum delay constraint in the current scenario, or if the specified ID does not refer to a maximum delay constraint, this command fails.

Do not specify both maximum delay arguments and the constraint ID.

```
remove_max_delay [-from from_list] [-to to_list] [-through through_list]
```

```
remove_max_delay -id constraint_ID
```

Arguments

Parameter	Type	Description
from	list of strings	Specifies a list of timing path starting points. A valid timing starting point is a clock, a primary input, an inout port, or a clock pin of a sequential cell.
through	list of strings	Specifies a list of pins, ports, cells, or nets through which the disabled paths must pass.
to	list of strings	Specifies a list of timing path ending points. A valid timing ending point is a clock, a primary output, an inout port, or a data pin of a sequential cell.
id	integer	Specifies the ID of the maximum delay constraint to remove from the current scenario. You must specify either the exact maximum delay arguments to remove or the constraint ID that refers to the maximum delay constraint to remove.

Error Codes

Error Code	Description
None	Invalid arguments -from/-to/-through.
None	Only one argument is needed.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions
You cannot use wildcards when specifying a clock or port name, either alone or in an accessor command.

Example
The following example specifies a range of maximum delay constraints to remove.

```
remove_max_delay -through U0/U1:Y
```

Related Examples on GitHub

- remove_max_delay

See Also

- 7.60 set_max_delay
- 7.61 set_min_delay
- 7.43 remove_min_delay

7.43 remove_min_delay

Description
This Tcl command removes a minimum delay constraint from the current timing scenario by specifying either its exact arguments or its ID. If the arguments do not match a minimum delay constraint in the current scenario, or if the specified ID does not refer to a minimum delay constraint, this command fails.

Do not specify both minimum delay arguments and the constraint ID.

```
remove_min_delay [-from from_list] [-to to_list] [-through through_list]
```

```
remove_min_delay -id constraint_ID
```

Arguments

Parameter	Type	Description
from	list of strings	Specifies a list of timing path starting points. A valid timing starting point is a clock, a primary input, an inout port, or a clock pin of a sequential cell.
through	list of strings	Specifies a list of pins, ports, cells, or nets through which the disabled paths must pass.

.....continued		
Parameter	**Type**	**Description**
to	list of strings	Specifies a list of timing path ending points. A valid timing ending point is a clock, a primary output, an inout port, or a data pin of a sequential cell.
id	integer	Specifies the ID of the minimum delay constraint to remove from the current scenario. You must specify either the exact minimum delay arguments to remove or the constraint ID that refers to the minimum delay constraint to remove.

Error Codes

Error Code	Description
None	Invalid arguments `-from/-to/-throug.`
None	Only one argument is needed.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions
You cannot use wildcards when specifying a clock or port name, either alone or in an accessor command.

Example
The following example specifies a range of minimum delay constraints to remove.

```
remove_min_delay -through U0/U1:Y
```

Related Examples on GitHub

- remove_min_delay

See Also

- 7.61 set_min_delay

7.44 remove_multicycle_path

Description
This Tcl command removes a multicycle path constraint from the current timing scenario by specifying either its exact arguments or its ID. If the arguments do not match a multicycle path constraint in the current scenario, or if the specified ID does not refer to a multicycle path constraint, this command fails.

Note: Do not specify both multicycle path arguments and the constraint ID.

```
remove_multicycle_path [-from from_list] [-to to_list] [-through through_list]
```

```
remove_multicycle_path -id constraint_ID
```

Arguments

Parameter	Type	Description
from	list of strings	Specifies a list of timing path starting points. A valid timing starting point is a clock, a primary input, an inout port, or a clock pin of a sequential cell.
through	list of strings	Specifies a list of pins, ports, cells, or nets through which the disabled paths must pass.
to	list of strings	Specifies a list of timing path ending points. A valid timing ending point is a clock, a primary output, an inout port, or a data pin of a sequential cell.
id	string	Specifies the ID of the multicycle path constraint to remove from the current scenario. You must specify either the exact multicycle path arguments to remove or the constraint ID that refers to the multicycle path constraint to remove.

Error Codes

Error Code	Description
None	Only one argument is needed.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions

You cannot use wildcards when specifying a clock name, either alone or in an accessor command such as get_pins or get_ports.

Example

The following example specifies all multicycle paths between reg1 and reg2 to 3 cycles for setup check.

```
remove_multicycle_path -from [get_pins {reg1}] -to [get_pins {reg2}]
```

Related Examples on GitHub

- remove_multicycle_path

See Also

- 7.62 set_multicycle_path

7.45 remove_output_delay

Description

This Tcl command removes an output delay by specifying both the clocks and port names or the ID of the output_delay constraint to remove. If the clocks and port names do not match an output delay constraint in the current scenario, or if the specified ID does not refer to an output delay constraint, this command fails.

Note: Do not specify both the clock and port names and the constraint ID.

```
remove_output_delay -clock clock_name port_pin_list
```

```
remove_output_delay -id constraint_ID
```

Arguments

Parameter	Type	Description
clock	string	Specifies the clock name to which the specified output delay value is assigned. **Note:** You must specify clock name as `{CLK}`, not `[get_clocks {CLK}]`.
port_pin_list	list of strings	Specifies the port names to which the specified output delay value is assigned.
id	integer	Specifies the ID of the clock with the output_delay value to remove from the current scenario. You must specify either both a clock name and list of port names or the output_delay constraint ID.

Error Codes

Error Code	Description
None	Parameter -clock has illegal value.
None	Invalid clock/port arguments.
None	Only one argument is needed.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions

You cannot use wildcards when specifying a clock or port names, either alone or in an accessor command.

Example
The following example removes the output delay from `CLK1` on port `out1`.

```
remove_output_delay -clock {CLK1} [get_ports out1]
```

Related Examples on GitHub
- remove_output_delay

See Also
- 7.64 set_output_delay

7.46 remove_scenario

Description
This Tcl command removes a scenario from the constraint database and removes it to the list of scenarios.

```
remove_scenario name
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of the scenario to delete. This is mandatory.

Error Codes

Error Code	Description
None	Required parameter _AtclParam0_ is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions
You cannot use wildcards when specifying a set name to remove.

Example
The following command removes the scenario named my_scenario.

```
remove_scenario my_scenario
```

Related Examples on GitHub
- remove_scenario

See Also

- 7.8 create_scenario
- 7.5 clone_scenario
- 7.48 rename_scenario

7.47 remove_set

Description

This Tcl command removes a set of paths from analysis. Only user-created sets can be deleted.

```
remove_set -name name
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of the set of paths to delete.

Error Codes

Error Code	Description
None	Required parameter -name is missing.
None	Unable to find set.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions

You cannot use wildcards when specifying a set name.

Example

The following command removes the set named my_set.

```
remove_set -name my_set
```

Related Examples on GitHub

- remove_set

See Also

- 7.9 create_set

7.48 rename_scenario

Description
This Tcl command renames an existing timing scenario to a new name. You must provide a unique name (that is, it cannot already be used by another timing scenario) for the new name.

Note: It is recommended to use the organize_tool_files command instead of this command.

```
rename_scenario old_name new_name
```

Arguments

Parameter	Type	Description
old_name	string	Specifies the name of the existing timing scenario to be renamed.
new_name	string	Specifies the new name for the new scenario.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following command renames the "my_old_scenario" scenario name into a "my_new_scenario" new name.

```
rename_scenario my_old_scenario my_new_scenario
```

Related Examples on GitHub
- rename_scenario

See Also
- 7.8 create_scenario
- 7.5 clone_scenario
- 7.46 remove_scenario

7.49 report

Description
This SmartTime-specific tcl command specifies the type of reports to be generated and the type of Analysis(Max Delay or Min Delay) Performed to generate the reports. Using this command you can generate the following types of reports:

- Timer report – This report displays the timing information organized by clock domain.
- Timing Violations report – This flat slack report provides information about constraint violations.

- Bottleneck report – This report displays the points in the design that contribute to the most timing violations.
- Datasheet report – This report describes the characteristics of the pins, I/O technologies, and timing properties in the design.
- Constraints Coverage report – This report displays the overall coverage of the timing constraints set on the current design.
- Combinational Loop report – This report displays loops found during initialization.
- Clock Domain Crossing report – This report analyzes timing paths that cross from one clock domain (the source clock) to another clock domain (the destination clock).

If the specified parameter/value is not correct incorrect this command fails.

```
report -type (timing | violations | datasheet | bottleneck | constraints_coverage |
combinational_loops | cdc) \
-analysis <max|min> \
-format (csv|text) \
<filename> \
timing options \
-max_parallel_paths <number> \
-max_paths <number> \
-print summary (yes|no) \
-use_slack_threshold (yes|no) \
-slack threshold <double> \
-print_paths (yes|no) \
-max_expanded_paths <number> \
-include_user_sets (yes|no) \
-include_clock_domains (yes|no) \
-select_clock_domains <clock name list> \
-limit_max_paths (yes|no) \
-include_pin_to_pin (yes|no) \
bottleneck options \
-cost_type (path_count|path_cost) \
-max_instances <number> \
-from <port/pin pattern> \
-to <port/pin pattern> \
-set_type <set_type> \
-set_name <set name> \
-clock <clock name> \
-from_clock <clock name> \
-to_clock <clock name> \
-in_to_out \
```

Arguments

Parameter	Type	Description
type	string	Specifies the type of the report to be generated. It is mandatory. • timing - Timing Report • violations - Timing Violation Report • datasheet - Datasheet Report • bottleneck - Bottleneck Report • constraints_coverage - Constraints Coverage Report • combinational_loops - Combinational Loops Report
analysis	string	Specifies the type of Analysis(Max Delay or Min Delay) Performed to generate the reports. It is optional. **Note:** This argument should not be used to generate datasheet reports. The command may fail if this argument is used to generate datasheet report. • max - Timing report considers maximum analysis (default). • min - Timing report considers minimum analysis.

.....continued

Parameter	Type	Description
format	string	Specifies the format in which the report is generated. It is optional. • text - Generates a text report (default). • csv - Generates the report in a comma-separated value format which you can import into a spreadsheet. **Note:** CDC type generates report in CSV format only.
filename	string	Specifies the file name of the generated report. It is mandatory.

Table 7-1. Timing Options and Values

Parameter/Value	Description
max_parallel_paths <number>	Specifies the max number of parallel paths. Parallel paths are timing paths with the same start and end points.
max_paths <number>	Specifies the max number of paths to display for each set. This value is a positive integer value greater than zero. Default is 100.
print_summary (yes|no)	Yes to include and No to exclude the summary section in the timing report.
use_slack_threshold (yes|no)	Yes to include slack threshold and no to exclude threshold in the timing report. The default is to exclude slack threshold.
slack_threshold <double>	Specifies the threshold value to consider when reporting path slacks. This value is in nanoseconds (ns). By default, there is no threshold (all slacks reported).
print_paths <yes|no>	Specifies whether the path section (clock domains and in-to-out paths) will be printed in the timing report. Yes to include path sections (default) and no to exclude path sections from the timing report.
max_expanded_paths <number>	Specifies the max number of paths to expand per set. This value is a positive integer value greater than zero. Default is 100.
include_user_sets (yes|no)	If yes, the user set is included in the timing report. If no, the user set is excluded in the timing report.
include_clock_domains (yes|no)	Yes to include and no to exclude clock domains in the timing report.
select_clock_domains <clock_name_list>	Defines the clock domain to be considered in the clock domain section. The domain list is a series of strings with domain names separated by spaces. Both the summary and the path sections in the timing report display only the listed clock domains in the clock_name_list.
limit_max_paths (yes|no)	Yes to limit the number of paths to report. No to specify that there is no limit to the number of paths to report (the default).
include_pin_to_pin (yes|no)	Yes to include and no to exclude pin-to-pin paths in the timing report.

Table 7-2. Bottleneck Options and Values

cost_type <path_count|path_cost>	Specifies the cost_type as either path_count or path_cost. For path_count, instances with the greatest number of path violations will have the highest bottleneck cost. For path_cost, instances with the largest combined timing violations will have the highest bottleneck cost.
max_instances <number>	Specifies the maximum number of instances to be reported. Default is 10.
from <port/pin pattern>	Reports only instances that lie on violating paths that start at locations specified by this option.
to <port/pin pattern>	Reports only instances that lie on violating paths that end at locations specified by this option.
clock <clock name>	This option allows pruning based on a given clock domain. Only instances that lie on these violating paths are reported.
set_name <set name>	Displays the bottleneck information for the named set. You can either use this option or use both -clock and -type. This option allows pruning based on a given set. Only paths that lie within the named set will be considered towards bottleneck.
set_type <set_type>	This option can only be used in combination with the -clock option, and not by itself. The options allows you to filter which type of paths should be considered towards the bottleneck: • reg_to_reg - Paths between registers in the design • async_to_reg - Paths from asynchronous pins to registers • reg_to_async - Paths from registers to asynchronous pins • external_recovery - The set of paths from inputs to asynchronous pins • external_removal - The set of paths from inputs to asynchronous pins • external_setup - Paths from input ports to registers • external_hold - Paths from input ports to registers • clock_to_out - Paths from registers to output ports
from_clock <clock name>	Reports only bottleneck instances that lie on violating timing paths of the inter-clock domain that starts at the source clock specified by this option. This option can only be used in combination with - to_clock.
to_clock <clock name>	Reports only instances that lie on violating paths that end at locations specified by this option.
in_to_out	Reports only instances that lie on violating paths that begin at input ports and end at output ports.

Return Type	Description
file	Generates SmartTime report file with the specified format.

Error Codes

Error Code	Description
None	Parameter -type has illegal value
None	Required parameter -type is missing
None	Required parameter _AtclParam0_ is missing

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example generates a timing violation report named timing_viol.txt. The report considers an analysis using maximum delays and does not filter paths based on slack threshold. It reports two paths per section and one expanded path per section:

```
report \
-type violations \
-analysis max \
-use_slack_threshold no \
-limit_max_paths yes \
-max_paths 2 \
-max_expanded_paths 1 \
 timing_viol.txt
```

The following example generates a datasheet report named datasheet.csv in CSV format:

```
report -type datasheet -format csv datasheet.csv
```

Related Examples on GitHub

- report

7.50 save

Description
This SmartTime-specific command saves all changes made prior to this command. This includes changes made on constraints, options, and sets.

```
save
```

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following script sets the maximum number of paths reported by list_paths to 10, reads an SDC file, and save both the option and the constraints into the design project.

```
set_options -limit_max_paths 10
read_sdc somefile.sdc
save
```

Related Examples on GitHub

- save

7.51 set_clock_groups

Description
This is an SDC command which disables timing analysis between the specified clock groups. No paths are reported between the clock groups in both directions. Paths between clocks in the same group continue to be reported.

Note: If you use the same name and the same exclusive flag of a previously defined clock group to create a new clock group, the previous clock group is removed and a new one is created in its place. The exclusive flags for the arguments above are all mutually exclusive. Only one can be specified.

```
set_clock_groups [-name name ] \
[-physically exclusive | -logically exclusive | -asynchronous] \
[-comment comment_string ] \
-group clock_list
```

Arguments

Parameter	Type	Description
name	string	Name given to the clock group. Optional.
physically_exclusive	None	Specifies that the clock groups are physically exclusive with respect to each other. Examples are multiple clocks feeding a register clock pin. The exclusive flags are all mutually exclusive. Only one can be specified.
logically_exclusive	None	Specifies that the clocks groups are logically exclusive with respect to each other. Examples are clocks passing through a mux.
asynchronous	None	Specifies that the clock groups are asynchronous with respect to each other, as there is no phase relationship between them. **Note:** The exclusive flags are all mutually exclusive. Only one can be specified.
group	list of strings	Specifies a list of clocks. There can any number of groups specified in the set_clock_groups command.

Return Type	Description
integer	Returns the ID of the clock group.

Error Codes

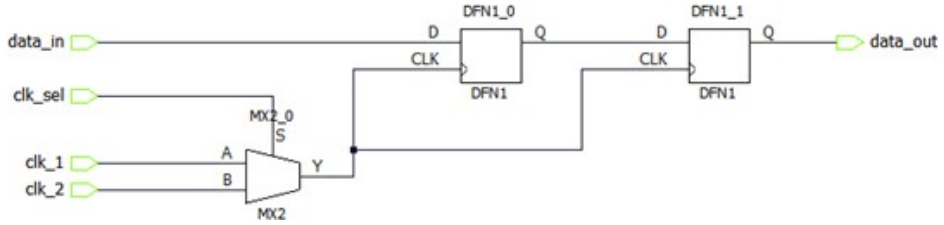
Error Code	Description
None	Invalid set_clock_groups constraint - only one of -physically_exclusive, -logically_exclusive or -asynchronou should be used.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

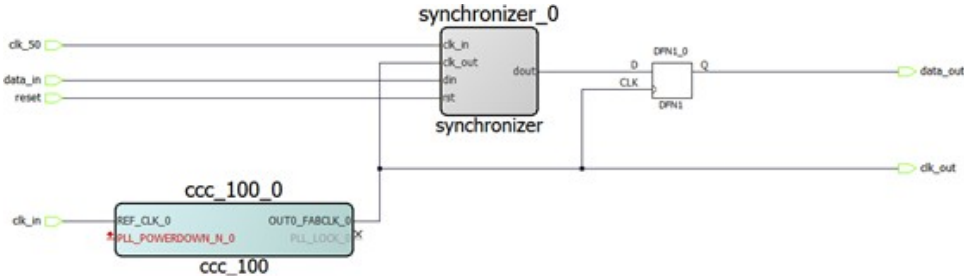
1. Here, there are three synchronous clocks receiving data from an asynchronous clock.



SDC:

```
create_clock -name clk_1 -period 5 [ get_ports clk_1 ]
create_clock -name clk_2 -period 10 [ get_ports clk_2 ]
set_clock_groups -logically_exclusive -group clk_1 -group clk_2
```

2. Here, there are three synchronous clocks receiving data from an asynchronous clock.



SDC:

```
create_clock -name clk_in -period 10 [ get_ports clk_in ]
create_clock -name clk_50 -period 20 [ get_ports clk_50 ]
create_generated_clock -name ccc_100 -divide_by 2 \
-source [ get_pins ccc_100_0/ccc_100_0/pll_inst_0/REF_CLK_0 ] \
[ get_pins ccc_100_0/ccc_100_0/pll_inst_0/OUT0 ] \
create_generated_clock -name clk_out -divide_by 1 \
-source [ get_pins { ccc_100_0/ccc_100_0/pll_inst_0/OUT0 } ] \
[ get_ports clk_out ]
set_clock_groups -asynchronous -group { clk_in ccc_100 clk_out } -group clk_50
```

Related Examples on GitHub

- set_clock_groups

See Also

- 7.17 list_clock_groups
- 7.35 remove_clock_groups

7.52 set_clock_latency

Description

This Tcl command defines the delay between an external clock source and the definition pin of a clock within SmartTime.

Clock source latency defines the delay between an external clock source and the definition pin of a clock within SmartTime. It behaves much like an input delay constraint. You can specify both an "early" delay and a "late" delay for this latency, providing an uncertainty which SmartTime propagates through its calculations. Rising and falling edges of the same clock can have different latencies. If only one value is provided for the clock source latency, it is taken as the exact latency value, for both rising and falling edges.

```
set_clock_latency -source [-rise] [-fall] [-early] [-late] delay clock
```

Arguments

Parameter	Type	Description
source	None	Specifies the source latency on a clock pin, potentially only on certain edges of the clock.
rise	None	Specifies the edge for which this constraint will apply. If neither or both rise and fall are passed, the same latency is applied to both edges.
fall	None	Specifies the edge for which this constraint will apply. If neither or both fall and rise are passed, the same latency is applied to both edges.
invert	None	Specifies that the generated clock waveform is inverted with respect to the reference clock.
late	None	Optional. Specifies that the latency is late bound on the latency. The appropriate bound is used to provide the most pessimistic timing scenario. However, if the value of `-late` is less than the value of `-early`, optimistic timing takes place which could result in incorrect analysis. If neither or both `-early` and `-late` are provided, the same latency is used for both bounds, which results in the latency having no effect for single clock domain setup and hold checks.
early	None	Optional. Specifies that the latency is early bound on the latency. The appropriate bound is used to provide the most pessimistic timing scenario. However, if the value of `-late` is less than the value of `-early`, optimistic timing takes place which could result in incorrect analysis. If neither or both `-early` and `-late` are provided, the same latency is used for both bounds, which results in the latency having no effect for single clock domain setup and hold checks.
delay	floating point	Specifies the latency value for the constraint.

.....continued		
Parameter	**Type**	**Description**
clock	string	Specifies the clock to which the constraint is applied. This clock must be constrained.

Error Codes

Error Code	Description
Error: SDC0061	Invalid clock latency constraint: Parameter has illegal value invoked from within command.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example sets an early clock source latency of 0.4 on the rising edge of main_clock. It also sets a clock source latency of 1.2, for both the early and late values of the falling edge of main_clock. The late value for the clock source latency for the falling edge of main_clock remains undefined.

```
set_clock_latency -source -rise -early 0.4 { main_clock }
```

```
set_clock_latency -source -fall 1.2 { main_clock }
```

Related Examples on GitHub
- set_clock_latency

See Also
- 7.6 create_clock
- 7.7 create_generated_clock

7.53 set_clock_to_output

Description
This SDC command defines the timing budget available inside the FPGA for an output relative to a clock.

```
set_clock_to_output delay_value -clock clock_ref [-max] [-min] output_list
```

Arguments

Parameter	Type	Description
delay_value	integer	Specifies the clock to output delay in nanoseconds. This time represents the amount of time available inside the FPGA between the active clock edge and the data change at the output port.
clock	string	Specifies the reference clock to which the specified clock to output is related. This is a mandatory argument.
max	None	Specifies that delay_value refers to the maximum clock to output at the specified output. If you do not specify `-max` or `-min` options, the tool assumes maximum and minimum clock to output delays to be equal.
min	None	Specifies that delay_value refers to the minimum clock to output at the specified output. If you do not specify `-max` or `-min` options, the tool assumes maximum and minimum clock to output delays to be equal.
output_list	list of strings	Provides a list of output ports in the current design to which `delay_value` is assigned. If you need to specify more than one object, enclose the objects in braces ({}).

Error Codes

Error Code	Description
None	Required parameter -clock is missing

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example sets an output delay of 0.3 ns for port Q relative to the CLK1 clock.

```
set_clock_to_output -max 0.3 -clock { clk } [ get_ports { Q } ]
```

Related Examples on GitHub

- set_clock_to_output

7.54 set_clock_uncertainty

Description
This Tcl command specifies simple clock uncertainty for single clock and clock-to-clock uncertainty between two clocks (from and to).

The set_clock_uncertainty command sets the timing uncertainty of clock networks. It can be used to model clock jitter or add guard band in timing analysis. Either simple clock uncertainty or clock-to-clock uncertainty can be specified. Simple clock uncertainty can be set on a clock or on any pin in the clock network. It will then apply to any path with the capturing register in the forward cone of the uncertainty. If multiple simple uncertainty applies to a register, the last one (in the propagation order from the clock source to the register) is used. Clock-to-clock uncertainty applies to inter-clock paths. Both “from” clock and “to” clock must be specified. Clock-to-clock uncertainty has higher priority than simple uncertainty. If both are set (a clock-to-clock uncertainty and a simple clock uncertainty on the “to” clock), the simple clock uncertainty will be ignored for inter-clock paths, only the clock-to-clock uncertainty will be used.

```
set_clock_uncertainty [-setup] [-hold] uncertainty [object_list -from from_clock |
-rise_from rise_from_clock | -fall_from fall_from_clock -to to_clock | -rise_to rise_to_clock
|
-fall_to fall_to_clock ]
```

Arguments

Parameter	Type	Description
uncertainty	floating point	Specifies the time in nanoseconds that represents the amount of variation between two clock edges.
object_list	list of strings	Specifies a list of clocks, ports, or pins for simple uncertainty; the uncertainty is applied either to destination flops clocked by one of the clocks in the object list option, or destination flops whose clock pins are in the fanout of a port or a pin specified in the object_list option.
from	list of strings	Specifies that the clock-to-clock uncertainty applies to both rising and falling edges of the source clock list. Only one of the -from, -rise_from, or -fall_from arguments can be specified for the constraint to be valid.
rise_from	list of strings	Specifies that the clock-to-clock uncertainty applies only to rising edges of the source clock list. Only one of the -from, -rise_from, or -fall_from arguments can be specified for the constraint to be valid.
fall_from	list of strings	Specifies that the clock-to-clock uncertainty applies only to falling edges of source clock list. Only one of the -from, -rise_from, or -fall_from arguments can be specified for the constraint to be valid.
from_clock/rise_from_clock/ fall_from_clock	list of strings	Specifies the list of clock names as the uncertainty source.
to	list of strings	Specifies that the clock-to-clock uncertainty applies to both rising and falling edges of the destination clock list. Only one of the -to, -rise_to, or -fall_to arguments can be specified for the constraint to be valid.
rise_to	list of strings	Specifies that the clock-to-clock uncertainty applies only to rising edges of the destination clock list. Only one of the -to, -rise_to, or -fall_to arguments can be specified for the constraint to be valid.
fall_to	list of strings	Specifies that the clock-to-clock uncertainty applies only to falling edges of the destination clock list. Only one of the -to, -rise_to, or -fall_to arguments can be specified for the constraint to be valid.

.....continued

Parameter	Type	Description
to_clock/rise_to_clock/ fall_to_clock	list of strings	Specifies the list of clock names as the uncertainty destination.
setup	None	Specifies that the uncertainty applies only to setup checks. If none or both `-setup` and `-hold` are present, the uncertainty applies to both setup and hold checks.
hold	None	Specifies that the uncertainty applies only to hold checks. If none or both `-setup` and `-hold` are present, the uncertainty applies to both setup and hold checks.

Return Type	Description
integer	Returns the ID of the clock uncertainty constraint.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

Simple Clock Uncertainty constraint examples.

The following example specifies uncertainty of 2 ns.

```
set_clock_uncertainty 2 [get_clocks clk]
```

The following example specifies setup uncertainty of 2 ns.

```
set_clock_uncertainty 2 -setup [get_clocks clk]
```

Clock to Clock Uncertainty constraint examples:

The following example specifies uncertainties of 10ns between Clk1 and Clk2 clock domains.

```
set_clock_uncertainty 10 -from [get_clocks { Clk1 }] -to [get_clocks { Clk2 }]
```

The following example specifies setup uncertainties between Clk1 and {Clk2 Clk3} clock domains with specific edges.

```
set_clock_uncertainty 0 -from [get_clocks { Clk1 }] -fall_to [get_clocks { Clk2 Clk3 }] -setup
```

```
set_clock_uncertainty 4.3 -fall_from [get_clocks { Clk1 Clk2 }] -rise_to *
```

```
set_clock_uncertainty 0.1 -rise_from [ get_clocks { Clk1 Clk2 }] \
-fall_to [get_clocks { Clk3 Clk4 }] -setup
```

```
set_clock_uncertainty 5 -rise_from [get_clocks {Clk1}]  -to [ get_clocks {*} ]
```

Related Examples on GitHub

- set_clock_uncertainty

See Also

- 7.19 list_clock_uncertainties
- 7.37 remove_clock_uncertainty

7.55 set_current_scenario

Description

This Tcl command specifies the timing scenario for the Timing Analyzer to use. All commands that follow this command will apply to the specified timing scenario. A timing scenario is a set of timing constraints used with a design. If the specified scenario is already the current one, this command has no effect.

After setting the current scenario, constraints can be listed, added, or removed, the checker can be invoked on the set of constraints, and so on.

This command uses the specified timing scenario to compute timing analysis.

Note: It is recommended to use the organize_tool_files command instead of this command.

```
set_current_scenario name
```

Arguments

Parameter	Type	Description
name	string	Specifies the name of the timing scenario to which to apply all commands from this point on.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

The following command sets "scenario_A" as current scenario of the timing scenario.

```
set_current_scenario scenario_A
```

Related Examples on GitHub

- set_current_scenario

See Also

- 7.8 create_scenario
- 7.13 get_current_scenario
- 7.46 remove_scenario

• 7.48 rename_scenario

7.56 set_disable_timing

Description
This Tcl command disables timing arcs within a cell(instance) and returns the ID of the created constraint if the command succeeded. To specify a Disable Timing constraint, open the Set Constraint to Disable Timing Arcs dialog box in the following way: Choose Disable Timing from the Constraints drop-down menu (Constraints > Disable Timing).
Note: This constraint is for the Place and Route tool and the Verify Timing tool. It is ignored by the Synthesis tool.

```
set_disable_timing -from value -to value name
```

Arguments

Parameter	Type	Description
from	string	Specifies the starting point for the timing arc. The `-from` and `-to` arguments must either both be present or both omitted for the constraint to be valid.
to	string	Specifies the ending point for the timing arc. The `-from` and `-to` arguments must either both be present or both omitted for the constraint to be valid.
name	string	Specifies the instance(cell) name for which the disable timing arc constraint will be created.

Return Type	Description
integer	Returns the ID of created constraint.

Error Codes

Error Code	Description
None	Required parameter _AtclParam0_ is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example disables timing arcs within A to Y.

```
set_disable_timing -from A -to Y a2
```

Related Examples on GitHub

- set_disable_timing

See Also

- 7.38 remove_disable_timing

7.57 set_external_check

Description

This SDC command defines the external setup and hold delays for an input relative to a clock.

```
set_external_check delay_value -clock clock_ref [-setup] [-hold] input_list
```

Arguments

Parameter	Type	Description
delay_value	integer	Specifies the clock to output delay in nanoseconds. This time represents the amount of time available inside the FPGA between the active clock edge and the data change at the output port.
clock	string	Specifies the reference clock to which the specified clock to output is related. This is a mandatory argument.
setup or hold	None	Specifies that delay_value refers to the setup/hold check at the specified input. This is a mandatory argument if `-hold` is not used. You must specify either the `-setup` or `-hold` option.
input_list	list of strings	Provides a list of input ports in the current design to which `delay_value` is assigned. If you need to specify more than one object, enclose the objects in braces ({}).

Return Type	Description
integer	Returns the ID of the external setup and hold constraint.

Error Codes

Error Code	Description
None	Required parameter -clock is missing.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example sets an external setup delay with 0.2 ns for A input port.

```
set_external_check -setup 0.2 -clock { clk } [ get_ports { A } ]
```

Related Examples on GitHub
- set_external_check

See Also
- 7.59 set_input_delay

7.58 set_false_path

Description
This Tcl command identifies paths that are considered false and excluded from the timing analysis in the current timing scenario. The `set_false_path` command identifies specific timing paths as being false. The false timing paths are paths that do not propagate logic level changes. This constraint removes timing requirements on these false paths so that they are not considered during the timing analysis. The path starting points are the input ports or register clock pins, and the path ending points are the register data pins or output ports. This constraint disables setup and hold checking for the specified paths.

The false path information always takes precedence over multiple cycle path information and overrides maximum delay constraints. If more than one object is specified within one `-through` option, the path can pass through any objects.

You must specify at least one of the `-from`, `-to`, or `-through` arguments for this constraint to be valid.

```
set_false_path [-ignore_errors] [-from from_list ] [-through through_list ] [-to to_list ]
```

Arguments

Parameter	Type	Description
ignore_errors	None	Specifies to avoid reporting errors for derived constraints targeting the logic that becomes invalid due to logic optimization. It is an optional argument. Some IPs may have extra logic present depending on other IPs used in the design but the synthesis tool will remove this logic if fewer IPs were used. In such cases, the implementation flow will halt without `-ignore_errors` flag. **Note:** It is not recommended to use this flag outside similar use cases.
from	list of strings	Specifies a list of timing paths starting points. A valid timing starting point is a clock, a primary input, an inout port, or a clock pin of a sequential cell.
through	list of strings	Specifies a list of pins or nets through which the disabled paths must pass.
to	list of strings	Specifies a list of timing paths ending points. A valid timing ending point is a clock, a primary output, an inout port, or a data pin of a sequential cell.

Return Type	Description
None	None

Error Codes

Error Code	Description
Error: SDC0021	Invalid false path constraint: the `-from` value is incorrect.
Error: SDC0022	Invalid false path constraint: the `-from` is empty.
Error: SDC0024	Invalid false path constraint: the `-to` is empty.
Error: SDC0026	Invalid false path constraint: the `-through` is empty.
Warning:	cell (`get_cells`) is incorrect type; `-through` objects must be of type net (`get_nets`), or pin (`get_pins`). **Note:** Constraint will be disabled.
Warning:	port (`get_ports`) is incorrect type; `-through` objects must be of type net (`get_nets`), or pin (`get_pins`). **Note:** Constraint will be disabled.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

The following example specifies all paths from clock pins of the registers in clock domain `clk1` to data pins of a specific register in clock domain `clk2` as false paths.

```
set_false_path -from [get_clocks {clk1}] -to reg_2:D
```

The following example specifies all paths through the pin U0/U1:Y to be false.

```
set_false_path -through U0/U1:Y
```

The following example specifies a derived false path constraint through the `PCIe_Demo_0/SYSRESET_POR/POWER_ON_RESET_N` pin.

```
set_false_path -ignore_errors -through [ get_pins {PCIe_Demo_0/SYSRESET_POR/
POWER_ON_RESET_N } ]
```

Related Examples on GitHub

- set_false_path

See Also

- 7.39 remove_false_path

7.59 set_input_delay

Description
This Tcl command creates an input delay on a port list by defining the arrival time of an input relative to a clock in the current scenario.

The `set_input_delay` command sets input path delays on input ports relative to a clock edge. This usually represents a combinational path delay from the clock pin of a register external to the current design. For in/out (bidirectional) ports, you can specify the path delays for both input and output modes. The tool adds input delay to path delay for paths starting at primary inputs.

A clock is a singleton that represents the name of a defined clock constraint. This can be:

- a single port name used as source for a clock constraint.
- a single pin name used as source for a clock constraint; for instance `reg1:CLK`. This name can be hierarchical (for instance, `toplevel/block1/reg2:CLK`).
- an object accessor that will refer to one clock: `[get_clocks {clk}]`.

Notes:

- The behavior of the -add_delay option is identical to that of PrimeTime(TM).
- If, using the `-add_delay` mechanism, multiple constraints are otherwise identical, except they specify different `-max` or `-min` values
  - the surviving `-max` constraint will be the maximum of the `-max` values.
  - the surviving `-min` constraint will be the minimum of the `-min` values.

```
set_input_delay delay_value -clock clock_ref [-max] [-min] [-clock_fall] [-rise] [-fall] [-
add_delay] \
input_list
```

Arguments

Parameter	Type	Description
delay_value	real	Specifies the arrival time in nanoseconds that represents the amount of time for which the signal is available at the specified input after a clock edge.
clock	string	Specifies the clock reference to which the specified input delay is related. This is a mandatory argument. If you do not specify `-max` or `-min` options, the tool assumes the maximum and minimum input delays to be equal.
max	None	Specifies that the `delay_value` refers to the longest path arriving at the specified input. If you do not specify `-max` or `-min` options, the tool assumes maximum and minimum input delays to be equal.
min	None	Specifies that the `delay_value` refers to the shortest path arriving at the specified input. If you do not specify `-max` or `-min` options, the tool assumes maximum and minimum input delays to be equal.
clock_fall	None	Specifies that the delay is relative to the falling edge of the clock reference. The default is the rising edge.
rise	None	Specifies that the delay is relative to a rising transition on the specified port(s). If `-rise` or `-fall` is not specified, then rising and falling delays are assumed to be equal.

.....continued		
Parameter	**Type**	**Description**
fall	None	Specifies that the delay is relative to a falling transition on the specified port(s). If `-rise` or `-fall` is not specified, then rising and falling delays are assumed to be equal.
add_delay	None	Specifies that this input delay constraint should be added to an existing constraint on the same port(s). The `-add_delay` option is used to capture information on multiple paths with different clocks or clock edges leading to the same input port(s).
input_list	list of string	Provides a list of input ports in the current design to which `delay_value` is assigned. If you need to specify more than one object, enclose the objects in braces ({}).

Return Type	Description
integer	Returns the ID of the clock input delay constraint.

Error Codes

Error Code	Description
Error: SDC0004	clk does not match any clock name or source.
Error: SDC0015	port list [get_ports {CLK_0_D}] is incorrect.
Error: SDC0054	Invalid IO delay constraint: the min delay is greater than max delay.
Error: SDC0061	Parameter _AtclParam0_ has illegal value.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example sets an input delay of 1.2ns for port data1 relative to the rising edge of `CLK1`.

```
set_input_delay 1.2 -clock [get_clocks CLK1] [get_ports data1]
```

The following example sets a different maximum and minimum input delay for port IN1 relative to the falling edge of `CLK2`.

```
set_input_delay 1.0 -clock_fall -clock CLK2 -min {IN1}
```

```
set_input_delay 1.4 -clock_fall -clock CLK2 -max {IN1}
```

The following example demonstrates an override condition of two constraints. The first constraint is overridden because the second constraint specifies a different clock for the same input.

```
set_input_delay 1.0 -clock CLK1 -max {IN1}
```

```
set_input_delay 1.4 -clock CLK2 -max {IN1}
```

The next example is almost the same as the previous one, however, in this case, the user has specified `-add_delay`, so both constraints will be honored.

```
set_input_delay 1.0 -clock CLK1 -max {IN1}
```

```
set_input_delay 1.4 -add_delay -clock CLK2 -max {IN1}
```

The following example is more complex:

- All constraints are for an input to port PAD1 relative to a rising edge clock CLK2. Each combination of `{-rise, -fall} x {-max, -min}` generates an independent constraint. But the max rise delay of 5 and the max rise delay of 7 interfere with each other.
- For a -max option, the maximum value overrides all lower values. Thus the first constraint will be overridden and the max rise delay of 7 will survive.

```
set_input_delay 5 -max -rise -add_delay [get_clocks CLK2] [get_ports PAD1] # will be
overridden
```

```
set_input_delay 3 -min -fall -add_delay [get_clocks CLK2] [get_ports PAD1]
```

```
set_input_delay 3 -max -fall -add_delay [get_clocks CLK2] [get_ports PAD1]
```

```
set_input_delay 7 -max -rise -add_delay [get_clocks CLK2] [get_ports PAD1]
```

Related Examples on GitHub

- set_input_delay

See Also

- 7.64 set_output_delay
- 7.41 remove_input_delay
- 7.45 remove_output_delay

7.60 set_max_delay

Description

This Tcl command specifies the required maximum delay for timing paths in the current design. The path length for any startpoint in `from_list` to any endpoint in `to_list` must be less than the `delay_value`. The timing engine automatically derives the individual maximum delay targets from clock waveforms and port input or output delays. For more information, refer to the `create_clock`, `set_input_delay`, and `set_output_delay` commands. The maximum delay constraint is a timing exception. This constraint overrides the default single cycle timing relationship for one or more timing paths. This constraint also overrides a multi-cycle path constraint.

You must specify at least one of the `-from` , `-to`, or `-through` arguments for this constraint to be valid.

```
set_max_delay delay_value [-from from_list ] [-to to_list ] [-through through_list ]
```

Arguments

Parameter	Type	Description
delay_value	floating point	Specifies a floating point number in nanoseconds that represents the required maximum delay value for specified paths. • If the path starting point is on a sequential device, the tool includes clock skew in the computed delay. • If the path starting point has an input delay specified, the tool adds that delay value to the path delay. • If the path ending point is on a sequential device, the tool includes clock skew and library setup time in the computed delay. • If the ending point has an output delay specified, the tool adds that delay to the path delay.
from	list of strings	Specifies a list of timing path starting points. A valid timing starting point is a clock, a primary input, an inout port, or a clock pin of a sequential cell.
to	list of strings	Specifies a list of timing path ending points. A valid timing ending point is a clock, a primary output, an inout port, or a data pin of a sequential cell.
through	list of strings	Specifies a list of pins or nets through which the timing paths must pass.

Return Type	Description
integer	Returns the ID of the clock maximum delay constraint.

Error Codes

Error Code	Description
Error: SDC0021	Invalid max delay constraint: the `-from` value is incorrect.
Error: SDC0022	Invalid max delay constraint: the `-from` is empty.
Error: SDC0023	Invalid max delay constraint: the `-to` value is incorrect.
Error: SDC0024	Invalid max delay constraint: the `-to` is empty.
Error: SDC0026	Invalid max delay constraint: the `-through` is empty
Error: SDC0061	Invalid max delay constraint: Missing or Illegal parameter/value.
Warning	cell (get_cells) is incorrect type;"`-through`" objects must be of type net (`get_nets`), or pin (`get_pins`). **Note:** Constraint will be disabled.
Warning	port (get_ports) is incorrect type;"`-through`" objects must be of type net (`get_nets`), or pin (`get_pins`). **Note:** Constraint will be disabled.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example sets a maximum delay by constraining all paths from `ff1a:CLK or ff1b:CLK` to `ff2e:D` with a delay less than 5 ns.

```
set_max_delay 5 -from {ff1a:CLK ff1b:CLK} -to {ff2e:D}
```

The following example sets a maximum delay by constraining all paths to output ports whose names start by "out" with a delay less than 3.8 ns.

```
set_max_delay 3.8 -to [get_ports out*]
```

Related Examples on GitHub
- set_max_delay

See Also
- 7.60 set_max_delay
- 7.42 remove_max_delay

7.61 set_min_delay

Description
This Tcl command specifies the required minimum delay for timing paths in the current design. The path length for any startpoint in `from_list` to any endpoint in `to_list` must be less than the `delay_value`. The timing engine automatically derives the individual minimum delay targets from clock waveforms and port input or output delays. For more information, refer to the `create_clock`, `set_input_delay`, and `set_output_delay`commands. The minimum delay constraint is a timing exception. This constraint overrides the default single cycle timing relationship for one or more timing paths. This constraint also overrides a multi-cycle path constraint.

You must specify at least one of the `-from`, `-to`, or `-through` arguments for this constraint to be valid.

```
set_min_delay delay_value [-from from_list ] [-to to_list ] [-through through_list ]
```

Arguments

Parameter	Type	Description
delay_value	floating point	Specifies a floating point number in nanoseconds that represents the required minimum delay value for specified paths. • If the path starting point is on a sequential device, the tool includes clock skew in the computed delay. • If the path starting point has an input delay specified, the tool adds that delay value to the path delay. • If the path ending point is on a sequential device, the tool includes clock skew and library setup time in the computed delay. • If the ending point has an output delay specified, the tool adds that delay to the path delay.
from	list of strings	Specifies a list of timing path starting points. A valid timing starting point is a clock, a primary input, an inout port, or a clock pin of a sequential cell.
to	list of strings	Specifies a list of timing path ending points. A valid timing ending point is a clock, a primary output, an inout port, or a data pin of a sequential cell.
through	list of string	Specifies a list of pins or nets through which the timing paths must pass.

Return Type	Description
integer	Returns the ID of the clock minimum delay constraint.

Error Codes

Error Code	Description
Error: SDC0021	Invalid min delay constraint: the `-from` value is incorrect.
Error: SDC0022	Invalid min delay constraint: the `-from` is empty.
Error: SDC0023	Invalid min delay constraint: the `-to` value is incorrect.
Error: SDC0024	Invalid min delay constraint: the `-to` is empty.
Error: SDC0026	Invalid min delay constraint: the `-through` is empty.
Error: SDC0061	Invalid min delay constraint: Missing or Illegal parameter/value.
Warning	port (get_ports) is incorrect type;"`-through`" objects must be of type net (`get_nets`), or pin (`get_pins`).

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following example sets a minimum delay by constraining all paths from `ff1a:CLK or ff1b:CLK to ff2e:D` with a delay less than 5 ns.

```
set_min_delay 5 -from {ff1a:CLK ff1b:CLK} -to {ff2e:D}
```

The following example sets a minimum delay by constraining all paths to output ports whose names start by "out" with a delay less than 3.8 ns.

```
set_min_delay 3.8 -to [get_ports out*]
```

Related Examples on GitHub
- set_min_delay

See Also
- 7.61 set_min_delay
- 7.42 remove_max_delay

7.62 set_multicycle_path

Description
Defines a path that takes multiple clock cycles in the current scenario. Setting multiple cycle paths constraint overrides the single cycle timing relationships between sequential elements by specifying the number of cycles that the data path must have for setup or hold checks. If you change the multiplier, it affects both the setup and hold checks.

False path information always takes precedence over multiple cycle path information. A specific maximum delay constraint overrides a general multiple cycle path constraint. If you specify more than one object within one `-through` option, the path passes through any of the objects.

You must specify at least one of the `-from` , `-to`, or `-through` arguments for this constraint to be valid.

```
set_multicycle_path ncycles [-setup] [-hold] [-setup_only] [-from from_list] \
[-through through_list ] [-to to_list]
```

Arguments

Parameter	Type	Description
ncycles	integer	Specifies an integer value that represents a number of cycles the data path must have for setup or hold check. The value is relative to the starting point or ending point clock, before data is required at the ending point. Number of cycles must be greater than `1`. If you set ncycles as `2.2` or `4/2` or `"8a"` then it is being truncated as `2` or `4` or `8`, and no warning is reported.
setup	None	Optional. Applies the cycle value for the setup check only. This option does not affect the hold check. The default hold check will be applied unless you have specified another set_multicycle_path command for the hold value.
hold	None	Optional. Applies the cycle value for the hold check only. This option does not affect the setup check. **Note:** If you do not specify `-setup` or `-hold`, the cycle value is applied to the setup check and the default hold check is 0 not ncycles -1 .

.....continued		
Parameter	**Type**	**Description**
setup_only	None	Optional. Specifies that the path multiplier is applied to setup paths only. The default value for hold check (which is 0) is applied.
from	list of strings	Specifies a list of timing path starting points. A valid timing starting point is a clock, a primary input, an inout port, or a clock pin of a sequential cell.
through	list of strings	Specifies a list of pins or nets through which the multiple cycle paths must pass.
to	list of strings	Specifies a list of timing path ending points. A valid timing ending point is a clock, a primary output, an inout port, or a data pin of a sequential cell.

Error Codes

Error Code	Description
Error: SDC0004	clk does not match any clock name or source.
Error: SDC0015	port list [get_ports { CLK_0_D }] is incorrect.
Error: SDC0054	Invalid IO delay constraint: the min delay is greater than max delay.
Error: SDC0061	Parameter _AtclParam0_ has illegal value.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Exceptions
Multiple priority management is not supported in Microchip SoC designs. All multiple cycle path constraints are handled with the same priority.

Example
The following example sets all paths between reg1 and reg2 to 3 cycles for setup check. Hold check is measured at the previous edge of the clock at reg2.

```
set_multicycle_path 3 -from [get_pins {reg1}] -to [get_pins {reg2}]
```

The following example specifies that four cycles are needed for setup check on all paths starting at the registers in the clock domain ck1. Hold check is further specified with two cycles instead of the three cycles that would have been applied otherwise.

```
set_multicycle_path 4 -setup -from [get_clocks {ck1}]
```

```
set_multicycle_path 2 -hold -from [get_clocks {ck1}]
```

The following example specifies that four cycles are needed for setup only check on all paths starting at the registers in the clock domain REF_CLK_0.

```
set_multicycle_path -setup_only 4 -from [ get_clocks { REF_CLK_0 } ]
```

Related Examples on GitHub

- set_multicycle_path

See Also

- 7.44 remove_multicycle_path

7.63 set_options

Description

This SmartTime-specific Tcl command sets options for timing analysis which can be changed in the SmartTime Options dialog box in the SmartTime GUI. All of the options from SmartTime are passed on to place-and-route tool, and some affect timing-driven place-and-route.

```
set_options \
[-max_opcond value ] \
[-min_opcond value ] \
[-interclockdomain_analysis value ] \
[-use_bibuf_loopbacks value ] \
[-enable_recovery_removal_checks value ] \
[-break_at_async value ] \
[-filter_when_slack_below value ] \
[-filter_when_slack_above value ] \
[-remove_slack_filters] \
[-limit_max_paths value ] \
[-expand_clock_network value ] \
[-expand_parallel_paths value ] \
[-analysis_scenario value ] \
[-tdpr_scenario value ] \
[-reset]
```

Arguments

Parameter	Type	Description
max_opcond	string	Sets the operating condition to use for Maximum Delay Analysis. The acceptable values for max_opcond for PolarFire can be the following: • slow_lv_ht - use slow_lv_ht conditions for maximum delay analysis • slow_lv_lt - use slow_lv_lt conditions for maximum delay analysis • fast_hv_lt - use fast_hv_lt conditions for maximum delay analysis Default is slow_lv_lt. max_opcond for SmartFusion2, IGLOO2 and RTG4 can be as following: • worst - use worst case conditions for maximum delay analysis • typical - use typical conditions for maximum delay analysis • best - use best case conditions for maximum delay analysis Default is worst.
min_opcond	string	Sets the operating condition to use for Minimum Delay Analysis. The acceptable values for min_opcond for PolarFire can be the following: • slow_lv_ht - use slow_lv_ht conditions for minimum delay analysis • slow_lv_lt - use slow_lv_lt conditions for minimum delay analysis • fast_hv_lt - use fast_hv_lt conditions for minimum delay analysis Default is fast_hv_lt. min_opcond for SmartFusion2, IGLOO2 and RTG4 can be as following: • worst - use worst case conditions for minimum delay analysis • typical - use typical conditions for minimum delay analysis • best - use best case conditions for minimum delay analysis Default is best.
interclockdomain_analysis	string	Enables or disables inter-clock domain analysis. Value can be the following: • yes - enables inter-clock domain analysis • no - disables inter-clock domain analysis Default is no. Timing-driven place-and-route is affected by this option.

.....continued

Parameter	Type	Description
use_bibuf_loopbacks	string	Instructs the timing analysis whether to consider loopback path in bidirectional buffers (D->Y, E->Y) as false-path {no}. Default is no; i.e., loopback are false paths. Values can be the following: • yes - enables loopback in bibufs • no - disables loopback in bibufs
enable_recovery_removal_checks	string	Enables recovery checks to be included in max-delay analysis and removal checks in min-delay analysis. Default is no. Values can be the following: • yes - enables recovery an removal checks • no - disables recovery and removal checks
break_at_async	string	Specifies whether or not timing analysis is allowed to cross asynchronous pins (clear, reset of sequential elements). Default is yes. Values can be the following: • yes - enables breaking paths at asynchronous ports • no - disables breaking paths at asynchronous ports. Timing-driven place-and-route is affected by this option.
filter_when_slack_below	floating point	Specifies a minimum slack value for paths reported by list_paths. Not set by default.
filter_when_slack_above	floating point	Specifies a maximum slack value for paths reported by list_paths. Not set by default.
remove_slack_filters	None	Removes the slack minimum and maximum set using `-filter_when_slack_below` and `-filter_when_slack_above`.
limit_max_paths	integer	Specifies the maximum number of paths reported by list_paths. Default is 20. Number must be greater than 0.
expand_clock_network	string	Specify whether or not clock network details are reported in expand_path. Default is yes. Values can be the following: • yes - enables expanded clock network information in paths • no - disables expanded clock network information in paths.
expand_parallel_paths	integer	Specify the number of parallel paths {paths with the same ends} to include in expand_path. Default is 1. Number must be greater than 0.
analysis_scenario	string	Specify the constraint scenario to be used for timing analysis. Default scenario is Primary.
tdpr_scenario	string	Specify the constraint scenario to be used for timing-driven place-and-route. Default scenario is Primary. Timing-driven place-and-route is affected by this option.
reset	None	Reset all options to the default values, except those for analysis and TDPR scenarios, which remain unchanged.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+

.....continued	
Supported Families	**Supported Libero SoC Versions**
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following script commands the timing engine to use best operating conditions for both max-delay analysis and min-delay analysis:

```
set_options -max_opcond {best} -min_opcond {best}
```

```
set_options -max_opcond {fast_hv_lt} -min_opcond {fast_hv_lt}
```

The following script changes the scenario used by timing-driven place-and-route and saves the change in the Libero project for place-and-route tools to see the change.

```
set_options -tdpr_scenario {My_TDPR_Scenario}
```

Related Examples on GitHub
- set_options

7.64 set_output_delay

Description
This Tcl command defines the output delay of an output relative to a clock in the current scenario.

The `set_output_delay` command sets output path delays on output ports relative to a clock edge. Output ports have no output delay unless you specify it. For in/out (bidirectional) ports, you can specify the path delays for both input and output modes. The tool adds output delay to path delay for paths ending at primary outputs.

Notes:
- The behavior of the -add_delay option is identical to that of PrimeTime(TM).
- If, using the `-add_delay` mechanism, multiple constraints are otherwise identical, except they specify different `-max` or `-min` values.
  - the surviving `-max` constraint will be the maximum of the `-max` values.
  - the surviving `-min` constraint will be the minimum of the `-min` values.

```
set_output_delay [-max] [-min] delay_value -clock clock_ref [-clock_fall] [-rise] [-fall] \
[-add_delay] output_list
```

Arguments

Parameter	Type	Description
delay_value	float	Specifies the amount of time before a clock edge for which the signal is required. This represents a combinational path delay to a register outside the current design plus the library setup time (for maximum output delay) or hold time (for minimum output delay).

.....continued

Parameter	Type	Description
clock	string	Specifies the clock reference to which the specified output delay is related. This is a mandatory argument.
max	None	Specifies that `delay_value` refers to the longest path from the specified output. If you do not specify `-max` or `-min` options, the tool assumes the maximum and minimum output delays to be equal.
min	None	Specifies that `delay_value` refers to the shortest path from the specified output. If you do not specify `-max` or `-min` options, the tool assumes the maximum and minimum output delays to be equal.
clock_fall	None	Specifies that the delay is relative to the falling edge of the clock reference. The default is the rising edge.
rise	None	Specifies that the delay is relative to a rising transition on the specified port(s). If `-rise` or `-fall` is not specified, then rising and falling delays are assumed to be equal.
fall	None	Specifies that the delay is relative to a falling transition on the specified port(s). If `-rise` or `-fall` is not specified, then rising and falling delays are assumed to be equal.
add_delay	None	Specifies that this output delay constraint should be added to an existing constraint on the same port(s). The `-add_delay` option is used to capture information on multiple paths with different clocks or clock edges leading to the same output port(s). **Notes:** • The behavior of the `-add_delay` option is identical to that of PrimeTime(TM). • If, using the `-add_delay` mechanism, multiple commands are otherwise identical, except they specify different `-max` or `-min` values. • the surviving `-max` constraint will be the maximum of the `-max` values. • the surviving `-min` constraint will be the minimum of the `-min` values.
output_list	list of string	Provides a list of output ports in the current design to which `delay_value` is assigned. If you need to specify more than one object, enclose the objects in braces ({}).

Return Type	Description
integer	Returns the ID of the clock output delay constraint.

Error Codes

Error Code	Description
Error: SDC0004	Invalid output delay constraint: clk does not match any clock name or source.
Error: SDC0015	Invalid output delay constraint: port list is incorrect.
Error: SDC0054	Invalid IO delay constraint: the min delay is greater than max delay.

.....continued	
Error Code	**Description**
Error: SDC0061	Invalid output delay constraint: Missing or Illegal parameter/value.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example

The following example sets an output delay of 1.2ns for port OUT1 relative to the rising edge of CLK1.

```
set_output_delay 1.2 -clock [get_clocks CLK1] [get_ports OUT1]
```

The following example sets a different maximum and minimum output delay for port OUT1 relative to the falling edge of CLK2.

```
set_output_delay -min {OUT1} 1.0 -clock_fall -clock CLK2
set_output_delay -max {OUT1} 1.4 -clock_fall -clock CLK2
```

The following example demonstrates an override condition of two constraints. The first constraint is overridden because the second constraint specifies a different clock for the same output.

```
set_output_delay 1.0 {OUT1} -clock CLK1 -max
set_output_delay 1.4 {OUT1} -clock CLK2 -max
```

The next example is almost the same as the previous one, however, in this case, the user has specified `-add_delay`, so both constraints will be honored.

```
set_output_delay 1.0 {OUT1} -clock CLK1 -max
set_output_delay 1.4 {OUT1} -add_delay -clock CLK2 -max
```

The following example is more complex:

- All constraints are for an output to port PAD1 relative to a rising edge clock CLK2. Each combination of {-rise, -fall} x {-max, -min} generates an independent constraint. But the max rise delay of 5 and the max rise delay of 7 interfere with each other.
- For a -max option, the maximum value overrides all lower values. Thus the first constraint will be overridden and the max rise delay of 7 will survive.

```
set_output_delay 5 [get_clocks CLK2] [get_ports PAD1] -max -rise -add_delay # will be
overridden
set_output_delay 3 [get_clocks CLK2] [get_ports PAD1] -min -fall -add_delay
set_output_delay 3 [get_clocks CLK2] [get_ports PAD1] -max -fall -add_delay
set_output_delay 7 [get_clocks CLK2] [get_ports PAD1] -max -rise -add_delay
```

Related Examples on GitHub

- set_output_delay

See Also

- 7.59 set_input_delay

- 7.41 remove_input_delay

7.65 write_sdc

Description
This Tcl command writes timing constraints into an SDC file. If multiple constraint scenarios are defined, `-scenario` allows the user to specify which scenario to write. By default, the current scenario is written.

```
write_sdc \
-scenario scenario_name \
-pin_separator ( : | / ) \
file name
```

Arguments

Parameter	Type	Description
scenario	string	Specifies the scenario to write. By default the current scenario is used.
pin_separator	char	Specify the pin separator used in the SDC file. It can be either ':' or '/'.
file name	string	Specify the SDC file name.

Supported Families

Supported Families	Supported Libero SoC Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
RTG4	v12.4+
SmartFusion2	v12.4+
IGLOO2	v12.4+

Example
The following script merges two SDC files and writes the result into a third SDC file.

```
read_sdc first.sdc
read_sdc -add second.sdc
write_sdc margin.sdc
```

See Also
- 7.32 read_sdc

8. SmartPower Tcl Commands

8.1 smartpower_add_new_scenario

Tcl command; creates a new scenario.

```
smartpower_add_new_scenario -name {value} -description {value} -mode {value}
```

8.1.1 Arguments

`-name {value}`

Specifies the name of the new scenario.

`-description {value}`

Specifies the description of the new scenario.

`-mode {<operating mode>:<duration>}+`

Specifies the mode(s) and duration(s) for the specified scenario.

8.1.2 Examples

This example creates a new scenario called myscenario: `smartpower_add_new_scenario -name "MyScenario" -mode "Custom_1:50.00" "Custom_2:25.00" -mode "Active:25.00"`

8.2 smartpower_add_pin_in_domain

```
smartpower_add_pin_in_domain -pin_name {pin_name} -pin_type {value} -domain_name
{domain_name} -domain_type {value}
```

Tcl command; adds a pin into a clock or set domain.

8.2.1 Arguments

`-pin_name {pin_name}`

Specifies the name of the pin to add to the domain.

`-pin_type {value}`

Specifies the type of the pin to add. The following table shows the acceptable values for this argument:

Value	Description
clock	The pin to add is a clock pin
data	The pin to add is a data pin

`-domain_name {domain_name}`

Specifies the name of the domain in which to add the specified pin.

`-domain_type {value}`

Specifies the type of domain in which to add the specified pin. The following table shows the acceptable values for this argument:

Value	Description
clock	The domain is a clock domain

set	The domain is a set domain

8.2.2 Notes

- The `domain_name` must be a name of an existing domain.
  - The `pin_name` must be a name of a pin that exists in the design.

8.2.3 Examples

The following example adds a clock pin to an existing Clock domain:

```
smartpower_add_pin_in_domain -pin_name { XCMP3/U0/U1:Y } -pin_type {clock} -
domain_name
{clk1} -domain_type {clock}
```

The following example adds a data pin to an existing Set domain:

```
smartpower_add_pin_in_domain -pin_name {XCMP3/U0/U1:Y} -pin_type {data} -domain_name
{myset} -domain_type {set}
```

8.3 smartpower_battery_settings

This SmartPower Tcl command sets the battery capacity in SmartPower. The battery capacity is used to compute the battery life of your design.

```
smartpower_battery_settings -capacity {decimal value}
```

8.3.1 Parameters

`-capacity {decimal value}` Value must be a positive decimal. This parameter is mandatory.

8.3.2 Exceptions

None

8.3.3 Returns

This command does not return a value.

8.3.4 Usage

The following table lists the parameters for the command, their types, and the values they can be set to.

smartpower_battery_settings	Type	Value	Description
capacity	Decimal	Positive decimal	Specify the battery capacity in mA*Hours

8.3.5 Example

This example sets the battery capacity to 1800 mA * Hours.

```
smartpower_battery_settings –capacity {1800}
```

8.4 smartpower_change_clock_statistics

```
smartpower_change_clock_statistics -domain_name {value} -clocks_freq {value} -
clocks_proba {value} -registers_freq {value} -registers_proba {value} -set_reset_freq
{value} -set_reset_proba {value} -primaryinputs_freq {value} -primaryinputs_proba
{value} - combinational_freq {value} -combinational_proba {value}
```

Tcl command; changes the default frequencies and probabilities for a specific domain.

8.4.1 Arguments

`-domain_name{value}`

Specifies the domain name in which to initialize frequencies and probabilities.

`-clocks_freq {value}`

Specifies the user input frequency in Hz, KHz, or MHz for all clocks.

`-clocks_proba {value}`

Specifies the user input probability in % for all clocks.

`-registers_freq {value}`

Specifies the user input frequency (in Hz, KHz, or MHz) or the toggle rate (in %). If the unit is not provided and toggle rate is active, the value is handled as a toggle rate; if toggle rate is not active, the value is handled as a frequency.

`-registers_proba {value}`

Specifies the user input probability in % for all registers.

`-set_reset_freq {value}`

Specifies the user input frequency (in Hz, KHz, or MHz) or the toggle rate (in %). If the unit is not provided and toggle rate is active, the value is handled as a toggle rate; if toggle rate is not active, the value is handled as a frequency.

`-set_reset_proba {value}`

Specifies the user input probability in % for all set/reset nets.

`-primaryinputs_freq {value}`

Specifies the user input frequency (in Hz, KHz, or MHz) or the toggle rate (in %). If the unit is not provided and toggle rate is active, the value is handled as a toggle rate; if toggle rate is not active, the value is handled as a frequency.

`-primaryinputs_proba {value}`

Specifies the user input probability in % for all primary inputs.

`-combinational_freq {value}`

Specifies the user input frequency (in Hz, KHz, or MHz) or the toggle rate (in %). If the unit is not provided and toggle rate is active, the value is handled as a toggle rate; if toggle rate is not active, the value is handled as a frequency.

`-combinational_proba {value}`

Specifies the user input probability in % for all combinational combinational output.

Note: This command is associated with the functionality of Initialize frequencies and probabilities dialog box.

8.4.2 Examples

The following example initializes all clocks with:

```
smartpower_change_clock_statistics -domain_name {my_domain} -clocks_freq {10 MHz} -
clocks_proba {20} -registers_freq {10 MHz} -registers_proba {20} -set_reset_freq {10
MHz} -set_reset_proba {20} -primaryinputs_freq {10 MHz} -primaryinputs_proba {20} -
combinational_freq {10 MHz} -combinational_proba {20}
```

8.5 smartpower_change_setofpin_statistics

`smartpower_change_setofpin_statistics -domain_name {value} -data_freq {value} -data_proba {value}`

Tcl command; changes the default frequencies and probabilities for a specific set.

8.5.1 Arguments

`-domain_name{value}`

Specifies the domain name in which to initialize data frequencies and probabilities.

`-data_freq {value}`

Specifies the user input data frequency in Hz, KHz, or MHz for all sets of pins.

`-data_proba {value}`

Specifies the user input data probability in % for all sets of pins.

8.5.2 Notes

This command is associated with the functionality of Initialize frequencies and probabilities dialog box.

8.5.3 Examples

The following example initializes all clocks withs:

```
smartpower_change_setofpin_statistics -domain_name {my_domain} -data_freq {10 MHz} -
data_proba {20}
```

8.6 smartpower_commit

Tcl command; saves the changes to the design file.

```
smartpower_commit
```

8.6.1 Arguments

None

8.6.2 Examples

```
smartpower_commit
```

8.7 smartpower_compute_vectorless

This Tcl command executes a vectorless analysis of the current operating mode.

8.7.1 Arguments

None

8.7.2 Example

```
smartpower_compute_vectorless
```

8.8 smartpower_create_domain

Tcl command; creates a new clock or set domain.

```
smartpower_create_domain -domain_type {value} -domain_name {domain_name}
```

8.8.1 Arguments

`-domain_type {value}`

Specifies the type of domain to create. The following table shows the acceptable values for this argument:

Value	Description
clock	The domain is a clock domain
set	The domain is a set domain

`-domain_name {domain_name}`

Specifies the name of the new domain.

8.8.2 Notes

The `domain name` cannot be the name of an existing domain. The domain `type` must be either clock or set.

8.8.3 Examples

The following example creates a new clock domain named "clk2": `smartpower_create_domain -domain_type {clock} -domain_name {clk2}` The following example creates a new set domain named "myset": `smartpower_create_domain -domain_type {set} -domain_name {myset}`

8.9 smartpower_edit_scenario

Tcl command; edits a scenario.

`smartpower_edit_scenario -name {value} -description {value} -mode {value} -new_name {value}`

8.9.1 Arguments

`-name {value}`

Specifies the name of the scenario.

`-description {value}`

Specifies the description of the scenario.

`-mode {<operating mode>:<duration>}`

Specifies the mode(s) and duration(s) for the specified scenario.

`-new_name {value}`

Specifies the new name for the scenario

8.9.2 Examples

This example edits the name of myscenario to finalscenario:

```
smartpower_edit_scenario -name myscenario -new_name finalscenario
```

8.10 smartpower_export_mpe_report

Description
This command exports the Microsemi Power Estimation (MPE) report in XML format.

```
smartpower_export_mpe_report -filename {file_name.xml}
```

Arguments

Parameter	Type	Description
-file_name {file_name}	string	Mandatory. Name of the XML file to be exported.

Example

```
smartpower_export_mpe_report -filename mpe_report.xml
```

8.11 smartpower_import_vcd

```
import_vcd -file "VCD file" [-opmode "mode name"] [-with_vectorless "TRUE | FALSE"]
[- partial_parse\ "TRUE | FALSE"] [-start_time "decimal value"] [-end_time "decimal
value"]
\
[-auto_detect_top_level_name "TRUE | FALSE"] [-top_level_name "top level name"] [-
glitch_filtering\ "false | auto | true"] [-glitch_threshold "integer value"] [-
stop_time "decimal value"]
```

This SmartPower Tcl command imports into SmartPower a VCD file generated by a simulation tool. SmartPower extracts the frequency and probability information from the VCD.

8.11.1 Parameters

`-file "VCD file"`

Value must be a file path. This parameter is mandatory.

`[-opmode "mode name"]`

Value must be a string. This parameter is optional.

`[-with_vectorless "TRUE | FALSE"]`

Value must be a boolean. This parameter is optional.

`[-partial_parse "TRUE | FALSE"]`

Value must be a boolean. This parameter is optional.

`[-start_time "decimal value"]`

Value must be a positive decimal. This parameter is optional.

`[-end_time "decimal value"]`

Value must be a positive decimal. This parameter is optional.

`[-auto_detect_top_level_name "TRUE | FALSE"]` Value must be a boolean. This parameter is optional.
`[-top_level_name "top level name"]`

Value must be a string. This parameter is optional.

`[-glitch_filtering "false | auto | true"]`

Value must be one of false | auto | true. This parameter is optional.

`[-glitch_threshold "integer value"]`

Value must be a positive integer. This parameter is optional.

8.11.2 Exceptions

None

8.11.3 Returns

This command does not return a value. Usage

This section lists all the parameters for the command, their types, and the values they can be set to. The default value is always listed first.

smartpower_import_vcd	Type	Values	Description
file	String	Path to a VCD file	Path to a VCD file.

opmode	String	Operating mode name “Active” by default	Operating mode in which the VCD will be imported. If the mode doesn’t exist, it will be created.
with_vectorless	Boolean	TRUE|FALSE	Specify the method to set the frequency and probability information for signals not annotated by the VCD TRUE: use the vectorless analysis FALSE: use average value computed from the VCD.
partial_parse	Boolean	FALSE|TRUE	Enable partial parsing of the VCD. Start time and end time need to be specified when TRUE.
start_time	Decimal value	positive decimal nanoseconds (ns)	Specify the starting timestamp of the VCD extraction in ns. It must be lower than the specified end_time. It must be lower than the last timestamp in the VCD file.
end_time	Decimal value	positive decimal nanoseconds (ns)	Specify the end timestamp of the VCD extraction in ns. It must be higher than the specified start_time.
auto_detect_top_level_name	Boolean	TRUE|FALSE	Enable the auto detection of the top level name in the VCD file. Top_level_name needs to be specified when FALSE.
top_level_name	Boolean	Full hierarchical name	Specify the full hierarchical name of the instance of the design in the VCD file.
glitch_filtering	Boolean	Auto|FALSE|TRUE	AUTO: Enable glitch filtering with predefined thereshold based on the family TRUE: Enable glitch filtering, glitch_threshold must be specified FALSE: Disable glitch filtering.
glitch_threshold	Integer	Positive integer	Specify the threshold in ps below which glitches are filtered out.

8.11.4 Examples

The Tcl command below imports the power.vcd file generated by the simulator into SmartPower:

```
smartpower_import_vcd -file "../../simulation/power.vcd"
```

The Tcl command below extracts information between 1ms and 2ms in the simulation, and stores the information into a custom mode:

```
smartpower_import_vcd -file "../../simulation/power.vcd" -partial_parse TRUE -
start_time 1000000 -end_time 2000000 -opmode "power_1ms_to_2ms"
```

8.12 smartpower_init_do

```
smartpower_init_do -with {value} -opmode {value} -clocks {value} -registers {value}
- set_reset {value} -primaryinputs {value} -combinational {value} -enables {value} -
othersets
```

```
{value}
```

Tcl command; initializes the frequencies and probabilities for clocks, registers, set/reset nets, primary inputs, combinational outputs, enables and other sets of pins, and selects a mode for initialization.

8.12.1 Arguments

`-with{value}`

This sets the option of initializing frequencies and probabilities with vectorless analysis or with fixed values. The following table shows the acceptable values for this argument:

Value	Description
vectorless	Initializes frequencies and probabilities with vectorless analysis.
fixed	Initializes frequencies and probabilities with fixed values.

`-opmode {value}`

Optional; specifies the mode in which to initialize frequencies and probabilities. The value must be Active or Flash*Freeze.

`-clocks {value}`

This sets the option of initializing frequencies and probabilities for all clocks. The following table shows the acceptable values for this argument:

Value	Description
true	Initializes frequencies and probabilities for all clocks
false	Does not initialize frequencies and probabilities for all clocks

`-registers {value}`

This sets the option of initializing frequencies and probabilities for all registers. The following table shows the acceptable values for this argument:

Value	Description
true	Initializes frequencies and probabilities for all registers
false	Does not initialize frequencies and probabilities for all registers

`-set_reset {value}`

This sets the option of initializing frequencies and probabilities for all set/reset nets. The following table shows the acceptable values for this argument:

Value	Description
true	Initializes frequencies and probabilities for all set/reset nets
false	Does not initialize frequencies and probabilities for all set/reset nets

`-primaryinputs{value}`

This sets the option of initializing frequencies and probabilities for all primary inputs. The following table shows the acceptable values for this argument:

Value	Description
true	Initializes frequencies and probabilities for all primary inputs

false	Does not initialize frequencies and probabilities for all primary inputs

`-combinational {value}`

This sets the option of initializing frequencies and probabilities for all combinational outputs. The following table shows the acceptable values for this argument:

Value	Description
true	Initializes frequencies and probabilities for all combinational outputs
false	Does not initialize frequencies and probabilities for all combinational outputs

`-enables {value}`

This sets the option of initializing frequencies and probabilities for all enable sets of pins. The following table shows the acceptable values for this argument:

Value	Description
true	Initializes frequencies and probabilities for all enable sets of pins
false	Does not initialize frequencies and probabilities for all enable sets of pins

`-othersets {value}`

This sets the option of initializing frequencies and probabilities for all other sets of pins. The following table shows the acceptable values for this argument:

Value	Description
true	Initializes frequencies and probabilities for all other sets of pins
false	Does not initialize frequencies and probabilities for all other sets of pins

Note: This command is associated with the functionality of Initialize frequencies and probabilities dialog box.

8.12.2 Examples

The following example initializes all clocks with:

```
smartpower_init_do -with {vectorless} -opmode {my_mode} -clocks {true} -registers {true}
-asynchronous {true} -primaryinputs {true} -combinational {true} -enables {true} -othersets {true}
```

8.13 smartpower_init_set_clocks_options

```
smartpower_init_set_clocks_options -with_clock_constraints {value} -with_default_values {value} -freq {value} -duty_cycle {value}
```

Tcl command; initializes the clock frequency options of all clock domains.

8.13.1 Arguments

`-with_clock_constraints {value}`

This sets the option of initializing the clock frequencies with frequency constraints from SmartTime. The following table shows the acceptable values for this argument:

Value	Description

true	Sets initialize clock frequencies with clock constraints ON
false	Sets initialize clock frequencies with clock constraints OFF

`-with_default_values {value}`

This sets the option of initializing the clock frequencies with a user input default value. The following table shows the acceptable values for this argument:

Value	Description
true	Sets initialize clock frequencies with default values ON
false	Sets initialize clock frequencies with default values OFF

`-freq {value}`

Specifies the user input frequency in Hz, KHz, or MHz.

`-duty_cycle {value}`

Specifies the user input duty cycles in %.

8.13.2 Notes

This command is associated with the functionality of Initialize frequencies and probabilities dialog box.

8.13.3 Examples

The following example initializes all clocks after executing `smartpower_init_do` with `-clocks {true}`:

```
smartpower_init_set_clocks_options -with_clock_constraints {true} -with_default_values {true} -freq {10 MHz} -duty_cycle {20}
```

8.14 smartpower_init_set_combinational_options

Tcl commands; initializes the frequency and probability of all combinational outputs.

```
smartpower_init_set_combinational_options -freq {value} -proba {value}
```

8.14.1 Arguments

`-freq {value}`

Specifies the user input frequency (in Hz, KHz, or MHz) or the toggle rate (in %). If the unit is not provided and toggle rate is active, the value is handled as a toggle rate; if toggle rate is not active, the value is handled as a frequency.

`-proba {value}`

Specifies the user input probability in %.

8.14.2 Notes

This command is associated with the functionality of Initialize frequencies and probabilities dialog box.

8.14.3 Examples

The following example initializes all combinational signals after executing `smartpower_init_do` with `-combinational {true}`:

```
smartpower_init_set_combinational_options -freq {10 MHz} -proba {20}
```

8.15 smartpower_init_set_enables_options

Tcl command; initializes the clock frequency of all enable clocks with the initialization options.

```
smartpower_init_set_enables_options -freq {value} -proba {value}
```

8.15.1 Arguments

`-freq {value}`

Specifies the user input frequency (in Hz, KHz, or MHz).

`-proba {value}`

Specifies the user input probability in %.

8.15.2 Notes

This command is associated with the functionality of Initialize frequencies and probabilities dialog box.

8.15.3 Examples

The following example initializes all clocks after executing `smartpower_init_do` with `-enables {true}`:

```
smartpower_init_set_enables_options -freq {10 MHz} -proba {20}
```

8.16 smartpower_init_set_primaryinputs_options

Tcl command; initializes the frequency and probability of all primary inputs.

```
smartpower_init_set_primaryinputs_options -freq {value} -proba {value}
```

8.16.1 Arguments

`-freq {value}`

Specifies the user input frequency (in Hz, KHz, or MHz) or the toggle rate (in %). If the unit is not provided and toggle rate is active, the value is handled as a toggle rate; if toggle rate is not active, the value is handled as a frequency.

`-proba {value}`

Specifies the user input probability in %.

8.16.2 Notes

This command is associated with the functionality of Initialize frequencies and probabilities dialog box.

8.16.3 Examples

The following example initializes all primary inputs after executing `smartpower_init_do` with `-primaryinputs {true}`:

```
smartpower_init_set_primaryinputs_options -freq {10 MHz} -proba {20}
```

8.17 smartpower_init_set_registers_options

Tcl command; initializes the frequency and probability of all register outputs.

```
smartpower_init_set_registers_options -freq {value} -proba {value}
```

8.17.1 Arguments

`-freq {value}`

Specifies the user input frequency (in Hz, KHz, or MHz) or the toggle rate (in %). If the unit is not provided and toggle rate is active, the value is handled as a toggle rate; if toggle rate is not active, the value is handled as a frequency.

`-proba {value}`

Specifies the user input probability in %.

8.17.2 Notes

This command is associated with the functionality of Initialize frequencies and probabilities dialog box.

8.17.3 Exceptions

None

8.17.4 Examples

The following example initializes all register outputs after executing `smartpower_init_do` with `-registers {true}`:

```
smartpower_init_set_registers_options -freq {10 MHz} -proba {20}
```

8.18 smartpower_init_setofpins_values

Tcl command; initializes the frequency and probability of all sets of pins.

```
smartpower_init_setofpins_values -domain_name {name} -freq {value} -proba {value}
```

8.18.1 Arguments

`-domain_name{name}`

Specifies the set of pins that will be initialized. The following table shows the acceptable values for this argument:

Value	Description
IOsEnableSet	Specifies that the IOsEnableSet set of pins will be initialized
MemoriesEnableSet	Specifies that the MemoriesEnableSet set of pins will be initialized

`-freq {value}`

Specifies the user input frequency in Hz, MHz, or KHz.

`-proba {value}`

Specifies the user input probability in %.

8.18.2 Notes

This command is associated with the functionality of Initialize frequencies and probabilities dialog box.

8.18.3 Examples

The following example initializes all primary inputs after executing `smartpower_init_do` with `-othersets {true}`:

```
smartpower_init_setofpins_values -domain_name {IOsEnableSet} -freq {10 MHz} -proba
{20}
```

8.19 smartpower_remove_all_annotations

Tcl command; removes all initialization annotations for the specified mode.

```
value}
```

8.19.1 Arguments

`-opmode {value}`

Removes all initialization annotations for the specified mode, where value must be Active or Flash*Freeze.

8.19.2 Notes

This command is associated with the functionality of Initialize frequencies and probabilities dialog box.

8.19.3 Examples

The following example initializes all clocks with opmode Acitve:

```
smartpower_remove_all_annotations -opmode {Active}
```

8.20 smartpower_remove_file

Tcl command; removes a VCD file from the specified mode or all operating mode. Frequency and probability information of signals annotated by the VCD are set back to the default value.

```
remove file
-file {value} \
-format {value} \
-opmode {value} \
```

8.20.1 Arguments

`-file {value}`

Specifies the file to be removed. This is mandatory.

`-format VCD`

Specifies that the type to be removed is a VCD file. This is mandatory.

`[-opmode {value}]`

Specifies the operating mode. This is optional. The following table shows the acceptable values for this argument:

Value	Description
Active	The operating mode is set to active
Static (PolarFire)	The operating mode is set to Static
Flash*Freeze	The operating mode is set to Flash*Freeze

8.20.2 Examples

This example removes the file test.vcd from the Active mode.

```
smartpower_remove_file -file "test.vcd" -format VCD -opmode "Active"
```

This example removes the VCD file power1.vcd from all operating modes:

```
smartpower_remove_file -file “power1.vcd” -format VCD
```

8.21 smartpower_remove_scenario

Tcl command; removes a scenario from the current design.

```
smartpower_remove_scenario -name {value}
```

8.21.1 Arguments

`-name {value}`

Specifies the name of the scenario.

8.21.2 Examples

This example removes a scenario from the current design:

```
smartpower_remove_scenario -name myscenario
```

8.22 smartpower_report_power

```
smartpower_report_power\ [-powerunit {value}] \
[-frequnit {value}] \ [-opcond {value}] \
[-opmode {value}] \ [-toggle {value}] \
[-power_summary {value}] \ [-rail_breakdown{value}] \ [-type_breakdown{ value}] \ [-clock_breakdown{value}] \
[-thermal_summary {value}] \ [-battery_life {value}] \
[-opcond_summary {value}] \ [-clock_summary {value}] \ [-style {value}] \
[-sortorder {value}] \ [-sortby {value}] \
[-instance_breakdown {value}] \ [-power_threshold {value}] \
[-filter_instance {value}] \ [-min_power {number}] \
[-max_instance {integer >= 0}] \ [-activity_sortorder {value}] \ [-activity_sortby {value}] \
[-activity_summary {value}] \
[-frequency_threshold {value}] \ [-filter_pin {value}] \
[-min_frequency {value}] \ [-max_pin {value}] \
[-enablerates_sortorder {value}] \ [-enablerates_sortby {value}] \
[-enablerates_summary {value}] \
```

Tcl command; creates a Power report, which enables you to determine if you have any power consumption problems in your design. It includes information about the global device and SmartPower preferences selection, and hierarchical detail (including gates, blocks, and nets), with a block-by-block, gate-by-gate, and net-by-net power summary SmartPower results.

```
[-with_annotation_coverage {value}] \
{filename}
```

8.22.1 Arguments

`-powerunit {value}`

Specifies the unit in which power is set. The following table shows the acceptable values for this argument:

Value	Description
W	The power unit is set to watts
mW	The power unit is set to milliwatts
uW	The power unit is set to microwatts

`-frequnit {value}`

Specifies the unit in which frequency is set. The following table shows the acceptable values for this argument:

Value	Description

Hz	The frequency unit is set to hertz
kHz	The frequency unit is set to kilohertz
MHz	The frequency unit is set to megahertz

`-opcond {value}`

Specifies the operating condition. The following table shows the acceptable values for this argument:

Value	Description
worst	The operating condition is set to worst case
typical	The operating condition is set to typical case
best	The operating condition is set to best case

`-opmode {value}`

Specifies the operating mode. The following table shows the acceptable values for this argument:

Value	Description
Active	The operating mode is set to Active
Standby	The operating mode is set to Standby
Flash*Freeze	The operating mode is set to Flash*Freeze

`-toggle {value}`

Specifies the toggle. The following table shows the acceptable values for this argument:

Value	Description
true	The toggle is set to true
false	The toggle is set to false

`-power_summary {value}`

Specifies whether to include the power summary, which shows the static and dynamic values in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the power summary in the report
false	Does not include the power summary in the report

`-rail_breakdown {value}`

Specifies whether to include the breakdown by rail summary in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the breakdown by rail summary in the report
false	Does not include the breakdown by rail summary in the report

`-type_breakdown {value}`

Specifies whether to include the breakdown by type summary in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the breakdown by type summary in the report
false	Does not include the breakdown by type summary in the report

`-clock_breakdown {value}`

Specifies whether to include the breakdown by clock domain in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the breakdown by clock domain summary in the report
false	Does not include the breakdown by clock domain summary in the report

`-thermal_summary {value}`

Specifies whether to include the thermal summary in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the thermal summary in the report
false	Does not include the thermal summary in the report

`-battery_life {value}`

Specifies whether to include the battery life summary in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the battery life summary in the report
false	Does not include the battery life summary in the report

`-opcond_summary {value}`

Specifies whether to include the operating conditions summary in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the operating conditions summary in the report
false	Does not include the operating conditions summary in the report

`-clock_summary {value}`

Specifies whether to include the clock domains summary in the report. The following table shows the acceptable values for this argument:

Value	Description

true	Includes the clock summary in the report
false	Does not include the clock summary in the report

`-style {value}`

Specifies the format in which the report will be exported. The following table shows the acceptable values for this argument:

Value	Description
Text	The report will be exported as Text file
CSV	The report will be exported as CSV file

`-sortby {value}`

Specifies how to sort the values in the report. The following table shows the acceptable values for this argument:

Value	Description
power values	Sorts based on the power values
alphabetical	Sorts in an alphabetical order

`-sortorder {value}`

Specifies the sort order of the values in the report. The following table shows the acceptable values for this argument:

Value	Description
ascending	Sorts the values in ascending order
descending	Sorts the values in descending order

`-instance_breakdown {value}`

Specifies whether to include the breakdown by instance in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the breakdown by instance in the report
false	Does not include the breakdown by instance in the report

`-power_threshold {value}`

This specifies whether to include only the instances that consume power above a certain minimum value. When this command is set to true, the `-min_power` argument must also be used to specify that only the instances that consume power above this minimum power value are the ones that are included in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the power threshold in the report
false	Does not include the power threshold in the report

`-filter_instance {value}`

This specifies whether to have a limit on the number of instances to include in the Power report. When this command is set to true, the `-max_instance` argument must also be used to specify the maximum number of instances to be included into the Power report. The following table shows the acceptable values for this argument:

Value	Description
true	Indicates that you want to have a limit on the number of instances to include in the Power report
false	Indicates that you do not want to have a limit on the number of instances to include in the Power report

`-min_power {number}`

Specifies which block to expand based on the minimum power value of a block.

`-max_instance {integer >= 0}`

Sets the maximum number of instances to a specified integer greater than or equal to 0 (zero). This will limit the maximum number of instances to be included in the Power report.

`-activity_sortorder {value}`

Specifies the sort order for the activity summary. The following table shows the acceptable values for this argument:

Value	Description
ascending	Sorts the values in ascending order
descending	Sorts the values in descending order

`-activity_sortby {value}`

Specifies how to sort the values for the activity summary. The following table shows the acceptable values for this argument:

Value	Description
pin name	Sorts based on the pin name
net name	Sorts based on thepnet name
domain	Sorts based on the clock domain
frequency	Sorts based on the clock frequency
source	Sorts based on the clock frequency source

`-activity_summary {value}`

Specifies whether to include the activity summary in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the activity summary in the report
false	Does not include the activity summary in the report

`-frequency_threshold {value}`

Specifies whether to add a frequency threshold. The following table shows the acceptable values for this argument:

Value	Description
true	Adds a frequency threshold
false	Does not add a frequency threshold

`-filter_pin {value}`

Specifies whether to filter by maximum number of pins. The following table shows the acceptable values for this argument:

Value	Description
true	Filters by maximum number of pins
false	Des not filter by maximum number of pins

`-min_frequency {value}`

Sets the minimum frequency to {decimal value [unit { Hz | KHz | MHz}]}.

`-max_pin {value}`

Sets the maximum number of pins.

`-enablerates_sortorder {value}`

Specifies the sort order for the probabilities summary. The following table shows the acceptable values for this argument:

Value	Description
ascending	Sorts the values in ascending order
descending	Sorts the values in descending order

`-enablerates_sortby {value}`

Specifies how to sort the values for the probabilities summary. The following table shows the acceptable values for this argument:

Value	Description
pin name	Sorts based on the pin name
net name	Sorts based on the net name
domain	Sorts based on the clock domain
frequency	Sorts based on the clock frequency
source	Sorts based on the clock frequency source

`-enablerates_summary {value}`

Specifies whether to include the probabilities summary in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the activity summary in the report
false	Does not include the activity summary in the report

`-with_annotation_coverage {value}`

Specifies whether to include the annotation coverage summary in the report. The following table shows the acceptable values for this argument:

Value	Description
true	Includes the annotation coverage summary in the report

Value	Description
false	Does not include the annotation coverage summary in the report

`{filename}`

Specifies the name of the report.

8.22.2 Notes

- The following arguments have been removed. Running the script will trigger a warning message: Warning: Invalid argument: -argname "argvalue" Ignored. Ignore the warning.
  `-annotated_pins {value}`

  `-stat_pow {value}`

  `-dyn_pow {value}`
- Flash*Freeze, Sleep, and Shutdown are available only for certain families and devices.
  - Worst and Best are available only for certain families and devices.

8.22.3 Examples

This example generates a Power report named report.rpt.

```
smartpower_report_power -powerunit "uW" -frequnit "MHz" -opcond "Typical" -opmode
"Active" -toggle "TRUE" -rail_breakdown "TRUE" -battery_life "TRUE" -style "Text" -
power_summary "TRUE" -activity_sortby "Source" text_report.txt
```

8.23 smartpower_set_mode_for_pdpr

This SmartPower Tcl command sets the operating mode used by the Power Driven Place and Route (PDPR) tool during power optimization.

```
smartpower_set_mode_for_pdpr -opmode {value}
```

8.23.1 Parameters

`-opmode {value}`

Value must be a valid operating mode. This parameter is mandatory.

Sets the operating mode for your power driven place and route.

8.23.2 Exceptions

None

8.23.3 Return Value

This command does not return a value.

8.23.4 Examples

This example sets the Active mode as the operating mode for Power Driven Place and Route.

```
set_mode_for_pdpr -opmode "Active"
```

This example creates a custom mode and set it to be used by Power Driven Place and Route (PDPR).

```
smartpower_add_new_custom_mode –name “MyCustomMode” \
–description “for PDPR” –base_mode “Active” smartpower_set_mode_for_pdpr –opmode
“MyCustomMode
```

8.24 smartpower_set_operating_condition

Tcl command; sets the operating conditions used in SmartPower to one of the pre-defined types.

```
smartpower_set_operating_condition -opcond {value}
```

8.24.1 Arguments

Examples

`-opcond {value}`

Specifies the value of the operating condition. The following table shows the acceptable values for this argument:

Value	Description
best	Sets the operating conditions to best
typical	Sets the operating conditions to typical
worst	Sets the operating conditions to worst

This example sets the operating conditions to best:

```
smartpower_set_operating_condition -opcond {best}
```

8.25 smartpower_set_operating_conditions

```
smartpower_set_operating_conditions "still_air | 1.0_mps | 2.5_mps | custom" -heatsink
"None | custom | 10mm_Low_Profile | 15mm_Medium_Profile | 20mm_High_Profile" -
boardmodel "None_Conservative | JEDEC_2s2p" [-teta_ja "decimal value"] [-teta_sa
"decimal value"]
```

Tcl command; sets the operating conditions used in SmartPower.

8.25.1 Arguments

`-still_air {value}`

Specifies the value for the still air operating condition. The following table shows the acceptable values for this argument:

Value	Description
1.0_mps	Sets the operating conditions to best
2.5_mps	Sets the operating conditions to typical
custom	Sets the operating conditions to worst

`-heatsink {value}`

Specifies the value of the operating condition. The following table shows the acceptable values for this argument:

Value	Description

none	No heat sink
custom	Sets a custom heat sink size
10mm_Low_Profile	10 mm heat sink
15mm_Low_Profile	15 mm heat sink
20mm_High_Profile	20 mm heat sink

`-boardmodel {value}`

Specifies your board model. The following table shows the acceptable values for this argument:

Value	Description
None_Conservative	No board model, conservative routing
JEDEC_2s2p	JEDEC 2s2p board model

`-teta_ja {decimal_value}`

Optional; sets your teta ja value; must be a positive decimal

`-teta_sa {decimal_value}`

Optional; sets your teta sa value; must be a positive decimal.

8.25.2 Examples

This example sets the operating conditions to best:

```
set_operating_conditions -airflow "still_air" -heatsink "None" -boardmodel
"None_Conservative "
```

8.26 smartpower_set_process

Tcl command; sets the process used in SmartPower to one of the pre-defined types.

`smartpower_set_process -process {value}`

8.26.1 Arguments

`-process {value}`

Specifies the value of the operating condition. The following table shows the acceptable values for this argument:

Value	Description
Typical	Sets the process for SmartPower to typical
Maximum	Sets the process for SmartPower to maximum

8.26.2 Examples

This example sets the operating conditions to typical:

`smartpower_set_process -process {Typical}`

8.27 smartpower_set_temperature_opcond

Tcl command; sets the temperature in the operating conditions to one of the pre-defined types.

`smartpower_set_temperature_opcond -use{value}`

8.27.1 Arguments

`-use{value}`

Specifies the temperature in the operating conditions. The following table shows the acceptable values for this argument:

Value	Description
oprange	Sets the temperature in the operating conditions as specified in your Project Settings.
design	Sets the temperature in the operating conditions as specified in the SmartPower design-wide operating range. Applies to SmartPower only.
mode	Sets the temperature in the operating conditions as specified in the SmartPower mode-specific operating range. Applies to SmartPower only.

Examples

This example sets the temperature in the operating conditions as specified in the custom mode-settings:

```
smartpower_set_temperature_opcond -use{mode}
```

8.28 smartpower_set_thermalmode

Tcl command; sets the mode of computing junction temperature.

```
smartpower_set_thermalmode -mode {value}
```

8.28.1 Arguments

`-mode{value}`

Specifies the mode in which the junction temperature is computed. The following table shows the acceptable values for this argument:

Value	Description
ambient	The junction temperature will be iteratively computed with total static power
opcond	The junction temperature will be given as one of the operating condition range values specified in the device selection

Note:

To compute the junction temperature, set the `smartpower_set_thermalmode`, `smartpower_set_tambient` and `smartpower_set_cooling` commands. The junction temperature will be updated when an output command is executed, such as `report(Power)`.

8.28.2 Examples

The following example sets the computing of the junction temperature to ambient mode:

```
smartpower_set_thermalmode -mode {ambient}
```

8.29 smartpower_set_voltage_opcond

Tcl command; sets the voltage in the operating conditions.

```
smartpower_set_voltage_opcond -voltage{value} -use{value}
```

Arguments

-

`voltage{value`

}

Specifies the voltage supply in the operating conditions. The following table shows the acceptable values for this argument:

Value	Description
VDD	Sets the voltage operating conditions for VDD
VDD18	Sets the voltage operating conditions for VDD18
VDDAUX	Sets the voltage operating conditions for VDDAUX
VDDI 1.1	Sets the voltage operating conditions for VDD 1.1
VDDI 1.2	Sets the voltage operating conditions for VDDI 1.2
VDDI 1.35	Sets the voltage operating conditions for VDDI 1.35
VDDI 1.5	Sets the voltage operating conditions for VDDI 1.5
VDDI 1.8	Sets the voltage operating conditions for VDDI 1.8
VDDI 2.5	Sets the voltage operating conditions for VDDI 2.5
VDDI 3.3	Sets the voltage operating conditions for VDDI 3.3
VDD25	Sets the voltage operating conditions for VDD25
VDDA	Sets the voltage operating conditions for VDDA
VDDA25	Sets the voltage operating conditions for VDDA25

`-use{value}`

Specifies the voltage in the operating conditions for each voltage supply. The following table shows the acceptable values for this argument:

Value	Description
oprange	Sets the voltage in the operating conditions as specified in your Project Settings.
design	Sets the voltage in the operating conditions as specified in the SmartPower design-wide operating range. Applies to SmartPower only.
mode	Sets the voltage in the operating conditions as specified in the SmartPower mode-specific operating range. Applies to SmartPower only.

8.29.1 Examples

This example sets the VCCA as specified in the SmartPower mode-specific settings:

```
smartpower_set_voltage_opcond -voltage{vcca} -use{mode}
```

8.30 smartpower_temperature_opcond_set_design_wide

```
smartpower_temperature_opcond_set_design_wide -best{value} -typical{value} -
worst{value} - thermal_mode{value}
```

Tcl command; sets the temperature for SmartPower design-wide operating conditions.

8.30.1 Arguments

Examples

-best{*value*}

Specifies the best temperature (in degrees Celsius) used for design-wide operating conditions.

-typical{*value*}

Specifies the typical temperature (in degrees Celsius) used for design-wide operating conditions.

-worst{*value*}

Specifies the worst temperature (in degrees Celsius) used for design-wide operating conditions.

-thermal_mode{*value*}

Specifies the mode in which the junction temperature is computed. The following table shows the acceptable values for this argument:

Value	Description
ambient	The junction temperature will be iteratively computed with total static power
opcond	The junction temperature will be given as one of the operating condition range values specified in the device selection

This example sets the temperature for design-wide operating conditions to Best 20, Typical 30, and Worst 60:

```
smartpower_temperature_opcond_set_design_wide -best{20} -typical{30} -worst{60}
```

8.31 smartpower_temperature_opcond_set_mode_specific

```
smartpower_temperature_opcond_set_mode_specific -opmode{value} -thermal_mode{value} -
best{value} -typical{value} -worst{value} -thermal_mode{value}
```

Tcl command; sets the temperature for SmartPower mode-specific operating conditions.

8.31.1 Arguments

-opmode {*value*}

Specifies the operating mode. The following table shows the acceptable values for this argument:

Value	Description
Active	The operating mode is set to Active
Static	The operating mode is set to Static

Value	Description
Flash*Freeze	The operating mode is set to Flash*Freeze

-thermal_mode{*value*}

Specifies the mode in which the junction temperature is computed. The following table shows the acceptable values for this argument:

Value	Description
ambient	The junction temperature will be iteratively computed with total static power
opcond	The junction temperature will be given as one of the operating condition range values specified in the device selection

-best{*value*}

Specifies the best temperature (in degrees Celsius) for the selected mode.

`-typical{value}`

Specifies the typical temperature (in degrees Celsius) for the selected mode.

`-worst{value}`

Specifies the worst temperature (in degrees Celsius) for the selected mode.

8.31.2 Examples

This example sets the temperature for mode-specific operating conditions for mode1:

```
smartpower_temperature_opcond_set_mode_specific -mode{mode1} -best{20} -typical{30} -
worst{60}
```

8.32 smartpower_voltage_opcond_set_design_wide

```
smartpower_voltage_opcond_set_design_wide -voltage{value} -best{value} -typical{value}
- worst{value}
```

Tcl command; sets the voltage settings for SmartPower design-wide operating conditions.

8.32.1 Arguments

`-voltage{value}`

Specifies the voltage supply in the operating conditions. The following table shows the acceptable values for this argument:

Value	Description
VDD	Sets the voltage operating conditions for VDD
VDDI 2.5	Sets the voltage operating conditions for VDDI 2.5
VPP	Sets the voltage operating conditions for VPP

Value	Description
VCCA	Sets the voltage operating conditions for VCCA
VCCI 3.3	Sets the voltage operating conditions for VCCI 3.3
VCCI 2.5	Sets the voltage operating conditions for VCCI 2.5
VCCI 1.8	Sets the voltage operating conditions for VCCI 1.8
VCCI 1.5	Sets the voltage operating conditions for VCCI 1.5
VCC33A	Sets the voltage operating conditions for VCC33A
VCCDA	Sets the voltage operating conditions for VCCDA

`-best{value}`

Specifies the best voltage used for design-wide operating conditions.

`-typical{value}`

Specifies the typical voltage used for design-wide operating conditions.

`-worst{value}`

Specifies the worst voltage used for design-wide operating conditions.

8.32.2 Examples

This example sets VCCA for design-wide to best 20, typical 30 and worst 40:

```
smartpower_voltage_opcond_set_design_wide -voltage{VCCA} -best{20} -typical{30} -
worst{40}
```

8.33 smartpower_voltage_opcond_set_mode_specific

```
smartpower_voltage_opcond_set_mode_specific -opmode{value} -voltage{value} -
best{value} - typical{value} -worst{value}
```

Tcl command; sets the voltage settings for SmartPower mode-specific use operating conditions.

8.33.1 Arguments

`-opmode {value}`

Use this option to specify the mode from which the operating conditions are extracted to generate the report.

Value	Description
Active	The operating mode is set to Active
Static	The operating mode is set to Static
Flash*Freeze	The operating mode is set to Flash*Freeze

`-voltage{value}`

Specifies the voltage in the operating conditions. The following table shows the acceptable values for this argument:

Value	Description
VDD	Sets the voltage operating conditions for VDD
VDD18	Sets the voltage operating conditions for VDD18
VDDAUX	Sets the voltage operating conditions for VDDAUX
VDDI 1.1	Sets the voltage operating conditions for VDD 1.1
VDDI 1.2	Sets the voltage operating conditions for VDDI 1.2
VDDI 1.35	Sets the voltage operating conditions for VDDI 1.35
VDDI 1.5	Sets the voltage operating conditions for VDDI 1.5
VDDI 1.8	Sets the voltage operating conditions for VDDI 1.8
VDDI 2.5	Sets the voltage operating conditions for VDDI 2.5
VDDI 3.3	Sets the voltage operating conditions for VDDI 3.3
VDD25	Sets the voltage operating conditions for VDD25
VDDA	Sets the voltage operating conditions for VDDA
VDDA25	Sets the voltage operating conditions for VDDA25

`-best{value}`

Specifies the best voltage used for mode-specific operating conditions.

`-typical{value}`

Specifies the typical voltage used for mode-specific operating conditions.

`-worst{value}`

Specifies the worst voltage used for mode-specific operating conditions.

8.33.2 Examples

This example sets the voltage for the static mode and sets best to 20, typical to 30 and worst to 40:

```
smartpower_voltage_opcond_set_mode_specific -opmode{active} -voltage{VCCA} -best{20} -
typical{30} -worst{40}
```

9. FlashPro Express Tcl Commands

9.1 close_project

Description
This Tcl command closes the FlashPro or FlashPro Express project. Equivalent to clicking the Project menu, and choosing Close Job Project.

```
close_project
```

Arguments

Parameter	Type	Description
None	None	None

Return Type	Description
None	None

Error Codes

Error Code	Description
None	None

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
This command closes the FlashPro/FlashPro Express project:

```
close_project
```

See Also
- open_project

9.2 configure_flashpro3_prg

Description
This Tcl command changes FlashPro3 programmer settings. You can configure FlashPro programmer in Libero SoC Software or via this command in FlashPro Express software. You will be able to set VPUMP voltage for the programmer and force TCK drop down list of frequencies you want to use for the programming. Similarly other programmer such as FlashPro4/5/6 can also be selected and related options can be set.

```
configure_flashpro3_prg [-vpump {ON | OFF}] \
[-clk_mode {discrete_clk | free_running_clk}] \
```

```
[-force_freq {ON | OFF}] \
[-freq {freq}]
```

Arguments

Parameter	Type	Description
vpump	string	Enables FlashPro programmer to drive VPUMP. Set to ON to drive VPUMP. Valid values are ON or OFF.
clk_mode	string	Specifies free running or discrete TCK. Valid value is "discrete_clk" or "free_running_cl".
force_freq	string	Forces the FlashPro software to use the TCK frequency specified by the software rather than the TCK frequency specified in the programmer file. Valid values are ON or OFF.
freq	integer	Specifies the TCK frequency in MHz. It can be between 1 MHz to 6 MHz.

Return Type	Description
None	None

Error Codes

Error Code	Description

None	Parameter 'parameter_name' is not defined. Valid command formatting is
	`configure_flashpro3_prg [-vpump "ON | OFF"] \`
	`[-clk_mode "free_running_clk | discrete_clk"] \`
	`[-force_freq "ON | OFF"] \`
	`[-freq "freq"]`

None	Invalid 'freq' argument value: (expecting 1000000, 2000000, 3000000, 4000000 or 6000000).
None	Invalid 'clk_mode' argument value: '' (expecting free_running_clk or discrete_clk).

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example

The following example sets the VPUMP option to ON, TCK to free running, and uses the TCK frequency specified in the programmer file (force_freq is set to OFF):

```
configure_flashpro3_prg -vpump {ON} \
-clk_mode {free_running_clk} \
-force_freq {OFF} -freq {4}
```

The following example sets VPUMP to ON, TCK to discrete, forces the FlashPro software to use the TCK frequency specified in the software (-force_freq is set to ON) at a frequency of 2 MHz:

```
configure_flashpro3_prg -vpump {ON} \
-clk_mode {discrete_clk} \
-force_freq {ON} -freq {2}
```

See Also

- configure_flashpro4_prg
- configure_flashpro5_prg
- configure_flashpro6_prg

9.3 configure_flashpro4_prg

Description

This Tcl command changes FlashPro4 programmer settings. You can configure FlashPro programmer in Libero SoC Software or wia this command in FlashPro Express software. You will be able to set VPUMP voltage for the programmer and force TCK drop down list of frequencies you want to use for the programming. Similarly other programmer such as FlashPro3/5/6 can also be selected and related options can be set

```
configure_flashpro4_prg [-vpump {ON|OFF}] \
[-clk_mode {discrete_clk | free_running_clk}] \
[-force_freq {ON|OFF}] \
[-freq {freq}]
```

Arguments

Parameter	Type	Description
vpump	string	Enables FlashPro4 programmer to drive VPUMP. Set to ON to drive VPUMP. Valid values are ON or OFF.
clk_mode	string	Specifies free running or discrete TCK. Valid value is "discrete_clk" or "free_running_cl".
force_freq	string	Forces the FlashPro software to use the TCK frequency specified by the software rather than the TCK frequency specified in the programmer file. Valid values are ON or OFF.
freq	integer	Specifies the TCK frequency in MHz. It can be between 1 MHz to 6 MHz.

Return Type	Description
None	None

Error Codes

Error Code	Description

None	Parameter 'parameter_name' is not defined. Valid command formatting is
	`configure_flashpro4_prg [-vpump "ON | OFF"] \`
	`[-clk_mode "free_running_clk | discrete_clk"] \`
	`[-force_freq "ON | OFF"] \`
	`[-freq "freq"]`

None	Invalid argument value: (expecting 1000000, 2000000, 3000000, 4000000, 5000000 or 6000000).
None	Invalid 'clk_mode' argument value: '' (expecting free_running_clk or discrete_clk).

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+

.....continued	
Supported Families	**Supported FlashPro Express Versions**
SmartFusion2	v12.4+

Example
The following example sets the VPUMP option to ON and uses a free running TCK at a frequency of 4 MHz (force_freq is set to OFF):

```
configure_flashpro4_prg -vpump {ON} \
-clk_mode {free_running_clk} \
-force_freq {OFF} \
-freq {4}
```

The following example sets the VPUMP option to ON, uses a discrete TCK and sets force_freq to ON at 2 MHz:

```
configure_flashpro4_prg -vpump {ON} \
-clk_mode {discrete_clk} \
-force_freq {ON} \
-freq {2}
```

See Also
- configure_flashpro3_prg
- configure_flashpro5_prg
- configure_flashpro6_prg

9.4 configure_flashpro5_prg

Description
This Tcl command changes FlashPro5 programmer settings. You can configure FlashPro programmer in Libero SoC Software or wia this command in FlashPro Express software. You will be able to set TCK/SCK frequency from the drop down list of frequencies you want to use for the programming.

```
configure_flashpro5_prg \
         -force_freq {ON} \
         -freq {freq}
```

Arguments

Parameter	Type	Description
force_freq	string	Forces the FlashPro software to use the TCK frequency specified by the software rather than the TCK frequency specified in the programmer file. Valid values are ON, OFF (default).
freq	integer	Specifies the TCK frequency in MHz. The default frequency is 4 MHz. It can be between 1 MHz to 6 MHz or it can be 10,15 or 30 MHz.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Error: freq: Invalid argument value: '50000000' (expecting 1000000, 2000000, 3000000, 4000000, 5000000, 6000000, 10000000, 15000000 or 30000000).
None	Error: force_freq: Invalid argument value: '' (expecting ON or OFF).

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
SmartFusion2	v12.4+

9.5 configure_flashpro6_prg

Description
This Tcl command changes FlashPro6 programmer settings. You can configure FlashPro6 programmer in Libero SoC Software or via this command in FlashPro Express software. You will be able to set VPUMP voltage for the programmer and force JTAG (Joint Test Action Group) or SPI (Serial Peripheral Interface bus) clock from the TCK (JTAG clock) or SCK (SPI clock) drop down list of frequencies you want to use for the programming. Similarly other programmer such as FlashPro5/4/3 can also selected and related options can be set.

```
configure_flashpro6_prg [-vpump {ON | OFF} ] \
[-force_freq {ON | OFF}] \
[-freq "freq"] \
[-force_sck_freq {ON | OFF} \
[-sck_freq "sck_freq"]
```

Arguments

Parameter	Type	Description
vpump	string	Enables FlashPro5 programmer to drive VPUMP. Set to ON to drive VPUMP. Valid value is ON (default) or OFF.
force_freq	string	Forces the FlashPro software to use the TCK frequency specified by the software rather than the TCK frequency specified in the programmer file. Valid values are ON, OFF(default).
freq	integer	Specifies the TCK frequency in MHz. TCK is used with a maximum frequency of 20 MHz, and the default frequency is 4 MHz. It can takes the value between 1 MHz to 6 MHz or it can be 10,15 or 30 MHz.
force_sck_freq	string	Forces the FlashPro software to use the SCK frequency. Valid values are ON, OFF(default).
sck_freq	floating	Specifies the SCK frequency in MHz. SCK is used with a maximum frequency of 40 MHz, and the default frequency is 20 MHz. Limitation of the SCK frequency for the selected programmer: 1.00, 2.00, 2.50, 3.33, 4.00, 5.00, 6.67, 8.00, 10.00, 13.33, 20.00 MHz

Return Type	Description
None	None

Error Codes

Error Code	Description

None	Parameter 'parameter_name' is not defined. Valid command formatting is
	`configure_flashpro6_prg [-vpump "ON | OFF"] \`
	`[-force_freq "ON | OFF"] \`
	`[-freq "freq"] \`
	`[-force_sck_freq "ON | OFF"] \`
	`[-sck_freq "sck_freq"]'`

None	Invalid 'freq' argument value: '4' (expecting 1000000, 2000000, 3000000, 4000000, 5000000, 6000000, 7000000, 8000000, 9000000, 10000000, 11000000, 12000000, 13000000, 14000000, 15000000, 16000000, 17000000, 18000000, 19000000 or 20000000).
None	Invalid 'sck_freq' argument value: '2' (expecting 400000, 500000, 600000, 700000, 800000, 1000000, 2000000, 2500000, 3330000, 4000000, 5000000, 6670000, 8000000, 10000000, 13330000, 20000000 or 40000000)

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
PolarFire SoC	v12.4+
SmartFusion2	v12.4+

Example
The following example sets TCK at a frequency of 4 MHz and sets force_freq to OFF:

```
configure_flashpro6_prg -force_freq {OFF} -freq {4}
```

The following example sets SCK at a frequency of 2 MHz and sets force_sck_freq to ON:

```
configure_flashpro6_prg -force_sck_freq {ON} -sck_freq {2}
```

See Also
- configure_flashpro3_prg
- configure_flashpro4_prg
- configure_flashpro5_prg

9.6 create_job_project

Description
This Tcl command creates FlashPro Express job using the programming job exported from Libero.

```
create_job_project -job_project_location {job project location} \
-job_file {path and name of job file} \
-overwrite value
```

Arguments

Parameter	Type	Description
job_project_location	string	Specifies the location for your FlashPro Express job (*.job) project. Must include the complete path to the *.pro file. If you do not provide the full path, FlashPro infers that you want to open the project from your current working directory.
job_file	string	Path to the Libero job file (*.job) that is used as input to create the Flashpro Express job project.
overwrite	boolean	Set value to TRUE, true or 1 to overwrite your existing job project. Valid values are: TRUE, true, 1, FALSE. false or 0.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	None

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following example creates a job project named test.job in the \fpexpress directory. It does not overwrite the existing job project:

```
create_job_project \
-job_project_location {D:\fpexpress} \
-job_file {D:\test\designer\test\export\test.job} -overwrite 0
```

See Also
- open_project

9.7 dump_tcl_support

Description
This Tcl command loads the list of supported FlashPro or FlashPro Express Tcl commands.

```
dump_tcl_support -file {filename}
```

Arguments

Parameter	Type	Description
file	string	Specifies absolute or relative path and the name of the file.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	None

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following example dumps your Tcl commands inot the file 'tcldump.tcl' and generates 'tcldump_syntax_error.tcl':

```
dump_tcl_support -file {tcldump.tcl}
```

9.8 enable_prg

Description
This Tcl command enables or disables the programmer specified. It will give error if the programmer specified for the -name option is not connected to that machine.

```
enable_prg \
-name {programmer_id} \
-enable {value}
```

Arguments

Parameter	Type	Description
name	string	Specify the programmer ID.
enable	boolean	Specify 1 or TRUE to enable the programmer, specify 0 or FALSE to disable the programmer. This is mandatory.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Programmer mentioned in the -name parameter is not connected to the machine
None	The programmer with name 'programmer ID' does not exist.
None	Required parameter 'enable' is missing.

.....continued

Error Code	Description
None	Required parameter 'name' is missing.
None	Parameter 'param_name' is not defined. Valid command formatting is enable_prg [-name "name"]+ -enable "TRUE | FALSE".

Supported Families

Supported Families	Supported
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following command enables the programmer with programmer ID 13802A15:

```
enable_prg -name {13802A15} -enable 1
```

9.9 enable_prg_type

Description
This Tcl command enables or disables the programmer of the type specified in the option -prg_type. If there are multipe programmers of same type connected to the machine, then use the enable_prg TCL command.

```
enable_prg_type \
-prg_type {programmer_type} \
-enable <value>
```

Arguments

Parameter	Type	Description
prg_type	string	Specify one of the following programmer type: FP | FP3 | FP4 | FP5 | FP6 | FP6Lite | PP. This is mandatory.
enable	boolean	Specifies 1 or TRUE to enable programmer, specifies 0 or FALSE to disables programmer. This is mandatory.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'prg_type' is missing.
None	Required parameter 'enable' is missing.
None	'type' is an invalid programming type. Please specify one of the following valid programming types: (FP|FP3|FP4|FPLite|PP)

.....continued

Error Code	Description
None	Parameter 'param_name' is not defined. Valid command formatting is enable_prg_type -prg_type "prg_type" -enable "TRUE | FALSE".

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following command enables the programmer with programmer type FP6:

```
enable_prg_type -prg_type FP6 -enable TRUE
```

See Also
- enable_prg
- ping_prg

9.10 export_script

Description
This Tcl command explicitly exports the Tcl command equivalents of the current FlashPro Express session. With this command you can re-execute the same commands interactively or in batch.

You must supply a file name with the -file parameter and the -relative_path parameter to specify whether an absolute or relative path is used in the exported script file.

```
export_script \
-file {absolute or relative path to exported file} \
-relative_path <value>
```

Arguments

Parameter	Type	Description
file	string	Specifies the absolute or relative path and name for the exported TCL script.
relative_path	boolean	Sets your option to use a relative or absolute path in the exported script; use 1 for relative path, 0 for absolute path.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Parameter 'file' has illegal value.

.....continued

Error Code	Description
None	Required parameter 'relative_path' is missing.

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following command exports the Tcl command equivalents of the current FlashPro Express session in exported.tcl:

```
export_script -file {./exported.tcl} -relative_path 1
```

9.11 open_project

Description
This Tcl command opens FlashPro Express project (*.pro) that was created in FPExpress.

```
open_project -project {path and name of the project file} \
[-connect_programmers value]
```

Arguments

Parameter	Type	Description
project	string	Specify the path of the FPExpress project with extension *.pro. Project path should include the complete path to the *.pro file. If you do not provide the full path, FlashPro Express infers that you want to open the project from your current working directory.
connect_programmers	boolean	Valid values are: TRUE, true, 1, FALSE. false or 0. This is optional. Default is 1.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'project' is missing.

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+

.....continued	
Supported Families	**Supported FlashPro Express Versions**
SmartFusion2	v12.4+

Example
This command opens the 'FPPrj1.pro' project from the FPProject1 directory:

```
open_project -project {./FPProject1/FPPrj1.pro} -connect_programmers 1
```

See Also
- close_project

9.12 ping_prg

Description
This is the FlashPro-specific tcl command that pings one or more programmers. Right-click a programmer and choose Ping.

Note: You can click the Refresh/Rescan for Programmers button to quickly ping new programmers.

```
ping_prg -name {name}
```

Arguments

Parameter	Type	Description
name	string	Specifies the programmer to be pinged. Repeat this argument for multiple programmers. See example below. This is mandatory.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'name' is missing.
None	Parameter 'name' is missing or has invalid value.
None	The programmer with name does not exist.

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following example pings the programmers 'FP300085' and 'FP30086':

```
ping_prg -name {FP300085} -name {FP300086}
```

See Also

- ping_prg

9.13 refresh_prg_list

Description

This Tcl command refreshes the programmer list. This is most often used to have FlashPro or FlashPro Express detect a programmer that you have just connected.

```
refresh_prg_list
```

Arguments

Parameter	Type	Description
None	None	None

Return Type	Description
None	None

Error Codes

Error Code	Description
None	None

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example

This command refreshes programmers list:

```
refresh_prg_list
```

9.14 remove_prg

Description

This Tcl command removes the programmer from the programmer list. Right-click a programmer to Ping, Self-Test, Scan, Check Chain or Remove it from the list.

```
remove_prg -name { name }
```

Arguments

Parameter	Type	Description
name	string	Specifies the programmer to be removed. You can repeat this argument for multiple programmers.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'name' is missing.
None	The programmer with name 'prg_name' does not exist.
None	Parameter 'param_name' is not defined. Valid command formatting is 'remove_prg [-name "name"]+'

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following example removes the programmer '03178' from the programmer list:

```
set prg_name "03178"
remove_prg -name "$prg_name"
```

9.15 run_selected_actions

Description
FlashPro-specific Tcl command. Runs the selected action on the specified programmer and returns the exit code from the action. If no programmer name is specified, the action is run on all connected programmers. Only one exit code is returned, so return code cannot be used when action is run on more than one programmer. A programming file must be loaded.

```
run_selected_actions [-name "name"] \
[-force_rtg4_erase "TRUE | FALSE"] \
[-prog_spi_flash "TRUE | FALSE"] \
[-disable_prog_design "TRUE | FALSE"] \
[-spi_flash_image "spi_flash_image"] \
[-spi_flash_action "spi_flash_action"]
```

Arguments

Parameter	Type	Description
name	string	Optional argument that specifies the programmer name. You can repeat this argument for multiple programmers.

.....continued

Parameter	Type	Description
force_rtg4_erase	boolean	This action is for RT4G device only. When the value of this option is set to "TRUE" the erase action will be run on RT4G device when erase action is executed in batch mode using TCL script.
disable_prog_design	boolean	Specify 1 or "TRUE" to disable device programming, specify 0 or "FALSE" to enable device programming.
prog_spi_flash	boolean	Specify 1 or "TRUE" to enable spi flash programming, sepcify 0 or "FALSE" to disable spi flash programming.
spi_flash_image	string	Provide the path of spi flash image (bin) file.
spi_flash_action	string	The value can be one of these: PROGRAM_SPI_IMAGE, VERIFY_SPI_IMAGE, READ_SPI_IMAGE, ERASE_SPI_FLASH.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	The programmer with name doesn't exist.
None	Parameter 'name' has illegal value.
None	Parameter 'argument_name' is not defined. Valid command formatting is 'run_selected_actions [-name "name"]* [-force_rtg4_erase "TRUE | FALSE"] \ [-prog_spi_flash "TRUE | FALSE"] [-disable_prog_design "TRUE | FALSE"] \ [-spi_flash_image "spi_flash_image"] [-spi_flash_action "spi_flash_action"]

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example

The following example runs the selected actionS on the programmers 'FP30085' and 'FP30086':

```
run_selected_actions -name {FP300085} -name {FP300086}
```

See Also

- set_programming_action

9.16 save_log

Description
This Tcl command saves your FlashPro or FlashPro Express log file. Equivalent to clicking the Project menu, and choosing Export Log File.

```
save_log -file {absolute or relative path of log file}
```

Arguments

Parameter	Type	Description
file	string	Specifies absolute or relative path and the name of the log file.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	None

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following example saves the log file with the name 'my_logfile.log':

```
save_log -file {my_logfile.log}
```

9.17 save_project

Description
This Tcl command saves the FlashPro or FlashPro Express project. Equivalent to clicking the Project menu, and choosing Save Job Project.

```
save_project
```

Arguments

Parameter	Type	Description
None	None	None

Return Type	Description
None	None

Error Codes

Error Code	Description
None	None

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
Saves the FlashPro/FlashPro Express job in your current working directory:

```
save_project
```

9.18 scan_chain_prg

Description
This Tcl command shows how the devices are ordered in the chain in the Log window. The scan chain operation scans and analyzes the JTAG chain connected to programmer(s) you have selected and checks that chain scanned matches the chain configured in FlashPro Express. To scan a chain: Right-click the programmer you want to scan and choose Scan and check chain. i.e. Device 1: 2A54CF1 Mfr: Microsemi Part: M2AA090T or Device 2: Unknown. In single mode, this command runs scan chain on a programmer. In chain mode, this command runs scan and check chain on a programmer if devices have been added in the grid.

```
scan_chain_prg -name { programmer_name }
```

Arguments

Parameter	Type	Description
name	string	Specifies the programmer name. This is mandatory.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'name' is missing.
None	Parameter 'param_name' is not defined. Valid command formatting is 'scan_chain_prg [-name "name"]
None	The programmer with name 'prg_name' does not exist

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following example runs scan chain on a single programmer (single mode) named 'E21428R':

```
set prg_name {E21428R}
scan_chain_prg -name "$prg_name"
```

9.19 self_test_prg

Description
This Tcl command Runs Self-Test on a programmer. Right-click a programmer to Ping, Self-Test, Scan, Check Chain or Remove it from the list.

Note: You must connect the programmer to the self-test board that comes with your programmer before performing a self-test. Self-test is not supported with FlashPro5/4 programmers. These programmers are rigorously tested at the factory during production.

```
self_test_prg -name { name }
```

Arguments

Parameter	Type	Description
name	string	Specifies the programmer name to run Self-Test. You can repeat this argument for multiple programmers.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'name' is missing.
None	The programmer with name 'prg_name' does not exist.
None	Parameter 'param_name' is not defined. Valid command formatting is 'self_test_prg [-name "name"]+'
None	programmer 'prg_name' : Self Test FAILED. Make sure the programmer is connected to the Loopback Test Board.

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following example runs Self-Test on a '03A178' programmer from the programmer list:

```
set prg_name "03A178"
self_test_prg -name "$prg_name"
```

9.20 set_prg_name

Description
This Tcl command changes the user name of a programmer. Enter the new programmer name in the Programmer window to rename the programmer. By default, the programmer name is the same as the programmer ID.

```
set_prg_name -name { name } -new_name { new_name }
```

Arguments

Parameter	Type	Description
name	string	Identifies the old programmer name.
new_name	string	Specifies the new programmer name.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Parameter 'param_name' is not defined. Valid command formatting is set_prg_name -name "name" -new_name "new_name"
None	Required parameter 'new_name' is missing.
None	The programmer with name 'some_device_name' does not exist.

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following example changes the name of the programmer 'FP300086' to 'FP3Prg2':

```
set prj_name "FP300086"
set_prg_name -name "$prj_name" -new_name {FP3Prg2}
```

9.21 set_programming_action

Description
This is the FlashPro-specific tcl command. Selects the action for a device. The device name parameter must be specified only in chain programming mode. A programming file must be loaded. The device must be a Microchip device.

```
set_programming_action [ -name {name} ] -action { action }
```

Arguments

Parameter	Type	Description
name	string	Specifies the device name. It is optional.
action	string	Specifies the action. This is mandatory. Valid values are: • PROGRAM - Programs all selected family features: FPGA Array, targeted eNVM clients and security settings. • ENC_DATA_AUTHENTICATION - Encrypted bitstream authentication data. This action is only visible if every device in the chain contains encrypted bitstream files. Selecting this action causes each bitstream file to be checked for authentication. • ERASE - Erases the selected family features: FPGA Array and security. • DEVICE_INFO - Displays the IDCODE, the design name, the checksum, and device security settings and programming environment information programmed into the device. • READ_IDCODE - Reads the device ID code from the device. • VERIFY - Verifies all selected family features: FPGA Array, targeted eNVM clients and security settings. • VERIFY_DIGEST - Calculates the digests for the components included in the bitstream and compares them against the programmed values.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'action' is missing.
None	You must specify the device name parameter for the command 'set_programming_action' in chain programming mode.
None	The device with name does not exist.

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following example sets the programming action in single programming mode::

```
set_programming_action -action {PROGRAM}
```

And in chain programming mode:

```
set_programming_action -name {MyDevice1} -action {ERASE}
```

See Also
- set_programming_action

9.22 set_programming_file

Description
This Tcl command Sets the programming file for a device. Either the -file or the -no_file flag must be specified. A programming file must be loaded(exported from Libero). The device must be a Microchip device.

```
set_programming_file [-name {name}] \
-file {absolute or relative path and the name of file} \
[-no_file]
```

Arguments

Parameter	Type	Description
name	string	Specifies the device name. This argument must be specified only in chain programming mode. It is optional.
file	stirng	Specifies the absolute or relative path and the name of programming file either stp or ppd file. stp and ppd are exported from libero - export bitstream dialog.
no_file	none	Specifies to unload the current programming file.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	You must specify the device name parameter for the command 'set_programming_file' in chain programming mode.
None	You must either specify the 'file' or the 'no_file' parameter.

.....continued

Error Code	Description
None	Parameter 'file' has illegal value.
None	The device with name 'device_name' does not exist.

Supported Families

Supported Families	Supported FlashPro Express Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Exceptions
Must be a Microchip device.

Example
The following command sets "e:/test/test.ppd" programming file in single programming mode:

```
set_programming_file -file {e:/test/test.ppd}
```

The following command sets "e:/test/test.stp" programming file in chain programming mode:

```
set_programming_file -name {MyDevice1} -file {e:/test/test.stp}
set_programming_file -name {MyDevice1} -no_file
```

9.23 set_programming_interface

Description
This Tcl command is used to select JTAG or SPI_SLAVE interface for programming.

```
set_programming_interface -interface {interface}
```

Arguments

Parameter	Type	Description
interface	string	Specifies JTAG or SPI_SLAVE programming.

Return Type	Description
None	None

Error Codes

Error Code	Description

Supported Families

Supported Families	Supported Versions
PolarFire	v12.4+

.....continued

Supported Families	Supported Versions
SmartFusion2	v12.4+

Example
The following example selects SPI_SLAVE programming:

```
set_programming_interface -interface {spi_slave}
```

See Also

-

9.24 set_spi_flash_action

Description
This Tcl command specifies SPI Flash programming action. You can program, verify or erase SPI Flash using this command.

```
set_spi_flash_action [-name {name}] -spi_flash_action {action}
```

Arguments

Parameter	Type	Description
name	string	Specifies the device name. It is optional.
spi_flash_action	string	Specifies one of the following actions: PROGRAM_SPI_IMAGE, VERIFY_SPI_IMAGE, READ_SPI_IMAGE, ERASE_SPI_FLASH. • PROGRAM_SPI_IMAGE: This action will erase the entire SPI flash then program the SPI image. • VERIFY_SPI_IMAGE: This action verifies the SPI Image on the SPI Flash. • READ_SPI_IMAGE: This action reads the SPI Image from the SPI Flash. • ERASE_SPI_FLASH: This action erases the entire SPI Flash.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'spi_flash_action' is missing.
None	You must specify the device name parameter for the command 'set_spi_flash_action' in chain programming mode.

.....continued

Error Code	Description
None	The action 'prg_action' is not supported. You must select the programming action from this list: 'PROGRAM_SPI_IMAGE, VERIFY_SPI_IMAGE, READ_SPI_IMAGE, ERASE_SPI_FLASH'.
None	The device with 'device_name' name does not exist.

Supported Families

Supported Families	Supported Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following example sets the "VERIFY_SPI_IMAGE" SPI Flash programming action in single programming mode, verifies the SPI Image on the SPI Flash:

```
set_spi_flash_action -name {MPFS250T_ES} \
                     -spi_flash_action {VERIFY_SPI_IMAGE}
```

9.25 set_spi_flash_file

Description
This Tcl command specifies SPI Flash programming file(.bin) to be associated with the device.

```
set_spi_flash_file [-name {name}] \
-file {path and the name of the programming file} \
-no_file
```

Arguments

Parameter	Type	Description
name	string	This argument must be specified only in chain programming mode. It is optional.
file	string	Specifies the SPI Flash programming file *.bin.
no_file	none	Specifies to unload/unspecify the SPI Flash programming file.

Return Type	Description
None	None

Error Codes

Error Code	Description
None	Required parameter 'action' is missing.
None	The device with name does not exist.

Supported Families

Supported Families	Supported Versions
PolarFire	v12.4+
SmartFusion2	v12.4+

Example
The following example sets the "VERIFY_SPI_IMAGE" SPI Flash programming action in single programming mode:

```
set_spi_flash_file -name {MPFS250T_ES} \
                   -spi_flash_action {VERIFY_SPI_IMAGE}
```

10. SmartDebug Tcl Commands

10.1 add_probe_insertion_point

This Tcl command adds probe points to be connected to user-specified I/Os for probe insertion flow.

```
add_probe_insertion_point -net net_name -driver driver -pin package_pin_name -port
port name
```

10.1.1 Arguments

`-net net_name`

Name of the net used for probe insertion.

`-driver driver`

Driver of the net.

`-pin package_pin_name`

Package pin name (i.e. I/O to which the net will be routed during probe insertion).

`-port port_name`

User-specified name for the probe insertion point.

10.1.2 Example

```
add_probe_insertion_point -net {count_out_c[0]} -driver
{Counter_8bit_0_count_out[0]:Q} - pin {H5} -port {Probe_Insert0}
```

10.2 add_to_probe_group

Tcl command; adds the specified probe points to the specified probe group.

```
add_to_probe_group -name probe_name -group group_name
```

10.2.1 Arguments

`-name probe_name`

Specifies one or more probes to add.

`-group group_name`

Specifies name of the probe group.

10.2.2 Example

```
add_to_probe_group -name out[5]:out[5]:Q \
-name grp1.out[3]:out[3]:Q \
-name out.out[1].out[1]:Q \
-group my_new_grp
```

10.3 construct_chain_automatically

This Tcl command automatically starts chain construction for the specified programmer.

```
construct_chain_automatically -name {programmer_name}
```

10.3.1 Arguments

`-name`

Specify the device (programmer) name. This argument is mandatory.

10.3.2 Example

For a single programmer:

```
construct_chain_automatically -name {21428}
```

See Also scan_chain_prg enable_device set_debug_programmer set_device_name set_programming_file set_programming_action run_selected_actions

10.4 create_probe_group

Tcl command; creates a new probe group.

```
create_probe_group -name group_name
```

10.4.1 Arguments

`-name group_name`

Specifies the name of the new probe group.

PolarFire

10.4.2 Example

```
create_probe_group -name my_new_grp
```

10.5 delete_active_probe

Tcl command; deletes either all or the selected active probes.

Note: You cannot delete an individual probe from the Probe Bus.

```
delete_active_probe -all | -name probe_name
```

10.5.1 Arguments

`-all`

Deletes all active probe names.

`-name probe_name`

Deletes the selected probe names.

10.5.2 Example

```
delete -all <- deletes all active probe names delete -name out[5]:out[5]:Q \
-name my_grp1.out[1]:out[1]:Q #deletes the selected probe names delete -name my_grp1 \
-name my_bus #deletes the group, bus and their members.
```

10.6 enable_device

This Tcl command enables or disables a device in the chain. When the device is disabled, it is bypassed. The device must be a Microsemi device.

```
enable_device -name {device_name} -enable {1 | 0}
```

10.6.1 Arguments

`-name`

Specify the device name. This argument is mandatory.

`-enable`

Specify the enable device. This argument is mandatory.

10.6.2 Example

```
enable_device -name {MPF300 (T_ES|TS_ES)} -enable 1
```

10.6.2.1 See Also

construct_chain_automatically scan_chain_prg

set_debug_programmer set_device_name set_programming_file set_programming_action run_selected_actions

10.7 event_counter

The event_counter Tcl command runs on signals that are assigned to channel A on the live probe, and displays the total events. It can be run before or after setting the live probe signal to channel A. The user specifies the duration to run the event_counter command.

`event_counter -run -stop –after` *`duration_in_seconds`*

10.7.1 Arguments

`-run`

Run event_counter.

`-stop`

Stop event_counter.

`-after` *`duration_in_seconds`*

Duration to stop event_counter. Specified by the user. This argument is required when `-stop` is specified.

10.7.2 Example

```
set_live_probe -probeA {count_out_c[0]:Counter_8bit_0_count_out[0]:Q} -probeB {}
event_counter -run event_counter -stop -after 10
```

10.8 export_smart_debug_data

`export_smart_debug_data [`*`device_components`*`] [`*`bitstream_components`*`] [-file_name {`*`file`*`} [- export_dir {`*`dir`*`}] [-force_rtg4_otp `*`0 | 1`*`]`

Tcl command; exports debug data for the SmartDebug application.

The command corresponds to the Export SmartDebug Data tool in Libero. The command creates a file with the extension “ddc” that contains data based on selected options. This file is used by SmartDebug (standalone application) to create a new SmartDebug project, or it can be imported into a device in SmartDebug (standalone application).

- If you do not specify any design components, all components available in the design will be included by default except the bitstream components.
- The generate_bitstream parameter is required if you want to generate bitstream file and include it in the exported file.
  - You must specify the bitstream components you want to include in the generated bitstream file or all available components will be included.
  - If you choose to include bitstream, and the design has custom security, the custom security bitstream component must be included.

10.8.1 Arguments

device_components

The following device components can be selected. Specify "1" to include the component, and "0" if you do not want to include the component.

-probes <1|0>

-package_pins <1|0>

-memory_blocks <1|0>

-security_data <1|0>

-chain <1|0>

-programmer_settings <1|0>

-ios_states <1|0>

bitstream_components

The following bitstream components can be selected. Specify "1" to include the component, and "0" if you do not want to include the component.

-generate_bitstream <1|0>

-bitstream_security <1|0>

-bitstream_fabric <1|0>

-bitstream_snvm <1|0>

-file_name *file*

Name of exported file with extension “ddc”.

-export_dir *dir*

Location where DDC file will be exported. If omitted, design export folder will be used.

10.8.2 Example

The following examples shows the export_smart_debug_data command with all parameters.

SmartFusion2, IGLOO2, RTG4 example:

```
export_smart_debug_data \
-file_name {sd1} \
-export_dir {d:\sd_prj\test3T\designer\sd1\export} \
-force_rtg4_otp 1 \
-probes 1 \
-package_pins 0 \
-memory_blocks 1 \
-envm_data 0 \
-security_data 1 \
-chain 1 \
-programmer_settings 1 \
-ios_states 1 \
-generate_bitstream 0 \
-bitstream_security 0 \
-bitstream_fabric 0 \
```

```
-bitstream_envm 0
```

PolarFire example:

```
export_smart_debug_data \
-file_name "top" \
-export_dir "./" \
-probes 1 \
-package_pins 0 \
-memory_blocks 1 \
-security_data 1 \
-chain 1 \
-programmer_settings 1 \
-ios_states 1 \
-generate_bitstream 1 \
-bitstream_security 0 \
-bitstream_fabric 1 \
-bitstream_snvm 1
```

The following example shows the command with no parameters:

```
export_smart_debug_data
```

10.9 fhb_control

```
fhb_control
-halt -clock_domain clkDomName(s)/all
-run -clock_domain clkDomName(s)
-step number_of_steps -clock_domain clkDomName(s)
-reset -clock_domain clkDomName(s)
-arm_trigger –trigger_signal liveProbePoint –trigger_edge_select rising -delay value -
clock_domain clkDomName(s)
```

This Tcl command provides FPGA Hardware Breakpoint (FHB) feature capability for SmartDebug.

-disarm trigger -clock_domain *clkDomName(s)/all*
-capture_waveform *number_of_steps* -vcd_file *target_file_name*
-clock_domain_status -clock_domain *clkDomName(s)/all*

10.9.1 Arguments

`-halt`

Specifies to halt the clock.

`-clock_domain` *clkDomName(s)/all*

Specifies clock domain names to halt. Can be single or multiple clock domains, halted in order specified by user.

`-run`

Specifies to run the clock.

-clock_domain *clkDomName(s)*

Specifies clock domain names to run. Can be single or multiple clock domains, releasing the user clock based on order specified.

-step *number_of_steps*

Specifies to step the clock “number_of_steps” times. Minimum value is 1.

-clock_domain *clkDomName(s)*

Specifies clock domain names to step. Can be single or multiple clock domains.

-reset

Specifies to reset FHB configuration for the specified clock domain.

-clock_domain *clkDomName(s)*

Specifies clock domain names to reset. Can be single or multiple clock domains.

-arm_trigger

Specifies to arm FHB configuration for the specified clock domain.

-trigger_signal *liveProbePoint*

Set the trigger signal to arm the FHBs.

-trigger_edge_select *rising*

Specifies the trigger signal edge to arm the FHBs. FHBs will be armed on rising edge of trigger signal.

-delay *value*

-clock_domain *clkDomName(s)*

Specifies clock domain names to be armed by the trigger signal. Can be single or multiple clock domains.

-disarm_trigger

Specifies to disarm FHB configuration for the specified clock domain.

-clock_domain *clkDomName(s)*

Specifies clock domain names to be reset by the trigger signal. Can be single or multiple clock domains.

-capture_waveform *number_of_steps*

Specifies to capture waveform of all the added signals to active probes in the specified clock domain for number_of_steps.

- vcd_file *target_file_name*

Target file to save the data and see the waveform.

-clock_domain_status *clkDomName(s)/all*

Specifies to read and display status of specified clock domain(s). Can be single or multiple clock domains.

10.9.2 Examples

```
fhb_control -halt -clock_domain {“FCCC_0/GL0_INST “ “FCCC_0/GL1_INST” } fhb_control
-run -clock_domain {“FCCC_0/GL0_INST “ “FCCC_0/GL1_INST” } fhb_control -step -
clock_domain {“FCCC_0/GL0_INST “ “FCCC_0/GL1_INST” } fhb_control -reset -clock_domain
{“FCCC_0/GL0_INST “ “FCCC_0/GL1_INST” }

fhb_control -arm_trigger -trigger_signal {q_0_c[14]:count_1_q[14]:Q}

-trigger_edge_select {rising} - delay 0 - clock_domain {"FCCC_0/GL0_INST"}

fhb_control -disarm_trigger -trigger_signal {q_0_c[14]:count_1_q[14]:Q}

-trigger_edge_select {rising} - delay 0 - clock_domain {"FCCC_0/GL0_INST"} fhb_control
-capture_waveform {10} -vcd_file {D:/wvf_location/waveform.vcd}
```

```
fhb_control – clock_domain_status – clock_domain { "FCCC_0/GL0_INST" "FCCC_0/GL1_INST"
"FCCC_0/GL2_INST" }
```

10.10 frequency_monitor

The frequency_monitor Tcl command calculates the frequency of a signal that is assigned to live probe A.

```
run_frequency_monitor -signal signal_name –time duration
```

10.10.1 Arguments

`-signal signal_name`

Specifies the signal name.

`–time duration`

Specifies the duration to run the command. The value can be 0.1, 1, 5, 8, or 10.

10.11 get_programmer_info

This Tcl command lists the IDs of all FlashPRO programmers connected to the machine.

```
get_programmer_info
```

This command takes no arguments.

10.11.1 Example

```
set a [get_programmer_info]
```

10.12 load_active_probe_list

Tcl command; loads the list of probes from the file.

```
load_active_probe_list –file file_path
```

10.12.1 Arguments

`-file file_path`

The input file location.

10.12.2 Example

```
load_active_probe_list -file "./my_probes.txt"
```

10.13 loopback_mode

This Tcl command applies loopback to a specified lane.

```
loopback_mode -lane {Physical_Location} -apply -type {loopback_type}
```

10.13.1 Arguments

`-lane {Physical_Location}`

Specify the physical location of the lane.

`-apply`

Apply specified loopback to specified lane.

`-type {loopback_type}`

Specify the loopback type to apply.

10.13.2 Examples

```
loopback_mode -lane {Q3_LANE2} -apply -type {EQ-NearEnd} loopback_mode -lane
{Q3_LANE0} -apply -type {EQ-FarEnd} loopback_mode -lane {Q0_LANE0} -apply -type
{CDRFarEnd} loopback_mode -lane {Q0_LANE1} -apply -type {NoLpbk} loopback_mode -lane
{Q1_LANE2} -apply -type {EQ-FarEnd} loopback_mode -lane {Q1_LANE0} -apply -type
{NoLpbk} loopback_mode -lane {Q2_LANE2} -apply -type {EQ-NearEnd} loopback_mode -lane
{Q2_LANE3} -apply -type {CDRFarEnd}
```

10.14 move_to_probe_group

Tcl command; moves the specified probe points to the specified probe group.

Note: Probe points related to a bus cannot be moved to another group.

```
move_to_probe_group -name probe_name -group group_name
```

10.14.1 Arguments

`-name probe_name`

Specifies one or more probes to move.

`-group group_name`

Specifies name of the probe group.

10.14.2 Example

```
move_to_probe_group -name out[5]:out[5]:Q \
-name grp1.out[3]:out[3]:Q \
-group my_grp2
```

10.15 optimize_dfe

This Tcl command supports the Optimize DFE feature in SmartDebug.

```
optimize_dfe -dfe_algorithm <type of dfe algorithm> -lane <lane(s) configured in the
design>
```

10.15.1 Arguments

`-dfe_algorithm`

This command executes Dfe Algorithm with type of dfe algorithm and lanes as input. Algorithm selection has two options:

`software_based` – executes DfeSs.tcl script `xcvr_based` –executes internal Dfe Auto Calibration. This argument is mandatory.

`-lane`

List of lane(s) configured in the design. This argument is mandatory.

10.15.2 Examples

```
optimize_dfe -lane {"Q2_LANE0"} -dfe_algorithm {software_based} optimize_dfe -lane
{"Q2_LANE0"} -dfe_algorithm {xcvr_based} optimize_dfe -lane {"Q2_LANE0" "Q0_LANE0"} -
dfe_algorithm {xcvr_based}
```

10.16 pcie_config_space

This Tcl command displays the value of the entered parameter in the SmartDebug log window and return thes

register:field value to the Tcl.

pcie_config_space -pcie_block_name {*pcie_block_name*} -param_name {*param name*}

10.16.1 Arguments

-pcie_block_name {*pcie_block_name*}

Complete logical hierarchy of the PCIE block whose status is to be read from the device. This parameter is mandatory.

-param_name {*param name*}

Parameter name to read from the device. This parameter is mandatory.

10.16.2 Example

pcie_config_space -pcie_block_name {sb_0/CM1_Subsystem/my_pcie_0} -param_name

{neg_max_payload}

Output Display in SmartDebug window: 512 bytes

Return value to the tcl script: 0x2

10.17 pcie_ltssm_status

This Tcl command displays the current LTSSM state from the PLDA core in the SmartDebug log window and returns the register:field value to the Tcl.

pcie_ltssm_status -pcie_block_name {*pcie_block_name*}

10.17.1 Arguments

-pcie_block_name {*pcie_block_name*}

Complete logical hierarchy of the PCIE block whose status is to be read from the device. This parameter is mandatory.

10.17.2 Example

pcie_ltssm_status -pcie_block_name {sb_0/CM1_Subsystem/my_pcie_0}

Output Display in SmartDebug window: Configuration.Linkwidth.start

Return value to the tcl script: 0x2

10.18 plot_eye

This Tcl command is used to plot eye and export eye plots.

plot eye -lane {*lane_instance_name*} -export_dir {*location_path*}

10.18.1 Arguments

-lane

Specify the lane instance name.

-export_dir

Specify the path to the location where the file is to be exported.

10.18.2 Example

plot_eye -lane {Q2_LANE0} - export_dir {E:\designs\G5\SERDES\ export.txt}

10.19 program_probe_insertion

This Tcl command runs the probe insertion flow on the selected nets.

```
program_probe_insertion
```

This command takes no arguments.

10.20 read_active_probe

`read_active_probe [-deviceName device_name] [-name probe_name]` [-group_name `bus_name`| `group_name`] [- value_type b|h]`[-file file_path]`

Tcl command; reads active probe values from the device. The target probe points are selected by the select_active_probe command.

10.20.1 Arguments

`-deviceName device_name`

Parameter is optional if only one device is available in the current configuration.

`-name probe_name`

Instead of all probes, read only the probes specified. The probe name should be prefixed with bus or group name if the probe is in the bus or group.

`-group_name bus_name | group_name`

Instead of all probes, reads only the specified buses or groups specified here.

`-value_type b | h`

Optional parameter, used when the read value is stored into a variable as a string. b = binary

h = hex

`-file file_path`

Optional. If specified, redirects output with probe point values read from the device to the specified file.

Note: When the user tries to read at least one signal from the bus/group, the complete bus or group is read. The user is presented with the latest value for all the signals in the bus/group.

10.20.2 Example

`read_active_probe -group_name {bus1} read_active_probe -group_name {group1}` To save into variable:

`set a [read_active_probe -group_name {bus_name} -value_type h]` #save read data in hex string

If read values are stored into a variable without specifying value_type parameter, it saves values as a binary string by default.

Example

`set a [read_active_probe ]` #sets variable a as binary string of read values after read_active_probe command.

10.21 read_lsram

Tcl command; reads a specified block of large SRAM from the device.

10.21.1 Physical block

`read_lsram -name block_name -fileName file_name`

10.21.2 Arguments

`-name block_name`

Specifies the name for the target block.

`-fileName file_name`

Optional; specifies the output file name for the data read from the device.

10.21.3 Exceptions

- Array must be programmed and active
- Security locks may disable this function

10.21.4 Example

Reads the LSRAM Block Fabric_Logic_0/U2/F_0_F0_U1/ramtmp_ramtmp_0_0/INST_RAM1K20_IP from the PolarFire device and writes it to the file output.txt.

```
read_lsram -name {Fabric_Logic_0/U2/F_0_F0_U1/ramtmp_ramtmp_0_0/INST_RAM1K20_IP} -
fileName {output.txt}
```

10.21.5 Logical block

`read_lsram -logicalBlockName block_name -port port_name`

10.21.6 Arguments

`-logicalBlockName block_name`

Specifies the name for the user defined memory block.

`-port port_name`

Specifies the port for the memory block selected. Can be either Port A or Port B.

10.21.7 Example

```
read_lsram -logicalBlockName {Fabric_Logic_0/U2/F_0_F0_U1} -port {Port A}
```

10.22 read_usram

Tcl command; reads a uSRAM block from the device.

10.22.1 Physical block

`read_usram [-name block_name] -fileName file_name`

10.22.2 Arguments

`-name block_name`

Specifies the name for the target block.

`-fileName file_name`

Optional; specifies the output file name for the data read from the device.

10.22.3 Exceptions

- Array must be programmed and active
- Security locks may disable this function

10.22.4 Example

Reads the uSRAM Block Fabric_Logic_0/U3/F_0_F0_U1/ramtmp_ramtmp_0_0/INST_RAM64x12_IP from the PolarFire device and writes it to the file sram_block_output.txt.

```
read_usram -name {Fabric_Logic_0/U3/F_0_F0_U1/ramtmp_ramtmp_0_0/INST_RAM64x12_IP} -
fileName {output.txt}
```

10.22.5 Logical block

```
read_usram -logicalBlockName block_name -port port_name
```

10.22.6 Arguments

`-logicalBlockName block_name`

Specifies the name of the user defined memory block.

`-port port_name`

Specifies the port of the memory block selected. Can be either Port A or Port B.

10.22.7 Example

```
read_usram -logicalBlockName {Fabric_Logic_0/U3/F_0_F0_U1} -port {Port A}
```

10.23 remove_from_probe_group

Tcl command; removes the specified probe points from the group. That is, the removed probe points won't be associated with any probe group.

Note: Probes cannot be removed from the bus.

```
remove_from_probe_group -name probe_name
```

10.23.1 Arguments

`-name probe_name`

Specifies one or more probe points to remove from the probe group.

10.23.2 Example

The following command removes two probes from my_grp2.

```
Move_out_of_probe_group -name my_grp2.out[3]:out[3]:Q \
-name my_grp2.out[3]:out[3]:Q
```

10.24 remove_probe_insertion_point

This Tcl command deletes an added probe from the probe insertion UI.

```
remove_probe_insertion_point –net net_name -driver driver
```

10.24.1 Arguments

`-net net_name`

Name of the existing net which is added using the `add_probe_insertion_point` command.

`-driver driver`

Driver of the net.

10.24.2 Example

```
remove_probe_insertion_point -net {count_out_c[0]} -driver
{Counter_8bit_0_count_out[0]:Q}
```

10.25 run_selected_actions

This Tcl command is used to run the selected action for a device.

```
run_selected_actions
```

This command takes no arguments.

10.25.1 Example

```
set_programming_action -name {MPF300(T_ES|TS_ES)} -action {DEVICE_INFO}
set_programming_action -name {M2S/M2GL090(T|TS|TV)} -action {ERASE}
```

10.25.1.1 See Also

construct_chain_automatically

scan_chain_prg enable_device set_debug_programmer set_device_name set_programming_file set_programming_action

10.26 save_active_probe_list

Tcl command; saves the list of active probes to a file.

```
save_active_probe_list -file file_path
```

10.26.1 Arguments

`-file` *`file_path`*

The output file location.

10.26.2 Example

```
save_active_probe_list -file "./my_probes.txt"
```

10.27 scan_chain_prg

In single mode, this Tcl command runs scan chain on a programmer. In chain mode, this Tcl command runs scan and check chain on a programmer if devices have been added in the grid.

```
scan_chain_prg -name {programmer_name}
```

10.27.1 Arguments

`-name`

Specify the device (programmer) name. This argument is mandatory.

10.27.2 Example

```
scan_chain_prg -name {21428}
```

10.27.2.1 See Also

construct_chain_automatically enable_device set_debug_programmer set_device_name set_programming_file set_programming_action run_selected_actions

10.28 scan_ecc_memories

This Tcl command scans and reports any errors detected in the PolarFire, PolarFire SoC, and RTG4 TPSRAM blocks that have ECC enabled.

```
scan_ecc_memories -filename {file name}
```

10.28.1 Arguments

`-filename`

Specify the name of the file where output is redirected. This argument is mandatory.

10.28.2 Example

```
scan_ecc_memories -filename {./output.txt}
```

10.29 select_active_probe

Tcl command; manages the current selection of active probe points to be used by active probe READ operations. This command extends or replaces your current selection with the probe points found using the search pattern.

`select_active_probe [-deviceName device_name] [-name probe_name_pattern] [-reset true|false]`

10.29.1 Arguments

`-deviceName device_name`

Parameter is optional if only one device is available in the current configuration..

`-name probe_name_pattern`

Specifies the name of the probe. Optionally, search pattern string can specify one or multiple probe points. The pattern search characters “*” and “?” also can be specified to filter out the probe names.

-reset *true | false*

Optional parameter; resets all previously selected probe points. If name is not specified, empties out current selection.

10.29.2 Example

The following command selects three probes. In the below example, “grp1” is a group and “out” is a bus..

```
Select_active_probe -name out[5]:out[5]:Q Select_active_probe -name
out.out[1]:out[1]:Q \
-name out.out[3]:out[3]:Q \
-name out.out[5]:out[5]:Q
```

10.30 set_live_probe

Tcl command; set_live_probe channels A and/or B to the specified probe point(s). At least one probe point must be specified. Only exact probe name is allowed (i.e. no search pattern that may return multiple points).

`set_live_probe [-deviceName device_name] [-probeA probe_name] [-probeB probe_name]`

10.30.1 Arguments

`-deviceName device_name`

Parameter is optional if only one device is available in the current configuration or set for debug (see SmartDebug user guide for details).

`-probeA probe_ name`

Specifies target probe point for the probe channel A.

-probeB *probe_ name*

Specifies target probe point for the probe channel B.

10.30.2 Exceptions

- The array must be programmed and active
- Active probe read or write operation will affect current settings of Live probe since they use same probe circuitry inside the device
- Setting only one Live probe channel affects the other one, so if both channels need to be set, they must be set from the same call to set_live_probe
- Security locks may disable this function
- In order to be available for Live probe, ProbeA and ProbeB I/O's must be reserved for Live probe respectively

10.30.3 Example

Sets the Live probe channel A to the probe point A12 on device MPF300TS_ES.

```
set_live_probe [-deviceName MPF300TS_ES] [-probeA A12]
```

10.31 set_debug_programmer

This Tcl command is used to set the debug programmer.

```
set_debug_programmer -name {programmer_name}
```

10.31.1 Arguments

`-name`

Specify the programmer. This argument is mandatory.

10.31.2 Example

```
set_debug_programmer -name {S201YQST1V}
```

10.31.2.1 See Also

construct_chain_automatically scan_chain_prg

enable_device set_device_name set_programming_file set_programming_action

run_selected_actions

10.32 set_programming_action

This Tcl command is used to select the action for a device.

```
set_programming_action [-name {device_name}] -action {procedure_action}
```

10.32.1 Arguments

`-name`

Specify the device name. This argument is mandatory.

`-action`

Specify the programming action. This argument is mandatory.

10.32.2 Example

```
set_programming_action -name {MPF300(T_ES|TS_ES)} -action {DEVICE_INFO}
set_programming_action -name {M2S/M2GL090(T|TS|TV)} -action {ERASE}
```

10.32.2.1 See Also

construct_chain_automatically scan_chain_prg

enable_device set_debug_programmer set_device_name set_programming_file run_selected_actions

10.33 set_programming_file

This Tcl command is used to set the programming file for a device. Either the file or the no_file flag must specified. A programming file must be loaded. The device must be a Microsemi device.

```
set_programming_file -name {device_name} -file {stapl_file_name_with_path}
```

10.33.1 Arguments

`-name`

Specify the device name. This argument is mandatory.

`-file`

Specify the *file* path. This argument is mandatory.

10.33.2 Example

```
set_programming_file -name {MPF300(T_ES|TS_ES)} -file
{D:/export/CM1_PCIE_TOP_default_uic_12_200_0_12.stp}
```

10.33.2.1 See Also

construct_chain_automatically scan_chain_prg

enable_device set_debug_programmer

set_device_name set_programming_action run_selected_actions

10.34 smartbert_test

This Tcl command is used for the following:

- Start a Smart BERT test
- Stop a Smart BERT test
- Reset error count

10.34.1 smartbert_test -start

This Tcl command starts a Smart BERT test with a specified pattern on a specified lane.

```
smartbert_test -start -pattern {pattern_type} -lane {Physical_Location}
```

10.34.2 Arguments

`-start`

Start the Smart BERT test.

`pattern {pattern_type}`

Specify the pattern type of the Smart BERT test.

`-lane{Physical_Location}`

Specify the physical location of the lane.

`-EQ-NearEndLoopback`

Enable EQ-Near End Loopback on specified lane.

10.34.3 Examples

```
smartbert_test -start -pattern {prbs9} -lane {Q0_LANE3} smartbert_test -start -pattern
{prbs23} -lane {Q3_LANE2} smartbert_test -start -pattern {prbs7} -lane {Q3_LANE1}
```

```
smartbert_test -start -pattern {prbs31} -lane {Q1_LANE2} -EQ-NearEndLoopback
smartbert_test -start -pattern {prbs9} -lane {Q2_LANE2} -EQ-NearEndLoopback
smartbert_test -start -pattern {prbs15} -lane {Q2_LANE3} -EQ-NearEndLoopback
```

10.34.4 smartbert_test -stop

This Tcl command stops a Smart BERT test on a specified lane.

```
smartbert_test -stop -lane {Physical_Location}
```

10.34.5 Arguments

`-stop`

Stop the smart BERT test.

`-lane {Physical_Location}`

Specify the physical location of the lane.

10.34.6 Examples

```
smartbert_test -stop -lane {Q0_LANE0} smartbert_test -stop -lane {Q0_LANE3}
smartbert_test -stop -lane {Q3_LANE2}
```

```
smartbert_test -stop -lane {Q3_LANE1} smartbert_test -stop -lane {Q1_LANE2}
smartbert_test -stop -lane {Q2_LANE2} smartbert_test -stop -lane {Q2_LANE3}
```

10.34.7 smartbert_test -reset_counter

This Tcl command resets a lane error counter.

```
smartbert_test -reset_counter -lane {Physical_Location}
```

10.34.8 Arguments

`-reset_counter`

Reset lane error counter on hardware and cumulative error count on the UI.

`-lane {Physical_Location}`

Specify the physical location of the lane.

10.34.9 Examples

```
smartbert_test -reset_counter -lane {Q0_LANE0} smartbert_test -reset_counter -
lane {Q3_LANE2} smartbert_test -reset_counter -lane {Q2_LANE3} smartbert_test -
reset_counter -lane {Q2_LANE2} smartbert_test -reset_counter -lane {Q1_LANE2}
smartbert_test -reset_counter -lane {Q3_LANE1}
```

10.35 static_pattern_transmit

This Tcl command starts and stops a Static Pattern Transmit.

```
static_pattern_transmit -start -lane {Physical_Location} -pattern {pattern_type} -
value
```

```
{user_pattern_value}
```

10.35.1 static_pattern_transmit -start

Parameters

`-start`

Start the Static Pattern Transmit.

`-lane {Physical_Location}`

Specify physical location of lane.

`-pattern {pattern_type}`

Specify pattern_type of Static Pattern Transmit.

-value {*user_pattern_value*

Specify user_pattern_value in hex if pattern_type selected is custom.

10.35.2 Examples

```
static_pattern_transmit -start -lane {Q0_LANE0} -pattern {fixed}
static_pattern_transmit -start -lane {Q0_LANE2} -pattern {maxrunlength} -value
{} static_pattern_transmit -start -lane {Q3_LANE2} -pattern {custom} -value
{df} static_pattern_transmit -start -lane {Q3_LANE0} -pattern {fixed} -value {}
static_pattern_transmit -start -lane {Q1_LANE1} -pattern {custom} -value {4578}
static_pattern_transmit -start -lane {Q1_LANE2} -pattern {fixed} -value {}

static_pattern_transmit -start -lane {Q2_LANE2} -pattern {maxrunlength} -value {}
static_pattern_transmit -start -lane {Q2_LANE1} -pattern {custom} -value {abcdef56}
```

10.35.3 static_pattern_transmit -stop

```
static_pattern_transmit -stop -lane {Physical_Location}
```

10.35.4 Parameters

-stop

Stop the Static Pattern Transmit.

-lane {*Physical_Location*}

Specify physical location of lane.

10.35.5 Examples

10.36 ungroup

```
static_pattern_transmit -stop -lane {Q0_LANE0} static_pattern_transmit -stop -lane
{Q0_LANE2} static_pattern_transmit -stop -lane {Q3_LANE2} static_pattern_transmit
-stop -lane {Q3_LANE0} static_pattern_transmit -stop -lane {Q1_LANE1}
static_pattern_transmit -stop -lane {Q1_LANE2} static_pattern_transmit -stop -lane
{Q2_LANE2} static_pattern_transmit -stop -lane {Q2_LANE1}
```

Tcl command; disassociates the probes as a group.

```
nngroup -name group_name
```

10.36.1 Arguments

-name *group_name*

Name of the group.

10.36.2 Example

```
ungroup –name my_grp4
```

10.37 unset_live_probe

Tcl command; discontinues the debug function and clears live probe A, live probe B, or both probes (Channel A/Channel B). An all zeros value is shown in the oscilloscope.

```
unset_live_probe -probeA 1 -probeB 1 [-deviceName device_name]
```

10.37.1 Arguments

-probeA

Live probe Channel A.

`-probeB`

Live probe Channel B.

`-deviceName device_name`

Parameter is optional if only one device is available in the current configuration or set for debug (see the SmartDebug User Guide for Libero or the SmartDebug User Guide for PolarFire for details).

10.37.2 Exceptions

- The array must be programmed and active.
- Active probe read or write operation affects current of Live Probe settings, because they use the same probe circuitry inside the device.
- Security locks may disable this function.

10.37.3 Example

The following example unsets live probe Channel A from the device MPF300TS_ES.

```
unset_live_probe -probeA 1[-deviceName MPF300TS_ES]
```

10.38 uprom_read_memory

This Tcl command reads a uPROM memory block from the device.

`read_uprom_memory -startAddress {hex_value} -words {integer_value}`

10.38.1 Arguments

`-startAddress hex_value`

Specifies the start address of the uPROM memory block.

`-words integer_value`

Specifies the number of 9-bit words.

10.38.2 Example

```
read_uprom_memory -startAddress {0xA} -words {100}
```

10.39 write_active_probe

`write_active_probe [-deviceName device_name] -name probe_name -value true|false`

`-group_name group_bus_name -group_value "hex-value" | "binary-value"`

Tcl command; sets the target probe point on the device to the specified value. The target probe point name must be specified.

10.39.1 Arguments

`-deviceName device_name`

Parameter is optional if only one device is available in the current configuration.

`-name probe_name`

Specifies the name for the target probe point. Cannot be a search pattern.

`-value true | false hex-value | binary-value`

Specifies values to be written. True = High

False = Low

`-group_name group_bus_name`

Specify the group or bus name to write to complete group or bus.

`-group_value "hex-value" | "binary-value"`

Specify the value for the complete group or bus.

Hex-value format : " <size>'h<value>" Binary-value format: " <size>'b<value>"

10.39.2 Example

```
write_active_probe -name out[5]:out[5]:Q -value true <-- write to a single probe
write_active_probe -name grp1.out[3]:out[3]:Q -value low <-- write to a probe in the group
write_active_probe -group_name grp1 -group_value "8'hF0" <-- write the value to complete group
write_active_probe -group_name out -group_value "8'b11110000" \
-name out[2]:out[2]:Q -value true <-- write multiple probes at the same time.
```

10.40 write_lsram

Tcl command; writes a word into the specified large TPSRAM location.

10.40.1 Physical block

`write_lsram -name block_name] -offset offset_value -tpsramValue integer_value`

10.40.2 Arguments

`-name block_name`

Specifies the name for the target block.

`-offset offset_value`

Offset (address) of the target word within the memory block.

`-tpsramValue integer_value`

Word to be written to the target location. Minimum value is 0 and may go up to 511. TPSRAM block has an aspect ratio of 512x40 (ECC disabled) and 512x33 (ECC enabled). SmartDebug enhanced the physical block view to read and write as 40-bit and 33-bit data.

10.40.3 Exceptions

- Array must be programmed and active
- The maximum value that can be written depends on the configuration of memory blocks
- Security locks may disable this function

10.40.4 Example

```
write_lsram -name {PF_TPSRAM_0/top_PF_TPSRAM_0_PF_TPSRAM_R0C0/INST_RAM1K20_IP} -offset
38 -tpsramValue {DEAFBEEDDA}
write_lsram -logicalBlockName block_name -port port_name -offset 1 offset_value -
logicalValue
hexadecimal_value
```

10.40.5 Logical block

Arguments

`-logicalBlockName block_name`

Specifies the name of the user defined memory block.

`-port port_name`

Specifies the port of the memory block selected. Can be either Port A or Port B.

`-offset offset_value`

Offset (address) of the target word within the memory block.

`-logicalValue hexadecimal_value`

Specifies the hexadecimal value to be written to the memory block. Size of the value is equal to the width of the output port selected.

10.40.6 Example

```
write_lsram -logicalBlockName {Fabric_Logic_0/U2/F_0_F0_U1} -port {Port A} -offset 1 -
logicalValue {00FFF}
```

10.41 write_usram

Tcl command; writes a 12-bit word into the specified uSRAM location.

10.41.1 Physical block

`write_usram –name block_name] –offset offset_value –value integer_value`

10.41.2 Arguments

`-name block_name`

Specifies the name for the target block.

`-offset offset_value`

Offset (address) of the target word within the memory block.

`-value integer_value`

1. bit value to be written.

10.41.3 Exceptions

- Array must be programmed and active
- The maximum value that can be written is 0x1FF
- Security locks may disable this function

10.41.4 Example

Writes a value of 0x291 to the device PolarFire in the Fabric_Logic_0/U3/F_0_F0_U1/ramtmp_ramtmp_0_0/INST_RAM64x12_IP with an offset of 0.

```
write_lsram -name {Fabric_Logic_0/U3/F_0_F0_U1/ramtmp_ramtmp_0_0/INST_RAM64x12_IP} -
offset 0 -value 291
```

`write_usram -logicalBlockName block_name -port port_name -offset offset_value -logicalValue`

`hexadecimal_value`

10.41.5 Logical block

Arguments

`-logicalBlockName block_name`

Specifies the name of the user defined memory block.

`-port port_name`

Specifies the port of the memory block selected. Can be either Port A or Port B.

`-offset offset_value`

Offset (address) of the target word within the memory block.

-logicalValue *hexadecimal_value*

Specifies the hexadecimal value to be written to the memory block. Size of the value is equal to the width of the output port selected.

10.41.6 Example

```
write_usram -logicalBlockName {Fabric_Logic_0/U3/F_0_F0_U1} -port {Port A} -offset 1 -
logicalValue {00FFF}
```

10.42 xcvr_read_register

This Tcl command reads SCB registers and their field values. Read value is in hex format. This command is used in SmartDebug Signal Integrity.

xcvr_read_register –inst_name <*inst_name*> –reg_name [<*reg_name*> | <*reg_name*:*field_name*>]

10.42.1 Arguments

-inst_name <*inst_name*>

Specify the lane instance name used in the design.

-reg_name <*reg_name*> or <*reg_name*:*field_name*>

Specify the <*reg_name*> for register name or <*reg_name*>:<*field_name*> for the register's field.

10.42.2 Examples

Reading pcslane's 32-bit register LNTV_R0:

```
xcvr_read_register -inst_name {CM1_PCIe_SS_0/PF_PCIE_0/LANE1} -reg_name {LNTV_R0}
```

Output:

```
Register Name: LNTV_R0 value: 0x12
The 'xcvr_write_register' command succeeded.
```

10.42.3 Reading Register LNTV_R0 field LNTV_RX_GEAR (i.e. 0th bit of 32-bit register):

```
xcvr_read_register -inst_name {CM1_PCIe_SS_0/PF_PCIE_0/LANE1} -reg_name
{LNTV_R0:LNTV_RX_GEAR}
```

Output:

```
Register Name: LNTV_R0:LNTV_RX_GEAR, Value: 0x0 The 'xcvr_read_register' command
succeeded.
```

10.42.4 Exception:

SOFT_RESET Register

The SOFT_RESET register is an SCB read/write register containing information such as block ID and Map IDs. It is also used to provide a pulsed reset to the SCB registers. It is a group-specific register.

The SOFT_RESET register is available with the four groups (pma_lane, pma_cmn, pcslane, and pcscmn). To read or write this register or its field value, "group name" must be added before "SOFT_RESET".

-reg_name <*group name*>_<*SOFT_RESET*> for register name

or

[<*group name*>_<*SOFT_RESET*>:field_name] for register field name

where <*group name*> can be PCS, PCSCMN, PMA, or PMA_CMN.

10.42.5 Examples

Reading all four groups' SOFT_RESET register and its field BLOCKID

10.42.5.1 Reading the PCS SOFT_RESET register and its field BLOCKID (i.e. 16th to 31st bit):

```
xcvr_read_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PCS_SOFT_RESET}
```

Output:

```
Register Name: PCS_SOFT_RESET, Value: 0x300100 The 'xcvr_read_register' command
succeeded.
```

10.42.5.2 Reading field BLOCKID:

```
xcvr_read_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PCS_SOFT_RESET:BLOCKID}
```

Output:

```
Register Name: PCS_SOFT_RESET:BLOCKID, Value: 0x30 The 'xcvr_read_register' command
succeeded.
```

10.42.5.3 Reading PCSCMN's SOFT_RESET register and its field BLOCKID (i.e. 16th to 31st bit):

```
xcvr_read_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PCSCMN_SOFT_RESET}
```

```
Register Name: PCSCMN_SOFT_RESET, Value: 0x340100 The 'xcvr_read_register' command
succeeded.
```

10.42.5.4 Reading field BLOCKID:

```
xcvr_read_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PCSCMN_SOFT_RESET:BLOCKID}
```

Output:

```
Register Name: PCSCMN_SOFT_RESET:BLOCKID, Value: 0x34 The 'xcvr_read_register' command
succeeded.
```

10.42.5.5 Reading PMA's SOFT_RESET register and its field BLOCKID (i.e. 16th to 31st bit):

```
xcvr_read_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PMA_SOFT_RESET}
```

Output:

```
Register Name: PMA_SOFT_RESET, Value: 0x1300100 The 'xcvr_read_register' command
succeeded.
```

10.42.5.6 Reading field BLOCKID:

```
xcvr_read_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PMA_SOFT_RESET:BLOCKID}
```

Output:

```
Register Name: PMA_SOFT_RESET:BLOCKID, Value: 0x130 The 'xcvr_read_register' command
succeeded.
```

10.42.5.7 Reading PMA_CMN's SOFT_RESET register and it's field BLOCKID (i.e. 16th to 31st bit):

```
xcvr_read_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PMA_CMN_SOFT_RESET}
```

Output:

```
Register Name: PMA_CMN_SOFT_RESET, Value: 0x1340100 The 'xcvr_read_register' command
succeeded.
```

10.42.5.8 Reading field BLOCKID:

```
xcvr_read_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PMA_CMN_SOFT_RESET:BLOCKID}
```

Output:

```
Register Name: PMA_CMN_SOFT_RESET:BLOCKID, Value: 0x134 The 'xcvr_read_register'
command succeeded.
```

10.42.5.9 See Also

xcvr_write_register

10.43 xcvr_write_register

```
xcvr_write_register -inst_name <inst_name> -reg_name [<reg_name> |
<reg_name:field_name>] - value {write_value}
```

This Tcl command writes SCB registers and their field values. Write value is in hex format. This command is used in SmartDebug Signal Integrity.

10.43.1 Arguments

`-inst_name <inst_name>`

Specify the lane instance name used in the design.

`-reg_name <reg_name> or <reg_name:field_name>`

Specify the <*reg_name*> for register name or <*reg_name*>:<*field_name*> for the register's field.

`-value <write_value>`

Specify the value in hex format.

10.43.2 Examples

Writing pcscmn's 32-bit register GSSCLK_CTRL

```
xcvr_write_register -inst_name {CM1_PCIe_SS_0/PF_PCIE_0/LANE1} -reg_name {GSSCLK_CTRL}
- value 0xffffffff
```

Output:

```
Register Name: GSSCLK_CTRL value: 0xffffffff The 'xcvr_write_register' command
succeeded.
```

10.43.3 Writing Register GSSCLK_CTRL field MCLK_GSSCLK_2_SEL i.e. 16th to 20th bits (5 bits) of 32-bit register

```
xcvr_write_register -inst_name {CM1_PCIe_SS_0/PF_PCIE_0/LANE1} \
-reg_name {GSSCLK_CTRL:MCLK_GSSCLK_2_SEL} -value 0x6
```

Output:

```
Register Name: GSSCLK_CTRL:MCLK_GSSCLK_2_SEL value: 0x6 The 'xcvr_write_register'
command succeeded.
```

10.43.4 Exception:

SOFT_RESET Register

The SOFT_RESET register is an SCB read/write register containing information such as block ID and Map IDs. It is also used to provide a pulsed reset to the SCB registers. It is a group-specific register.

The SOFT_RESET register is available with the four groups (`pma_lane`, `pma_cmn`, `pcslane`, and `pcscmn`). To read or write this register or its field value, "group name" must be added before "SOFT_RESET".

`-reg_name <group name>_<SOFT_RESET>` for register name

or

`[<group name>_<SOFT_RESET>:field_name]` for register field name

where <*group name*> can be PCS, PCSCMN, PMA, or PMA_CMN

10.43.5 Examples

Writing all four groups' SOFT_RESET register and its field PERIPH

10.43.5.1 Writing to the PCS SOFT_RESET register (32-bits) and its field PERIPH (i.e. 8th bit):

```
xcvr_write_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PCS_SOFT_RESET}
-value 0xffffffff
```

Output:

```
Register Name: PCS_SOFT_RESET value: 0xffffffff The 'xcvr_write_register' command
succeeded.
```

10.43.5.2 Writing to field PERIPH:

```
xcvr_write_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PCS_SOFT_RESET:PERIPH} -value 0x1
```

Output:

```
Register Name: PCS_SOFT_RESET:PERIPH value: 0x1 The 'xcvr_write_register' command
succeeded.
```

10.43.5.3 Writing to PCSCMN's SOFT_RESET register (32-bits) its field PERIPH (i.e. 8th bit):

```
xcvr_write_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PCSCMN_SOFT_RESET} -value 0xffffffff
```

Output:

```
Register Name: PCSCMN_SOFT_RESET value: 0xffffffff The 'xcvr_write_register' command
succeeded.
```

10.43.5.4 Writing to field PERIPH:

```
xcvr_write_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PCSCMN_SOFT_RESET:PERIPH} -value 0x1
```

Output:

```
Register Name: PCSCMN_SOFT_RESET:PERIPH value: 0x1 The 'xcvr_write_register' command
succeeded.
```

10.43.5.5 Writing to PMA's SOFT_RESET register its field PERIPH (i.e. 8th bit):

```
xcvr_write_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PMA_SOFT_RESET}
-value 0xffffffff
```

Output:

```
Register Name: PMA_SOFT_RESET value: 0xffffffff The 'xcvr_write_register' command
succeeded.
```

10.43.5.6 Writing to field PERIPH:

```
xcvr_write_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name
{PMA_SOFT_RESET:PERIPH} -value 0x1
```

Output:

```
Register Name: PMA_SOFT_RESET:PERIPH value: 0x1 The 'xcvr_write_register' command
succeeded.
```

10.43.5.7 Writing to PMA_CMN's SOFT_RESET register its field PERIPH (i.e. 8th bit):

```
xcvr_write_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name

{PMA_CMN_SOFT_RESET} -value 0xffffffff
```

Output:

```
Register Name: PMA_CMN_SOFT_RESET value: 0xffffffff

The 'xcvr_write_register' command succeeded.
```

10.43.5.8 Writing to field PERIPH:

```
xcvr_write_register -inst_name SmartBERT_L4_0/PF_XCVR_0/LANE0 -reg_name

{PMA_CMN_SOFT_RESET:PERIPH} -value 0x1
```

Output:

```
Register Name: PMA_CMN_SOFT_RESET:PERIPH value: 0x1 The 'xcvr_write_register' command
succeeded.
```

10.43.5.9 See Also

xcvr_read_register

11. Configure JTAG Chain Tcl Commands

These commands take a script that contains JTAG chain configuration-specific Tcl commands and passes them to FlashPro Express for execution.

Note that these commands cannot be executed directly from Libero.

11.1 add_actel_device

Adds an Actel device to the chain. Either the *file* or *device* parameter must be specified. Chain programming mode must have been set.

```
add_actel_device [-file {filename}] [-device {device}] -name{name}
```

11.1.1 Arguments

Where:

-file{*filename*}

Specifies a programming filename.

-device{*device*}

Specifies the device family (such as MPF300).

-name{*name*}

Specifies the device user name.

11.1.2 Exceptions

None

11.1.3 Example

```
add_actel_device -file {e:/design/stp/TOP.stp} -name {MyDevice1} add_actel_device -
device {MPF300} -name {MyDevice2}
```

11.2 add_non_actel_device

Adds a non-Actel device in the chain. Either the file, or (-tck And -ir) parameters must be specified. The Chain programming mode must have been set.

```
add_non_actel_device [-file {file}] [-ir {ir}] [-tck {tck}] [-name {name}]
```

11.2.1 Arguments

-file {*filename*}

Specifies a BSDL file.

-ir {*ir*}

Specifies the IR length.

-tck {*tck*}

Specifies the maximum TCK frequency (in MHz).

-name {*name*}

Specifies the device user name.

11.2.2 Exceptions

None

11.2.3 Examples

```
add_non_actel_device -file {e:/design/bsdl/DeviceX.bsdl } -name {MyDevice3}
add_non_actel_device -ir 8 - tck 5 -name {MyDevice4}
```

11.3 add_non_actel_device_to_database

Imports settings via a BSDL file that adds non-Actel or non-Microsemi devices to the device database so that they are recognized during scan chain and auto-construction operations.

```
add_non_actel_device_to_database [-file {bsdl_filename}]
```

11.3.1 Arguments

-file {bsdl_filename}

Specifies the path to the BSDL file and the BSDL filename add to the database.

11.3.2 Supported Families

All non-Microsemi and non-Actel families

11.3.3 Exceptions

N/A

11.3.4 Examples

The following example uses a BSDL file to add a non-Microsemi (1502AS J44) device to the device database:

```
add_non_actel_device_to_database -file {c:/bsdl/atmel/1502AS_J44.bsd}
```

The following example uses a BSDL file to add a non-Microsemi (80200) device to the device database:

```
add_non_actel_device_to_database -file {c:/bsdl/intel/80200_v1.0.bsd}
```

11.4 construct_chain_automatically

Automatically starts chain construction for the specified programmer.

```
construct_chain_automatically[(-name {name})+]
```

11.4.1 Arguments

-name {*name*}

Specifies the programmer(s) name(s).

11.4.2 Exceptions

N/A

11.4.3 Example

Example for one programmer:

```
construct_chain_automatically -name {21428}
```

Example for two programmers:

```
construct_chain_automatically -name {21428} -name {00579}
```

11.5 copy_device

Copies a device in the chain to the clipboard. Chain programming mode must be set. See the paste_device command for more information.

```
copy_device (-name {name})*
```

11.5.1 Arguments

`-name {name}`

Specifies the device name. Repeat this argument to copy multiple devices.

11.5.2 Exceptions

None

11.5.3 Example

The example copies the device 'mydevice1' to the same location with a new name 'mydevice2'.

```
copy_device -name {MyDevice1} -name {MyDevice2}
```

11.6 cut_device

Removes one or more devices from the chain. It places the removed device in the clipboard. Chain programming mode must be set to use this command. See the `paste_device` command for more information.

```
cut_device (-name {name})*
```

11.6.1 Arguments

`-name {name}`

Specifies the device name. You can repeat this argument for multiple devices.

11.6.2 Exceptions

None

11.6.3 Example

The following example removes the devices 'mydevice1' and 'mydevice2' from the chain.

```
cut_device -name {MyDevice1} -name {MyDevice2}
```

11.7 enable_device

Enables or disables a device in the chain (if the device is disabled, it is bypassed). Chain programming mode must be set. The device must be a Microsemi device.

```
enable_device -name {name} -enable {TRUE|FALSE}
```

11.7.1 Arguments

`-name {name}`

Specifies your device name

`-enable {TRUE|FALSE}`

Specifies whether the device is to be enabled or disabled. If you specify multiple devices, this argument applies to all specified devices. (TRUE = enable. FALSE = disable)

11.7.2 Exceptions

None

11.7.3 Example

The following example disables the device 'mydevice1' in the chain.

```
enable_device -name {MyDevice1} -enable {FALSE}
```

11.8 paste_device

Pastes the devices that are on the clipboard in the chain, immediately above the *position_name* device, if this parameter is specified. Otherwise it places the devices at the end of the chain. The chain programming mode must be enabled.

```
paste_device [-position_name {position_name}]
```

11.8.1 Arguments

`-position_name {position_name}`

Optional argument that specifies the name of a device in the chain.

11.8.2 Exceptions

None

11.8.3 Examples

The following example pastes the devices on the clipboard immediately above the device 'mydevice3' in the chain.

```
paste_device –position_name {MyDevice3}
```

11.9 remove_device

Removes the device from the chain. Chain programming mode must be set.

```
remove_device (-name {name})*
```

11.9.1 Arguments

`-name {name}`

Specifies the device name. You can repeat this argument for multiple devices.

11.9.2 Exceptions

None

11.9.3 Example

Remove a device 'M2S050T' from the chain:

```
remove_device (-name {M2S050T})*
```

11.10 remove_non_actel_device_from_database

Removes settings for non-Microsemi or non-Actel device from the device database.

```
remove_non_actel_device_from_database [-name {device_name}]
```

11.10.1 Arguments

`-name {device_name}`

Specifies the non-Actel or non-Microsemi device name to be removed from the database. You can repeat this argument for multiple devices.

11.10.2 Supported Families

Non-Microsemi and non-Actel devices

11.10.3 Exceptions

None

11.10.4 Example

The following example removes the F1502AS_J44 device from the database:

```
remove_non_actel_device_from_database -name {F1502AS_J44}
```

The following example removes the SA2_PROCESSOR device from the database:

```
remove_non_actel_device_from_database -name {SA2_PROCESSOR}
```

11.11 select_libero_design_device

This command selects the Libero design device for the Programming Connectivity and Interface tool within Libero. This command is needed when the tool cannot automatically resolve the Libero design device when there are two or more identical devices that match the Libero design device in the configured JTAG chain.

```
select_libero_design_device -name {device_name}
```

11.11.1 Arguments

```
-name {device_name}
```

Specifies a user-assigned unique device name in the JTAG chain.

11.11.2 Exceptions

None

11.11.3 Example

```
select_libero_design_device -name {M2S050TS (2)} select_libero_design_device -name
{my_design_device}
```

11.11.4 Note

This Tcl command is typically used in a Tcl command script file that is passed to the Libero run_tool command.

```
run_tool -name {CONFIGURE_CHAIN} -script {<flashPro_cmd>.tcl}
```

11.12 set_bsdl_file

Sets a BSDL file to a non-Microsemi device in the chain. Chain programming mode must have been set. The device must be a non-Microsemi device.

```
set_bsdl_file -name {name} -file {file}
```

11.12.1 Arguments

```
name {name}
```

Specifies the device name.

```
-file {file}
```

Specifies the BSDL file.

11.12.2 Supported Families

Any non-Microsemi device supported by FlashPro Express.

11.12.3 Exceptions

None

11.12.4 Example

The following example sets the BSDL file /design/bsdl/NewBSDL2.bsdl to the device 'MyDevice3':

```
set_bsdl_file -name {MyDevice3} -file {e:/design/bsdl/NewBSDL2.bsdl}
```

11.13 set_device_ir

Sets the IR length of a non-Microsemi device in the chain. Chain programming mode must be set. The device must be a non-Microsemi device.

```
set_device_ir -name {name} -ir {ir}
```

11.13.1 Arguments

`-name {name}`

Specifies the device name.

`-ir {ir}`

Specifies the IR length.

11.13.2 Supported Families

Any non-Microsemi device supported by FlashPro Express.

11.13.3 Exceptions

None

11.13.4 Example

The following example sets the IR length to '2' for the non-Microsemi device 'MyDevice4':

```
set_device_ir -name {MyDevice4} -ir {2}
```

11.14 set_device_name

Changes the user name of a device in the chain. Chain programming mode must be set .

```
set_device_name -name {name} -new_name {new_name}
```

11.14.1 Arguments

`-name {name}`

Identifies the old device name.

`-new_name {new_name}`

Specifies the new device name.

11.14.2 Exceptions

None

11.14.3 Example

The following example changes the user name of the device from 'MyDevice4' to 'MyDevice5':

```
set_device_name -name {MyDevice4} -new_name {MyDevice5}
```

11.15 set_device_order

Sets the order of the devices in the chain to the order specified. Chain programming mode must have been set. Unspecified devices will be at the end of the chain.

```
set_device_order (-name {name})*
```

11.15.1 Arguments

`-name {name}`

Specifies the device name. To specify a new order you must repeat this argument and specify each device name in the order desired.

11.15.2 Exceptions

None

11.15.3 Example

The following example sets the device order for 'MyDevice1', 'MyDevice2', 'MyDevice3', and 'MyDevice4'. 'MyDevice2' is unspecified so it moves to the end of the chain.

```
set_device_order -name {MyDevice3} -name {MyDevice1} -name {MyDevice4}
```

the new order is:

```
MyDevice3 MyDevice1 MyDevice4 MyDevice2
```

11.16 set_device_tck

Sets the maximum TCK frequency of a non-Microsemi device in the chain. Chain programming mode must be set. The device must be a non-Microsemi device.

```
set_device_tck -name {name} -tck {tck}
```

11.16.1 Arguments

`-name {name}`

Specifies the device name.

`-tck {tck}`

Specifies the maximum TCK frequency (in MHz).

11.16.2 Supported Families

Any non-Microsemi device supported by FlashPro Express.

11.16.3 Exceptions

None

11.16.4 Example

The following example sets the maximum TCK frequency of the non-Microsemi device 'MyDevice4':

```
set_device_tck -name {MyDevice4} -tck {2.25}
```

11.17 set_device_type

Changes the family of a Microsemi device in the chain. The device must be a Microsemi device. The device parameter below is now optional.

```
set_device_type -name {name} -type {type}
```

11.17.1 Arguments

`-name {name}`

Identifies the name of the device you want to change.

`-type {type}`

Specifies the device family.

11.17.2 Exceptions

None

11.17.3 Example

The following example sets the device 'MyDevice2' to the type MPF300.

```
set_device_type -name {MyDevice2} -type {MPF300}
```

11.18 set_programming_action

This Tcl command is used to select the action for a device.

```
set_programming_action [-name {device_name}] -action {procedure_action}
```

11.18.1 Arguments

`-name`

Specify the device name. This argument is mandatory.

`-action`

Specify the programming action. This argument is mandatory.

11.18.2 Example

```
set_programming_action -name {MPF300(T_ES|TS_ES)} -action {DEVICE_INFO}
set_programming_action -name {M2S/M2GL090(T|TS|TV)} -action {ERASE}
```

11.18.2.1 See Also

construct_chain_automatically scan_chain_prg

enable_device set_debug_programmer set_device_name set_programming_file run_selected_actions

11.19 set_programming_file

This Tcl command is used to set the programming file for a device. Either the file or the no_file flag must specified. A programming file must be loaded. The device must be a Microsemi device.

```
set_programming_file -name {device_name} -file {stapl_file_name_with_path}
```

11.19.1 Arguments

`-name`

Specify the device name. This argument is mandatory.

`-file`

Specify the *file* path. This argument is mandatory.

11.19.2 Example

```
set_programming_file -name {MPF300(T_ES|TS_ES)} -file
{D:/export/CM1_PCIE_TOP_default_uic_12_200_0_12.stp}
```

11.19.2.1 See Also

construct_chain_automatically scan_chain_prg

enable_device set_debug_programmer set_device_name set_programming_file run_selected_actions

12. System Builder Tcl Commands

12.1 is_synthesis_enabled

Description
Tcl query to determine if a project can be synthesized. The source is an `.edn` file.

```
is_synthesis_enabled
```

Arguments

Return Type	Description
1	Synthesis is enabled.
0	Synthesis is disabled.

Example

```
set p [is_synthesis_enabled]
puts "$p"
```

12.2 sb_configure_page

Description
This Tcl command is used to configure the parameters of a System Builder component page.

```
sb_configure_page \
-component_name {component_name} \
-page_name {page_name} \
[-params {param_list}]
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.
-page_name {page_name}	string	Mandatory. Name of the system builder page being configured.
-params {param_list}	string	Mandatory. List of parameters being configured within given page.

Example

```
sb_configure_page -component_name {sb} -page_name{DEVICEFEATURES} -params {USE_ENVM:1}
```

12.3 sb_add_core

Description
This Tcl command is used to add cores to Master/Slave subsystems in the **Peripherals** page of the System Builder component.
Notes: There are two types of cores that can be added using the `sb_add_core` command.

1. One type is of actual cores like CoreI2C, CoreGPIO with fixed core versions available in the **Peripherals** page. If, for example, CoreI2C is added to a subsystem with core_name specified as i2c_peripheral and a core count of 4, then the actual instance names of CoreI2C added will be i2c_peripheral_0, i2c_peripheral_1, i2c_peripheral_2 and i2c_peripheral_3 in the generated design.
2. Second type of cores are Fabric AMBA Slave and Fabric AMBA Master. Adding them to a subsystem will configure the bus core of the subsystem to enable and expose master/slave interfaces of the bus core on the generated System Builder component to be connected to actual masters/slaves in the fabric. If, for example, Fabric AMBA Slave configured as AHBLite(`AMBA_INTERFACE_TYPE:AHBLITE`) is added to a subsystem with core_name specified as ahb_slave and a core count of 4, then the CoreAHBLite bus core of that subsystem will be configured to enable and expose 4 AHBLite slave bus interfaces with names ahb_slave, ahb_slave_1, ahb_slave_2 and ahb_slave_3 on the System Builder component to be connected to actual slave peripherals in the fabric.

```
sb_add_core \
-component_name {component_name} \
-core_vlnv {vendor:library:name:version} \
[-core_name {core_name}] \
-subsystem_name {subsystem_name}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.
-core_vlnv {vendor:library:name:version}	string	Mandatory. Version identifier of the core being instantiated in the SmartDesign.
-core_name {core_name}	string	Optional. Name of the instance of the core in the System Builder component. If no details are provided, the instance name will be automatically defined as `<vlnv_core_name>_n` (where `n` starts at 0).
-subsystem_name {subsystem_name}	string	Mandatory. Name of the subsystem the core is being added to.

Example

```
sb_add_core -component_name {sb} -core_vlnv {Actel:SystemBuilder:AMBA_SLAVE:0.0.102} -
core_name {AMBA_SLAVE_0} -subsystem_name {FIC0_Master_Subsystem}
```

12.4 sb_configure_core

Description
This Tcl command is used to configure cores that are already added to Master/Slave subsystem in the **Peripherals** page of the System Builder component.

```
sb_configure_core \
-component_name {component_name} \
```

```
[-core_name {core_name}] \
[-params {param_list}]
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.
-core_name {core_name}	string	Optional. Name of the instance of the core in the System Builder component.
-params {param_list}	string	Mandatory. List of the configuration parameters for the selected core.

Example

```
sb_configure_core -component_name {sb} -core_name {AMBA_SLAVE_0} -params
{"AMBA_INTERFACE_TYPE:AHBLITE" "NUM_OF_INTERRUPTS:0"}
```

12.5 sb_configure_core_count

Description
This command is used to specify the number of instances of a core already added to Master/Slave subsystem in the **Peripherals** page of the System Builder component. All instances will have the same configuration.
Notes: There are two types of cores that can be added using the `sb_add_core` command.

1. One type is of actual cores like CoreI2C, CoreGPIO with fixed core versions available in the **Peripherals** page. If, for example, CoreI2C is added to a subsystem with core_name specified as i2c_peripheral and a core count of 4, then the actual instance names of CoreI2C added will be i2c_peripheral_0, i2c_peripheral_1, i2c_peripheral_2 and i2c_peripheral_3 in the generated design.
2. Second type of cores are Fabric AMBA Slave and Fabric AMBA Master. Adding them to a subsystem will configure the bus core of the subsystem to enable and expose master/slave interfaces of the bus core on the generated System Builder component to be connected to actual masters/slaves in the fabric. If, for example, Fabric AMBA Slave configured as AHBLite(`AMBA_INTERFACE_TYPE:AHBLITE`) is added to a subsystem with core_name specified as ahb_slave and a core count of 4, then the CoreAHBLite bus core of that subsystem will be configured to enable and expose 4 AHBLite slave bus interfaces with names ahb_slave, ahb_slave_1, ahb_slave_2 and ahb_slave_3 on the System Builder component to be connected to actual slave peripherals in the fabric.

```
sb_configure_core_count \
-component_name {component_name} \
-core_vlnv {vendor:library:name:version} \
[-core_name {core_name}] \
-subsystem_name {subsystem_name}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.
-core_vlnv {vendor:library:name:version}		Mandatory. Version identifier of the core being instantiated in the SmartDesign.

.....continued		
Parameter	**Type**	**Description**
-core_name {core_name}	string	Optional. Name of the instance of the core in the System Builder component. If no details are provided, the instance name will be automatically defined as `<vlnv_core_name>_n` (where `n` starts at 0).
-subsystem_name {subsystem_name}	string	Mandatory. Name of the subsystem the core is being added to.

Example

```
sb_configure_core_count -component_name {sb} -core_name {AMBA_SLAVE_0} -count {2}
```

12.6 sb_move_core

Description

This Tcl command is used to move cores from one Subsystem to another in the **Peripherals** page of System Builder. In the exported Tcl description file of a System Builder component, the only scenario where this command will be seen is when the Fabric DDR is used and is moved from its default 'Fabric DDR Subsystem' to a different Subsystem in the **Peripherals** page.

Note: In a System Builder design, if the Fabric External DDR Memory (FDDR) is enabled on the **Device Features** page, then the Fabric DDR Subsystem is automatically enabled with the core Fabric_DDR_RAM added to it in the **Peripherals** page.

A Fabric AMBA Master configured as AXI or AHBLite can be added to the Fabric DDR Subsystem to enable a fabric master to access external DDR memory using FDDR. Alternatively, the Fabric DDR RAM can also be moved (drag and drop) to other Subsystems in the Peripherals page so that the master(s) in that Subsystem will also be able to access external DDR memory using FDDR.

For example, the Fabric_DDR_RAM can be moved from its default 'Fabric DDR Subsystem' to 'MSS FIC_0 - MSS Master Subsystem' (FIC0_Master_Subsystem) which will enable Cortex-M3 in the MSS to access external DDR memory using FDDR via FIC_0 Master address space. The Tcl command `sb_move_core` will be used to capture the action of moving the Fabric_DDR_RAM to a different Subsystem in the exported Tcl description for a System Builder component.

```
sb_move_core \
-component_name {component_name} \
[-core_name {core_name}] \
-subsystem_name {subsystem_name}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.
-core_name {core_name}	string	Optional. Name of the instance of the core in the System Builder component.
-subsystem_name {subsystem_name}	string	Mandatory. Name of the subsystem the core is being added to.

Example

```
sb_move_core -component_name {sb} -core_name {Fabric_DDR_RAM} -subsystem_name
{FIC0_Master_Subsystem}
```

12.7 sb_enable_core

Description
This Tcl command is used to enable the cores/bus interfaces in various subsystems of the **Peripherals** page of the System Builder component (excluding the MSS Peripherals).

```
sb_enable_core \
-component_name {component_name} \
[-core_name {core_name}]
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.
-core_name {core_name}	string	Optional. Name of the instance of the core in the System Builder component.

Example

```
sb_enable_core -component_name {sb} -core_name {FIC_0_AMBA_MASTER}
```

12.8 sb_disable_core

Description
This Tcl command is used to disable the cores/bus interfaces in various subsystems of the **Peripherals** page of the System Builder component (excluding the MSS Peripherals).

```
sb_disable_core \
-component_name {component_name} \
[-core_name {core_name}]
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.
-core_name {core_name}	string	Optional. Name of the instance of the core in the System Builder component.

Example

```
sb_disable_core -component_name {sb} -core_name {FIC_0_AMBA_MASTER}
```

12.9 sb_enable_peripheral

Description
This Tcl command is used to enable the MSS peripherals in various subsystems of the **Peripherals** page of the System Builder component.

Note: By default, MSS_GIO, MSS_USB, MSS_MAC and MSS_CAN are disabled in all devices. Use the `sb_enable_peripheral` command to enable the required peripheral.

```
sb_enable_peripheral \
-component_name {component_name} \
[-peripheral_name {peripheral_name}]
```

Arguments

Parameter	Type	Description
-component_name {component_name}		Mandatory. Name of the system builder component.
-peripheral_name {peripheral_name}		Mandatory. Name of the MSS peripheral instance to be enabled.

Example

```
sb_enable_peripheral -component_name {sb} -peripheral_name {MSS_SPI_0}
```

12.10 sb_disable_peripheral

Description
This Tcl command is used to disable the MSS peripherals in various subsystems of the **Peripherals** page of the System Builder component.
Note: By default, MSS_MMUART_0/1, MSS_I2C_0/1, MSS_SPI_0/1 are enabled in all devices. Use the `sb_disable_peripheral` command to disable the peripherals that are not needed.

```
sb_disable_peripheral \
-component_name {component_name} \
[-peripheral_name {peripheral_name}]
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.
-peripheral_name {peripheral_name}	string	Mandatory. Name of the MSS peripheral instance to be disabled.

Example

```
sb_disable_peripheral -component_name {sb} -peripheral_name {MSS_SPI_0}
```

12.11 sb_configure_peripheral

Description
This Tcl command is used to configure the MSS peripherals in various subsystems of the **Peripherals** page of the System Builder component.

```
sb_configure_peripheral \
-component_name {component_name} \
```

```
[-peripheral_name {peripheral_name}]
[-params {param_list}]
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.
-peripheral_name {peripheral_name}	string	Mandatory. Name of the MSS peripheral instance to be enabled.
-params {param_list}	string	Mandatory. List of the configuration parameters for the selected MSS peripheral.

Example

```
sb_configure_peripheral -component_name {sb} -peripheral_name {MM_UART_0} -params
{"MODE:MODE_ASYNC" "MODE_DUPLEX:MODE_FULL" "MODEM_MUX:MUX_IO" "TX_RX_MUX:MUX_IO"
"USE_MODEM:false" }
```

12.12 sb_set_fic_direct_mode

Description
This Tcl command is used to set/unset the Fabric Interface Controller (FIC) direct mode in the **Peripherals** page of the System Builder component.

```
sb_set_fic_direct_mode \
-component_name {component_name} \
-mode {true|false}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.
-mode {true|false}		Mandatory. Set the mode to 'true' to enable the direct mode, else 'false'.

Example

```
sb_set_fic_direct_mode -component_name {sb} -mode {true}
```

12.13 sb_configure_envm

Description
This Tcl command is used to specify a `.cfg` file with all clients information in the **ENVM** tab of the **Memories** page in the System Builder component.

```
sb_configure_envm \
-component_name {component_name} \
-cfg_file {file_path}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.
-cfg_file <file_path>	string	Mandatory. Path of the configuration file (.cfg) used to configure the ENVM.

Example

```
sb_configure_envm -component_name {sb} -cfg_file{./ENVM.cfg}
```

12.14 open_sb_component

Description
This Tcl command opens a System Builder component. You must open a System Builder component before you can configure any of its pages using the `sb_*` commands.

Note: To create a System Builder design, the `create_and_configure_core` command must be used, but with empty params. This creates an ungenerated System Builder component with default configuration. Then before configuring any of the System Builder pages, the System Builder component needs to be opened using the `open_sb_component` command.

```
open_sb_component \
-component_name {component_name}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.

Example

```
open_sb_component -component_name
```

12.15 generate_sb_component

Description
This Tcl command is used to generate a System Builder component.

```
generate_sb_component \
-component_name {component_name}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.

Example

```
generate_sb_component -component_name {sb}
```

12.16 close_sb_component

Description
Close a System Builder component. You must close the System Builder component after you are done with configuring all its pages and generating it.

```
close_sb_component \
-component_name {component_name}
```

Arguments

Parameter	Type	Description
-component_name {component_name}	string	Mandatory. Name of the system builder component.

Example

```
close_sb_component -component_name {sb}
```

13. Revision History

Revision	Date	Description
B	4/2021	The following changes were made in this revision: • Updated 7. SmartTime Tcl Commands with references to the Libero® SoC Design Suite Tcl Examples GitHub repository. • Updated 2. Project Manager Tcl Commands • Updated 4. HDL Tcl Commands • Updated 9. FlashPro Express Tcl Commands • Added the 6. MSS Tcl Commands
A	11/2020	Initial Revision

14. Microchip FPGA Technical Support

Microchip FPGA Products Group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, and worldwide sales offices. This section provides information about contacting Microchip FPGA Products Group and using these support services.

14.1 Customer Service

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

- From North America, call **800.262.1060**
- From the rest of the world, call **650.318.4460**
- Fax, from anywhere in the world, **650.318.8044**

14.2 Customer Technical Support

Microchip FPGA Products Group staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions about Microchip FPGA Products. The Customer Technical Support Center spends a great deal of time creating application notes, answers to common design cycle questions, documentation of known issues, and various FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

You can communicate your technical questions through our Web portal and receive answers back by email, fax, or phone. Also, if you have design problems, you can upload your design files to receive assistance. We constantly monitor the cases created from the web portal throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

Technical support can be reached at soc.microsemi.com/Portal/Default.aspx.

For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), log in at soc.microsemi.com/Portal/Default.aspx, go to the **My Cases** tab, and select **Yes** in the ITAR drop-down list when creating a new case. For a complete list of ITAR-regulated Microchip FPGAs, visit the ITAR web page.

You can track technical cases online by going to My Cases.

14.3 Website

You can browse a variety of technical and non-technical information on the Microchip FPGA Products Group home page, at www.microsemi.com/soc.

14.4 Outside the U.S.

Customers needing assistance outside the US time zones can either contact technical support at (https://soc.microsemi.com/Portal/Default.aspx) or contact a local sales office.

Visit About Us for sales office listings and corporate contacts.

The Microchip Website

Microchip provides online support via our website at www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods being used in attempts to breach the code protection features of the Microchip devices. We believe that these methods require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Attempts to breach these code protection features, most likely, cannot be accomplished without violating Microchip's intellectual property rights.
- Microchip is willing to work with any customer who is concerned about the integrity of its code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is "unbreakable." Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Trademarks

ISBN: 978-1-5224-7922-2

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: www.microchip.com/support Web Address: www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Australia - Sydney** Tel: 61-2-9868-6733 **China - Beijing** Tel: 86-10-8569-7000 **China - Chengdu** Tel: 86-28-8665-5511 **China - Chongqing** Tel: 86-23-8980-9588 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 **China - Hong Kong SAR** Tel: 852-2943-5100 **China - Nanjing** Tel: 86-25-8473-2460 **China - Qingdao** Tel: 86-532-8502-7355 **China - Shanghai** Tel: 86-21-3326-8000 **China - Shenyang** Tel: 86-24-2334-2829 **China - Shenzhen** Tel: 86-755-8864-2200 **China - Suzhou** Tel: 86-186-6233-1526 **China - Wuhan** Tel: 86-27-5980-5300 **China - Xian** Tel: 86-29-8833-7252 **China - Xiamen** Tel: 86-592-2388138 **China - Zhuhai** Tel: 86-756-3210040	**India - Bangalore** Tel: 91-80-3090-4444 **India - New Delhi** Tel: 91-11-4160-8631 **India - Pune** Tel: 91-20-4121-0141 **Japan - Osaka** Tel: 81-6-6152-7160 **Japan - Tokyo** Tel: 81-3-6880- 3770 **Korea - Daegu** Tel: 82-53-744-4301 **Korea - Seoul** Tel: 82-2-554-7200 **Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906 **Malaysia - Penang** Tel: 60-4-227-8870 **Philippines - Manila** Tel: 63-2-634-9065 **Singapore** Tel: 65-6334-8870 **Taiwan - Hsin Chu** Tel: 886-3-577-8366 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 **Thailand - Bangkok** Tel: 66-2-694-1351 **Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4485-5910 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-72400 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-72884388 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820