PlanAhead ソフトウェア
Tcl コマンド リファレンス ガイド

UG789 (2011 年 3 月 1 日)

XILINX®

本資料は英語版 (v.13.1) を翻訳したもので、内容に相違が生じる場合には原文を優先します。
資料によっては英語版の更新に対応していないものがあります。
日本語版は参考用としてご使用の上、最新情報につきましては、必ず最新英語版をご参照ください。

XILINX®

第 1 章

カテゴリ別 SDC および Tcl コマンド

プロジェクト コマンド

- add_files：アクティブなファイルセットへソースを追加します。
- archive_project：現在のプロジェクトを圧縮します。
- change_msg_type：メッセージ タイプを ID 別に変更します。
- close_design：現在のデザインを閉じます。
- close_project：現在開いているプロジェクトを閉じます。
- config_run：run の各プログラム オプションを設定します。
- create_fileset：新規ファイルセットを作成します。
- create_project：プロジェクトを新規作成します。
- create_run：現在のプロジェクトの合成またはインプリメンテーション run を定義します。
- current_fileset：現在のファイルセットを表示します。
- current_project：現在のプロジェクトを設定または表示します。
- current_run：現在の run を設定または表示します。
- delete_fileset：ファイルセットを削除します。
- delete_run：既存の run を削除します。
- get_files：ソース ファイルのリストを表示します。
- get_filesets：現在のプロジェクトのファイルセットのリストを表示します。
- get_parts：ソフトウェアで使用可能なパーツのリストを表示します。
- get_projects：プロジェクトのリストを表示します。
- get_runs：run (実行パターン) のリストを表示します。
- help：1 つまたは複数の項目に対するヘルプを表示します。
- import_as_run：NCD とオプションの TWX を run としてインポートします。
- import_files：ファイル/ディレクトリをアクティブなファイルセットにインポートします。
- import_ip：IP ファイルをインポートしてファイルセットに追加します。
- import_xise：現在のプロジェクトにザイリンクス ISE プロジェクト ファイルの設定をインポートします。
- import_xst：指定した XST ファイルをインポートします。
- launch_runs：run のセットを起動します。

- open_impl_design : インプリメント済みデザインを開きます。
- open_io_design : IO デザインを開きます。
- open_netlist_design : 合成またはネットリスト デザインを開きます。
- open_project : PlanAhead プロジェクト ファイル (.ppr) を開きます。
- open_rtl_design : RTL デザインを開きます。
- refresh_design : 現在のデザインを最新情報に更新します。
- reimport_files : 期限切れのファイルがあれば、それをインポートし直します。
- remove_files : ファイルセットからファイルまたはディレクトリを削除します。
- reorder_files : アクティブなファイルセットでソース ファイルの順序を変更します。
- reset_run : 既存の run をリセットします。
- save_design : 現在のデザインを保存します。
- save_design_as : 現在のデザインを新しい制約セットとして保存します。
- save_project_as : 現在のプロジェクトを新しい名前で保存します。
- set_speed_grade : タイミング スピード グレード
- update_file : リモート ファイルの内容でインポートされたファイルをアップデートします。
- upgrade_ip : コンフィギャブル IP を最新 IP にアップデートします。
- wait_on_run : 指定した run が終了するまで Tcl コマンドの実行を停止します。
- write_bitstream : 現在のデザインのビットストリームを書き出します。

ファイル入力および出力コマンドファイル

- read_chipscope_cdc : ChipScope Core Inserter の CDC ファイルをインポートします。
- read_csv : パッケージ ピンとポート配置情報をインポートします。
- read_pxml : PXML ファイルからパーティション定義をインポートします。
- read_twx : Trace STA ツールからタイミング結果を読み込みます。
- read_ucf : ファイルから物理制約をインポートします。
- read_verilog : 1 つまたは複数の Verilog ファイルを読み込みます。
- read_vhdl : 1 つまたは複数の VHDL ファイルを読み込みます。
- read_xdl : ファイルから配置情報をインポートします。
- write_bitstream : 現在のデザインのビットストリームを書き出します。
- write_chipscope_cdc : デバッグ ポートに接続されているネットをエクスポートします。
- write_csv : パッケージ ピンとポート配置情報をエクスポートします。
- write_edf : 現在のネットリストを EDIF ファイルとしてエクスポートします。
- write_ibis : 現在のフロアプランの IBIS モデルを書き出します。
- write_ncd : 配置を NCD ファイルにエクスポートします。
- write_pcf : 変換された制約を物理制約ファイル (.pcf) にエクスポートします。
- write_sdf : イベント シミュレーションで平坦な SDF 遅延ファイルを生成します。
- write_timing : タイミング結果のセットをファイルにエクスポートします。
- write_ucf : UCF 情報をファイルまたはディレクトリにエクスポートします。
- write_verilog : 現在のネットリストを Verilog 形式でエクスポートします。
- write_vhdl : 現在のネットリストのポートを VHDL 形式でエクスポートします。
- write_xdc : XDC 情報をファイルまたはディレクトリにエクスポートします。

オブジェクト コマンド

- filter : リストにフィルタをかけて、新しいリストを表示します。
- get_cells : 現在のデザインのセルすべてのリストを表示します。
- get_clocks : 現在のデザインのクロックのリストを表示します。
- get_debug_ports : 現在のデザインの ChipScope デバッグ ポートのリストを表示します。
- get_designs : 現在のデザインに含まれるデザインのリストを表示します。
- get_files : ソース ファイルのリストを表示します。
- get_filesets : 現在のプロジェクトのファイルセットのリストを表示します。
- get_generated_clocks : 現在のデザインの生成済みクロックのリストを表示します。
- get_interfaces : 現在のデザインの I/O ポート インターフェイスのリストを表示します。
- get_iobanks : IO バンクのリストを表示します。
- get_lib_cells : 現在のデザインのライブラリ セルのリストを表示します。
- get_lib_pins : 現在のデザインのライブラリ セル ピンのリストを表示します。
- get_libs : 現在のデザインのライブラリのリストを表示します。
- get_msg_limit : メッセージ数制限を表示します。
- get_nets : 現在のデザインのネットのリストを表示します。
- get_package_pins : パッケージ ピンのリストを表示します。
- get_param : パラメータ値を表示します。
- get_parts : ソフトウェアで使用可能なパーツのリストを表示します。
- get_path_groups : 現在のデザインのパス グループのリストを表示します。
- get_pblocks : 現在のデザインの Pblock のリストを表示します。
- get_pins : 現在のデザインのピンのリストを表示します。
- get_ports : 現在のデザインのポートのリストを表示します。
- get_projects : プロジェクトのリストを表示します。
- get_property : オブジェクトのプロパティを表示します。
- get_reconfig_modules : 現在のプロジェクトのリコンフィギャブル モジュールのリストを表示します。
- get_runs : run (実行パターン) のリストを表示します。
- get_selected_objects : 選択したオブジェクトを表示します。
- get_sites : サイトのリストを表示します。
- list_property : オブジェクトのプロパティをリストします。
- report_property : オブジェクトのプロパティをレポートします。
- set_property : オブジェクトのプロパティを設定します。

プロパティおよびパラメータ コマンド

- create_property : オブジェクトのクラスのプロパティを作成します。
- filter : リストにフィルタをかけて、新しいリストを表示します。
- get_param : パラメータ値を表示します。
- get_property : オブジェクトのプロパティを表示します。
- list_param : すべてのパラメータ名を表示します。
- list_property : オブジェクトのプロパティをリストします。
- report_param : すべてのパラメータに関する情報を表示します。
- report_property : オブジェクトのプロパティをレポートします。
- set_param : パラメータ値を設定します。
- set_property : オブジェクトのプロパティを設定します。

SDC コマンド

- all_clocks : 現在のデザインのクロックすべてのリストを表示します。
- all_inputs : 現在のデザインの入力ポートすべてのリストを表示します。
- all_outputs : 現在のデザインの出力ポートすべて のリストを表示します。
- all_registers : 現在のデザインのレジスタ セルおよびピンすべてのリストを表示します。
- create_clock : クロック オブジェクトを作成します。
- create_generated_clock : 生成済みのクロック オブジェクトを作成します。
- current_design : 現在のデザインを設定または表示します。
- current_instance : 現在のインスタンスを設定または表示します。
- get_cells : 現在のデザインのセルすべてのリストを表示します。
- get_clocks : 現在のデザインのクロックのリストを表示します。
- get_lib_cells : 現在のデザインのライブラリ セルのリストを表示します。
- get_lib_pins : 現在のデザインのライブラリ セル ピンのリストを表示します。
- get_libs : 現在のデザインのライブラリのリストを表示します。
- get_nets : 現在のデザインのネットのリストを表示します。
- get_pins : 現在のデザインのピンのリストを表示します。
- get_ports : 現在のデザインのポートのリストを表示します。
- group_path : コスト ファンクション計算のパスをグループ化します。
- set_case_analysis : 入力を 1、0、rising、falling のいずれかに指定します。
- set_clock_gating_check : クロック ゲート付きチェックをキャプチャします。
- set_clock_groups : 専用または非同期のクロック グループを設定します。
- set_clock_latency : 実際または予測済みのクロック レイテンシをキャプチャします。
- set_clock_sense : ポートまたはピンのクロック エッジを設定します。
- set_clock_transition : 予測済みクロック遷移をキャプチャします。

- set_clock_uncertainty : False パスを定義します。
- set_data_check : データ間チェックを作成します。
- set_disable_timing : タイミング アークをディスエーブルにします。
- set_false_path : False パスを定義します。
- set_hierarchy_separator : 階層区切り文字を設定します。
- set_ideal_latency : 理想的なレイテンシを指定します。
- set_ideal_network : 理想的なネットワークを指定します。
- set_input_delay : ポートまたはピンの入力遅延を設定します。
- set_load : ポートおよびネットのキャパシタンスを設定します。
- set_logic_dc : ポート/ピンのロジック DC を設定します。
- set_logic_one : ポート/ピンのロジック 1 を設定します。
- set_logic_zero : ポート/ピンのロジック 0 を設定します。
- set_max_delay : タイミング パスの最大遅延を指定します。
- set_max_fanout : ポートまたはセルの最大ファンアウトを設定します。
- set_max_time_borrow : ラッチ用に借りることのできる時間を制限します。
- set_min_delay : タイミング パスの最小遅延を指定します。
- set_multicycle_path : マルチサイクル パスを定義します。
- set_operating_conditions : プロセス、温度、および電圧を設定します。
- set_output_delay : ポートまたはピンの出力遅延を設定します。
- set_propagated_clock : 伝搬されるクロック レイテンシを指定します。
- set_timing_derate : 定数遅延のディレーティング係数を指定します。
- set_units : チェックするユニットを設定します。

XDC コマンド – サポートされる SDC コマンドすべてを含む

- all_clocks : 現在のデザインのクロックすべてのリストを表示します。
- all_cpus : 現在のデザインの CPU セルのリストを表示します。
- all_dsps : 現在のデザインの DSP セルのリストを表示します。
- all_hsios : 現在のデザインの HSIO セルのリストを表示します。
- all_inputs : 現在のデザインの入力ポートすべてのリストを表示します。
- all_mults : 現在のデザインの MULT セルのリストを表示します。
- all_outputs : 現在のデザインの出力ポートすべて のリストを表示します。
- all_rams : 現在のデザインの RAM セルのリストを表示します。
- all_registers : 現在のデザインのレジスタ セルおよびピンすべてのリストを表示します。
- config_timing_corners : シングル/マルチ コーナーのタイミング解析を設定します。
- config_timing_pessimism : 共通ノード パーミッション解析を設定します。
- create_clock : クロック オブジェクトを作成します。
- create_generated_clock : 生成済みのクロック オブジェクトを作成します。

- create_operating_conditions : 新しい動作状態を作成します。
- create_pblock : 新規 Pblock を作成します。
- current_design : 現在のデザインを設定または表示します。
- current_instance : 現在のインスタンスを設定または表示します。
- get_cells : 現在のデザインのセルすべてのリストを表示します。
- get_clocks : 現在のデザインのクロックのリストを表示します。
- get_debug_cores : 現在のデザインの ChipScope デバッグ コアのリストを表示します。
- get_debug_ports : 現在のデザインの ChipScope デバッグ ポートのリストを表示します。
- get_designs : 現在のデザインに含まれるデザインのリストを表示します。
- get_generated_clocks : 現在のデザインの生成済みクロックのリストを表示します。
- get_interfaces : 現在のデザインの I/O ポート インターフェイスのリストを表示します。
- get_iobanks : IO バンクのリストを表示します。
- get_lib_cells : 現在のデザインのライブラリ セルのリストを表示します。
- get_lib_pins : 現在のデザインのライブラリ セル ピンのリストを表示します。
- get_libs : 現在のデザインのライブラリのリストを表示します。
- get_nets : 現在のデザインのネットのリストを表示します。
- get_package_pins : パッケージ ピンのリストを表示します。
- get_path_groups : 現在のデザインのパス グループのリストを表示します。
- get_pins : 現在のデザインのピンのリストを表示します。
- get_ports : 現在のデザインのポートのリストを表示します。
- get_property : オブジェクトのプロパティを表示します。
- get_sites : サイトのリストを表示します。
- group_path : コスト ファンクション計算のパスをグループ化します。
- remove_disable_timing : タイミング アークをイネーブルにします。
- set_case_analysis : 入力を 1、0、rising、falling のいずれかに指定します。
- set_clock_gating_check : クロック ゲート付きチェックをキャプチャします。
- set_clock_groups : 専用または非同期のクロック グループを設定します。
- set_clock_latency : 実際または予測済みのクロック レイテンシをキャプチャします。
- set_clock_sense : ポートまたはピンのクロック エッジを設定します。
- set_clock_transition : 予測済みクロック遷移をキャプチャします。
- set_clock_uncertainty : False パスを定義します。
- set_data_check : データ間チェックを作成します。
- set_delay_model : タイミング遅延モデル
- set_disable_timing : タイミング アークをディスエーブルにします。
- set_false_path : False パスを定義します。
- set_hierarchy_separator : 階層区切り文字を設定します。
- set_ideal_latency : 理想的なレイテンシを指定します。
- set_ideal_network : 理想的なネットワークを指定します。

- set_input_delay : ポートまたはピンの入力遅延を設定します。
- set_input_jitter : クロック オブジェクトの入力ジッタを設定します。
- set_load : ポートおよびネットのキャパシタンスを設定します。
- set_logic_dc : ポート/ピンのロジック DC を設定します。
- set_logic_one : ポート/ピンのロジック 1 を設定します。
- set_logic_zero : ポート/ピンのロジック 0 を設定します。
- set_max_delay : タイミング パスの最大遅延を指定します。
- set_max_fanout : ポートまたはセルの最大ファンアウトを設定します。
- set_max_time_borrow : ラッチ用に借りることのできる時間を制限します。
- set_min_delay : タイミング パスの最小遅延を指定します。
- set_multicycle_path : マルチサイクル パスを定義します。
- set_operating_conditions : プロセス、温度、および電圧を設定します。
- set_output_delay : ポートまたはピンの出力遅延を設定します。
- set_propagated_clock : 伝搬されるクロック レイテンシを指定します。
- set_property : オブジェクトのプロパティを設定します。
- set_switching_activity : 指定したオブジェクトのスイッチ動作を設定します。
- set_system_jitter : システム ジッタを設定します。
- set_timing_derate : 定数遅延のディレーティング係数を指定します。
- set_units : チェックするユニットを設定します。

GUI 制御コマンド

- endgroup : グループ単位で前/次の動作を実行できるコマンド セットを終了します。
- get_selected_objects : 選択したオブジェクトを表示します。
- highlight_objects : オブジェクトを別の色でハイライトします。
- mark_objects : GUI でオブジェクトをマークします。
- redo : 前のコマンドを再実行します。
- select_objects : GUI でオブジェクトを選択します。
- start_gui : PlanAhead の GUI を起動します。
- startgroup : グループ単位で前/次の動作を実行できるコマンド セットを開始します。
- stop_gui : PlanAhead の GUI を閉じます。
- undo : 前のコマンドの実行を取り消します。
- unhighlight_objects : 現在ハイライトされているオブジェクトのハイライトを解除します。
- unmark_objects : 現在マークされているアイテムのマークを解除します。
- unselect_objects : 現在選択されているアイテムの選択を解除します。

ツール起動コマンド

- compxlib : シミュレーション ライブラリをコンパイルします。
- crossprobe_fed : BEL およびネットのパスを FPGA Editor にクロスプローブします。
- launch_chipscope_analyzer : run に対して ChipScope Analyzer ツールを起動します。
- launch_fpga_editor : run に対して FPGA Editor ツールを起動します。
- launch_impact : run に対して iMPACT コンフィギュレーション ツールを起動します。
- launch_isim : ISim シミュレータを使用してシミュレーションを実行します。
- launch_xpa : XPower Analyzer ツールを起動します。

レポート コマンド

- check_timing : 可能性のあるタイミング問題のタイミングをチェックします。
- create_slack_histogram : ヒストグラムを作成します。
- delete_timing_results : メモリからタイミング結果のセットを一掃します。
- get_msg_count : メッセージ カウントを表示します。
- get_msg_limit : メッセージ数制限を表示します。
- report_clock_interaction : 内部クロック タイミング パスとクロックなしのレジスタについてレポートします。
- report_constraint : デザインの制約に関連する情報を表示します。
- report_control_sets : デザイン特有の制御セットについてレポートします。
- report_debug_core : ChipScope デバッグ コアの詳細をレポートします。
- report_delay_calculation : セルまたはネット アークの遅延計算の詳細をレポートします。
- report_disable_timing : ディスエーブルにしたタイミング アークをレポートします。
- report_drc : DRC を実行します。
- report_io : デバイスのすべての IO サイトに関する情報を表示します。
- report_min_pulse_width : 最小パルス幅チェックをレポートします。
- report_param : すべてのパラメータに関する情報を表示します。
- report_power : 電力概算を実行し、レポートを表示します。
- report_property : オブジェクトのプロパティをレポートします。
- report_resources : リソース概算を実行し、レポートを表示します。
- report_ssn : 現在のパッケージおよびピン配置で SSN 解析を実行します。
- report_sso : 現在のパッケージおよびピン配置で WASSO 解析を実行します。
- report_stats : 統計をレポートします。
- report_timing : タイミング パスをレポートします。
- report_transformed_primitives : Unisim プリミティブ変換の詳細をレポートします。
- report_ucf_timing : タイミング パスをレポートします。
- report_utilization : デバイス使用率を計算し、レポートを表示します。
- reset_drc : DRC を削除します。
- reset_msg_limit : メッセージ数制限をリセットします。
- reset_path : 1 サイクルのビヘイビアに指定パスをリセットします。
- reset_ssn : メモリから SSN 結果を一掃します。
- reset_sso : メモリから WASSO 結果を一掃します。
- reset_timing : 現在のデザインのタイミング情報をリセットします。
- reset_ucf : ファイルから読み込んだフロアプラン制約を一掃します。
- set_msg_limit : メッセージ数制限を設定します。
- version : PlanAhead のバージョンおよびバージョンの日付を表示します。

フロアプラン コマンド

- add_cells_to_pblock : Pblock へセルを追加します。
- create_pblock : 新規 Pblock を作成します。
- delete_pblock : Pblock を削除します。
- delete_rpm : RPM を削除します。
- get_pblocks : 現在のデザインの Pblock のリストを表示します。
- place_pblocks : Pblock 配置ツールを実行します。
- remove_cells_from_pblock : Pblock からセルを削除します。
- resize_pblock : UCF の範囲を移動、サイズ変更、追加、削除します。
- swap_locs : 2 つの位置を入れ替えます。

ピン配置コマンド

- create_interface : I/O ポート インターフェイスを新規作成します。
- create_port : スカラー ポートまたはバス ポートを作成します。
- delete_interface : I/O ポート インターフェイスをプロジェクトから削除します。
- delete_port : ポートまたはポート バスのリストを削除します。
- make_diff_pair_ports : 差動ペアを 2 ポートにします。
- place_ports : ポートのセットを自動的に配置します。
- set_package_pin_val : 1 つまたは複数のパッケージ ピンのユーザー コラムを設定します。
- split_diff_pair_ports : 2 ポート間の差動ペア関係を削除します。

CORE Generator™ コマンド

- create_ip : コンフィギャブル IP のインスタンスを作成して、ファイルセットに追加します。
- create_ip_catalog : 最新バージョンの IP カタログを作成して、指定ディレクトリに保存します。
- generate_ip : コンフィギャブル IP を生成します。
- import_ip : IP ファイルをインポートしてファイルセットに追加します。
- reset_ip : コンフィギャブル IP をリセットします。

ChipScope™ コマンド

- connect_debug_port : ネットとピンをデバッグ ポート チャネルに接続します。
- create_debug_core : ChipScope デバッグ コアを新規作成します。
- create_debug_port : ChipScope デバッグ ポートを新規作成します。
- delete_debug_core : ChipScope デバッグ コアを削除します。
- delete_debug_port : ChipScope デバッグ ポートを削除します。
- disconnect_debug_port : デバッグ ポート チャネルからのネットとピンの接続を解除します。
- implement_debug_core : ChipScope デバッグ コアをインプリメントします。

パーティション コマンド

- config_partition : 指定 run のモジュール変数とステートを設定します。
- demote_run : 前にプロモートしたパーティションのプロモートを解除して、インポートに使用できないようにします。
- promote_run : 前にインプリメントしたパーティションをプロモートして、インポートおよびこの run やその他の run で再利用できるようにします。
- set_property : オブジェクトのプロパティを設定します。

パーシャル リコンフィギュレーション コマンド

- config_partition : 指定 run のモジュール変数とステートを設定します。
- create_reconfig_module : 新しいリコンフィギャブル モジュールを作成してセルに追加します。セルは、リコンフィギャブル パーティションとして表示されます。
- delete_reconfig_module : リコンフィギャブル モジュールを削除します。
- demote_run : 前にプロモートしたパーティションのプロモートを解除して、インポートに使用できないようにします。
- get_reconfig_modules : 現在のプロジェクトのリコンフィギャブル モジュールのリストを表示します。
- load_reconfig_modules : 指定 run から特定のリコンフィギャブル モジュールまたはすべてのモジュールを読み込みます。
- promote_run : 前にインプリメントしたパーティションをプロモートして、インポートおよびこの run やその他の run で再利用できるようにします。
- set_property : オブジェクトのプロパティを設定します。
- verify_config : インプリメントされた run を解析して、パーシャル リコンフィギュレーションに必要な規則にしたがっているかどうかを解析します。

XILINX®

第 2 章

アルファベット別 SDC および Tcl コマンド

この章では、アルファベット順にすべての PlanAhead™ ソフトウェアの SDC および Tcl コマンドをリストしています。

add_cells_to_pblock

Pblock へセルを追加します。

構文

```
add_cells_to_pblock [-add_primitives] [-clear_locs]
[-quiet] pblock cells ...
```

表示結果

なし

使用法

-add_primitives
指定したピンスタンスのすべてのプリミティブを Pblock へ割り当てます。

-clear_locs
インスタンスの位置制約を削除します。

-quiet
コマンド エラーを無視します。

pblock (必須)
セルを追加する Pblock を指定します。

cells (必須)
追加するセルを指定します。

add_files

アクティブなファイルセットへソースを追加します。

構文

```
add_files [-fileset arg ] [-norecurse] [-scan_for_includes]
[-quiet] [ files ...]
```

表示結果

追加されたファイル オブジェクトのリスト

使用法

–fileset
ファイルセット名を指定します。

–norecurse
指定ディレクトリで繰り返し検索します。

–scan_for_includes
ファイルセットの RTL ソースに含まれるファイルすべてをスキャンして追加します。

–quiet
コマンド エラーを無視します。

files
追加するファイルおよびディレクトリ名を指定します。

all_clocks

現在のデザインのクロックすべてのリストを表示します。

構文

```
all_clocks [-quiet]
```

表示結果

クロックのリスト

使用法

-quiet
コマンド エラーを無視します。

all_cpus

現在のデザインの CPU セルのリストを表示します。

構文

```
all_cpus [-quiet]
```

表示結果

CPU セル オブジェクトのリスト

使用法

-quiet

コマンド エラーを無視します。

all_dsps

現在のデザインの DSP セルのリストを表示します。

構文

```
all_dsps [-quiet]
```

表示結果

DSP セル オブジェクトのリスト

使用法

-quiet
コマンド エラーを無視します。

all_hsios

現在のデザインの HSIO セルのリストを表示します。

構文

```
all_hsios [-quiet]
```

結果

HSIO セル オブジェクトのリスト

使用法

-quiet

コマンド エラーを無視します。

all_inputs

現在のデザインの入力ポートすべてのリストを表示します。

構文

```
all_inputs [-quiet]
```

表示結果

ポート オブジェクトのリスト

使用法

-quiet
コマンド エラーを無視します。

all_mults

現在のデザインの MULT セルのリストを表示します。

構文

```
all_mults [-quiet]
```

結果

MULT セル オブジェクトのリスト

使用法

-quiet
コマンド エラーを無視します。

all_outputs

現在のデザインの出力ポートすべて のリストを表示します。

構文

```
all_outputs [-quiet]
```

表示結果

ポート オブジェクトのリスト

使用法

-quiet
コマンド エラーを無視します。

all_rams

現在のデザインの RAM セルのリストを表示します。

構文

```
all_rams [-quiet]
```

表示結果

RAM セル オブジェクトのリスト

使用法

-quiet
コマンド エラーを無視します。

all_registers

現在のデザインのレジスタ セルおよびピンすべてのリストを表示します。

構文

```
all_registers [-clock args ] [-rise_clock args ]
[-fall_clock args ] [-cells] [-data_pins] [-clock_pins]
[-async_pins] [-output_pins] [-level_sensitive] [-edge_triggered]
[-quiet]
```

表示結果

セルまたはピン オブジェクトのリスト

使用法

-clock
このクロックのレジスタを考慮します。

-rise_clock
クロックの立ち上がりエッジでトリガされるレジスタを考慮します。

-fall_clock
クロックの立ち下がりエッジでトリガされるレジスタを考慮します。

-cells
セルのリストを表示します (デフォルト)。

-data_pins
レジスタ データ ピンのリストを表示します。

-clock_pins
レジスタ クロック ピンのリストを表示します。

-async_pins
非同期プリセット/クリア ピンのリストを表示します。

-output_pins
レジスタ出力ピンのリストを表示します。

-level_sensitive
レベル センシティブ ラッチのみを考慮します。

-edge_triggered
エッジ トリガのフリップフロップのみを考慮します。

-quiet
コマンド エラーを無視します。

archive_project

現在のプロジェクトを圧縮します。

構文

```
archive_project [-force] [-exclude_run_results] [-quiet] [ file ]
```

結果

true

使用法

–force
既存の圧縮ファイルを上書きします。

–exclude_run_results
圧縮に run 結果を含めません。

–quiet
コマンド エラーを無視します。

file
圧縮ファイル名を指定します。

change_msg_type

メッセージ タイプを ID 別に変更します。

構文

```
change_msg_type -id arg -type arg [-quiet]
```

表示結果

なし

使用法

-id (必須)
「common-99」などの特定のメッセージ ID を指定します。

-type (必須)
「ERROR」など、変更するメッセージ タイプを指定します。

-quiet
コマンド エラーを無視します。

check_timing

可能性のあるタイミング問題のタイミングをチェックします。

構文

```
check_timing [-override_defaults args ] [-include args ]
[-exclude args ] [-verbose] [-quiet] [ check_list ]
```

表示結果

なし

使用法

–override_defaults
check_list を使用して timing_check_defaults 変数のチェックを上書きします。

–include
timing_check_defaults 変数に対して実行されるチェックのリストを追加します。

–exclude
timing_check_defaults 変数に対して実行されるチェックのリストを除外します。

–verbose
より詳細な出力になります。

–quiet
コマンド エラーを無視します。

check_list : {unconstrained_endpoints multiple_clock no_clock no_input_delay loops generated_clocks}
実行されるチェックのリストで、有効な値は unconstrained_endpoints、multiple_clock、no_clock,no_input_delay、loops、generated_clocks になります。

close_design

現在のデザインを閉じます。

構文

```
close_design [-quiet]
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

close_project

現在開いているプロジェクトを閉じます。

構文

```
close_project [-delete] [-quiet]
```

結果

なし

使用法

-delete
ディスクからもプロジェクトを削除します。

-quiet
コマンド エラーを無視します。

compxlib

シミュレーション ライブラリをコンパイルします。

構文

```
compxlib [-arch arg ] [-cfg] [-cfgopt arg ] [-dir arg ] [-e arg ]
[-exclude_sublib] [-exclude_superseded] [-info arg ] [-l arg ]
[-lib arg ] [-log arg ] [-p arg ] -s arg [-source_lib arg ]
[-verbose] [-w] [-64bit] [-quiet]
```

表示結果

なし

使用法

-arch デフォルト : all
デバイス アーキテクチャを選択します。

-cfg デフォルト : compxlib.cfg
コンフィギュレーション ファイルをデフォルト設定で生成します。

-cfgopt
simulator:language:library:options の形式でオプションを設定します。

-dir デフォルト : .
コンパイルした結果を保存するディレクトリ パスを指定します。

-e
前に compxlib でコンパイルしたライブラリのある既存ディレクトリを指定します。

-exclude_sublib
EDK の .pao ファイルで定義されるサブライブラリをコンパイルで除外します (EDK ライブラリのみ)。

-exclude_superseded
使用されていない EDK のライブラリをコンパイルで除外します (EDK ライブラリのみ)。

-info
コンパイル済みライブラリの情報を表示します。

-l デフォルト : all
この言語のライブラリをコンパイルします。

-lib デフォルト : all
コンパイルするライブラリを選択します。

-log デフォルト : compxlib.log
ユーザー独自のログ ファイルを作成します。

-p
このディレクトリのシミュレータの実行ファイルを使用します。

-s (必須)
このシミュレータ用のライブラリをコンパイルします。

-source_lib
指定した場合は、環境変数 XILINX (ISE 用) または XILINX_EDK (EDK 用) で指定されているデフォルト パスを検索する前に、このディレクトリでライブラリ ソース ファイルが検索されます。

-verbose
プログラム実行中にさらに多くのメッセージが表示されます。

-w
コンパイル済みライブラリが上書きされます。

-64bit
64 ビットのコンパイルが実行されます。

-quiet
コマンド エラーを無視します。

config_partition

指定 run のモジュール変数とステートを設定します。

構文

```
config_partition [-cell arg ] [-reconfig_module arg ] [-import]
[-implement] [-import_dir arg ] [-preservation arg ] [-quiet] run
```

結果

なし

使用法

-cell
指定した run でコンフィギュレーションするパーティション インスタンスを指定します。最上位のパーティションを変更する場合は、このオプションは指定しないでください。

-reconfig_module
この run のこのインスタンスに適用するさまざまなリコンフィギャブル モジュールを指定します。

-import
この run に対して動作をインポートするために、インスタンス (またはスタティック ロジック) を設定します。

-implement
この run に対して動作をインプリメントするために、インスタンス (またはスタティック ロジック) を設定します。

-import_dir
この前にインプリメントされたモジュールのインポート元のディレクトリを指定します。

-preservation
パーティションの保持レベルを設定します。有効な値は、routing (デフォルト)、placement、synthesis のいずれかです。

-quiet
コマンド エラーを無視します。

run (必須)
変更する run を指定します。

config_run

run の各プログラム オプションを設定します。

構文

```
config_run [-quiet] run  program  option  value
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

run (必須)
変更する run を指定します。

program (必須)
オプションを設定するプログラムの名前を指定します。

option (必須)
設定するオプションの名前を指定します。

value (必須)
オプションに対して設定する引数の値を指定します。

config_timing_corners

シングル/マルチ コーナーのタイミング解析を設定します。

構文

```
config_timing_corners [-corner arg ] [-multi_corner arg ]
[-delay_type arg ] [-quiet]
```

表示結果

なし

使用法

-corner
コーナー名。有効な値は Slow、Fast です。

-multi_corner
マルチ コーナー解析を実行します。有効な値は、on、off です。

-delay_type
指定したタイミング コーナーに対して解析されるパス遅延を指定します。有効な値は、none、max、min、min_max です。

-quiet
コマンド エラーを無視します。

config_timing_pessimism

共通ノード パーミッション解析を設定します。

構文

```
config_timing_pessimism [-enable] [-disable] [-transition arg ]
[-threshold arg ] [-common_node arg ] [-quiet]
```

表示結果

なし

使用法

-enable
CRPR (Common Reconvergence Pessimism Removal) をイネーブルにします。

-disable
CRPR (Common Reconvergence Pessimism Removal) をディスエーブルにします。

-transition
指定した遷移の悲観的設定を削除します。有効な値は、any_transition、same_transition です。

-threshold
共通ノードの悲観的しきい値を設定します。

-common_node
配線ネットワークの共通ノードに対して CRPR を実行します。値 : on、off

-quiet
コマンド エラーを無視します。

connect_debug_port

ネットとピンをデバッグ ポート チャネルに接続します。

構文

```
connect_debug_port [-channel_start_index arg ]
[-quiet] port  nets ...
```

表示結果

なし

使用法

-channel_start_index
チャネル インデックスからネットを接続します。

-quiet
コマンド エラーを無視します。

port (必須)
デバッグ ポート名を指定します。

nets (必須)
ネットまたはピンのリストを指定します。

create_clock

クロック オブジェクトを作成します。

構文

```
create_clock [-period arg ] [-name arg ] [-waveform args ] [-add]
[-quiet] [ objects ...]
```

結果

クロック名

使用法

-period デフォルト : 10.0
クロック周期 :値 0

-name
クロック名を指定します。

-waveform
クロック エッジを指定します。

-add
source_objects の既存クロックに追加します。

-quiet
コマンド エラーを無視します。

objects
クロック ソース ポート、ピンまたはネットを指定します。

create_debug_core

ChipScope デバッグ コアを新規作成します。

構文

```
create_debug_core [-quiet] name  type
```

結果

new debug_core object

使用法

-quiet
コマンド エラーを無視します。

name (必須)
新しいデバッグ コア インスタンスの名前を指定します。

type (必須)
新しいデバッグ コア インスタンスのタイプを指定します。

create_debug_port

ChipScope デバッグ ポートを新規作成します。

構文

```
create_debug_port [-quiet] name  type
```

結果

新しい debug_port オブジェクト

使用法

-quiet
コマンド エラーを無視します。

name (必須)
デバッグ コア インスタンスの名前を指定します。

type (必須)
デバッグ コア インスタンスのタイプを指定します。

create_fileset

新規ファイルセットを作成します。

構文

```
create_fileset [-srcset] [-constrset] [-simset] [-quiet] name
```

結果

新しいファイルセット オブジェクト

使用法

–srcset
ファイルセットをソース ファイルセットとして作成します。

–constrset
ファイルセットを制約ファイルセットとして作成します (デフォルト)。

–simset
ファイルセットをシミュレーション ソース ファイルセットとして作成します。

–quiet
コマンド エラーを無視します。

name (必須)
作成されるファイルセットの名前を指定します。

create_generated_clock

生成済みのクロック オブジェクトを作成します。

構文

```
create_generated_clock [-name arg ] [-source args ]
[-edges args ] [-divide_by arg ] [-multiply_by arg ]
[-combinational] [-duty_cycle arg ] [-invert] [-edge_shift args ]
[-add] [-master_clock arg ] [-quiet] [ objects ...]
```

表示結果

クロック名

使用法

-name
生成されたクロックの名前を指定します。

-source
マスター クロック ソース オブジェクト ピン/ポートを指定します。

-edges
エッジを指定します。

-divide_by デフォルト : 0
周波数除算係数 :値 = 1

-multiply_by デフォルト : 0
周波数乗算係数 :値 = 1

-combinational
組み合わせロジックを介して 1 クロックで分周されます。

-duty_cycle デフォルト : 0.0
周波数乗算のデューティ サイクル範囲 :0.0 ～ 100.0

-invert
信号を反転します。

-edge_shift
エッジ シフトを指定します。

-add
source_objects の既存クロックに追加します。

-master_clock
マスター ピンに複数のクロックがある場合は、このクロックを使用します。

-quiet
コマンド エラーを無視します。

objects
生成されたクロック ソース オブジェクト

create_interface

I/O ポート インターフェイスを新規作成します。

構文

```
create_interface [-parent arg ] [-quiet] name
```

表示結果

新しいインターフェイス オブジェクト

使用法

-parent
この親インターフェイスに新しいインターフェイスを割り当てます。

-quiet
コマンド エラーを無視します。

name (必須)
新しい I/O ポート インターフェイスの名前を指定します。

create_ip

コンフィギャブル IP のインスタンスを作成して、ファイルセットに追加します。

構文

```
create_ip [-srcset arg ] -module_name arg -vendor arg -library arg
-name arg -version arg [-quiet]
```

表示結果

追加されたファイル オブジェクトのリスト

使用法

-srcset
ソース セット名を指定します。

-module_name (必須)
プロジェクトに追加される新しい IP の名前を指定します。

-vendor (必須)
IP ベンダーの名前を指定します。

-library (必須)
IP ライブラリの名前を指定します。

-name (必須)
IP 名を指定します。

-version (必須)
IP のバージョン番号を指定します。

-quiet
コマンド エラーを無視します。

create_ip_catalog

最新バージョンの IP カタログを作成して、指定ディレクトリに保存します。

構文

```
create_ip_catalog -dir arg [-repositories args ] [-quiet]
```

表示結果

新しい IP カタログ ファイルへのパス

使用法

-dir (必須)
新しい IP カタログを書き込むディレクトリを指定します。

-repositories
新しい IP カタログに含まれる IP レポジトリのリストを指定します。

-quiet
コマンド エラーを無視します。

create_operating_conditions

新しい動作状態を作成します。

構文

```
create_operating_conditions -name arg [-library arg ]
[-process arg ] [-temperature arg ] [-voltage arg ]
[-tree_type arg ] [-calc_mode arg ] [-airflow arg ]
[-rail_voltages args ] [-quiet]
```

結果

なし

使用法

-name (必須)
動作状態名を指定します。

-library
ライブラリ名

-process デフォルト : 0.0
プロセス乗数 :0 ～ 100

-temperature デフォルト : ファミリによって異なる
周囲温度 (C) :-55 ～ 125

-voltage デフォルト : 0.0
電圧 (V) :0 ～ 1000

-tree_type デフォルト : balanced_tree
ツリー タイプ

-calc_mode デフォルト : nominal
計算モード : nominal または worst_case

-airflow デフォルト : ファミリによって異なる
エアフロー (LFM) :0 ～ 750

-rail_voltages
レール電圧の名前と値の組み合わせリスト (サポートされる値 : vccint、vccaux、vcco33、vcco25、vcco18、vcco15、vcco12)

-quiet
コマンド エラーを無視します。

create_pblock

新規 Pblock を作成します。

構文

```
create_pblock [-parent arg ] [-quiet] name
```

表示結果

新しい Pblock オブジェクト

使用法

-parent
新しい Pblock の親を指定します。

-quiet
コマンド エラーを無視します。

name (必須)
新しい Pblock の名前を指定します。

create_port

スカラー ポートまたはバス ポートを作成します。

構文

```
create_port -direction arg [-from arg ] [-to arg ] [-diff_pair]
[-interface arg ] [-quiet] name
```

表示結果

作成されたポート オブジェクトのリスト

使用法

-direction (必須)
ポートの方向を指定します。有効な引数は、in、out、inout です。

-from
新しいバスのインデックスの始まりを指定します。

-to
新しいバスのインデックスの終わりを指定します。

-diff_pair
ポートの差動ペアを作成します。

-interface
このインターフェイスに新しいインターフェイスを割り当てます。

-quiet
コマンド エラーを無視します。

name (必須)
ポート名

create_project

プロジェクトを新規作成します。

構文

```
create_project [-part arg ] [-force] [-quiet] name  dir
```

結果

新しいプロジェクト オブジェクト

使用法

-part
プロジェクトのデフォルトのザイリンクス パーツを設定します。

-force
既存のプロジェクト ディレクトリを上書きします。

-quiet
コマンド エラーを無視します。

name (必須)
プロジェクト名

dir (必須)
プロジェクト ファイルが保存されるディレクトリ

create_property

オブジェクトのクラスのプロパティを作成します。

構文

```
create_property [-type arg ] [-quiet] name [ class ]
```

結果

問題のない場合はプロパティ、エラーのあった場合は ""

使用法

-type デフォルト : string
作成するプロパティのタイプ。有効な値は string、int、double、bool です。

-quiet
コマンド エラーを無視します。

name (必須)
作成するプロパティの名前を指定します。

class デフォルト : string
作成するプロパティのオブジェクト タイプ。有効な値は、design、net、cell、pin、port、pblock です。

create_reconfig_module

新しいリコンフィギャブル モジュールを作成してセルに追加します。セルは、リコンフィギャブル パーティションとして表示されます。

構文

```
create_reconfig_module [-force] [-blackbox] [-quiet] name  cell
```

結果

新しい reconfig_module オブジェクト

使用法

–force
未決定の制約変更があっても、コマンドを実行します。変更は失われます。

–blackbox
ブラック ボックス リコンフィギャブル モジュールを作成します。ソース ファイルと制約ファイルはブラック ボックス リコンフィギャブル モジュールには追加されない可能性があります。

–quiet
コマンド エラーを無視します。

name (必須)
新しいリコンフィギャブル モジュールの名前

cells (必須)
新しいリコンフィギャブル モジュールを受信するセル

create_run

現在のプロジェクトの合成またはインプリメンテーション run を定義します。

構文

```
create_run [-srcset arg ] [-constrset arg ] [-parent_run arg ]
[-part arg ] -flow arg [-strategy arg ] [-quiet] name
```

表示結果

run オブジェクト

使用法

–srcset
run のソース ファイルセット

–constrset
run の制約ファイルセット

–parent_run
新しいインプリメンテーション run に関連する合成 run

–part
パーツ名

–flow (必須)
DesignAhead フロー名

–strategy
run に適用するストラテジ

–quiet
コマンド エラーを無視します。

name (必須)
新規 run の名前

create_slack_histogram

ヒストグラムを作成します。

構文

```
create_slack_histogram [-to args ] [-delay_type arg ]
[-num_bins arg ] [-slack_less_than arg ]
[-slack_greater_than arg ] [-group args ] [-report_unconstrained]
[-significant_digits arg ] [-scale arg ] [-results arg ] [-quiet]
```

結果

なし

使用法

-to
範囲の終わりのクロックを指定します。

-delay_type デフォルト : max
パス遅延のタイプ :値 : max、min、min_max

-num_bins デフォルト :10
ビンの最大数値 = 1

-slack_less_than デフォルト :1e+30
これよりも少ないスラックのパスを表示します。

-slack_greater_than デフォルト :-1e+30
これよりも多いスラックのパスを表示します。

-group
このグループのパスに対するレポートを制限します。

-report_unconstrained
制約の付いていないエンドポイントをレポートします。

-significant_digits デフォルト :3
表示する桁数 :範囲 :0 ～ 13

-scale デフォルト : linear
ヒストグラムを描画する尺度のタイプを指定します。有効な値は、linear か logarithmic です。

-results
この名前で GUI パネルへ結果を出力します。

-quiet
コマンド エラーを無視します。

crossprobe_fed

BEL およびネットのパスを FPGA Editor にクロスプローブします。

構文

```
crossprobe_fed [-run arg ] [-path args ] [-objects args ]
[-quiet]
```

表示結果

なし

使用法

–run
実行パターン

–path
プリミティブ セルおよびネットに接続されるパス

–objects
クロスプローブするセルおよびネットのリスト

–quiet
コマンド エラーを無視します。

current_design

現在のデザインを設定または表示します。

構文

```
current_design [-quiet] [ design ]
```

結果

デザイン オブジェクト

使用法

–quiet
コマンド エラーを無視します。

design
設定される現在のデザイン名

current_fileset

現在のファイルセットを表示します。

構文

```
current_fileset [-srcset] [-constrset] [-quiet]
```

結果

ファイルセット オブジェクト

使用法

–srcset
現在のソース ファイルセットを表示します。

–constrset
現在の制約ファイルセットを表示します。

–quiet
コマンド エラーを無視します。

current_instance

現在のインスタンスを設定または表示します。

構文

```
current_instance [-quiet] [ instance ]
```

表示結果

インスタンス名

使用法

–quiet
コマンド エラーを無視します。

instance
インスタンス名

current_project

現在のプロジェクトを設定または表示します。

構文

```
current_project [-quiet] [ project ]
```

結果

現在のまたは新しく設定されたプロジェクト オブジェクト

使用法

–quiet
コマンド エラーを無視します。

project
現在のプロジェクトとして設定されるプロジェクト (オプション)

current_run

現在の run を設定または表示します。

構文

```
current_run [-synthesis] [-implementation] [-quiet] [ run ]
```

表示結果

run オブジェクト

使用法

-synthesis
現在の合成 run を設定または表示します。

-implementation
現在のインプリメンテーション run を設定または表示します (synthesis が指定されない限り、これがデフォルト)。

-quiet
コマンド エラーを無視します。

run
現在の run として設定する run (オプション)

delete_debug_core

ChipScope デバッグ コアを削除します。

構文

```
delete_debug_core [-quiet] cores ...
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

cores (必須)
削除する ChipScope デバッグ コアを指定します。

delete_debug_port

ChipScope デバッグ ポートを削除します。

構文

```
delete_debug_port [-quiet] ports ...
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

ports (必須)
削除する ChipScope デバッグ ポートを指定します。

delete_fileset

ファイルセットを削除します。

構文

```
delete_fileset [-quiet] fileset
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

fileset (必須)
削除するファイルセットを指定します。

delete_interface

I/O ポート インターフェイスをプロジェクトから削除します。

構文

```
delete_interface [-all] [-quiet] interfaces ...
```

表示結果

なし

使用法

-all
インターフェイスに付属するすべてのポートおよびバスも削除します。

-quiet
コマンド エラーを無視します。

interfaces (必須)
削除する I/O ポート インターフェイスを指定します。

delete_pblock

Pblock を削除します。

構文

```
delete_pblock [-hier] [-quiet] pblocks ...
```

表示結果

なし

使用法

-hier
Pblock の子ブロックすべても削除します。

-quiet
コマンド エラーを無視します。

pblocks (必須)
削除する Pblock を指定します。

delete_port

ポートまたはポート バスのリストを削除します。

構文

```
delete_port [-quiet] ports ...
```

表示結果

なし

使用法

–quiet
コマンド エラーを無視します。

ports (必須)
削除するポートを指定します。

delete_reconfig_module

リコンフィギャブル モジュールを削除します。

構文

```
delete_reconfig_module [-quiet] reconfig_module
```

結果

なし

使用法

-quiet
コマンド エラーを無視します。

reconfig_module (必須)
削除するリコンフィギャブル モジュールを指定します。

delete_rpm

RPM を削除します。

構文

```
delete_rpm [-quiet] rpm
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

rpm (必須)
削除する RPM を指定します。

delete_run

既存の run を削除します。

構文

```
delete_run [-noclean_dir] [-quiet] run
```

表示結果

なし

使用法

-noclean_dir
ディスクからすべての出力ファイルおよびディレクトリを削除しません。

-quiet
コマンド エラーを無視します。

run (必須)
変更する run を指定します。

delete_timing_results

メモリからタイミング結果のセットを一掃します。

構文

```
delete_timing_results [-type arg ] [-quiet] name
```

表示結果

なし

使用法

-type
削除するタイミング結果のタイプ。有効な値は、timing_path、slack_histogram、clock_interaction です。

-quiet
コマンド エラーを無視します。

name (必須)
削除する結果のセット名

demote_run

前にプロモートしたパーティションのプロモートを解除して、インポートに使用できないようにします。

構文

```
demote_run [-run arg ] [-partition_names args ]
[-promote_dir arg ] [-quiet]
```

表示結果

なし

使用法

-run
デモートされるプロモート済み run

-partition_names
プロモートされるパーティションのリスト

-promote_dir
デモートされるディレクトリ

-quiet
コマンド エラーを無視します。

disconnect_debug_port

デバッグ ポート チャネルからのネットとピンの接続を解除します。

構文

```
disconnect_debug_port [-channel_index arg ] [-quiet] port
```

表示結果

なし

使用法

-channel_index
チャネル インデックスのネット接続を解除します。

-quiet
コマンド エラーを無視します。

port (必須)
デバッグ ポート名を指定します。

endgroup

グループ単位で前/次の動作を実行できるコマンド セットを終了します。

構文

```
endgroup [-quiet]
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

filter

リストにフィルタをかけて、新しいリストを表示します。

構文

```
filter [-regexp] [-nocase] [-quiet] [ objects ] [ filter ]
```

表示結果

新しいリスト

使用法

-regexp
=~ および !~ 演算子で正規表現が使用されます。

-nocase
-regexp を使用した場合に、大文字/小文字が関係なくなります。

-quiet
コマンド エラーを無視します。

objects
フィルタするオブジェクトのリスト

filter
式を使用してリストをフィルタします。

generate_ip

コンフィギャブル IP を生成します。

構文

```
generate_ip [-srcset arg ] -files args [-quiet]
```

表示結果

生成されたファイルのリスト

使用法

–srcset
ソース セット名を指定します。

–files (必須)
生成される IP ソース ファイルを指定します。

–quiet
コマンド エラーを無視します。

get_cells

現在のデザインのセルすべてのリストを表示します。

構文

```
get_cells [-hsc arg ] [-hierarchical] [-regexp] [-nocase]
[-filter arg ] [-of_objects args ] [-match_style arg ] [-quiet]
[ patterns ...]
```

表示結果

セル オブジェクトのリスト

使用法

-hsc デフォルト : /
階層区切り文字

-hierarchical
現在のインスタンスでレベルごとに検索します。

-regexp
パターンは完全な正規表現になります。

-nocase
大文字/小文字に関係なく実行します (-regexp が指定されたときのみ有効)。

-filter
式を使用してリストをフィルタします。

-of_objects
これらのピンまたはネットのセルを表示します。

-match_style デフォルト : sdc
パターン一致のスタイル。有効な値は ucf、sdc です。

-quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してセル名を一致させます。

get_clocks

現在のデザインのクロックのリストを表示します。

構文

```
get_clocks [-regexp] [-nocase] [-filter arg ] [-match_style arg ]
[-quiet] [ patterns ...]
```

表示結果

クロックのリスト

使用法

-regexp
パターンは完全な正規表現になります。

-nocase
大文字/小文字に関係なく実行します (-regexp が指定されたときのみ有効)。

-filter
式を使用してリストをフィルタします。

-match_style デフォルト : sdc
パターン一致のスタイル。有効な値は ucf、sdc です。

-quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してクロック名を一致させます。

get_debug_cores

現在のデザインの ChipScope デバッグ コアのリストを表示します。

構文

```
get_debug_cores [-filter arg ] [-regexp] [-nocase] [-quiet]
[ patterns ...]
```

表示結果

debug_core オブジェクトのリスト

使用法

-filter
式を使用してリストをフィルタします。

-regexp
パターンは正規表現になります。

-nocase
パターン一致は大文字/小文字に関係なくなります。

-quiet
コマンド エラーを無視します。

patterns
デバッグ ポートを検出するため一致させるパターンを指定します。

get_debug_ports

現在のデザインの ChipScope デバッグ ポートのリストを表示します。

構文

```
get_debug_ports [-filter arg ] [-regexp] [-nocase] [-quiet]
[ patterns ...]
```

表示結果

debug_port オブジェクトのリスト

使用法

–filter
式を使用してリストをフィルタします。

–regexp
パターンは正規表現になります。

–nocase
パターン一致は大文字/小文字に関係なくなります。

–quiet
コマンド エラーを無視します。

patterns
デバッグ ポートを検出するため一致させるパターンを指定します。

get_designs

現在のデザインに含まれるデザインのリストを表示します。

構文

```
get_designs [-regexp] [-nocase] [-filter arg ] [-quiet]
[ patterns ...]
```

表示結果

デザイン オブジェクトのリスト

使用法

-regexp
パターンは完全な正規表現になります。

-nocase
大文字/小文字に関係なく実行します (-regexp が指定されたときのみ有効)。

-filter
式を使用してリストをフィルタします。

-quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してデザイン名を一致させます。

get_files

ソース ファイルのリストを表示します。

構文

```
get_files [-regexp] [-nocase] [-filter arg ] [-of_objects args ]
[-quiet] [ patterns ...]
```

表示結果

ファイル オブジェクトのリスト

使用法

–regexp
パターンは完全な正規表現になります。

–nocase
大文字/小文字に関係ないパターン一致を実行します。

–filter
式を使用してリストをフィルタします。

–of_objects
これらのファイルセットのファイルを表示します。

–quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してファイル名を一致させます。

get_filesets

現在のプロジェクトのファイルセットのリストを表示します。

構文

```
get_filesets [-regexp] [-nocase] [-filter arg ] [-quiet]
[ patterns ...]
```

表示結果

ファイルセット オブジェクトのリスト

使用法

–regexp
パターンは完全な正規表現になります。

–nocase
大文字/小文字に関係ないパターン一致を実行します。

–filter
式を使用してリストをフィルタします。

–quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してファイルセット名を一致させます。

get_generated_clocks

現在のデザインの生成済みクロックのリストを表示します。

構文

```
get_generated_clocks [-regexp] [-nocase] [-filter arg ] [-quiet]
[ patterns ...]
```

表示結果

クロックのリスト

使用法

–regexp
パターンは完全な正規表現になります。

–nocase
大文字/小文字に関係なく実行します (–regexp が指定されたときのみ有効)。

–filter
式を使用してリストをフィルタします。

–quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対して生成されたクロック名を一致させます。

get_interfaces

現在のデザインの I/O ポート インターフェイスのリストを表示します。

構文

```
get_interfaces [-regexp] [-nocase] [-quiet] [ patterns ...]
```

表示結果

インターフェイス オブジェクトのリスト

使用法

–regexp
パターンは正規表現になります。

–nocase
パターン一致は大文字/小文字に関係なくなります。

–quiet
コマンド エラーを無視します。

patterns
I/O ポート インターフェイスを検出するために一致させるパターンを指定します。

get_iobanks

IO バンクのリストを表示します。

構文

```
get_iobanks [-filter arg ] [-regexp] [-nocase] [-quiet]
[ patterns ...]
```

表示結果

iobank オブジェクトのリスト

使用法

-filter
式を使用してリストをフィルタします。

-regexp
パターンは正規表現になります。

-nocase
パターン一致は大文字/小文字に関係なくなります。

-quiet
コマンド エラーを無視します。

patterns
IO バンクを検出するため一致させるパターンを指定します。

get_lib_cells

現在のデザインのライブラリ セルのリストを表示します。

構文

```
get_lib_cells [-regexp] [-nocase] [-filter arg ]
[-of_objects args ] [-quiet] [ patterns ...]
```

表示結果

ライブラリ セルのリスト

使用法

-regexp
パターンは完全な正規表現になります。

-nocase
大文字/小文字に関係なく実行します (-regexp が指定されたときのみ有効)。

-filter
式を使用してリストをフィルタします。

-of_objects
これらのセルのライブラリ セルを表示します。

-quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してライブラリ セル名を一致させます。

get_lib_pins

現在のデザインのライブラリ セル ピンのリストを表示します。

構文

```
get_lib_pins [-regexp] [-nocase] [-filter arg ] [-quiet]
[ patterns ...]
```

表示結果

ライブラリ セル ピンのリスト

使用法

–regexp
パターンは完全な正規表現になります。

–nocase
大文字/小文字に関係なく実行します (–regexp が指定されたときのみ有効)。

–filter
式を使用してリストをフィルタします。

–quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してライブラリ セル名を一致させます。

get_libs

現在のデザインのライブラリのリストを表示します。

構文

```
get_libs [-regexp] [-nocase] [-filter arg ] [-quiet]
[ patterns ...]
```

表示結果

ライブラリのリスト

使用法

–regexp
パターンは完全な正規表現になります。

–nocase
大文字/小文字に関係なく実行します (–regexp が指定されたときのみ有効)。

–filter
式を使用してリストをフィルタします。

–quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してライブラリ名を一致させます。

get_msg_count

メッセージ カウントを表示します。

構文

```
get_msg_count [-type arg ] [-id arg ] [-quiet]
```

表示結果

なし

使用法

-type デフォルト :ALL
「ERROR」など、表示するメッセージ タイプを指定します。

-id
common-99 などの表示するメッセージ ID を指定します。

-quiet
コマンド エラーを無視します。

get_msg_limit

メッセージ数制限を表示します。

構文

```
get_msg_limit [-type arg ] [-id arg ] [-quiet]
```

表示結果

なし

使用法

-type デフォルト :ALL

「ERROR」など、表示するメッセージ タイプを指定します。

-id

common-99 などの表示するメッセージ ID を指定します。

-quiet

コマンド エラーを無視します。

get_nets

現在のデザインのネットのリストを表示します。

構文

```
get_nets [-hsc arg ] [-hierarchical] [-regexp] [-nocase]
[-filter arg ] [-of_objects args ] [-match_style arg ] [-quiet]
[ patterns ...]
```

表示結果

ネット オブジェクトのリスト

使用法

-hsc デフォルト : /
階層区切り文字

-hierarchical
現在のインスタンスでレベルごとに検索します。

-regexp
パターンは完全な正規表現になります。

-nocase
大文字/小文字に関係なく実行します (-regexp が指定されたときのみ有効)。

-filter
式を使用してリストをフィルタします。

-of_objects
これらのピン/ポート、セルまたはクロックのネットを表示します。

-match_style デフォルト : sdc
パターン一致のスタイル。有効な値は ucf、sdc です。

-quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してネット名を一致させます。

get_package_pins

パッケージ ピンのリストを表示します。

構文

```
get_package_pins [-regexp] [-nocase] [-filter arg ] [-quiet]
[ patterns ...]
```

表示結果

パッケージ ピン オブジェクトのリスト

使用法

–regexp
パターンは完全な正規表現になります。

–nocase
大文字/小文字に関係ないパターン一致を実行します。

–filter
式を使用してリストをフィルタします。

–quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してパッケージ ピン名を一致させます。

get_param

パラメータ値を表示します。

構文

```
get_param [-quiet] name
```

表示結果

パラメータ値

使用法

-quiet
コマンド エラーを無視します。

name (必須)
パラメータ名

get_parts

ソフトウェアで使用可能なパーツのリストを表示します。

構文

```
get_parts [-regexp] [-nocase] [-filter arg ] [-quiet]
[ patterns ...]
```

表示結果

パーツ オブジェクトのリスト

使用法

–regexp
パターンは完全な正規表現になります。

–nocase
大文字/小文字に関係ないパターン一致を実行します。

–filter
式を使用してリストをフィルタします。

–quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してパーツ名を一致させます。

get_path_groups

現在のデザインのパス グループのリストを表示します。

構文

```
get_path_groups [-regexp] [-nocase] [-quiet] [ patterns ...]
```

表示結果

パス グループのリスト

使用法

-regexp
パターンは完全な正規表現になります。

-nocase
大文字/小文字に関係なく実行します (-regexp が指定されたときのみ有効)。

-quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してパス グループ名を一致させます。

get_pblocks

現在のデザインの Pblock のリストを表示します。

構文

```
get_pblocks [-regexp] [-nocase] [-quiet] [ patterns ...]
```

表示結果

Pblock オブジェクトのリスト

使用法

-regexp
パターンは正規表現になります。

-nocase
パターン一致は大文字/小文字に関係なくなります。

-quiet
コマンド エラーを無視します。

patterns
Pblock を検出するため一致するパターンを指定します。

get_pins

現在のデザインのピンのリストを表示します。

構文

```
get_pins [-hsc arg ] [-hierarchical] [-regexp] [-nocase] [-leaf]
[-filter arg ] [-of_objects args ] [-match_style arg ] [-quiet]
[ patterns ...]
```

表示結果

ピン オブジェクトのリスト

使用法

-hsc デフォルト : /
階層区切り文字

-hierarchical
現在のインスタンスでレベルごとに検索します。

-regexp
パターンは完全な正規表現になります。

-nocase
大文字/小文字に関係なく実行します (-regexp が指定されたときのみ有効)。

-leaf
-of_objects を使用した場合、ネットのリーフ/グローバル ピンを表示します。

-filter
式を使用してリストをフィルタします。

-of_objects
これらのセル、ネットまたはクロックのピンを表示します。

-match_style デフォルト : sdc
パターン一致のスタイル。有効な値は ucf、sdc です。

-quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してピン名を一致させます。

get_ports

現在のデザインのポートのリストを表示します。

構文

```
get_ports [-regexp] [-nocase] [-filter arg ] [-of_objects args ]
[-match_style arg ] [-quiet] [ patterns ...]
```

表示結果

ポート オブジェクトのリスト

使用法

-regexp
パターンは完全な正規表現になります。

-nocase
大文字/小文字に関係なく実行します (-regexp が指定されたときのみ有効)。

-filter
式を使用してリストをフィルタします。

-of_objects
これらのネット、クロックのポートを表示します。

-match_style デフォルト : sdc
パターン一致のスタイル。有効な値は ucf、sdc です。

-quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してポート名を一致させます。

get_projects

プロジェクトのリストを表示します。

構文

```
get_projects [-regexp] [-nocase] [-quiet] [ patterns ...]
```

結果

プロジェクト オブジェクトのリスト

使用法

-regexp
パターンは完全な正規表現になります。

-nocase
大文字/小文字に関係ないパターン一致を実行します。

-quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してプロジェクト名を一致させます。

get_property

オブジェクトのプロパティを表示します。

構文

```
get_property [-quiet] property_name  object
```

結果

プロパティ値

使用法

-quiet
コマンド エラーを無視します。

property_name (必須)
取り出す値のプロパティ名を指定します。

object (必須)
プロパティを示すオブジェクトを指定します。

get_reconfig_modules

現在のプロジェクトのリコンフィギャブル モジュールのリストを表示します。

構文

```
get_reconfig_modules [-filter arg ] [-of_objects args ] [-quiet]
[ patterns ...]
```

表示結果

reconfig_module オブジェクトのリスト

使用法

-filter
式を使用してリストをフィルタします。

-of_objects
これらのセルのリコンフィギャブル モジュールを表示します。

-quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してリコンフィギャブル モジュール名を一致させます。

get_runs

run (実行パターン) のリストを表示します。

構文

```
get_runs [-filter arg ] [-quiet] [ patterns ...]
```

表示結果

run オブジェクトのリスト

使用法

-filter
式を使用してリストをフィルタします。

-quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対して run 名を一致させます。

get_selected_objects

選択したオブジェクトを表示します。

構文

```
get_selected_objects [-primary] [-quiet]
```

結果

選択したオブジェクトのリスト

使用法

-primary
選択規則に従って選択されたオブジェクトは含みません。

-quiet
コマンド エラーを無視します。

get_sites

サイトのリストを表示します。

構文

```
get_sites [-regexp] [-filter arg ] [-range args ]
[-of_objects args ] [-quiet] [ patterns ...]
```

表示結果

サイト オブジェクトのリスト

使用法

–regexp
パターンは正規表現になります。

–filter
式を使用してリストをフィルタします。

–range
範囲内のサイト名を一致させます。

–of_objects
ここに渡されるオブジェクトのサイトを表示します。

–quiet
コマンド エラーを無視します。

patterns デフォルト : *
パターンに対してサイト名を一致させます。結合サイトでもパッケージ ピン名を一致させます。

group_path

コスト ファンクション計算のパスをグループ化します。

構文

```
group_path [-name arg ] [-weight arg ] [-default] [-from args ]
[-to args ] [-through args ] [-quiet]
```

表示結果

なし

使用法

-name
グループ名

-weight デフォルト : 1.0
コスト ファンクションの重さ :範囲 :0 ～ 100 (インプリメントなし)

-default
パスをデフォルト グループ (インプリメントなし) に移動します。

-from
from_list で開始されるパスを考慮します。

-to
to_list で終了するパスを考慮します。

-through
ピン、セル、またはネット (インプリメントなし) を通るパスを考慮します。

-quiet
コマンド エラーを無視します。

help

1 つまたは複数の項目に対するヘルプを表示します。

構文

```
help [-short] [-list] [-all] [-quiet] [ pattern ]
```

表示結果

なし

使用法

-short
短いコマンド ヘルプのみが表示されます。man ページのトピックは表示されません。

-list
一致するトピックがリストされます。

-all
一致するヘルプがすべて表示されます。デフォルトでは複数の一致がまとめられるようになっています。

-quiet
コマンド エラーを無視します。

pattern デフォルト : *
一致するトピックのヘルプを表示します。

highlight_objects

オブジェクトを別の色でハイライトします。

構文

```
highlight_objects [-color_index arg ] [-rgb args ] [-color arg ]
[-quiet] objects
```

結果

なし

使用法

–color_index
色インデックス

–rgb
色インデックス

–color
色の名前

–quiet
コマンド エラーを無視します。

objects (必須)
ハイライトするオブジェクト

implement_debug_core

ChipScope デバッグ コアをインプリメントします。

構文

```
implement_debug_core [-quiet] [ cores ...]
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

コア
削除する ChipScope デバッグ コアを指定します。

import_as_run

NCD とオプションの TWX を run としてインポートします。

構文

```
import_as_run [-run arg ] [-twx arg ] [-pcf arg ] [-quiet] ncd
```

表示結果

なし

使用法

-run
結果をこの run にインポートします。

-twx
インポートする TWX ファイル

-pcf
インポートする PCF ファイル

-quiet
コマンド エラーを無視します。

ncd (必須)
インポートする配線済み NCD ファイル

import_files

ファイル/ディレクトリをアクティブなファイルセットにインポートします。

構文

```
import_files [-fileset arg ] [-force] [-norecurse] [-flat]
[-relative_to arg ] [-quiet] files ...
```

表示結果

インポートされたファイル オブジェクトのリスト

使用法

–fileset
ファイルセット名を指定します。

–force
プロジェクト ディレクトリの同じ名前のファイルを上書きします。

–norecurse
ディレクトリ検索の繰り返しのデフォルト動作をディスエーブルにします。

–flat
ファイルをフラットなディレクトリ構造にインポートします。

–relative_to
指定した相対ディレクトリにファイルをインポートします。

–quiet
コマンド エラーを無視します。

files (必須)
ファイルセットにインポートするファイルの名前

import_ip

IP ファイルをインポートしてファイルセットに追加します。

構文

```
import_ip [-srcset arg ] -file arg -name arg [-quiet]
```

表示結果

追加されたファイル オブジェクトのリスト

使用法

-srcset
ソース セット名を指定します。

-file (必須)
インポートする IP ファイルの名前

-name (必須)
作成される新しい IP の名前

-quiet
コマンド エラーを無視します。

import_xise

現在のプロジェクトにザイリンクス ISE プロジェクト ファイルの設定をインポートします。

構文

```
import_xise [-quiet] file
```

表示結果

true

使用法

-quiet
コマンド エラーを無視します。

file (必須)
インポートする XISE プロジェクト ファイルの名前

import_xst

指定した XST ファイルをインポートします。

構文

```
import_xst [-quiet] file
```

結果

XST ファイルからインポートされたファイル オブジェクトのリスト

使用法

-quiet
コマンド エラーを無視します。

file (必須)
アップデートするインポート ファイル

launch_chipscope_analyzer

run に対して ChipScope Analyzer ツールを起動します。

構文

```
launch_chipscope_analyzer [-run arg ] [-csproject arg ] [-quiet]
```

表示結果

なし

使用法

-run
実行パターン

-csproject
ChipScope プロジェクト

-quiet
コマンド エラーを無視します。

launch_fpga_editor

run に対して FPGA Editor ツールを起動します。

構文

```
launch_fpga_editor [-run arg ] [-more_options arg ] [-mapped_ncd]
[-quiet]
```

表示結果

なし

使用法

-run
実行パターン

-more_options
コマンド ライン オプションをさらに指定します。

-mapped_ncd
マップされた NCD ファイルを使用します。

-quiet
コマンド エラーを無視します。

launch_impact

run に対して iMPACT コンフィギュレーション ツールを起動します。

構文

```
launch_impact [-run arg ] [-ipf arg ] [-quiet]
```

表示結果

なし

使用法

-run
実行パターン

-ipf
iMPACT プロジェクト

-quiet
コマンド エラーを無視します。

launch_isim

ISim シミュレータを使用してシミュレーションを実行します。

構文

```
launch_isim [-simset arg ] [-mode arg ] [-quiet]
```

表示結果

なし

使用法

-simset

シミュレーション ファイルセットの名前を指定します。

-mode デフォルト : behavioral

シミュレーション モード (behavioral、functional、timing)

-quiet

コマンド エラーを無視します。

launch_runs

run のセットを起動します。

構文

```
launch_runs [-jobs arg ] [-scripts_only] [-copy_sources]
[-all_placement] [-dir arg ] [-host args ] [-remote_cmd arg ]
[-email_to args ] [-email_all] [-pre_launch_script arg ]
[-post_launch_script arg ] [-force] [-quiet] runs ...
```

結果

なし

使用法

-jobs デフォルト : 1
ジョブ数

-scripts_only
スクリプトのみを生成します。

-copy_sources
指定した run ディレクトリに RTL ソースをコピーします。

-all_placement
配置が固定されているかどうかに関わらず、すべての配置を ISE にエクスポートします (デフォルトでは、固定された配置のみがエクスポートされます)。

-dir
ディレクトリを起動します。

-host
指定したジョブ数を使用して指定したリモート ホストで起動します。例 :-host {machine1 2} -host {machine2 4}

-remote_cmd デフォルト : ssh -q -o BatchMode=yes
リモート ホストへログインするコマンド

-email_to
ジョブが完了した際に通知する電子メール アドレスのリストを指定します。

-email_all
各ジョブの完了時に電子メールを送信します。

-pre_launch_script
各ジョブの起動前に実行するスクリプトを指定します。

-post_launch_script
各ジョブの完了後に実行するスクリプトを指定します。

-force
未決定の制約変更があっても、コマンドを実行します。変更は失われます (パーシャル リコンフィギュレーション デザインの場合)。

-quiet
コマンド エラーを無視します。

run (必須)
起動する run を指定します。

launch_xpa

XPower Analyzer ツールを起動します。

構文

```
launch_xpa [-run arg ] [-more_options arg ] [-mapped_ncd]
[-quiet]
```

表示結果

なし

使用法

-run
実行パターン

-more_options
コマンド ライン オプションをさらに指定します。

-mapped_ncd
マップされた NCD ファイルを使用します。

-quiet
コマンド エラーを無視します。

list_param

すべてのパラメータ名を表示します。

構文

```
list_param [-quiet]
```

表示結果

リスト

使用法

-quiet
コマンド エラーを無視します。

list_property

オブジェクトのプロパティをリストします。

構文

```
list_property [-quiet] object
```

結果

プロパティ名のリスト

使用法

-quiet
コマンド エラーを無視します。

object (必須)
プロパティを示すオブジェクトを指定します。

load_reconfig_modules

指定 run から特定のリコンフィギャブル モジュールまたはすべてのモジュールを読み込みます。

構文

```
load_reconfig_modules [-force] [-run arg ] [-quiet]
[ reconfig_modules ...]
```

結果

なし

使用法

-force
未決定の制約変更があっても、コマンドを実行します。変更は失われます。

-run
指定したディレクトリからのリコンフィギャブル モジュールの読み込みを実行します。

-quiet
コマンド エラーを無視します。

reconfig_modules
読み込むリコンフィギャブル モジュールを指定します。

make_diff_pair_ports

差動ペアを 2 ポートにします。

構文

```
make_diff_pair_ports [-quiet] ports ...
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

ports (必須)
結合するポートを指定します。

mark_objects

GUI でオブジェクトをマークします。

構文

```
mark_objects [-quiet] objects
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

objects (必須)
マークするオブジェクト

open_impl_design

インプリメント済みデザインを開きます。

構文

```
open_impl_design [-quiet] [ run ]
```

表示結果

デザイン オブジェクト

使用法

-quiet
コマンド エラーを無視します。

run
開く run を指定します。

open_io_design

IO デザインを開きます。

構文

```
open_io_design [-name arg ] [-part arg ] [-srcset arg ]
[-constrset arg ] [-quiet]
```

表示結果

デザイン オブジェクト

使用法

-name
デザイン名

-part
パーツ名

-srcset
使用するソース ファイルセット

-constrset
使用する制約ファイルセット

-quiet
コマンド エラーを無視します。

open_netlist_design

合成またはネットリスト デザインを開きます。

構文

```
open_netlist_design [-name arg ] [-part arg ] [-srcset arg ]
[-constrset arg ] [-quiet] [ run ]
```

表示結果

デザイン オブジェクト

使用法

-name
デザイン名

-part
パーツ名

-srcset
ソース ファイルセット

-constrset
制約ファイルセット

-quiet
コマンド エラーを無視します。

run
ネットリスト デザインで開く run

open_project

PlanAhead プロジェクト ファイル (.ppr) を開きます。

構文

```
open_project [-read_only] [-quiet] file
```

結果

開いたプロジェクト オブジェクト

使用法

-read_only
プロジェクトを読み出し専用モードで開きます。

-quiet
コマンド エラーを無視します。

file (必須)
読み込むプロジェクト ファイル

open_rtl_design

RTL デザインを開きます。

構文

```
open_rtl_design [-name arg ] [-part arg ] [-srcset arg ]
[-constrset arg ] [-quiet]
```

表示結果

デザイン オブジェクト

使用法

-name
デザイン名

-part
パーツ名

-srcset
run のファイルセット

-constrset
run のファイルセット

-quiet
コマンド エラーを無視します。

place_pblocks

Pblock 配置ツールを実行します。

構文

```
place_pblocks [-effort arg ] [-utilization arg ]
[-quiet] pblocks ...
```

表示結果

なし

使用法

-effort デフォルト :HIGH

Pblock ごとの配置ツールのエフォート レベル 値 :LOW、MEDIUM、HIGH

-utilization

配置ツールの Pblock ごとの使用率

-quiet

コマンド エラーを無視します。

pblocks (必須)

配置する Pblock のリスト

place_ports

ポートのセットを自動的に配置します。

構文

```
place_ports [-skip_unconnected_ports] [-check_only] [-quiet]
[ ports ...]
```

表示結果

なし

使用法

-skip_unconnected_ports
接続のないポートは配置しません。

-check_only
I/O クロック配置 DRC のみをチェックします。

-quiet
コマンド エラーを無視します。

ports
配置するポート (指定しない場合は、すべてのポートが配置されます)

promote_run

前にインプリメントしたパーティションをプロモートして、インポートおよびこの run やその他の run で再利用できるようにします。

構文

```
promote_run [-partition_names args ] [-promote_dir arg ]
[-description arg ] [-no_state_update] [-quiet] run
```

表示結果

なし

使用法

-partition_names
プロモートされるパーティションのリスト

-promote_dir
パスをプロモートします。

-description
詳細をプロモートします。

-no_state_update
プロモートされたパーティションのステートを Import に自動的にアップデートしません。

-quiet
コマンド エラーを無視します。

run (必須)
プロモートされるインプリメント済み run

read_chipscope_cdc

ChipScope Core Inserter の CDC ファイルをインポートします。

構文

```
read_chipscope_cdc [-quiet] file
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

file (必須)
ファイル名

read_csv

パッケージ ピンとポート配置情報をインポートします。

構文

```
read_csv [-quiet] file
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

file (必須)
ファイル名

read_pxml

PXML ファイルからパーティション定義をインポートします。

構文

```
read_pxml [-quiet] file
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

file (必須)
PXML ファイル名を指定します。

read_twx

Trace STA ツールからタイミング結果を読み込みます。

構文

```
read_twx [-cell arg ] [-pblock arg ] [-quiet] name  file
```

表示結果

なし

使用法

-cell
指定したセルに対してレポート ファイルの名前を解釈します。

-pblock
指定した Pblock に対してレポート ファイルの名前を解釈します。

-quiet
コマンド エラーを無視します。

name (必須)
結果のセット名

file (必須)
TRACE インポート ファイルの名前

read_ucf

ファイルから物理制約をインポートします。

構文

```
read_ucf [-cell arg ] [-quiet] file
```

表示結果

追加したファイルのリスト

使用法

-cell
このセルの制約をインポートします。

-quiet
コマンド エラーを無視します。

file (必須)
ファイル名

read_verilog

1 つまたは複数の Verilog ファイルを読み込みます。

構文

```
read_verilog [-library arg ] [-quiet] files ...
```

表示結果

追加されたファイル オブジェクトのリスト

使用法

-library デフォルト : work
ライブラリ名

-quiet
コマンド エラーを無視します。

files(必須)
Verilog ファイル名

read_vhdl

1 つまたは複数の VHDL ファイルを読み込みます。

構文

```
read_vhdl [-library arg ] [-quiet] files
```

結果

追加されたファイル オブジェクトのリスト

使用法

-library デフォルト : work
vhdl ライブラリ

-quiet
コマンド エラーを無視します。

files (必須)
ファイル名

read_xdl

ファイルから配置情報をインポートします。

構文

```
read_xdl [-pblock arg ] [-cell arg ] [-quiet] file
```

表示結果

なし

使用法

-pblock
この Pblock の配置をインポートします。

-cell
このセルの配置をインポートします。

-quiet
コマンド エラーを無視します。

file (必須)
配置ファイル名を指定します。

redo

前のコマンドを再実行します。

構文

```
redo [-list] [-quiet]
```

表示結果

再実行可能なタスクのリスト (-list を使用した場合)

使用法

-list
再実行可能なタスクのリストが表示されます。

-quiet
コマンド エラーを無視します。

refresh_design

現在のデザインを最新情報に更新します。

構文

```
refresh_design [-part arg ] [-quiet]
```

結果

なし

使用法

-part
このパーツに対してデザインを更新します。

-quiet
コマンド エラーを無視します。

reimport_files

期限切れのファイルがあれば、それをインポートし直します。

構文

```
reimport_files [-force] [-quiet] [ files ...]
```

表示結果

インポートされたファイル オブジェクトのリスト

使用法

-force

ローカル ファイルの方が新しくてもインポートをし直します。

-quiet

コマンド エラーを無視します。

files

インポートし直すファイルのリスト。ファイルを指定しない場合は、期限切れプロジェクトのファイルすべてがインポートしなおされます。

remove_cells_from_pblock

Pblock からセルを削除します。

構文

```
remove_cells_from_pblock [-quiet] pblock  cells ...
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

pblock (必須)
セルを削除する Pblock を指定します。

cells (必須)
追加するセルを指定します。

remove_disable_timing

タイミング アークをイネーブルにします。

構文

```
remove_disable_timing [-from arg ] [-to arg ] [-quiet]
[ objects ...]
```

結果

なし

使用法

-from
指定したセルのピンからイネーブルになります。

-to
指定したセルのピンまでイネーブルになります。

-quiet
コマンド エラーを無視します。

objects
セルまたはピン、ポート、ライブラリ セル、ライブラリ ピンのリスト

remove_files

ファイルセットからファイルまたはディレクトリを削除します。

構文

```
remove_files [-fileset arg ] [-quiet] [ files ...]
```

表示結果

削除されたファイルのリスト

使用法

–fileset
ファイルセット名を指定します。

–quiet
コマンド エラーを無視します。

files
削除するファイルおよびディレクトリ名を指定します。

reorder_files

アクティブなファイルセットでソース ファイルの順序を変更します。

構文

```
reorder_files [-fileset arg ] [-before arg ] [-after arg ]
[-front] [-back] [-auto] [-disable_unused] [-quiet] files ...
```

表示結果

なし

使用法

-fileset
順番を並び替えるファイルセットを指定します。

-before
このファイルの前にリストされたファイルを移動します。

-after
このファイルの後にリストされたファイルを移動します。

-front
リストされたファイルを前に移動します (デフォルト)。

-back
リストされたファイルを後に移動します。

-auto
指定したファイルセットを自動的に並び替えます。

-disable_unused
TOP デザイン ユニットに関連していないファイルをすべてディスエーブルにします。

-quiet
コマンド エラーを無視します。

files(必須)
移動するファイルを指定します。

report_clock_interaction

内部クロック タイミング パスとクロックなしのレジスタについてレポートします。

構文

```
report_clock_interaction [-delay_type arg ] [-asynchronous]
[-significant_digits arg ] [-results arg ] [-quiet]
```

結果

なし

使用法

-delay_type デフォルト : max
パス遅延のタイプ :値 : max、min、min_max

-asynchronous
非同期クロック間のパスのみをレポートします。

-significant_digits デフォルト :2
表示する桁数 :範囲 :0 ～ 13

-results
この名前で GUI パネルへ結果を出力します。

-quiet
コマンド エラーを無視します。

report_constraint

デザインの制約に関連する情報を表示します。

構文

```
report_constraint [-all_violators] [-verbose] [-path_type arg ]
[-max_delay] [-min_delay] [-recovery] [-removal]
[-clock_gating_setup] [-clock_gating_hold] [-max_transition]
[-min_transition] [-min_pulse_width] [-min_period] [-max_skew]
[-significant_digits arg ] [-nosplit] [-quiet]
```

結果

なし

使用法

-all_violators
すべての制約違反を表示します。

-verbose
詳細情報を表示します。

-path_type
パス レポートのフォーマット。使用できる値は、end、slack_only、command、summary、full、full_clock、full_clock_expanded、short (デフォルト : verbose==true の場合は 'full' else 'slack_only') です。

-max_delay
max_delay & セットアップのみ

-min_delay
min_delay & ホールドのみ

-recovery
非同期リカバリのみ

-removal
非同期削除のみ

-clock_gating_setup
クロック ゲートのセットアップのみ

-clock_gating_hold
クロック ゲートのホールドのみ

-max_transition
max_transition のみ

-min_transition
min_transition のみ

-min_pulse_width
min_pulse_width のみ

-min_period
min_period のみ

-max_skew
max_skew のみ

-significant_digits デフォルト : 2
表示する桁数。範囲は、0 ～ 13 です。

-nosplit
列がはみ出しても行を分割しません。

-quiet
コマンド エラーを無視します。

report_control_sets

デザイン特有の制御セットについてレポートします。

構文

```
report_control_sets [-detail] [-quiet] [ file ]
```

結果

レポート

使用法

-detail
各制御セットの詳細をレポートします。

-quiet
コマンド エラーを無視します。

file
出力ファイル

report_debug_core

ChipScope デバッグ コアの詳細をレポートします。

構文

```
report_debug_core [-quiet] file
```

結果

なし

使用法

-quiet
コマンド エラーを無視します。

file (必須)
ファイル名

report_delay_calculation

セルまたはネット アークの遅延計算の詳細をレポートします。

構文

```
report_delay_calculation [-from_pin args ] [-to_pin args ] [-min]
[-max] [-significant_digits arg ] [-quiet]
```

結果

なし

使用法

-from_pin
レポート範囲の始まりのピンを指定します。

-to_pin
レポート範囲の終わりのピンを指定します。

-min
最小遅延計算をレポートします。

-max
最大遅延計算をレポートします (デフォルト)。

-significant_digits デフォルト : 2
表示する桁数 :範囲 :0 ～ 13

-quiet
コマンド エラーを無視します。

report_disable_timing

ディスエーブルにしたタイミング アークをレポートします。

構文

```
report_disable_timing [-quiet]
```

結果

なし

使用法

-quiet
コマンド エラーを無視します。

report_drc

DRC を実行します。

構文

```
report_drc [-rules args ] [-file arg ] [-quiet] [ name ]
```

表示結果

なし

使用法

–rules
DRC 規則

–file
DRC レポート ファイル

–quiet
コマンド エラーを無視します。

name
GUI で表示される DRC 結果名

report_io

デバイスのすべての IO サイトに関する情報を表示します。

構文

```
report_io [-quiet] [ file ]
```

結果

レポート

使用法

–quiet
コマンド エラーを無視します。

file
出力ファイル

report_min_pulse_width

最小パルス幅チェックをレポートします。

構文

```
report_min_pulse_width [-path_type arg ]
[-significant_digits arg ] [-input_pins] [-verbose]
[-nosplit] [-quiet] [ objects ...]
```

結果

なし

使用法

-path_type デフォルト : full

パス レポートのフォーマット :値 : end、slack_only、command、summary、full、full_clock、full_clock_expanded、short

-significant_digits デフォルト : 2

表示する桁数 :範囲 :0 ～ 13

-input_pins

パスの入力ピンを表示します。

-verbose

詳細情報を表示します。

-nosplit

列がはみ出しても行を分割しません。

-quiet

コマンド エラーを無視します。

objects

最小パルス幅をチェックするオブジェクトのリスト

report_param

すべてのパラメータに関する情報を表示します。

構文

```
report_param [-quiet]
```

結果

パラメータ レポート

使用法

-quiet
コマンド エラーを無視します。

report_power

電力概算を実行し、レポートを表示します。

構文

```
report_power [-verbose] [-hierarchy] [-levels arg ] [-file arg ]
[-format arg ] [-xpe arg ] [-quiet]
```

結果

なし

使用法

-verbose
詳細なリソース概算をレポートします。

-hierarchy
消費電力を階層別にレポートします。

-level デフォルト : 1
レポートされる階層レベル数 :値 = 1

-file
ファイルへ結果を出力します。

-format デフォルト : table
消費電力レポートのフォーマット : table、xml

-xpe
XPE にインポートできるように結果を XML ファイルに出力します。

-quiet
コマンド エラーを無視します。

report_property

オブジェクトのプロパティをレポートします。

構文

```
report_property [-all] [-quiet] object
```

結果

プロパティ レポート

使用法

-all
設定していないものも含めてオブジェクトのプロパティすべてをレポートします。

-quiet
コマンド エラーを無視します。

object (必須)
プロパティを示すオブジェクトを指定します。

report_resources

リソース概算を実行し、レポートを表示します。

構文

```
report_resources [-verbose] [-hierarchy] [-levels arg ]
[-file arg ] [-format arg ] [-quiet]
```

結果

なし

使用法

-verbose
詳細なリソース概算をレポートします。

-hierarchy
階層ごとに概算をレポートします。

-level デフォルト : 1
レポートされる階層レベル数 :値 = 1

-file
ファイルへ結果を出力します。

-format デフォルト : table
リソース概算レポートのフォーマット : table、xml

-quiet
コマンド エラーを無視します。

report_ssn

現在のパッケージおよびピン配置で SSN 解析を実行します。

構文

```
report_ssn [-file arg ] [-quiet] name
```

結果

SSN レポート

使用法

-file
出力ファイル名

-quiet
コマンド エラーを無視します。

name (必須)
結果のセット名

report_sso

現在のパッケージおよびピン配置で WASSO 解析を実行します。

構文

```
report_sso [-file arg ] [-board_thickness arg ]
[-via_diameter arg ] [-pad_to_via_breakout_length arg ]
[-breakout_width arg ] [-other_pcb_inductance arg ]
[-socket_inductance arg ] [-ground_bounce arg ]
[-output_cap arg ] [-quiet] name
```

表示結果

SSO レポート

使用法

–file
出力ファイル名

–board_thickness
PCB の厚さをミリ単位で表示します。

–via_diameter
ミリ単位のビア直径を終了します。

–pad_to_via_breakout_length
ミリ単位でパッドからビア ブレークアウトの長さを指定します。

–breakout_width
ミリ単位でブレークアウト幅を指定します。

–other_pcb_inductance
ナノヘンリー単位でほかの PCB 寄生インダクタンスを指定します。

–socket_inductance
ナノヘンリー単位でソケット インダクタンスを指定します。

–ground_bounce
ミリボルト単位で最大グランド バウンスを指定します。

–output_cap
ピコファラッド単位で各出力ドライバのキャパシタンスを指定します。

–quiet
コマンド エラーを無視します。

name (必須)
結果のセット名

report_stats

統計をレポートします。

構文

```
report_stats [-cell arg ] [-pblock arg ] [-clock_region arg ]
[-format arg ] [-level arg ] [-tables args ] [-quiet] file
```

結果

レポート

使用法

-cell
指定したセルの統計を記述します。

-pblock
指定した Pblock の統計を記述します。

-clock_region
指定したクロック領域の統計を記述します。

-format デフォルト : TABLE
レポート フォーマットを指定します。使用できる値は、TABLE、CSV、XML です。

-level デフォルト : 1
レポート レベル (-cell または -pblock と共に使用)

-tables
表のタイプを指定します。使用できる値は、rtlMacro、rtlPrimitive、rtlHierarchy、rtlMemory、rtlPower、rtlPower2、primitive、netBoundary、carryChain、physicalResource、io、RPM、clock、PRModule、PRModule、pblockOverlap、ioBank です。

-quiet
コマンド エラーを無視します。

file (必須)
出力ファイル名

report_timing

タイミング パスをレポートします。

構文

```
report_timing [-from args ] [-rise_from args ] [-fall_from args ]
[-to args ] [-rise_to args ] [-fall_to args ] [-through args ]
[-rise_through args ] [-fall_through args ] [-delay_type arg ]
[-max_paths arg ] [-nworst arg ] [-path_type arg ] [-input_pins]
[-nets] [-slack_less_than arg ] [-slack_greater_than arg ]
[-group args ] [-sort_by arg ] [-significant_digits arg ]
[-no_report_unconstrained] [-file arg ] [-append] [-console]
[-match_style arg ] [-results arg ] [-quiet]
```

結果

なし

使用法

-from
レポート範囲の始まり (ピン、ポート、クロック) を指定します。

-rise_from
ピン、ポート、またはクロックからの立ち上がりを指定します。

-fall_from
ピン、ポート、またはクロックからの立ち下がりを指定します。

-to
レポート範囲の終わり (ピン、ポート、クロック) を指定します。

-rise_to
立ち上がりの終わるピン、ポート、またはクロックを指定します。

-fall_to
立ち下がりの終わるピン、ポート、またはクロックを指定します。

-through
レポート範囲の通過ポイント (ピン、ポート、クロック) を指定します。

-rise_through
立ち上がりの途中のピン、ポート、またはクロックを指定します。

-fall_through
立ち下がりの途中のピン、ポート、クロックを指定します。

-delay_type デフォルト : max
パス遅延のタイプ :値 : max、min、min_max

-max_paths デフォルト : 1
出力されるパス グループごとのパスの最大数。値 = 1 です。

-nworst デフォルト : 1
エンドポイントまでの N ワーストパスをリストします。値 = 1 です。

-path_type デフォルト : full_clock_expanded
パス レポートのフォーマット。使用できる値は、end、summary、short、full、full_clock、full_clock_expanded です。

-input_pins
パスの入力ピンを表示します。

-nets
ネット名をリストします。

-slack_less_than デフォルト : 1e+30
これよりも少ないスラックのパスを表示します。

-slack_greater_than デフォルト : -1e+30
これよりも多いスラックのパスを表示します。

-group
このグループのパスに対するレポートを制限します。

-sort_by デフォルト : slack
パスの並び替え順。使用できる値は、group、slack です。

-significant_digits デフォルト :3
表示する桁数。範囲は、0 ～ 13 です。

-no_report_unconstrained
制約の付いていないパスをレポートしません。

-file
ファイルへ結果を出力します。

-append
ファイルに結果が追加されます。上書きはされません。

-console
コンソールに結果を表示します。

-match_style デフォルト : ucf
パターン一致のスタイル。有効な値は ucf、sdc です。

-results
この名前で GUI パネルへ結果を出力します。

-quiet
コマンド エラーを無視します。

report_transformed_primitives

Unisim プリミティブ変換の詳細をレポートします。

構文

```
report_transformed_primitives [-quiet] file
```

結果

なし

使用法

-quiet
コマンド エラーを無視します。

file (必須)
レポート ファイル名を指定します。

report_ucf_timing

タイミング パスをレポートします。

構文

```
report_ucf_timing [-name arg ] [-speed arg ] [-from args ]
[-to args ] [-thru args ] [-min_max arg ] [-transition arg ]
[-sort_by arg ] [-interconnect_type arg ] [-maxpaths arg ]
[-nworst arg ] [-required_times] [-quiet]
```

結果

なし

使用法

-name デフォルト : results_1
この名前で GUI パネルへ結果を出力します。

-speed
スピード グレード名を指定します。

-from
開始ピンまたはインスタンス名でフィルタします。

-to
終了ピンまたはインスタンス名でフィルタします。

-thru
中間ピンまたはインスタンス名でフィルタします。

-min_max
max (セットアップ) または min (ホールド) タイミング解析を実行します。

-transition
立ち上がりまたは立ち下がり遷移をレポートします。

-sort_by
並び替えの基準を指定します。

-interconnect_type
インターコネクトの遅延タイプ

-maxpaths デフォルト : 1
多くのパスのレポートをこの数に制限します。

-nworst デフォルト : 1
エンドポイントごとに多くのパスのレポートをこの数に制限します

-required_times
すべての必要な時間を計算します。

-quiet
コマンド エラーを無視します。

report_utilization

デバイス使用率を計算し、レポートを表示します。

構文

```
report_utilization [-append] [-of_areagroup arg ] [-of_hier arg ]
[-quiet] [ file ]
```

結果

レポート

使用法

-append
ファイルに結果が追加されます。上書きはされません。

-of_areagroup
指定したエリア グループ ブロックの使用率がレポートされます。

-of_hier
指定したデザイン階層の使用率がレポートされます。

-quiet
コマンド エラーを無視します。

file
出力ファイル

reset_drc

DRC を削除します。

構文

```
reset_drc [-quiet] [ name ]
```

結果

なし

使用法

-quiet
コマンド エラーを無視します。

name
DRC 結果名

reset_ip

コンフィギャブル IP をリセットします。

構文

```
reset_ip [-srcset arg ] -files args [-quiet]
```

結果

リセットされたファイルのリスト

使用法

–srcset
ソース セット名を指定します。

–files (必須)
リセットされる IP ソース ファイルを指定します。

–quiet
コマンド エラーを無視します。

reset_msg_limit

メッセージ数制限をリセットします。

構文

```
reset_msg_limit [-type arg ] [-id arg ] [-quiet]
```

結果

なし

使用法

-type デフォルト :ALL
「ERROR」など、リセットするメッセージ タイプを指定します。

-id
common-99 などのリセットするメッセージ ID を指定します。

-quiet
コマンド エラーを無視します。

reset_path

1 サイクルのビヘイビアに指定パスをリセットします。

構文

```
reset_path [-setup] [-hold] [-rise] [-fall] [-from args ]
[-rise_from args ] [-fall_from args ] [-to args ]
[-rise_to args ] [-fall_to args ] [-through args ]
[-rise_through args ] [-fall_through args ] [-quiet]
```

結果

なし

使用法

-setup
パスのセットアップ タイミング解析をリセットします。

-hold
パスのホールド タイミング解析をリセットします。

-rise
定義済みパスの立ち上がり遅延のみをリセットします。

-fall
定義済みパスの立ち下がり遅延のみをリセットします。

-from
パスの開始点またはクロックのリストを指定します。

-rise_from
開始点またはクロックのリストから立ち上がるパスに適用されます。

-fall_from
開始点またはクロックのリストから立ち下がるパスに適用されます。

-to
パスの終了点またはクロックのリストを指定します。

-rise_to
終了点またはクロックのリストで立ち上がるパスに適用されます。

-fall_to
終了点またはクロックのリストで立ち下がるパスに適用されます。

-through
スルー ピン、セル、またはネットのリストを指定します。

-rise_through
立ち上がりスルー ピン、セル、またはネットに適用されます。

-fall_through
立ち下がりスルー ピン、セル、またはネットに適用されます。

-quiet
コマンド エラーを無視します。

reset_run

既存の run をリセットします。

構文

```
reset_run [-noclean_dir] [-quiet] run
```

結果

なし

使用法

-noclean_dir
ディスクからすべての出力ファイルおよびディレクトリを削除しません。

-quiet
コマンド エラーを無視します。

run (必須)
変更する run を指定します。

reset_ssn

メモリから SSN 結果を一掃します。

構文

```
reset_ssn [-quiet] name
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

name (必須)
結果のセット名

reset_sso

メモリから WASSO 結果を一掃します。

構文

```
reset_sso [-quiet] name
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

name (必須)
結果のセット名

reset_timing

現在のデザインのタイミング情報をリセットします。

構文

```
reset_timing [-quiet]
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

reset_ucf

ファイルから読み込んだフロアプラン制約を一掃します。

構文

```
reset_ucf [-quiet] file
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

file (必須)
ファイル名

resize_pblock

UCF の範囲を移動、サイズ変更、追加、削除します。

構文

```
resize_pblock [-add args ] [-remove args ] [-from args ]
[-to args ] [-replace] [-locs arg ] [-quiet] pblock
```

表示結果

なし

使用法

-add
サイト範囲を追加します。

-remove
サイト範囲を削除します。

-from
移動するサイト範囲を指定します。

-to
サイト範囲の移動先を指定します。

-replace
すべての既存の範囲を削除します。

-locs デフォルト : keep_all
LOC の処理方法を指定します。

-quiet
コマンド エラーを無視します。

pblock (必須)
サイズを変更する Pblock を指定します。

save_design

現在のデザインを保存します。

構文

```
save_design [-force] [-quiet]
```

表示結果

なし

使用法

-force
デザインを強制的に保存します。必要であれば、ターゲットの UCF に上書きします。

-quiet
コマンド エラーを無視します。

save_design_as

現在のデザインを新しい制約セットとして保存します。

構文

```
save_design_as [-dir arg ] [-quiet] name
```

表示結果

なし

使用法

-dir
ローカル ファイルすべてをこのディレクトリに保存します。

-quiet
コマンド エラーを無視します。

name (必須)
新しい制約ファイルセットの名前を指定します。

save_project_as

現在のプロジェクトを新しい名前で保存します。

構文

```
save_project_as [-force] [-quiet] name  dir
```

結果

保存したプロジェクト オブジェクト

使用法

–force
既存のプロジェクト ディレクトリを上書きします。

–quiet
コマンド エラーを無視します。

name (必須)
保存するプロジェクトの新しい名前

dir (必須)
プロジェクト ファイルが保存されるディレクトリ

select_objects

GUI でオブジェクトを選択します。

構文

```
select_objects [-quiet] objects
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

objects (必須)
選択するオブジェクト

set_case_analysis

入力を 1、0、rising、falling のいずれかに指定します。

構文

```
set_case_analysis [-quiet] [ value ] objects ...
```

表示結果

なし

使用法

–quiet
コマンド エラーを無視します。

value デフォルト : 1
ピンの論理値。使用できる値は、0、1、rising、falling、zero、one、rise、fall です。

objects (必須)
ポートまたはピンのリスト

set_clock_gating_check

クロック ゲート付きチェックをキャプチャします。

構文

```
set_clock_gating_check [-setup arg ] [-hold arg ] [-rise] [-fall]
[-high] [-low] [-quiet] [ objects ...]
```

表示結果

なし

使用法

-setup デフォルト : 0
クロック ゲート付きセットアップ タイム値 = 0

-hold デフォルト : 0
クロック ゲート付きホールド タイム値 = 0

-rise
定義済みチェックのの立ち上がり値のみを指定します。

-fall
定義済みチェックのの立ち下がり値のみを指定します。

-high
クロック波形の High に対するチェックを指定します。

-low
クロック波形の Low に対するチェックを指定します。

-quiet
コマンド エラーを無視します。

objects
クロック、ポート、ピン、またはセルのリスト

set_clock_groups

専用または非同期のクロック グループを設定します。

構文

```
set_clock_groups -name arg [-logically_exclusive]
[-physically_exclusive] [-asynchronous] [-allow_paths]
[-group args ] [-quiet]
```

表示結果

なし

使用法

-name (必須)
クロック グループの名前を指定します。

-logically_exclusive
論理的に排他的なクロック グループを指定します。

-physically_exclusive
物理的に排他的なクロック グループを指定します。

-asynchronous
非同期クロック グループを指定します。

-allow_paths
制約どおりにパスを維持します。

-group
クロック リストを指定します。

-quiet
コマンド エラーを無視します。

set_clock_latency

実際または予測済みのクロック レイテンシをキャプチャします。

構文

```
set_clock_latency [-clock args ] [-rise] [-fall] [-min] [-max]
[-source] [-late] [-early] [-quiet] delay  objects ...
```

表示結果

なし

使用法

-clock
関連するクロックのリスを指定します。

-rise
クロックの立ち上がりレイテンシを指定します。

-fall
クロックの立ち下がりレイテンシを指定します。

-min
クロックの立ち上がりおよび立ち下がりの最小状態のレイテンシを指定します。

-max
クロックの立ち上がりおよび立ち下がりの最大状態のレイテンシを指定します。

-source
クロックの立ち上がりおよび立ち下がりのソース レイテンシを指定します。

-late
クロックの立ち上がりおよび立ち下がりの後期ソース レイテンシを指定します。

-early
クロックの立ち上がりおよび立ち下がりの早期ソース レイテンシを指定します。

-quiet
コマンド エラーを無視します。

delay (必須)
クロック レイテンシ

objects (必須)
クロック、ポート、またはピンのリスト

set_clock_sense

ポートまたはピンのクロック エッジを設定します。

構文

```
set_clock_sense [-positive] [-negative] [-stop_propagation]
[-pulse arg ] [-clocks args ] [-quiet] [ pins ...]
```

表示結果

なし

使用法

–positive
正ユネイト (反転なし) のクロック エッジを指定します。

–negative
負ユネイト (反転あり) のクロック エッジを指定します。

–stop_propagation
指定したピンからのクロック伝搬を停止します。

–pulse
パルス クロック エッジを指定します。

–clocks
クロック リストを指定します。

–quiet
コマンド エラーを無視します。

pins
ポート/ピンのリストを指定します。

set_clock_transition

予測済みクロック遷移をキャプチャします。

構文

```
set_clock_transition [-rise] [-fall] [-min] [-max]
[-quiet] transition clocks ...
```

結果

なし

使用法

-rise
クロックの立ち上がり遷移を指定します。

-fall
クロックの立ち下がり遷移を指定します。

-min
クロックの立ち上がりおよび立ち下がりの最小遷移を指定します。

-max
クロックの立ち上がりおよび立ち下がりの最大遷移を指定します。

-quiet
コマンド エラーを無視します。

transition (必須)
クロック ピンの遷移時間

clocks (必須)
クロックのリスト

set_clock_uncertainty

False パスを定義します。

構文

```
set_clock_uncertainty [-setup] [-hold]
[-from args ] [-rise_from args ] [-fall_from args ]
[-to args ] [-rise_to args ] [-fall_to args ]
[-quiet] uncertainity [ objects ...]
```

結果

なし

使用法

-setup
セットアップ チェックのクロック誤差を指定します。

-hold
ホールド チェックのクロック誤差を指定します。

-from
クロック間誤差のソース クロックを指定します。

-rise_from
立ち上がりエッジでクロック間誤差のソース クロックを指定します。

-fall_from
立ち下がりエッジでクロック間誤差のソース クロックを指定します。

-to
クロック間誤差のデスティネーション クロックを指定します。

-rise_to
立ち上がりエッジでクロック間誤差のデスティネーション クロックを指定します。

-fall_to
立ち下がりエッジでクロック間誤差のデスティネーション クロックを指定します。

-quiet
コマンド エラーを無視します。

uncertainity (必須)
クロック ネットワークの誤差

objects
クロック、セル、ポート、またはピンのリスト

set_data_check

データ間チェックを作成します。

構文

```
set_data_check [-from args ] [-to args ] [-rise_from args ]
[-fall_from args ] [-rise_to args ] [-fall_to args ] [-setup]
[-hold] [-clock args ] [-quiet] value
```

結果

なし

使用法

-from
データ チェックを開始するデータのピン/ポート

-to
データ チェックを終了するデータのピン/ポート

-rise_from
データ チェックを開始するデータのピン/ポートの立ち上がり

-fall_from
データ チェックを開始するデータのピン/ポートの立ち下がり

-rise_to
データ チェックを終了するデータのピン/ポートの立ち上がり

-fall_to
データ チェックを終了するデータのピン/ポートの立ち下がり

-setup
データ チェックのセットアップ タイムを指定します。

-hold
データ チェックのホールド タイムを指定します。

-clock
チェックの関連ピン/ポートのクロック ドメインを指定します。

-quiet
コマンド エラーを無視します。

value (必須)
定義したチェックのセットアップまたはホールド タイム

set_delay_model

タイミング遅延モデル

構文

```
set_delay_model [-interconnect arg ] [-quiet]
```

表示結果

なし

使用法

-interconnect デフォルト : estimated

タイミング解析に使用されるインターコネクト遅延モデル値 : estimated、none

-quiet

コマンド エラーを無視します。

set_disable_timing

タイミング アークをディスエーブルにします。

構文

```
set_disable_timing [-from arg ] [-to arg ] [-quiet] objects ...
```

表示結果

なし

使用法

–from
指定したセルのピンからイネーブルになります。

–to
指定したセルのピンまでイネーブルになります。

–quiet
コマンド エラーを無視します。

objects (必須)
セルまたはピン、ポート、ライブラリ セル、ライブラリ ピンのリスト

set_false_path

False パスを定義します。

構文

```
set_false_path [-setup] [-hold] [-rise] [-fall] [-reset_path]
[-from args ] [-rise_from args ] [-fall_from args ]
[-to args ] [-rise_to args ] [-fall_to args ] [-through args ]
[-rise_through args ] [-fall_through args ] [-quiet]
```

結果

なし

使用法

-setup
パスのセットアップ タイミング解析を削除します。

-hold
パスのホールド タイミング解析を削除します。

-rise
定義済みパスの立ち上がり遅延のみを削除します。

-fall
定義済みパスの立ち下がり遅延のみを削除します。

-reset_path
False パスを設定する前にこのパスをリセットします。

-from
パスの開始点またはクロックのリストを指定します。

-rise_from
開始点またはクロックのリストから立ち上がるパスに適用されます。

-fall_from
開始点またはクロックのリストから立ち下がるパスに適用されます。

-to
パスの終了点またはクロックのリストを指定します。

-rise_to
終了点またはクロックのリストで立ち上がるパスに適用されます。

-fall_to
終了点またはクロックのリストで立ち下がるパスに適用されます。

-through
スルー ピン、セル、またはネットのリストを指定します。

-rise_through
立ち上がりスルー ピン、セル、またはネットに適用されます。

-fall_through
立ち下がりスルー ピン、セル、またはネットに適用されます。

-quiet
コマンド エラーを無視します。

set_hierarchy_separator

階層区切り文字を設定します。

構文

```
set_hierarchy_separator [-quiet] [ separator ]
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

separator デフォルト : /
階層区切り文字

set_ideal_latency

理想的なレイテンシを指定します。

構文

```
set_ideal_latency [-rise] [-fall] [-min] [-max]
[-quiet] value  objects ...
```

表示結果

なし

使用法

-rise
理想的な立ち上がりレイテンシを指定します。

-fall
理想的な立ち下がりレイテンシを指定します。

-min
理想的な最小レイテンシを指定します。

-max
理想的な最大レイテンシを指定します。

-quiet
コマンド エラーを無視します。

value (必須)
理想的なレイテンシ値。値 = 0 です。

objects (必須)
ポートまたはピンのリスト

set_ideal_network

理想的なネットワークを指定します。

構文

```
set_ideal_network [-no_propagate] [-quiet] objects ...
```

結果

なし

使用法

-no_propagate
ロジック ゲートを介して伝播しないようにします。

-quiet
コマンド エラーを無視します。

objects (必須)
ピン、ポート、またはネットのリスト

set_input_delay

ポートまたはピンの入力遅延を設定します。

構文

```
set_input_delay [-clock args ] [-reference_pin args ]
[-clock_fall] [-level_sensitive] [-rise] [-fall]
[-max] [-min] [-add_delay] [-network_latency_included]
[-source_latency_included] [-quiet] [ delay ] objects ...
```

表示結果

なし

使用法

-clock
相対クロック

-reference_pin
相対ピンまたはポート

-clock_fall
遅延はクロックの立ち下がりエッジに対応します。

-level_sensitive
遅延はレベル センシティブなラッチからになります。

-rise
立ち上がり遅延を指定します。

-fall
立ち下がり遅延を指定します。

-max
最大遅延を指定します。

-min
最小遅延を指定します。

-add_delay
既存の入力遅延を削除しません。

-network_latency_included
既に含まれているクロックのネットワーク レイテンシを指定します。

-source_latency_included
既に含まれているクロックのソース レイテンシを指定します。

-quiet
コマンド エラーを無視します。

delay デフォルト : 1.0
パス遅延

objects (必須)
ポート/ピンのリストを指定します。

set_input_jitter

クロック オブジェクトの入力ジッタを設定します。

構文

```
set_input_jitter [-quiet] clock_name  input_jitter
```

表示結果

クロック名

使用法

–quiet
コマンド エラーを無視します。

clock_name (必須)
クロック名を指定します。

input_jitter (必須)
入力ジッタ。値 = 0 です。

set_load

ポートおよびネットのキャパシタンスを設定します。

構文

```
set_load [-rise] [-fall] [-max] [-min] [-subtract_pin_load]
[-pin_load] [-wire_load] [-quiet] capacitance  objects ...
```

表示結果

なし

使用法

-rise
Specify the rise capacitance value (for ports only)

-fall
立ち下がりキャパシタンス値を指定します (ポートのみ)。

-max
最大キャパシタンス値を指定します。

-min
最小キャパシタンス値を指定します。

-subtract_pin_load
値からピン キャパシタンスを引算します (ネットのみ)。

-pin_load
ピン キャパシタンス (ポートのみ)

-wire_load
ワイヤー キャパシタンス (ポートのみ)

-quiet
コマンド エラーを無視します。

capacitance (必須)
キャパシタンス値

objects (必須)
ポートまたはネットのリスト

set_logic_dc

ポート/ピンのロジック DC を設定します。

構文

```
set_logic_dc [-quiet] objects ...
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

objects (必須)
ポートまたはピンのリスト

set_logic_one

ポート/ピンのロジック 1 を設定します。

構文

```
set_logic_one [-quiet] objects ...
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

objects (必須)
ポートまたはピンのリスト

set_logic_zero

ポート/ピンのロジック 0 を設定します。

構文

```
set_logic_zero [-quiet] objects ...
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

objects (必須)
ポートまたはピンのリスト

set_max_delay

タイミング パスの最大遅延を指定します。

構文

```
set_max_delay [-rise] [-fall] [-reset_path] [-from args ]
[-rise_from args ] [-fall_from args ] [-to args ]
[-rise_to args ] [-fall_to args ] [-through args ]
[-rise_through args ] [-fall_through args ] [-quiet] delay
```

表示結果

なし

使用法

-rise
遅延値が立ち上がりパス遅延に適用されます。

-fall
遅延値が立ち下がりパス遅延に適用されます。

-reset_path
最大遅延を設定する前にこのパスをリセットします。

-from
パスの開始点またはクロックのリストを指定します。

-rise_from
開始点またはクロックのリストから立ち上がるパスに適用されます。

-fall_from
開始点またはクロックのリストから立ち下がるパスに適用されます。

-to
パスの終了点またはクロックのリストを指定します。

-rise_to
終了点またはクロックのリストで立ち上がるパスに適用されます。

-fall_to
終了点またはクロックのリストで立ち下がるパスに適用されます。

-through
スルー ピン、セル、またはネットのリストを指定します。

-rise_through
立ち上がりスルー ピン、セル、またはネットに適用されます。

-fall_through
立ち下がりスルー ピン、セル、またはネットに適用されます。

-quiet
コマンド エラーを無視します。

delay (必須)
遅延値

set_max_fanout

ポートまたはセルの最大ファンアウトを設定します。

構文

```
set_max_fanout [-quiet] [ fanout ] [ ports ...]
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

fanout デフォルト : 0.0
最大ファンアウト

ポート
ポートのリスト

set_max_time_borrow

ラッチ用に借りることのできる時間を制限します。

構文

```
set_max_time_borrow [-quiet] delay  objects ...
```

表示結果

なし

使用法

–quiet
コマンド エラーを無視します。

delay (必須)
借りる制限値。値 = 0 です。

objects (必須)
クロック、セル、データ ピン、またはクロック ピンのリスト

set_min_delay

タイミング パスの最小遅延を指定します。

構文

```
set_min_delay [-rise] [-fall] [-reset_path] [-from args ]
[-rise_from args ] [-fall_from args ] [-to args ]
[-rise_to args ] [-fall_to args ] [-through args ]
[-rise_through args ] [-fall_through args ] [-quiet] delay
```

結果

なし

使用法

-rise
遅延値が立ち上がりパス遅延に適用されます。

-fall
遅延値が立ち下がりパス遅延に適用されます。

-reset_path
最小遅延を設定する前にこのパスをリセットします。

-from
パスの開始点またはクロックのリストを指定します。

-rise_from
開始点またはクロックのリストから立ち上がるパスに適用されます。

-fall_from
開始点またはクロックのリストから立ち下がるパスに適用されます。

-to
パスの終了点またはクロックのリストを指定します。

-rise_to
終了点またはクロックのリストで立ち上がるパスに適用されます。

-fall_to
終了点またはクロックのリストで立ち下がるパスに適用されます。

-through
スルー ピン、セル、またはネットのリストを指定します。

-rise_through
立ち上がりスルー ピン、セル、またはネットに適用されます。

-fall_through
立ち下がりスルー ピン、セル、またはネットに適用されます。

-quiet
コマンド エラーを無視します。

delay (必須)
遅延値

set_msg_limit

メッセージ数制限を設定します。

構文

```
set_msg_limit -count arg [-type arg ] [-id arg ] [-quiet]
```

表示結果

なし

使用法

-count (必須)
メッセージの制限カウントを設定します。

-type デフォルト : ALL
「ERROR」など、設定するメッセージ タイプを指定します。

-id
common-99 などのセットするメッセージ ID を指定します。

-quiet
コマンド エラーを無視します。

set_multicycle_path

マルチサイクル パスを定義します。

構文

```
set_multicycle_path [-setup] [-hold] [-rise] [-fall] [-start]
[-end] [-reset_path] [-from args ] [-rise_from args ]
[-fall_from args ] [-to args ] [-rise_to args ] [-fall_to args ]
[-through args ] [-rise_through args ] [-fall_through args ]
[-quiet] [ path_multiplier ]
```

結果

なし

使用法

-setup
セットアップ乗算器のみが設定されます。

-hold
ホールド乗算器のみが設定されます。

-rise
パスの終了ポイントの立ち上がり遅延に有効な乗算器

-fall
パスの終了ポイントの立ち下がり遅延に有効な乗算器

-start
パスの開始ポイントに対して計測される乗算器

-end
パスの終了ポイントに対して計測される乗算器

-reset_path
マルチサイクルを設定する前にこのパスをリセットします。

-from
パスの開始点またはクロックのリストを指定します。

-rise_from
開始点またはクロックのリストから立ち上がるパスに適用されます。

-fall_from
開始点またはクロックのリストから立ち下がるパスに適用されます。

-to
パスの終了点またはクロックのリストを指定します。

-rise_to
終了点またはクロックのリストで立ち上がるパスに適用されます。

-fall_to
終了点またはクロックのリストで立ち下がるパスに適用されます。

-through
スルー ピン、セル、またはネットのリストを指定します。

-rise_through
立ち上がりスルー ピン、セル、またはネットに適用されます。

-fall_through
立ち下がりスルー ピン、セル、またはネットに適用されます。

-quiet
コマンド エラーを無視します。

path_multiplier デフォルト : 1
パスのセットアップ タイミング解析を削除します。

set_operating_conditions

プロセス、温度、および電圧を設定します。

構文

```
set_operating_conditions [-analysis_type arg ] [-library args ]
[-max arg ] [-min arg ] [-max_library args ] [-min_library args ]
[-quiet] [ condition ]
```

表示結果

なし

使用法

-analysis_type
解析タイプ。値は、single、bc_wc、on_chip_variation です。

-library
状態を検索するライブラリ

-max
最大動作状態名

-min
最小動作状態名

-max_library
最大状態を検索するライブラリ

-min_library
最小状態を検索するライブラリ

-quiet
コマンド エラーを無視します。

condition
1 つの動作状態名

set_output_delay

ポートまたはピンの出力遅延を設定します。

構文

```
set_output_delay [-clock args ] [-reference_pin args ]
[-clock_fall] [-level_sensitive] [-rise] [-fall] [-max] [-min]
[-add_delay] [-group_path arg ] [-network_latency_included]
[-source_latency_included] [-quiet] [ delay ] objects ...
```

表示結果

なし

使用法

-clock
相対クロック

-reference_pin
相対ピンまたはポート

-clock_fall
遅延はクロックの立ち下がりエッジに対応します。

-level_sensitive
遅延はレベル センシティブなラッチからになります。

-rise
立ち上がり遅延を指定します。

-fall
立ち下がり遅延を指定します。

-max
最大遅延を指定します。

-min
最小遅延を指定します。

-add_delay
既存の入力遅延を削除しません。

-group_path
パスのグループ名を指定します。

-network_latency_included
既に含まれているクロックのネットワーク レイテンシを指定します。

-source_latency_included
既に含まれているクロックのソース レイテンシを指定します。

-quiet
コマンド エラーを無視します。

delay デフォルト : 1.0
パス遅延

objects (必須)
ポート/ピンのリストを指定します。

set_package_pin_val

1 つまたは複数のパッケージ ピンのユーザー コラムを設定します。

構文

```
set_package_pin_val -package_pins args -column arg -value arg [-quiet]
```

表示結果

なし

使用法

-package_pins (必須)
パッケージ ピン名

-column (必須)
ユーザー列名

value (必須)
設定する値

-quiet
コマンド エラーを無視します。

set_param

パラメータ値を設定します。

構文

```
set_param [-quiet] name  value
```

表示結果

新しく設定されたパラメータ値

使用法

-quiet
コマンド エラーを無視します。

name (必須)
パラメータ名

value (必須)
パラメータ値

set_propagated_clock

伝搬されるクロック レイテンシを指定します。

構文

```
set_propagated_clock [-quiet] objects ...
```

結果

なし

使用法

-quiet
コマンド エラーを無視します。

objects (必須)
クロック、ポート、またはピンのリスト

set_property

オブジェクトのプロパティを設定します。

構文

```
set_property [-quiet] property_name  property_value  objects ...
```

結果

問題のない場合は設定された値、エラーのあった場合は ""

使用法

-quiet
コマンド エラーを無視します。

property_name (必須)
設定するプロパティの名前

property_value (必須)
設定するプロパティの値

objects (必須)
プロパティを設定するオブジェクト

set_speed_grade

タイミング スピード グレード

構文

```
set_speed_grade [-quiet] value
```

結果

結果文字列

使用法

–quiet
コマンド エラーを無視します。

value (必須)
タイミング解析に使用されるスピード グレード

set_switching_activity

指定したオブジェクトのスイッチ動作を設定します。

構文

```
set_switching_activity -toggle_rate arg [-type args ] [-quiet]
```

表示結果

なし

使用法

–toggle_rate (必須)
トグル レート値。0% = 値 = 100%

–type
有効なタイプ値 (registers、inputs、outputs、inouts、ports、outputEnable、three_states、dsps、brams、bramWrite、bramEnable、clockEnable) のリスト

–quiet
コマンド エラーを無視します。

set_system_jitter

システム ジッタを設定します。

構文

```
set_system_jitter [-quiet] system_jitter
```

表示結果

system_jitter

使用法

-quiet
コマンド エラーを無視します。

system_jitter (必須)
システム ジッタ。値 = 0 です。

set_timing_derate

定数遅延のディレーティング係数を指定します。

構文

```
set_timing_derate [-early] [-late] [-clock] [-data] [-net_delay]
[-cell_delay] [-cell_check] [-quiet] derate  objects ...
```

表示結果

なし

使用法

-early
早期ディレート係数を指定します。

-late
後期ディレート係数を指定します。

-clock
クロック パス専用のディレート係数を指定します。

-data
データ パス専用のディレート係数を指定します。

-net_delay
ネット遅延専用のディレート係数を指定します。

-cell_delay
セル遅延専用のディレート係数を指定します。

-cell_check
セル タイミング チェック専用のディレート係数を指定します。

-quiet
コマンド エラーを無視します。

derate (必須)
ディレート係数 :値 = 0

objects (必須)
セル、ライブラリ セル、またはネットのリスト

set_units

チェックするユニットを設定します。

構文

```
set_units [-capacitance arg ] [-time arg ] [-current arg ]
[-voltage arg ] [-power arg ] [-resistance arg ] [-suffix arg ]
[-digits arg ] [-quiet]
```

表示結果

なし

使用法

-capacitance デフォルト : pF
キャパシタンス単位 (ファラッド)

-time デフォルト : ns
時間単位 (秒)

-current デフォルト : mA
電流単位 (アンペア)

-voltage デフォルト :V
電圧単位 (ボルト)

-power デフォルト : mW
消費電力単位 (ワット)

-resistance デフォルト : ohm
抵抗単位 (オーム)

-suffix
単位の接尾語

-digits デフォルト : 1
桁数

-quiet
コマンド エラーを無視します。

split_diff_pair_ports

2 ポート間の差動ペア関係を削除します。

構文

```
split_diff_pair_ports [-quiet] ports ...
```

表示結果

なし

使用法

-quiet
コマンド エラーを無視します。

ports (必須)
分割するポートを指定します。

start_gui

PlanAhead の GUI を起動します。

構文

```
start_gui
```

結果

なし

startgroup

グループ単位で前/次の動作を実行できるコマンド セットを開始します。

構文

```
startgroup [-try] [-quiet]
```

表示結果

int

使用法

-try
既にグループ単位で開始されている場合は、開始しません。

-quiet
コマンド エラーを無視します。

stop_gui

PlanAhead の GUI を閉じます。

構文

```
stop_gui
```

表示結果

なし

swap_locs

2 つの位置を入れ替えます。

構文

```
swap_locs [-quiet] aloc  bloc
```

表示結果

なし

使用法

–quiet
コマンド エラーを無視します。

aloc (必須)
1 つ目の位置 (ポート/セル/サイト – bloc と同じタイプである必要あり)

bloc (必須)
2 つ目の位置 (ポート/セル/サイト – aloc と同じタイプである必要あり)

undo

前のコマンドの実行を取り消します。

構文

```
undo [-list] [-quiet]
```

表示結果

実行不可能なタスクのリスト (-list を使用した場合)

使用法

-list
実行不可能なタスクのリストが表示されます。

-quiet
コマンド エラーを無視します。

unhighlight_objects

現在ハイライトされているオブジェクトのハイライトを解除します。

構文

```
unhighlight_objects [-color_index arg ] [-rgb args ]
[-color arg ] [-quiet] [ objects ]
```

表示結果

なし

使用法

-color_index
インデックスに色を付けます。

-rgb
インデックスに色を付けます。

-color
色の名前

-quiet
コマンド エラーを無視します。

objects
ハイライトを解除するオブジェクト

unmark_objects

現在マークされているアイテムのマークを解除します。

構文

```
unmark_objects [-quiet] [ objects ]
```

表示結果

なし

使用法

–quiet
コマンド エラーを無視します。

objects
マークを解除するオブジェクト

unselect_objects

現在選択されているアイテムの選択を解除します。

構文

```
unselect_objects [-quiet] [ objects ]
```

表示結果

なし

使用法

–quiet
コマンド エラーを無視します。

objects
選択を解除するオブジェクト

update_file

リモート ファイルの内容でインポートされたファイルをアップデートします。

構文

```
update_file -file arg -remote_file arg [-quiet]
```

表示結果

アップデートされたファイル

使用法

-file (必須)
アップデートするインポート ファイル

-remote_file (必須)
インポートするリモート ファイル

-quiet
コマンド エラーを無視します。

upgrade_ip

コンフィギャブル IP を最新 IP にアップデートします。

構文

```
upgrade_ip [-srcset arg ] [-version arg ] -files args [-quiet]
```

結果

アップデートされたファイルのリスト

使用法

–srcset
ソース セット名を指定します。

–version
アップデートする IP のバージョン

–files (必須)
アップデートする IP ソース ファイル

–quiet
コマンド エラーを無視します。

verify_config

インプリメントされた run を解析して、パーシャル リコンフィギュレーションに必要な規則にしたがっているかどうかを解析します。

構文

```
verify_config [-file arg ] [-verbose] [-quiet] runs ...
```

表示結果

パーシャル リコンフィギュレーション レポート

使用法

–file
出力レポート ファイル名

–verbose
ログ ファイルに詳細情報を出力します。

–quiet
コマンド エラーを無視します。

run (必須)
検証するインプリメント済み run のリスト

version

PlanAhead のバージョンおよびバージョンの日付を表示します。

構文

```
version [-build] [-quiet]
```

表示結果

PlanAhead のバージョン

使用法

-build
ビルドのバージョン情報を表示して、閉じます。

-quiet
コマンド エラーを無視します。

wait_on_run

指定した run が終了するまで Tcl コマンドの実行を停止します。

構文

```
wait_on_run [-timeout arg ] [-quiet] run
```

表示結果

なし

使用法

-timeout デフォルト : -1
run が完了するまで待機する最大時間 (分単位)

-quiet
コマンド エラーを無視します。

run (必須)
待機する run

write_bitstream

現在のデザインのビットストリームを書き出します。

構文

```
write_bitstream [-bitgen_options arg ] [-quiet] [ file ]
```

表示結果

なし

使用法

-bitgen_options
bitgen のコマンド ライン オプション

-quiet
コマンド エラーを無視します。

file
生成する BIT ファイルの名前

write_chipscope_cdc

デバッグ ポートに接続されているネットをエクスポートします。

構文

```
write_chipscope_cdc [-quiet] file
```

表示結果

出力ファイル名

使用法

-quiet
コマンド エラーを無視します。

file (必須)
ファイル名

write_csv

パッケージ ピンとポート配置情報をエクスポートします。

構文

```
write_csv [-mode arg ] [-quiet] file
```

表示結果

出力ファイル名

使用法

-mode デフォルト : port
有効な値は port です。

-quiet
コマンド エラーを無視します。

file (必須)
ファイル名

write_edf

現在のネットリストを EDIF ファイルとしてエクスポートします。

構文

```
write_edf [-pblocks args ] [-cell arg ] [-quiet] [ file ]
```

表示結果

出力ファイルまたはディレクトリ名

使用法

-pblocks
これらの Pblock のネットリストをエクスポートします (-cell とは併用できません)。

-cell
このセルのネットリストをエクスポートします (-pblocks とは併用できません)。

-quiet
コマンド エラーを無視します。

file
出力ファイル (-pblocks、-cell で指定したディレクトリ)

write_ibis

現在のフロアプランの IBIS モデルを書き出します。

構文

```
write_ibis [-allmodels] [-pin] [-truncate arg ] [-quiet] file
```

結果

出力ファイル名

使用法

–allmodels
アーキテクチャに使用できるバッファ モデルをすべて含みます。デフォルトでは、フロアプランに使用されるバッファ モデルのみが含まれます。

–pin
詳細なパッケージの各ピン記述 (RLC マトリクス) を含めます。

–truncate デフォルト : 20
出力ファイルの信号名の最大長を指定します。これより長い名前は切り捨てられます。有効な値は、20、40、0 です。値 0 の場合は、名前は切り捨てられません。IBIS 3.2 仕様に従って、デフォルトは 20 です。

–quiet
コマンド エラーを無視します。

file (必須)
出力ファイル名

write_ncd

配置を NCD ファイルにエクスポートします。

構文

```
write_ncd [-quiet] file
```

結果

出力ファイル名

使用法

-quiet
コマンド エラーを無視します。

file (必須)
配置を書き込むファイルの名前

write_pcf

変換された制約を物理制約ファイル (.pcf) にエクスポートします。

構文

```
write_pcf [-quiet] file
```

結果

出力ファイル名

使用法

-quiet
コマンド エラーを無視します。

file (必須)
配置を書き込むファイルの名前

write_sdf

ベント シミュレーションで平坦な SDF 遅延ファイルを生成します。

構文

```
write_sdf [-process_corner arg ] [-cell arg ] [-top_module arg ]
[-quiet] file
```

結果

なし

使用法

-process_corner デフォルト : slow

SDF 遅延の必要とされるプロセス コーナーを指定します。有効なタイは slow と fast です。

-cell デフォルト : デザイン全体

des.subblk.cpu のように、書き込むデザインのルートを指定します。

-top_module デフォルト : 新しい最上位モジュール名

最上位モジュール名を netlist などのカスタムの名前に置き換えます。

-quiet

コマンド エラーを無視します。

file (必須)

ファイル名

write_timing

タイミング結果のセットをファイルにエクスポートします。

構文

```
write_timing [-quiet] name  file
```

結果

なし

使用法

–quiet
コマンド エラーを無視します。

name (必須)
結果のセット名

file (必須)
結果を書き込むファイルの名前

write_ucf

UCF 情報をファイルまたはディレクトリにエクスポートします。

構文

```
write_ucf [-fixed_only] [-constraints arg ] [-pblocks args ]
[-cell arg ] [-quiet] [ file ]
```

結果

出力ファイルまたはディレクトリ名

使用法

-fixed_only

固定された配置のみがエクスポートされます (デフォルトでは固定の有無に関係なくエクスポートされます)。

-constraints デフォルト : valid

無効と示される制約を含めます。有効な値は valid、invalid、または all です。

-pblocks

これらの Pblock の配置をエクスポートします (-cell とは併用できません)。

-cell

このセルの配置をエクスポートします (-pblocks とは併用できません)。

-quiet

コマンド エラーを無視します。

file

出力ファイル (-pblocks、-cell で指定したディレクトリ)

write_verilog

現在のネットリストを Verilog 形式でエクスポートします。

構文

```
write_verilog [-cell arg ] [-mode arg ] [-nolib]
[-writealloverrides arg ] [-top_module arg ]
[-sdf_anno arg ] -sdf_path arg [-quiet] file
```

結果

出力ファイルまたはディレクトリ名

使用法

-cell デフォルト : デザイン全体
des.subblk.cpu のように、書き込むデザインのルートを指定します。

-mode デフォルト : design
有効な値は design、port、または timing_sim です。

-nolib
すべてのライブラリからのセルを 1 つのライブラリにまとめます。

-writealloverrides
デフォルト値と同じであっても、書き込みパラメータがザイリンクス プリミティブに上書きされます。

-top_module デフォルト : 新しい最上位モジュール名
最上位モジュール名を netlist などのカスタムの名前に置き換えます。

-sdf_anno
sdf_annotate システム タスク文が生成されるかどうかを指定します。

-sdf_path (必須)
SDF ファイルのディレクトリまでのパス

-quiet
コマンド エラーを無視します。

file (必須)
書き込むファイルまたはディレクトリ

write_vhdl

現在のネットリストのポートを VHDL 形式でエクスポートします。

構文

```
write_vhdl [-mode arg ] [-quiet] file
```

結果

出力ファイル名

使用法

-mode デフォルト : port
有効な値は port です。

-quiet
コマンド エラーを無視します。

file (必須)
出力ファイル名

write_xdc

XDC 情報をファイルまたはディレクトリにエクスポートします。

構文

```
write_xdc [-fixed_only] [-constraints arg ] [-pblocks args ]
[-cell arg ] [-quiet] file
```

結果

出力ファイルまたはディレクトリ名

使用法

-fixed_only
固定された配置のみがエクスポートされます (デフォルトでは固定の有無に関係なくエクスポートされます)。

-constraints
無効と示される制約を含めます。有効な値は valid、invalid、または all (デフォルトは valid) です。

-pblocks
これらの Pblock の配置をエクスポートします (-cell とは併用できません)。

-cell
このセルの配置をエクスポートします (-pblocks とは併用できません)。

-quiet
コマンド エラーを無視します。

file (必須)
出力ファイル (-pblocks、-cell で指定したディレクトリ)