

PlanAhead Tcl コマンド リファレンス ガイド

UG789 (v 13.3) 2011 年 10 月 19 日



Xilinx is disclosing this user guide, manual, release note, and/or specification (the “Documentation”) to you solely for use in the development of designs to operate with Xilinx hardware devices. You may not reproduce, distribute, republish, download, display, post, or transmit the Documentation in any form or by any means including, but not limited to, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of Xilinx. Xilinx expressly disclaims any liability arising out of your use of the Documentation. Xilinx reserves the right, at its sole discretion, to change the Documentation without notice at any time. Xilinx assumes no obligation to correct any errors contained in the Documentation, or to advise you of any corrections or updates. Xilinx expressly disclaims any liability in connection with technical support or assistance that may be provided to you in connection with the Information.

THE DOCUMENTATION IS DISCLOSED TO YOU “AS-IS” WITH NO WARRANTY OF ANY KIND. XILINX MAKES NO OTHER WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, REGARDING THE DOCUMENTATION, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT OF THIRD-PARTY RIGHTS. IN NO EVENT WILL XILINX BE LIABLE FOR ANY CONSEQUENTIAL, INDIRECT, EXEMPLARY, SPECIAL, OR INCIDENTAL DAMAGES, INCLUDING ANY LOSS OF DATA OR LOST PROFITS, ARISING FROM YOUR USE OF THE DOCUMENTATION.

© Copyright 2002–2012 Xilinx Inc. All Rights Reserved. XILINX, the Xilinx logo, the Brand Window and other designated brands included herein are trademarks of Xilinx, Inc. All other trademarks are the property of their respective owners. The PowerPC name and logo are registered trademarks of IBM Corp., and used under license. All other trademarks are the property of their respective owners.

本資料は英語版 (v.13.3) 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。

この資料に関するフィードバックおよびリンクなどの問題につきましては、jpn_trans_feedback@xilinx.com までお知らせください。いただきましたご意見を参考に早急に対応させていただきます。なお、このメール アドレスへのお問い合わせは受け付けておりません。あらかじめご了承ください。

概要

PlanAhead の Tcl 機能の概要

Tcl (Tool Command Language) は PlanAhead™ ツール環境に統合されているスクリプト言語です。Tcl は、デザイン制約および SDC (Synopsys® Design Constraints) に使用される半導体業界の標準言語です。

SDC は、Synopsys 社の Synplify やその他のベンダー ツールから FPGA 合成ツールへタイミング制約を渡すメカニズムで、タイミング制約の業界標準なので、スクリプト言語には Tcl が最も適しています。

Tcl を使用することで、自動スクリプトだけでなく、デザイン ツールに対するインタラクティブなクエリを実行できます。Tcl には、デザイン データベースからツール、デザイン設定、ステートなどに関する情報をインタラクティブに取り出す機能があります。たとえば、特定のタイミング解析レポートコマンドを検索し、インクリメンタル制約を適用し、その直後にクエリを実行して、ツールの手順を再実行せずに動作が予測どおりかどうかを確認できます。

次のセクションでは、PlanAhead での基本的な Tcl 機能について説明します。

注記： この章には、すべての Tcl コマンドに関する説明は含まれません。Tcl リソースへの参照および PlanAhead 環境での一般的な Tcl の機能に関する説明が含まれます。

Tcl ジャーナル ファイル

PlanAhead ツールを起動すると、デザイン セッション中に実行されるコマンドおよび操作の記録が PlanAhead.log ファイルに記述されます。また、PlanAhead.jou という、セッション中に実行された Tcl コマンドのみを記述したジャーナル ファイルも生成され、このファイルを使用して新しい Tcl スクリプトを作成できます。

注記： このファイルのバックアップ バージョンが planahead.jou_backup で、1 つ前のセッションで実行した Tcl コマンドの詳細が保存されています。

これらのファイルの保存場所に関する情報は、『[PlanAhead ユーザー ガイド](#)』(UG632) の付録 A を参照してください。

Tcl ヘルプ

Tcl の help コマンドを使用すると、サポートされる Tcl コマンドの概要が表示されます。

- ・ `help` : Tcl コマンド カテゴリのリストが返されます。

help

コマンド カテゴリは、File I/O などの特定のファンクションを実行するコマンド グループです。

- ・ `help -category category` : 指定のカテゴリに含まれるコマンドのリストを返します。

help -category object

この例では、オブジェクトを処理する Tcl コマンドのリストが返されます。

- ・ `help pattern` : 指定の検索パターンに一致したコマンドのリストを返します。この構文を使用すると、コマンド グループから特定のコマンドをすばやく検索できます。

help get_*

この例では、「get_」で始まる Tcl コマンドのリストが返されます。

- ・ `help command` : 指定のコマンドに関する詳細情報を返します。

help get_cells

この例では、`get_cells` コマンドの詳細が表示されます。

- ・ `help -short command` : 指定のコマンドの簡単な説明を返します。

help -short get_cells

PlanAhead ツールの起動

PlanAhead ツールには、主に次の 3 つの操作モードがあります。

- ・ GUI モード (デフォルト)
- ・ Tcl コマンドライン オプションを使用した PlanAhead ツールの起動 (バッチ モード)
- ・ Tcl シェル モード

次のセクションでは、バッチ モードと Tcl シェル モードについて説明します。

バッチ モード

PlanAhead ツールでは、起動時に次の 2 つの場所で Tcl 初期化スクリプトが検索されます。

1. `installdir/planAhead/scripts/init.tcl`
2. `userdir/Xilinx/PlanAhead/init.tcl`

説明 :

`installdir` : PlanAhead ツールのインストールディレクトリ

`userdir` : ユーザーのホーム ディレクトリ

- ◆ Windows の場合 : `%APPDATA%/Xilinx/PlanAhead/init.tcl`
- ◆ Linux の場合 : `$HOME/.Xilinx/PlanAhead/init.tcl`

init.tcl がいずれかまたは両方の場所で見つかった場合、インストール ディレクトリ、ホーム ディレクトリの順でこのファイルが使用されます。

- ・ インストール ディレクトリにある init.tcl ファイルを使用すると、企業またはデザイン グループですべてのユーザーに対して共通の初期化スクリプトをサポートできます。インストール ディレクトリから PlanAhead ツールを起動すると、そのディレクトリの init.tcl スクリプトが使用されます。
- ・ ホーム ディレクトリにある init.tcl を使用すると、各ユーザーがそれぞれコマンドを追加したり、デザイン要件を満たすようにツールのインストール ディレクトリに含まれるコマンドを変更できます。

この init.tcl スクリプトは標準の Tcl コマンド ファイルで、PlanAhead ツールでサポートされるどの Tcl コマンドも含めることができます。次の文を追加すると、init.tcl から別の Tcl スクリプト ファイルを参照できます。

```
source path_to_file/file_name.tcl
```

一般的な Tcl 構文のガイドライン

Tcl では、OS に関係なく Linux の表記規則 (/) が使用されます。

次のセクションでは、PlanAhead ツールで Tcl を使用する際の一般的な構文ガイドラインについて説明します。

Tcl スクリプトの実行

Tcl スクリプトを実行するには、次のコマンドを使用します。

```
source file_name
```

PlanAhead の GUI で Tcl スクリプトを実行するには、[Tools] → [Run Tcl Script] をクリックします。

一般的な構文構造

PlanAhead ツールの Tcl コマンドの一般的な構造は、次のとおりです。

```
command [optional_parameters] required_parameters
```

コマンド構文は、アンダースコア (_) で区切られた「動詞 - 名詞」および「動詞 - 形容詞 - 名詞」の形になります。

コマンドは、関連するコマンド同士に同じ接頭辞が付けられ、分類されています。

- ・ クエリを実行するコマンドには、通常 get_ が接頭辞として付いています。
- ・ 値やパラメーターを設定するコマンドには、通常 set_ が接頭辞として付いています。
- ・ レポートを生成するコマンドには、通常 report_ が接頭辞として付いています。

これらのコマンドは、グローバル名前空間に属しており、コマンドに付属するサブコマンドはありません。

構文例

次は、**get_cells -help** コマンドを実行した例です。

```
get_cells
```

Description:

Get a list of cells in the current design

Syntax:

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

Returns:

list of cell objects

Usage:

Name	Optional	Default	Description

-hsc	yes	/	Hierarchy separator
-hierarchical	yes		Search level-by-level in current instance
-regexp	yes		Patterns are full regular expressions
-nocase	yes		Perform case-insensitive matching (valid only when -regexp specified)
-filter	yes		Filter list with expression
-of_objects	yes		Get cells of these pins or nets
-match_style	yes	sdh	Style of pattern matching, valid values are ucf, sdh
-quiet	yes		Ignore command errors
patterns	yes	*	Match cell names against patterns

Categories:

SDC, XDC, Object

不明コマンド

Tcl には、通常サポートされるビルトイン コマンド、Tcl インタープリターに渡される PlanAhead 特有のコマンド、およびユーザー定義のプロシージャのリストが含まれます。

これらの既知のコマンドに含まれないコマンドは OS に送信され、exec コマンドからシェルで実行されます。これにより、OS 特有のシェル コマンドを実行できます。シェル コマンドがない場合、コマンドが見つからなかったことを示すエラー メッセージが表示されます。

リターン コード

Tcl コマンドの中には、オブジェクトのリストやコレクションなどの戻り値が出力されるものがあります。それ以外のコマンドでは、処理は実行されても、ユーザーが直接利用できるような値を返すとは限りません。Tcl インターフェイスを統合したツールの一部には、コマンドでエラーのない場合は 0、エラーがある場合は 1 を返すものもあります。

Tcl コマンドまたはスクリプトのエラーを正しく処理するには、Tcl ビルトイン コマンドの `catch` を使用する必要があります。一般的には、`catch` コマンドと番号付き情報/警告/エラー メッセージに基づいて、Tcl スクリプトのフローで問題を評価します。

PlanAhead ツールの Tcl コマンドでは、コマンドの完了時に `TCL_OK` または `TCL_ERROR` が返され、標準の Tcl メカニズムによりグローバル変数 `$ERRORINFO` が設定されます。

`$ERRORINFO` 変数を使用する場合は、Tcl コンソールでエラーがレポートされた後に次を入力します。

puts \$ERRORINFO

これにより、エラーの詳細情報が表示されます。たとえば、次のコード例では Tcl スクリプト (`procs.tcl`) が使用されており、ユーザー定義の手順 (`loads`) が実行されます。数行トランスクリプト メッセージが表示された後、5 行目にエラーが表示されます。

```
Line 1: PlanAhead % source procs.tcl
Line 2: PlanAhead% loads
Line 3: Found 180 driving FFs
Line 4: Processing pin a_reg_reg[1]/Q...
Line 5: ERROR: [HD-Tcl 53] Cannot specify '-patterns' with '-of_objects'.
Line 6: PlanAhead% puts $errorInfo
Line 7: ERROR: [HD-Tcl 53] Cannot specify '-patterns' with '-of_objects'. While executing
"get_ports -of objects $pin" (procedure "my_report" line 6) invoked from within procs.tcl
```

Tcl スクリプトファイルの `catch` 節に **puts \$errorInfo** を追加し、エラーが見つかったときに詳細を表示するようにしたり、Tcl コンソールでエラーが発生したときに必要に応じて「`puts $errorInfo`」と入力して特定のエラーの詳細を表示できます。

上記のコード例では、6 行目に「**puts \$errorInfo**」と入力することで、7 行目にエラーの詳細情報が表示されています。

Tcl スクリプトの実行

Tcl スクリプトは、コマンドライン オプションの 1 つとして指定するか、GUI の [Tools] → [Run Tcl Script] をクリックして実行できます。Tcl スクリプトを GUI から実行すると、進捗バーが表示され、スクリプトが終了するまですべての GUI 操作がブロックされます。

現在のところ、ランタイム中にスクリプトの実行を一時停止する方法はないので、標準的な OS 手法のプロセスを停止 (kill) する方法で強制終了するしかありません。この場合、最後に保存した後の作業が失われます。

ファースト クラスの Tcl オブジェクトとその関係

PlanAhead ツールの Tcl コマンドを使用すると、ネットリスト、デバイス、プロジェクトのオブジェクト モデルに直接アクセスできます。これらオブジェクトは「ファースト クラス」と呼ばれ、単なる文字列記述ではなく、操作およびクエリが可能であることを意味します。例外もありますが、通常はオブジェクトとしてクエリを実行できます。これらのオブジェクトには、クエリ可能なプロパティが含まれ、ほかのオブジェクトを取得できる関係があります。

オブジェクト タイプと定義

PlanAhead ツールには多くのオブジェクト タイプがありますが、ここでは基本的なタイプの定義と説明を示します。最も基本的で重要なオブジェクト タイプは、デザイン ネットリストのエンティティに関連するもので、次のものがあります。

セル

セルは、プリミティブまたはネットリスト内の階層のいずれかのインスタンスです。これには、フリップフロップ、LUT、I/O バッファ、RAM、DSP のほか、ほかのセルのコレクションのラッパーである階層インスタンスが含まれます。

ピン

ピンはセル上の論理接続ポイントです。ピンにより、セル内部が抽象化されて使用しやすくなります。ピンは、階層またはプリミティブのセル上に存在します。ピンには、クロックピン、データピン、リセットピン、フリップフロップの出力ピンなどが含まれます。

ポート

ポートは、特殊なタイプの階層ピン、最上位ネットリスト オブジェクト、モジュールまたはエンティティ上のピンです。ポートは通常 I/O パッドに接続され、FPGA デバイス外部に接続されます。

ネット

ネットは、物理的に直接相互接続される 1 つのワイヤまたは複数のワイヤです。ネットは階層またはフラットにできますが、常に一連のピンがまとめて分類されます。

クロック

クロックは、デザイン内の順序ロジックに伝搬される周期的な信号です。クロックはプライマリ クロックドメインにできるほか、DCM、PLL、MMCM などのクロック プリミティブで生成できます。クロックは UCF の TIMESPEC PERIOD 制約とほぼ同じで、スタティック タイミング解析アルゴリズムの基盤になっています。

オブジェクトのクエリ

ファーストクラス オブジェクトはすべて、通常次のように Tcl コマンド `get_` を使用してクエリできます。

`get_object_type pattern`

ここで `pattern` は検索パターンであり、必要に応じて階層区切り文字を使用して完全な名前を指定します。オブジェクトは通常、階層の各レベルで文字列パターンを一致させることによりクエリされます。検索パターンには次のようにワイルドカードも使用でき、オブジェクトを検索しやすくなっています。

`get_cells */inst_1`

このコマンドでは、最上位のすぐ下の階層で `inst_1` という名前のセルが検索されます。階層の各レベルで同じパターンを繰り返し使用して検索する場合は、次の構文を使用してください。

`get_cells -hierarchical inst_1`

このコマンドでは、`inst_1` に一致するインスタンスがすべての階層レベルで検索されます。

構文の詳細は、次のコマンドでヘルプ情報を参照してください。

- ・ `help get_cells`
- ・ `get_cells -help`

オブジェクト プロパティ

オブジェクトには、クエリを実行できるプロパティが含まれます。プロパティ名はオブジェクトタイプによって異なります。オブジェクトの特定のプロパティをクエリするには、次のコマンドを使用します。

get_property property_name object

次の例では、セル オブジェクトの lib_cell プロパティをクエリしており、指定のインスタンスがどの UniSim コンポーネントにマップされているかがわかります。

get_property lib_cell [get_cell inst_1]

指定したオブジェクトに使用可能なプロパティすべてを表示するには、**report_property** コマンドを使用します。

report_property [get_cells inst_1]

次の表に、特定のオブジェクトに対して返されるプロパティを示します。

特定のオブジェクトで返されるプロパティ

キー	値	データ型
bel	OLOGICE1.OUTFF	string
class	cell	string
iob	TRUE	string
is_blackbox	0	bool
is_fixed	0	bool
is_partition	0	bool
is_primitive	1	bool
is_reconfigurable	0	bool
is_sequential	1	bool
lib_cell	FD	string
LOC	OLOGIC_X1Y27	string
name	error	string
primitive_group	FD_LD	string
primitive_subgroup	flop	string
site	OLOGIC_X1Y27	string
type	FD & LD	string
XSTLIB	1	bool

プロパティの中には、読み出し専用のものであれば、ユーザー設定が可能なものもあります。UCF や HDL でアノテート可能な属性にマップされるプロパティは、通常 Tcl コマンドの **set_property** でユーザーが設定できます。

set_property loc OLOGIC_X1Y27 [get_cell inst_1]

プロパティに基づいたフィルター処理

オブジェクトをクエリする `get_*` コマンドには、そのオブジェクトのプロパティ値に基づいてクエリをフィルター処理するオプションがあります。このオプションは、非常に優れたオブジェクトクエリコマンド機能です。たとえば、プリミティブ タイプ FD のセルをすべてクエリするには、次を入力します。

```
get_cells * -hierarchical -filter "lib_cell == FD"
```

また、`=~` 演算子を使用すると、文字列パターンでフィルター処理できます。たとえば、デザインに含まれるすべてのフリップフロップ タイプをクエリするには、次を入力します。

```
get_cells * -hierarchical -filter "lib_cell =~ FD*"
```

OR (`||`) や AND (`&&`) を使用すると、複数のプロパティフィルターを組み合わせで検索できます。次の例では、デザイン内のすべてのセルから、フリップフロップ タイプで配置済みロケーション制約が設定されているものをクエリしています。

```
get_cells * -hierarchical -filter {lib_cell =~ FD* && loc != ""}
```

注記： この例では、フィルター オプションの値が `" "` ではなく、`{ }` で囲まれています。これはインタープリターによるコマンド変換を回避する標準的な Tcl 構文で、これにより `loc` プロパティに空の文字列を渡すことができます。

オブジェクトのリスト (コレクション)

複数のオブジェクトを返すコマンドは、通常ネイティブ Tcl リストのようなコンテナ (コレクション) を返します。この PlanAhead ツールの機能により、多数の Tcl オブジェクト処理を大幅に最適化できるので、Tcl ビルトイン コマンドの `foreach` で処理される `foreach_in_collection` のような特殊な反復コマンドを必要としません。

リストが大きい場合、ログ ファイルと GUI の Tcl コンソールでの表示が少し異なります。通常、`get_*` コマンドの結果に対して Tcl 変数を設定すると、リスト全体がコンソールとログ ファイルに表示されますが、リストが大きい場合、ツールのバッファのメモリ オーバーフローを回避するため、コンソールおよびログに出力される際にリストの一部が省略されます。

コンソールとログ ファイルでリストが省略されている場合、最後のエレメントが「...」と表示されますが、変数代入での実際のリストは正しいもので、最後のエレメントはエラーではありません。次は、1 つのセルをクエリした場合と、デザインに含まれるすべてのセルをクエリした場合 (リストが大きい場合) の例です。

```
get_cells inst_1
inst_1
get_cells * -hierarchical
XST_VCC XST_GND error readIngressFifo wbDataForInputReg fifoSelect_0 fifoSelect_1 fifoSelect_2 fifoSelect_3 ...
%set x [get_cells * -hierarchical]
XST_VCC XST_GND error readIngressFifo wbDataForInputReg fifoSelect_0 fifoSelect_1 fifoSelect_2 fifoSelect_3 ...
%lindex $x end
bftClk_BUFGB/bufg
%llength $x
4454
```

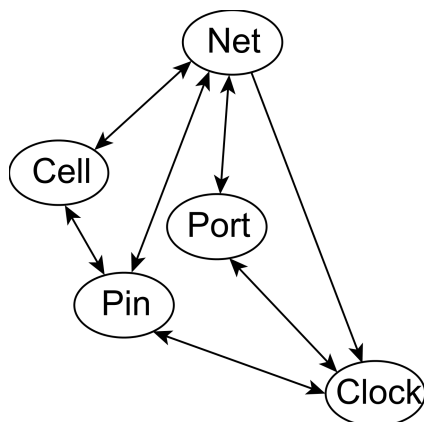
この例では、4000 セルすべてはコンソールに表示されず、「...」で省略されていますが、リストの最後のエレメントは Tcl 変数に正しく含まれています。

オブジェクトの関係

関連するオブジェクトは、`get_*` コマンドに `-of` オプションを使用してクエリできます。たとえば、あるセル ロジックに接続されたピンのリストを取得するには、次を入力します。

```
get_pins -of [get_cells inst_1]
```

次の図は、PlanAhead ツールのオブジェクト タイプとその関係を示します。オブジェクト間の矢印ではその `get_*` コマンドで `-of` オプションを使用して、論理接続をたどって接続されているオブジェクトへの Tcl リファレンスを取得できることを示しています。



エラー、警告、クリティカル警告、および情報メッセージ

各コマンドの結果を示すメッセージは、ログ ファイルと、GUI を使用中であればコンソールにも表示されます。これらのメッセージには識別しやすいように番号が付いています。ログ ファイルでは、INFO、WARNING、CRITICAL_WARNING、ERROR の後にサブシステム識別子と独自の番号が付きます。

次は、タイミング ライブラリを読み込んだ後に表示される INFO メッセージの例です。

```
INFO: [HD-LIB 1] Done reading timing library
```

このようなメッセージにより、ログ ファイルで特定の問題を検出しやすくなり、コマンド実行中の動作内容が理解しやすくなります。

通常、Tcl スクリプトからの Tcl コマンドでエラーが発生すると、続くコマンドの実行は停止されます。これは、回復不可能なエラー状況を避けるためです。これらのエラー状況を捕捉し、続行する Tcl ビルトイン コマンドがあります。一般的な Tcl メカニズムを使用したエラー処理方法については、Tcl リファレンスで `catch` コマンドを参照してください。

カテゴリ別 SDC および Tcl コマンド

カテゴリ

- ・ SDC コマンド
- ・ XDC コマンド
- ・ オブジェクト コマンド
- ・ プロジェクト コマンド
- ・ ファイル入力および出力コマンド
- ・ フロアプラン コマンド
- ・ ピン配置コマンド
- ・ 電力コマンド
- ・ パーシャル リコンフィギュレーション コマンド
- ・ プロパティおよびパラメーター コマンド
- ・ パーティション コマンド
- ・ GUI 制御コマンド
- ・ ツール起動コマンド
- ・ レポート コマンド
- ・ CORE Generator コマンド
- ・ ChipScope コマンド

SDC コマンド

- ・ `all_clocks` : 現在のデザインに含まれるすべてのクロックのリストを取得します。
- ・ `all_fanin` : 指定したシンクのファンインのピンまたはセルのリストを取得します。
- ・ `all_fanout` : 指定したソースのファンアウトのピンまたはセルのリストを取得します。
- ・ `all_inputs` : 現在のデザインの入力ポートすべてのリストを取得します。
- ・ `all_outputs` : 現在のデザインの出力ポートすべてのリストを取得します。
- ・ `all_registers` : 現在のデザインのレジスタ セルまたはピンのリストを取得します。
- ・ `current_design` : 現在のデザインを設定または取得します。
- ・ `current_instance` : 現在のインスタンスを設定または取得します。
- ・ `get_cells` : 現在のデザインのセルのリストを取得します。
- ・ `get_clocks` : 現在のデザインのクロックのリストを取得します。
- ・ `get_hierarchy_separator` : 階層区切り文字を取得します。
- ・ `get_lib_cells` : ライブラリ セルのリストを取得します。デフォルトでは、現在のデザインのパーツに関連するすべてのライブラリ セルが返されます。
- ・ `get_lib_pins` : ライブラリ セル ピンのリストを作成します。
- ・ `get_libs` : ライブラリのリストを作成します。
- ・ `get_nets` : 現在のデザインのネットのリストを取得します。
- ・ `get_pins` : 現在のデザインのピンのリストを取得します。
- ・ `get_ports` : 現在のデザインのポートのリストを取得します。
- ・ `get_timing_arcs` : タイミング アークのリストを取得します。
- ・ `get_timing_paths` : タイミング パスを取得します。
- ・ `report_operating_conditions` : 消費電力予測の動作条件値を取得します。
- ・ `reset_operating_conditions` : 消費電力予測の動作条件をツールのデフォルトにリセットします。
- ・ `set_hierarchy_separator` : 階層区切り文字を設定します。
- ・ `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_operating_conditions` : 消費電力予測の動作条件を設定します。
- ・ `set_units` : チェックするユニットを設定します。

XDC コマンド

- ・ `all_clocks` : 現在のデザインに含まれるすべてのクロックのリストを取得します。
- ・ `all_cpus` : 現在のデザインの CPU セルのリストを取得します。
- ・ `all_dsps` : 現在のデザインの DSP セルのリストを取得します。
- ・ `all_fanin` : 指定したシンクのファンインのピンまたはセルのリストを取得します。
- ・ `all_fanout` : 指定したソースのファンアウトのピンまたはセルのリストを取得します。
- ・ `all_hsios` : 現在のデザインの HSIO セルのリストを取得します。
- ・ `all_inputs` : 現在のデザインの入力ポートすべてのリストを取得します。
- ・ `all_outputs` : 現在のデザインの出力ポートすべてのリストを取得します。

- ・ `all_rams` : 現在のデザインの RAM セルのリストを取得します。
- ・ `all_registers` : 現在のデザインのレジスタ セルまたはピンのリストを取得します。
- ・ `config_timing_analysis` : タイミング解析の一般設定を指定します。
- ・ `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。
- ・ `config_timing_pessimism` : タイミング解析の共通ノードにおける不必要に悪い見積もり部分の削除を設定します。
- ・ `create_operating_conditions` : ライブラリの新しい動作条件を作成します。
- ・ `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取得します。
- ・ `current_instance` : 現在のインスタンスを設定または取得します。
- ・ `delete_power_results` : 指定した消費電力予測結果を削除します。
- ・ `delete_timing_results` : メモリからタイミング結果のセットを消去します。
- ・ `filter` : リストにフィルターを適用し、新しいリストを作成します。
- ・ `get_cells` : 現在のデザインのセルのリストを取得します。
- ・ `get_clocks` : 現在のデザインのクロックのリストを取得します。
- ・ `get_designs` : 現在のデザインに含まれるデザインのリストを取得します。
- ・ `get_generated_clocks` : 現在のデザインの生成済みクロックのリストを取得します。
- ・ `get_hierarchy_separator` : 階層区切り文字を取得します。
- ・ `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取得します。
- ・ `get_jobbanks` : I/O バンクのリストを取得します。
- ・ `get_lib_cells` : ライブラリ セルのリストを取得します。デフォルトでは、現在のデザインのパーツに関連するすべてのライブラリ セルが返されます。
- ・ `get_lib_pins` : ライブラリ セル ピンのリストを作成します。
- ・ `get_libs` : ライブラリのリストを作成します。
- ・ `get_nets` : 現在のデザインのネットのリストを取得します。
- ・ `get_package_pins` : パッケージ ピンのリストを取得します。
- ・ `get_path_groups` : 現在のデザインのパス グループのリストを取得します。
- ・ `get_pins` : 現在のデザインのピンのリストを取得します。
- ・ `get_ports` : 現在のデザインのポートのリストを取得します。
- ・ `get_property` : オブジェクトのプロパティを取得します。
- ・ `get_sites` : サイトのリストを取得します。
- ・ `get_timing_arcs` : タイミング アークのリストを取得します。
- ・ `get_timing_paths` : タイミング パスを取得します。
- ・ `report_default_switching_activity` : 指定したデフォルト タイプのスイッチング アクティビティを取得します。
- ・ `report_operating_conditions` : 消費電力予測の動作条件値を取得します。
- ・ `report_switching_activity` : 指定したオブジェクトのスイッチング アクティビティを取得します。

- ・ `reset_default_switching_activity` : デフォルト タイプのスイッチング アクティビティをリセットします。
- ・ `reset_operating_conditions` : 消費電力予測の動作条件をツールのデフォルトにリセットします。
- ・ `reset_property` : オブジェクトのプロパティをリセットします。
- ・ `reset_switching_activity` : 指定したオブジェクトのスイッチング アクティビティをリセットします。
- ・ `set_clock_gating` : 消費電力で最適化するデザインのクロック ゲーティング オプションを設定します。
- ・ `set_default_switching_activity` : 指定したタイプのデフォルトのスイッチング アクティビティを設定します。
- ・ `set_delay_model` : タイミング解析用のインターコネクト遅延モデルを設定します。
- ・ `set_hierarchy_separator` : 階層区切り文字を設定します。
- ・ `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_operating_conditions` : 消費電力予測の動作条件を設定します。
- ・ `set_property` : オブジェクトのプロパティを設定します。
- ・ `set_switching_activity` : 指定したオブジェクトまたはデフォルトのタイプのスイッチング アクティビティを設定します。
- ・ `set_units` : チェックするユニットを設定します。

オブジェクト コマンド

- ・ `filter` : リストにフィルターを適用し、新しいリストを作成します。
- ・ `get_cells` : 現在のデザインのセルのリストを取得します。
- ・ `get_clocks` : 現在のデザインのクロックのリストを取得します。
- ・ `get_debug_cores` : 現在のデザインの ChipScope デバッグ コアのリストを取得します。
- ・ `get_debug_ports` : 現在のデザインの ChipScope デバッグ ポートのリストを取得します。
- ・ `get_designs` : 現在のデザインに含まれるデザインのリストを取得します。
- ・ `get_files` : ソース ファイルのリストを取得します。
- ・ `get_filesets` : 現在のプロジェクトのファイルセットのリストを取得します。
- ・ `get_generated_clocks` : 現在のデザインの生成済みクロックのリストを取得します。
- ・ `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取得します。
- ・ `get_iobanks` : I/O バンクのリストを取得します。
- ・ `get_ipdefs` : 現在の IP カタログから IP のリストを取得します。
- ・ `get_ips` : 現在のデザインの IP のリストを取得します。
- ・ `get_lib_cells` : ライブラリ セルのリストを取得します。デフォルトでは、現在のデザインのパーツに関連するすべてのライブラリ セルが返されます。
- ・ `get_lib_pins` : ライブラリ セル ピンのリストを作成します。
- ・ `get_libs` : ライブラリのリストを作成します。
- ・ `get_nets` : 現在のデザインのネットのリストを取得します。
- ・ `get_package_pins` : パッケージ ピンのリストを取得します。

- ・ `get_parts` : ソフトウェアで使用可能なパーツのリストを取得します。
- ・ `get_path_groups` : 現在のデザインのパス グループのリストを取得します。
- ・ `get_pblocks` : 現在のデザインの Pblock のリストを取得します。
- ・ `get_pins` : 現在のデザインのピンのリストを取得します。
- ・ `get_ports` : 現在のデザインのポートのリストを取得します。
- ・ `get_projects` : プロジェクトのリストを取得します。
- ・ `get_property` : オブジェクトのプロパティを取得します。
- ・ `get_reconfig_modules` : 現在のプロジェクトのリコンフィギュラブル モジュールのリストを取得します。
- ・ `get_runs` : run (実行パターン) のリストを取得します。
- ・ `get_selected_objects` : 選択したオブジェクトを取得します。
- ・ `get_sites` : サイトのリストを取得します。
- ・ `get_timing_arcs` : タイミング アークのリストを取得します。
- ・ `get_timing_paths` : タイミング パスを取得します。
- ・ `list_property` : オブジェクトのプロパティをリストします。
- ・ `list_property_value` : オブジェクトの有効なプロパティ値をリストします。
- ・ `report_property` : オブジェクトのプロパティをレポートします。
- ・ `reset_property` : オブジェクトのプロパティをリセットします。
- ・ `set_property` : オブジェクトのプロパティを設定します。

プロジェクト コマンド

- ・ `add_files` : アクティブなファイルセットにソースを追加します。
- ・ `archive_project` : 現在のプロジェクトのアーカイブを作成します。
- ・ `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。
- ・ `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 を作成します。
- ・ `current_fileset` : 現在のファイルセットを取得します。
- ・ `current_project` : 現在のプロジェクトを設定または取得します。
- ・ `current_run` : 現在の run を設定または取得します。
- ・ `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。
- ・ `find_top` : 供給されているファイル、ファイルセット、またはアクティブ ファイルセットからトップ モジュールの候補を検索します。ランク付けされた候補のリストを返します。
- ・ `get_files` : ソース ファイルのリストを取得します。
- ・ `get_filesets` : 現在のプロジェクトのファイルセットのリストを取得します。
- ・ `get_ipdefs` : 現在の IP カタログから IP のリストを取得します。

- ・ `get_ips` : 現在のデザインの IP のリストを取得します。
- ・ `get_parts` : ソフトウェアで使用可能なパーツのリストを取得します。
- ・ `get_projects` : プロジェクトのリストを取得します。
- ・ `get_runs` : run (実行パターン) のリストを取得します。
- ・ `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` : ファイルまたは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_synplify` : 指定した Synplify プロジェクト ファイルをインポートします。
- ・ `import_xise` : 作成したプロジェクトに XISE 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 プロジェクト 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。
- ・ `link_design` : ネットリスト デザインを開きます。
- ・ `open_impl_design` : インプリメント済みデザインを開きます。
- ・ `open_io_design` : I/O ピン配置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_run` : ネットリストまたはインプリメンテーション デザインで run 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` : 最新でないファイル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_msg_severity` : 指定した ID のメッセージの重要度をリセットします。
- ・ `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作成します。
- ・ `set_msg_severity` : 指定した ID のメッセージの重要度を設定します。
- ・ `set_speed_grade` : タイミング解析に使用するスピード グレードを指定します。
- ・ `update_design` : 現在のデザインのネットリストをアップデートします。
- ・ `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` : コンフィギャラブル IP を最新バージョンにアップグレードします。
- ・ `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生成します。

ファイル入力および出力コマンド

- ・ `config_webtalk`：ソフトウェア、IP、およびデバイスの使用統計をザイリンクスに送信する WebTalk をイネーブル/ディスエーブルにします。
- ・ `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_edif`：1 つまたは複数の EDIF ファイルを読み込みます。
- ・ `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 スタティック タイミング解析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生成します。
- ・ `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_edif`：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用のフラットな 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を VHDL 形式でエクスポートします。
- ・ `write_xdc`：作成する XDC ファイルの名前を指定します。XDC ファイルのデフォルトのファイル拡張子は .xdc です。このコマンドを `-pblocks` または `-cell` を使用して呼び出した場合、file 引数は出力ディレクトリを指定します。

フロアプラン コマンド

- ・ `add_cells_to_pblock` : Pblock にセルを追加します。
- ・ `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 のリストを取得します。
- ・ `place_cell` : 1 つまたは複数のインスタンスを新しい場所に移動または配置します。サイトおよびセルは正しい順序でリストし、サイトの数とセルの数は同じである必要があります。
- ・ `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。
- ・ `reset_ucf` : ファイルから読み込んだフロアプラン制約を消去します。
- ・ `resize_pblock` : Pblock 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` : 2 つのポートから差動ペアを作成します。
- ・ `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 ピンのユーザー列を設定します。
- ・ `split_diff_pair_ports` : 2 ポート間の差動ペアの関係を削除します。

電力コマンド

- ・ `power_opt_design` : 高度なクロック ゲーティングを使用して、ダイナミック消費電力を最適化します。
- ・ `report_clock_gating` : クロック ゲーティングをレポートします。
- ・ `report_power` : 消費電力予測を実行し、レポートを表示します。
- ・ `reset_switching_activity` : 指定したオブジェクトのスイッチング アクティビティをリセットします。
- ・ `set_clock_gating` : 消費電力で最適化するデザインのクロック ゲーティング オプションを設定します。
- ・ `set_switching_activity` : 指定したオブジェクトまたはデフォルトのタイプのスイッチング アクティビティを設定します。

パーシャル リコンフィギュレーション コマンド

- ・ `config_partition`：指定の run で使用するモジュールとステートを設定します。
- ・ `create_reconfig_module`：新しいリコンフィギュラブル モジュールを作成してセルに追加します。セルは、リコンフィギュラブル パーティションとしてマークされます。
- ・ `delete_reconfig_module`：リコンフィギュラブル 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、インポートに使用されないようにします。
- ・ `get_reconfig_modules`：現在のプロジェクトのリコンフィギュラブル モジュールのリストを取得します。
- ・ `load_reconfig_modules`：指定の run から特定のリコンフィギュラブル 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。
- ・ `reset_property`：オブジェクトのプロパティをリセットします。
- ・ `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従っているかどうかを確認します。

プロパティおよびパラメーター 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`：リストにフィルターを適用し、新しいリストを作成します。
- ・ `get_param`：パラメーター値を取得します。
- ・ `get_property`：オブジェクトのプロパティを取得します。
- ・ `list_param`：すべてのパラメーター名を取得します。
- ・ `list_property`：オブジェクトのプロパティをリストします。
- ・ `list_property_value`：オブジェクトの有効なプロパティ値をリストします。
- ・ `report_param`：すべてのパラメーターに関する情報を取得します。
- ・ `report_property`：オブジェクトのプロパティをレポートします。
- ・ `reset_param`：パラメーターをリセットします。
- ・ `reset_property`：オブジェクトのプロパティをリセットします。
- ・ `set_param`：パラメーター値を設定します。
- ・ `set_property`：オブジェクトのプロパティを設定します。

パーティション コマンド

- ・ `config_partition` : 指定の run で使用するモジュール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、インポートに使用されないようにします。
- ・ `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。
- ・ `reset_property` : オブジェクトのプロパティをリセットします。
- ・ `set_property` : オブジェクトのプロパティを設定します。

GUI 制御コマンド

- ・ `endgroup` : グループ単位で実行を取り消し/やり直しできるコマンド シーケンスを終了します。
- ・ `get_selected_objects` : 選択したオブジェクトを取得します。
- ・ `highlight_objects` : オブジェクトを指定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` : 取り消されたコマンドをやり直します。
- ・ `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` : グループ単位で実行を取り消し/やり直しできるコマンド シーケンスを開始します。
- ・ `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

- ・ `compplib` : 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。
- ・ `data2mem` : データをメモリに変換します。
- ・ `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` : 発生する可能性のあるタイミング問題をチェックします。

- ・ `create_slack_histogram`：ヒストグラムを作成します。
- ・ `delete_clocknetworks_results`：メモリのクロック ネットワーク結果のセットを消去します。
- ・ `delete_timing_results`：メモリからタイミング結果のセットを消去します。
- ・ `get_msg_count`：メッセージ数を取得します。
- ・ `get_msg_limit`：メッセージ数の制限を取得します。
- ・ `report_clock_interaction`：クロックの相互関係をレポートします。
- ・ `report_clock_utilization`：デザインのクロック ネットに関する情報をレポートします。
- ・ `report_clocknetworks`：クロック ネットワークをレポートします。
- ・ `report_clocks`：クロックをレポートします。
- ・ `report_config_timing`：タイミング解析に影響する設定をレポートします。
- ・ `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_drc`：DRC を実行します。
- ・ `report_io`：デバイスのすべての I/O 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ーに関する情報を取得します。
- ・ `report_power`：消費電力予測を実行し、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`：リソース予測を実行し、レポートを表示します。
- ・ `report_route_status`：配線のステータスをレポートします。デフォルトでは、物理ネットの総数、配線済みネット数、未配線のネット数、部分的に配線済みのネット数がレポートされます。配線オブジェクトを指定すると、各ノードの名前と接続を含むその配線オブジェクトの接続がレポートされます。
- ・ `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_utilization`：デバイス使用率を算出し、レポートを表示します。
- ・ `reset_drc`：DRC の結果を削除します。
- ・ `reset_msg_limit`：メッセージ数制限をリセットします。
- ・ `reset_ssn`：メモリから SSN 結果を消去します。
- ・ `reset_sso`：メモリから WASSO 結果を消去します。
- ・ `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。

CORE Generator コマンド

- ・ `create_ip` : コンフィギュラブル 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` : コンフィギュラブル 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` : コンフィギュラブル 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 チャンネルに接続します。
- ・ `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 チャンネルからネットとピンの接続を解除します。
- ・ `get_debug_cores` : 現在のデザインの ChipScope デバッグ コアのリストを取得します。
- ・ `get_debug_ports` : 現在のデザインの ChipScope デバッグ ポートのリストを取得します。
- ・ `implement_debug_core` : ChipScope デバッグ コアをインプリメントします。
- ・ `launch_chipscope_analyzer` : run に対して ChipScope Analyzer ツールを起動します。
- ・ `read_chipscope_cdc` : ChipScope Core Inserter の CDC ファイルをインポートします。
- ・ `report_debug_core` : ChipScope デバッグ コアの詳細をレポートします。
- ・ `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。

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 SDC および Tcl コマンドをリストしています。

add_cells_to_pblock

Pblock にセルを追加します。

構文

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

戻り値

なし

使用法

名 前	必須/ オプション	デフォルト	説明
<code>-add_primitives</code>	オプション		指定したインスタンスのすべてのプリミティブを Pblock に割り当てます。
<code>-clear_locs</code>	オプション		インスタンスのロケーション制約を削除します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>pblock</code>	必須		セルを追加する Pblock を指定します。
<code>cells</code>	必須		追加するセルを指定します。

カテゴリ

フロアプラン

説明

指定したロジック インスタンスを Pblock に追加します。

Pblock にセルを追加したら、**place_pblocks** コマンドを使用して Pblock を FPGA のファブリック上に配置できます。Pblock が自動的に配置された場合、**resize_pblocks** コマンドを使用して Pblock を手動で移動し、サイズを変更できます。

Pblock からインスタンスを削除するには、**remove_cells_from_pblock** コマンドを使用します。

引数

-add_primitives : 指定したインスタンスのすべてのプリミティブを Pblock に割り当てます。ブロック モジュールを指定し、そのモジュールに含まれるすべてのインスタンスを自動的に指定の Pblock に追加できます。

-clear_locs : 既に配置されているセルのインスタンス ロケーション制約を消去します。これにより、フロアプランするために新しい Pblock を定義する際に、セルの LOC 制約をリセットできます。このオプションを指定しないと、配置が指定されているインスタンスを Pblock に追加する場合に配置が解除されません。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

pblock : 指定のインスタンスを追加する Pblock の名前を指定します。

cells : 指定の Pblock に追加する 1 つまたは複数のセル オブジェクトを指定します。

例

次の例では、pb_cpuEngine という Pblock を作成し、cpuEngine モジュールに含まれるすべてのプリミティブを追加し、配置済みのインスタンスの配置制約を消去します。

```
create_pblock pb_cpuEngine
add_cells_to_pblock pb_cpuEngine [get_cells cpuEngine] -add_primitives -clear_locs
```

関連項目

- [get_pblocks](#)
- [place_pblocks](#)
- [remove_cells_from_pblock](#)
- [resize_pblock](#)

add_files

アクティブなファイルセットにソースを追加します。

構文

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

戻り値

追加されたファイル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-fileset</code>	オプション		ファイルセット名を指定します。
<code>-norecurse</code>	オプション		下位ディレクトリでは検索を実行しないよう指定します。
<code>-scan_for_includes</code>	オプション		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。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>files</code>	オプション		追加するファイルおよびディレクトリ名を指定します。 <code>-scan_for_includes</code> を使用しない場合は指定する必要があります。

カテゴリ

プロジェクト

説明

1 つ以上のソース ファイルまたは 1 つ以上のディレクトリ内のソース ファイルを指定したファイルセットに追加します。

ファイルをローカルプロジェクト フォルダにコピーし、指定したファイルセットにも追加する `import_files` コマンドとは異なり、指定したファイルセットを参照することによってのみファイルを追加します。

引数

`-fileset name`：指定したソース ファイルを追加するファイルセットを指定します。指定したファイルセットが存在しない場合は、エラーが表示されます。ファイルセットを指定しない場合は、デフォルトでソース ファイルセットに追加されます。

files : 指定したファイルセットに追加するファイルまたはディレクトリ名のリストを指定します。ディレクトリ名を指定した場合は、そのディレクトリと下位ディレクトリに含まれる有効なソースファイルすべてが追加されます。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリで指定ファイルが検索されます。

-norecurse : 指定したディレクトリの下位ディレクトリでコマンドを実行しないよう指定します。デフォルトではこのオプションは指定されず、下位ディレクトリでもプロジェクトに追加可能なソース ファイルが検索されます。

-search_for_includes : Verilog ソース ファイルで **'include** 文を検索し、それらの参照ファイルも指定したファイルセットに追加します。デフォルトでは、**'include** ファイルは追加されません。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、rtl.v というファイルを現在のプロジェクトに追加します。

```
add_files rtl.v
```

この例ではパスが指定されていないので、現在の作業ディレクトリで rtl.v ファイルが検索されます。ファイルセットは指定されていないので、ファイルはデフォルトでソース ファイルセットに追加されます。

次の例は、**top.ucf** というファイルを **constrs_1** 制約ファイルセットに追加し、project_1 ディレクトリとその下位ディレクトリの有効なソース ファイルを追加します。

```
add_files -fileset constrs_1 -quiet c:/Design/top.ucf c:/Design/project_1
```

この例では、C:/Design ディレクトリの **top.ucf** ファイル、および **project_1** ディレクトリとその下位ディレクトリの制約ファイルが検出され、指定した **constrs_1** 制約セットに追加されます。

また、**-quiet** オプションが使用されているので、コマンドライン エラーは無視されます。

-norecurse オプションが指定されていれば、**project_1** ディレクトリの制約ファイルのみが追加され、下位ディレクトリは検索されません。

関連項目

[import_files](#)

all_clocks

現在のデザインに含まれるすべてのクロックのリストを取得します。

構文

```
all_clocks [-quiet]
```

戻り値

クロック オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

[SDC](#)、[XDC](#)

説明

現在のデザインで宣言されたすべてのクロックのリストを返します。デザインの特定のクロックのリストを取得するには、**get_clocks** コマンドを使用します。

クロックを定義するには、**create_clock** または **create_generated_clock** コマンドを使用します。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、サンプル CPU ネットリスト プロジェクトのすべてのクロックが表示されます。

```
% all_clocks
cpuClk wbClk usbClk phy_clk_pad_0_i phy_clk_pad_1_i fftClk
```

次の例では、返されたリストを別のコマンドに渡しています。

```
% set_propagated_clock [all_clocks]
```

この例では、**set_propagated_clock** コマンドがデザインのすべてのクロックに適用されます。

関連項目

[get_clocks](#)

all_cpus

現在のデザインの CPU セルのリストを取得します。

構文

```
all_cpus [-quiet]
```

戻り値

CPU セル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC

説明

現在のデザインの CPU セル オブジェクトのリストを取得します。現在のデザインで宣言されたすべての CPU セル オブジェクトのリストを作成します。

注記：このコマンドを実行すると、CPU セル オブジェクトのリストが返されます。

引数

-quiet (オプション) : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、現在のデザインの CPU オブジェクトすべてが返されます。

```
all_cpus
```

次の例では、返されたリストを別のコマンドに渡しています。

```
set_false_path -from [all_cpus] -to [all_registers]
```

関連項目

- ・ [all_dsps](#)
- ・ [all_hsios](#)
- ・ [all_registers](#)

all_dsps

現在のデザインの DSP セルのリストを取得します。

構文

```
all_dsps [-quiet]
```

戻り値

DSP セル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC

説明

現在のデザインの DSP セル オブジェクトのリストを返します。現在のデザインで宣言されたすべての DSP セル オブジェクトのリストを作成します。

引数

-quiet (オプション) : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、現在のデザインで定義されたすべての DSP のリストが返されます。

```
all_dsps
```

次の例では、返されたリストを別のコマンドに渡しています。

```
set_false_path -from [all_dsps] -to [all_registers]
```

関連項目

- [all_cpus](#)
- [all_hsios](#)
- [all_registers](#)

all_fanin

指定したシンクのファンインのピンまたはセルのリストを取得します。

構文

```
all_fanin [-startpoints_only] [-flat] [-only_cells] [-levels arg]
[-pin_levels arg] [-trace_arcs arg] [-quiet] [to]
```

戻り値

セルまたはピン オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-startpoints_only</code>	オプション		タイミング開始点のみを検出します。
<code>-flat</code>	オプション		階層を無視します。
<code>-only_cells</code>	オプション		セルのみを検出します。
<code>-levels</code>	オプション	0	処理されるセル レベルの最大数を指定します。
<code>-pin_levels</code>	オプション	0	処理されるピン レベルの最大数を指定します。
<code>-trace_arcs</code>	オプション		トレースするネットワーク アークのタイプを指定します。有効な値は timing、enabled、all です。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>to</code>	オプション		シンク ピン、ポート、またはネットのリストを指定します。

カテゴリ

SDC、XDC

説明

指定したシンクのファンインのポート、ピン、またはセルをレポートします。

注記：このコマンドを実行すると、セル、ピン、またはポート オブジェクトのリストが返されます。

引数

`-startpoints_only` (オプション)：タイミング開始点のみを検出します。

`-flat` (オプション)：デザインの階層を無視します。

`-only_cells` (オプション)：指定したシンクのファンイン パスにあるセル オブジェクトのみを返します。ピンまたはポートは返しません。

-levels *value* (オプション)：処理されるセル レベルの最大数を指定します。デフォルト値は 0 です。

-pin_levels *value* (オプション)：処理されるピン レベルの最大数を指定します。デフォルト値は 0 です。

-trace_arcs *value*：トレースするネットワーク アークのタイプを指定します。有効な値は timing、enabled、および all です。

-quiet (オプション)：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

to (オプション)：指定したピン、ポート、またはネットへのファンイン オブジェクトをレポートします。

例

次の例では、DAT ポートのタイミング ファンインがリストされます。

```
all_fanin DAT
```

関連項目

- ・ [current_design](#)
- ・ [all_fanout](#)

all_fanout

指定したソースのファンアウトのピンまたはセルのリストを取得します。

構文

```
all_fanout [-endpoints_only] [-flat] [-only_cells] [-levels arg]
[-pin_levels arg] [-trace_arcs arg] [-quiet] [from]
```

戻り値

セルまたはピン オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-endpoints_only	オプション		タイミング終了点のみを検出します。
-flat	オプション		階層を無視します。
-only_cells	オプション		セルのみを検出します。
-levels	オプション	0	処理されるセル レベルの最大数を指定します。
-pin_levels	オプション	0	処理されるピン レベルの最大数を指定します。
-trace_arcs	オプション		トレースするネットワーク アークのタイプを指定します。有効な値は timing、enabled、all です。
-quiet	オプション		コマンド エラーを無視します。
<i>from</i>	オプション		ソース ピン、ポート、またはネットのリストを指定します。

カテゴリ

SDC、XDC

説明

指定したソースのファンアウトのポート、ピン、またはセルをレポートします。

注記：このコマンドを実行すると、セルまたはピン オブジェクトのリストが返されます。

引数

-endpoints_only (オプション)：タイミング終了点のみを検出します。

-flat (オプション)：デザインの階層を無視します。

-only_cells (オプション)：指定したソースのファンアウト パスにあるセル オブジェクトのみを返します。

-levels *value* (オプション)：処理されるセル レベルの最大数を指定します。デフォルト値は 0 です。

-pin_levels *value* (オプション)：処理されるピン レベルの最大数を指定します。デフォルト値は 0 です。

-trace_arcs *value*：トレースするネットワーク アークのタイプを指定します。有効な値は timing、enabled、および all です。

-quiet (オプション)：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

from (オプション)：指定したソース ポート、ピン、またはネットのファンアウト パスにあるオブジェクトをレポートします。

例

次の例では、現在のデザインに含まれるポート DAT のタイミング ファンアウトがリストされます。

```
all_fanout -from DAT
```

関連項目

- ・ [current_design](#)
- ・ [all_fanin](#)

all_hsios

現在のデザインの HSIO セルのリストを取得します。

構文

```
all_hsios [-quiet]
```

戻り値

HSIO セル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC

説明

現在のデザインで宣言されたすべての高速 I/O (HSIO) セル オブジェクトのリストを返します。これらの HSIO セル オブジェクトは、変数に代入するか、別のコマンドに渡すことができます。

引数

-quiet (オプション) : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、現在のデザインの HSIO オブジェクトすべてが返されます。

```
all_hsios
```

次の例では、返されたリストを別のコマンドに渡しています。

```
set_false_path -from [all_hsios] -to [all_registers]
```

関連項目

- [all_cpus](#)
- [all_dsps](#)
- [all_registers](#)

all_inputs

現在のデザインの入力ポートすべてのリストを取得します。

構文

```
all_inputs [-quiet]
```

戻り値

ポート オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

[SDC](#)、[XDC](#)

説明

現在のデザインの入力ポート オブジェクトすべてのリストを返します。

引数

-quiet (オプション) : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、現在のデザインの入力ポート オブジェクトすべてが返されます。

```
all_inputs
```

次の例では、返されたリストを別のコマンドに渡しています。

```
set_input_delay 5 -clock REFCLK [all_inputs]
```

関連項目

[all_outputs](#)

all_outputs

現在のデザインの出力ポートすべてのリストを取得します。

構文

```
all_outputs [-quiet]
```

戻り値

ポート オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

SDC、XDC

説明

現在のデザインで宣言されたすべての出力ポート オブジェクトのリストを返します。

引数

-quiet (オプション) : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、現在のデザインの出力ポート オブジェクトすべてが返されます。

```
all_outputs
```

次の例では、返されたリストを別のコマンドに渡しています。

```
set_output_delay 5 -clock REFCLK [all_outputs]
```

関連項目

[all_inputs](#)

all_rams

現在のデザインの RAM セルのリストを取得します。

構文

```
all_rams [-quiet]
```

戻り値

RAM セル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC

説明

現在のデザインで宣言されたすべての RAM セル オブジェクトのリストを返します。これらの RAM セル オブジェクトは、変数に代入するか、別のコマンドに渡すことができます。

引数

-quiet (オプション) : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、現在のデザインの RAM オブジェクトすべてが返されます。

```
all_rams
```

関連項目

- [all_cpus](#)
- [all_dsps](#)
- [all_hsios](#)
- [all_registers](#)

all_registers

現在のデザインのレジスタ セルまたはピンのリストを取得します。

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]
```

戻り値

セルまたはピン オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-clock	オプション		指定したクロックが供給されるレジスタを取得します。
-rise_clock	オプション		クロックの立ち上がりエッジでトリガーされるレジスタを取得します。
-fall_clock	オプション		クロックの立ち下がりエッジでトリガーされるレジスタを取得します。
-cells	オプション		セルのリストを返します (デフォルト)。
-data_pins	オプション		レジスタのデータ ピンのリストを返します。
-clock_pins	オプション		レジスタのクロック ピンのリストを返します。
-async_pins	オプション		非同期プリセット/クリア ピンのリストを返します。
-output_pins	オプション		レジスタの出力ピンのリストを返します。
-level_sensitive	オプション		レベルで認識されるラッチのみを返します。
-edge_triggered	オプション		エッジでトリガーされるフリップフロップのみを返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

SDC、XDC

説明

現在のデザインのシーケンシャル セルまたはピンのリストを返します。このコマンドを実行すると、レジスタまたはピン オブジェクトのリストが返されます。デフォルトでは、デザインのすべてのレジスタ セルのリストが返されます。

返されるオブジェクトは、下に説明するさまざまな引数を使用することにより制限できます。指定のクロックに制限したり、指定のクロックの立ち上がりエッジまたは立ち下がりエッジでトリガーされるレジスタに制限したりすることが可能です。

また、ピン引数を使用することにより、レジスタ オブジェクトの代わりにレジスタのピンを返すことができます。

引数

-cells (オプション)：ピン オブジェクトではなくレジスタ セル オブジェクトを返します。これがデフォルトの動作です。

-clock args (オプション)：指定したクロックのファンアウトにクロック ピンが含まれるレジスタすべてのリストを返します。

- ・ **-rise_clock args** (オプション)：指定したクロックの立ち上がりエッジでトリガーされるレジスタのリストを返します。
- ・ **-fall_clock args** (オプション)：指定したクロックの立ち下がりエッジでトリガーされるレジスタのリストを返します。
- ・ 注記：クロックは、**-clock**、**-rise_clock**、または **-fall_clock** のいずれか 1 つで指定する必要があります。これらの引数を組み合わせて使用することはできません。

-level_sensitive (オプション)：レベルで認識されるレジスタまたはラッチを返します。

-edge_triggered (オプション)：エッジでトリガーされるレジスタまたはフリップフロップを返します。

-data_pins (オプション)：デザインに含まれるすべてのレジスタまたは検索条件を満たすレジスタのデータ ピンのリストを返します。

-clock_pins (オプション)：検索条件を満たすレジスタのクロック ピンのリストを返します。

-async_pins (オプション)：検索条件を満たすレジスタの非同期ピンを返します。

-output_pins (オプション)：検索条件を満たすレジスタの出力ピンのリストを返します。

注記：**-*_pins** 引数は、個別に使用する必要があります。複数の引数を同時に使用すると、1 つの引数のみが **-data_pins**、**-clock_pins**、**-async_pins**、**-output_pins** の優先順位で使用されます。

-quiet (オプション)：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、デザインの任意のクロックの立ち下がりエッジでトリガーされるレジスタのリストが返されます。

```
all_registers -fall_clock [all_clocks]
```

次の例では、返されたリストを別のコマンドに渡しています。

```
set_min_delay 2.0 -to [all_registers -clock CCLK -data_pins]
```

関連項目

[current_design](#)

archive_project

現在のプロジェクトのアーカイブを作成します。

構文

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

戻り値

true

使用法

名 前	必須/ オプション	デフォルト	説明
<code>-force</code>	オプション		既存のアーカイブ ファイルを上書きします。
<code>-exclude_run_results</code>	オプション		アーカイブに run の結果を含めません。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<i>file</i>	オプション		アーカイブ ファイル名を指定します。

カテゴリ

プロジェクト

説明

PlanAhead プロジェクトのアーカイブを作成して、プロジェクトのバックアップとして保存したり、リモート サイトに送信したりするために使用します。

デザイン階層を解析し、必要なソース ファイル、インクルード ファイル、リモート ファイルがをライブラリ ディレクトリからコピーし、制約ファイルをコピーし、さまざまな合成、シミュレーション、インプリメンテーション run の結果をコピーしてから、プロジェクトの ZIP ファイルを作成します。

引数

-force : 同じ名前の既存の ZIP ファイルを上書きします。ZIP ファイルが存在する場合に、**-force** が指定されていないと、エラー メッセージが表示されます。

-exclude_run_results : 合成またはインプリメンテーション run の結果を除外します。このコマンドにより、プロジェクト アーカイブのサイズを大幅に縮小できます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

file : **archive_project** コマンドで作成する ZIP ファイルの名前を指定します。*file* を指定しない場合、プロジェクトと同じ名前の ZIP ファイルが作成されます。

例

次の例では、現在のプロジェクトのアーカイブが作成されます。

```
archive_project
```

ファイル名が指定されていないので、プロジェクト アーカイブの名前は *project_name.zip* となります。

次の例は、現在のプロジェクトとして *project_3* を指定し、*proj3.zip* というプロジェクトのアーカイブを作成します。

```
current_project project_3  
archive_project -force -exclude_run_results proj3.zip
```

-force を使用しているので、*proj3.zip* ファイルが既に存在している場合は上書きされます。**-exclude_run_results** を使用しているので、合成 run またはインプリメンテーション run の結果はアーカイブに含まれません。プロジェクトで定義されたさまざまな run は含まれますが、その結果は含まれません。

関連項目

[current_project](#)

check_timing

発生する可能性のあるタイミング問題をチェックします。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-override_defaults</code>	オプション		timing_check_defaults 変数のチェックの代わりに check_list で指定したチェックを実行します。
<code>-include</code>	オプション		timing_check_defaults 変数のリストに指定したチェックを追加します。
<code>-exclude</code>	オプション		timing_check_defaults 変数のリストから指定したチェックを除外します。
<code>-verbose</code>	オプション		詳細情報を表示します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>check_list</code>	オプション	{unconstrained_endpoints multiple_clock no_clock no_input_delay no_output_delay loops generated_clocks}	実行されるチェックのリスト。有効な値は unconstrained_endpoints、multiple_clock、no_clock、no_output_delay、no_input_delay、loops、generated_clocks です。

カテゴリ

レポート

説明

ポート、ピン、およびパスのデザイン エレメントを現在のタイミング制約に対してチェックします。**report_timing** コマンドを実行する前にデザイン データおよびタイミング制約に問題がないかどうかを確認する場合に使用します。**check_timing** コマンドはデフォルトタイミング チェックを実行し、検出された違反のサマリをレポートします。違反に関する詳細を取得するには、**-verbose** オプションを使用します。

デフォルトのチェックは、次のとおりです。

- ・ **generated_clocks** : **create_generated_clock** コマンドで定義された派生クロックが、別の生成されたクロックを基準としているのではなく、**create_clock** コマンドで定義されたプライマリクロックを基準としていることをチェックします。
- ・ **loops** : デザインに組み合わせフィードバックループがないことをチェックします。
- ・ **multiple_clock** : レジスタクロックピンに複数のクロックが供給されていないことをチェックします。レジスタクロックピンに複数のクロックが供給されている場合、解析にどのクロックを使用するかが不明になります。その場合、**set_case_analysis** コマンドを使用してレジスタクロックピンに 1 つのクロックのみが伝播されるようにしてください。
- ・ **no_clock** : クロックが供給されていないレジスタをレポートします。レジスタにクロックが供給されていない場合、データピンでレジスタクロックピンに対するセットアップまたはホールドチェックが実行されません。
- ・ **no_input_delay** : 入力遅延制約が設定されていない入力ポートをレポートします。入力遅延は、**set_input_delay** コマンドを使用して指定できます。クロックに同期しない入力ポートでは、入力遅延はチェックされません。
- ・ **unconstrained_endpoints** : 制約が設定されていないタイミングパスエンドポイントをレポートします。エンドポイントがレジスタデータピンの場合、**create_clock** コマンドを使用して割り当てたクロックで制約されます。エンドポイントが出力ポートの場合、**set_output_delay** または **set_max_delay** コマンドを使用して割り当てた出力遅延で制約されます。

引数

-override_defaults {args} : デフォルトのタイミングチェックではなく、指定したタイミング制約チェックを実行します。上記に説明されているチェックから実行するチェックを指定します。

-include args : デフォルトチェックに加えて実行するチェックを実行します。上記に説明されているチェックから実行するチェックを指定します。

-exclude args : check_timing コマンドで実行されるチェックから指定のチェックを除外します。上記に説明されているチェックから除外するチェックを指定します。

-verbose : 指定したチェックの詳細な結果を表示します。このオプションを使用すると、さまざまなチェックで検出されたタイミング問題が発生する可能性のあるポートまたはパスの数だけでなく、ポートまたはパスの名前も表示されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドラインエラーは無視され、コマンドでエラーが発生した場合でもエラーメッセージは表示されません。

check_list : 実行するチェックの名前を指定します。上記に説明されているチェックから実行するチェックを指定します。

例

次の例では、デフォルトのタイミングチェックが実行され、検出された問題に関する詳細情報が表示されます。

```
check_timing -verbose
```

次の例では、デフォルトのタイミングチェックから指定のチェックが除外されて **check_timing** が実行されます。

```
check_timing -exclude {loops generated_clocks}
```

関連項目

[report_timing](#)

close_design

現在のデザインを閉じます。

構文

```
close_design [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

説明

PlanAhead で現在アクティブなデザインを閉じます。

デザインが変更されている場合でも、閉じる前にデザインを保存するかどうか尋ねるメッセージは表示されません。close_design コマンドを使用する前に、save_design または save_design_as コマンドを実行して変更を保存しておく必要があります。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、現在のプロジェクトが閉じられます。

```
close_design
```

PlanAhead で複数のデザインが開いている場合は、**close_design** コマンドを実行する前に **current_design** コマンドで現在のデザインを指定できます。

次の例では、現在のデザインを指定してから閉じています。

```
current_design rtl_1
close_design
```

rtl_1 デザインがアクティブ デザインとして指定され、**close_design** コマンドで閉じられます。

関連項目

- ・ [current_design](#)
- ・ [save_design](#)
- ・ [save_design_as](#)

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]
```

戻り値

なし

使用法

名 前	必須/ オプション	デフォルト	説明
-delete	オプション		プロジェクトをディスクからも削除します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

説明

PlanAhead で開いているプロジェクトを閉じます。

引数

-delete : プロジェクトを閉じた後、ハード ディスクからプロジェクト データを削除します。

注記：このオプションを使用する際には、注意が必要です。**close_project** コマンドに **-delete** オプションを使用した場合、確認メッセージは表示されません。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、アクティブ プロジェクトが閉じられます。

```
close_project
```

現在開いているプロジェクトが閉じられます。複数のプロジェクトが開いている場合は、現在のプロジェクトに対してのみ **close_project** コマンドが実行されます。現在のプロジェクトは、**current_project** コマンドで指定できます。

次の例では、project_1 をアクティブ プロジェクトとして指定してから閉じ、コンピューターのハード ディスクから削除します。

```
current_project project_1
close_project -delete
```

注記：コンピューターのハード ディスクからプロジェクト ファイルを削除する前に、確認メッセージは表示されません。**-delete** オプションを指定する前に、問題がないかどうかを確認してください。

関連項目

[current_project](#)

compplib

シミュレーション ライブラリをコンパイルします。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-arch	オプション	all	デバイス アーキテクチャを選択します。
-cfg	オプション	compplib.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	オプション		simulator:language:library:options の形式でコンフィギュレーション オプションを設定します。
-dir	オプション	.	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	オプション		以前に compplib でコンパイルしたライブラリが存在する既存ディレクトリを指定します。
-exclude_sublib	オプション		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	オプション		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)。
-force	オプション		コンパイル済みライブラリを上書きします。
-more	オプション		トピックの詳細なヘルプを表示します。
-info	オプション		コンパイル済みライブラリの情報を表示します。
-l	オプション	all	ライブラリをコンパイルする言語を指定します。
-lib	オプション	all	コンパイルするライブラリを選択します。
-log	オプション	compplib.log	ユーザー独自のログ ファイルを作成します。

名 前	必須/ オプション	デフォルト	説明
-p	オプション		指定したディレクトリからシミュレータの実行ファイルを使用します。
-s	オプション		指定したシミュレータ用にライブラリをコンパイルします。
-source_lib	オプション		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、指定したディレクトリでライブラリソース ファイルを検索します。
-verbose	オプション		詳細情報を表示します。
-w	オプション		コンパイル済みライブラリを上書きします。
-64bit	オプション		64 ビットのコンパイルを実行します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

ツール起動

config_partition

指定の run で使用するモジュール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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-cell	オプション		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。トップパーティションを変更する場合は、このオプションを指定しないでください。
-reconfig_module	オプション		指定の run の指定のインスタンスに適用するリコンフィギュラブル モジュールを指定します。
-import	オプション		指定の run で指定のインスタンス (またはスタティック ロジック) のアクションを import に設定します。
-implement	オプション		指定の run で指定のインスタンス (またはスタティック ロジック) のアクションを implement に設定します。
-import_dir	オプション		以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	オプション	routing	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、placement、synthesis です。
-quiet	オプション		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。

カテゴリ

パーシャル リコンフィギュレーション、パーティション

config_timing_analysis

タイミング解析の一般設定を指定します。

構文

```
config_timing_analysis [-disable_paths_between_unrelated_ucf_clocks arg]  
[-enable_input_delay_default_clock arg] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-disable_paths_between_unrelated_ucf_clocks</code>	オプション		関連のない UCF クロック間のタイミング パスをディスエーブルにします。有効な値は true、false です。このオプションは SDC 制約ではサポートされていません。
<code>-enable_input_delay_default_clock</code>	オプション		内部定義されたクロックからの SDC のクロックなし入力遅延をイネーブルにします。有効な値は true、false です。このオプションは UCF 制約ではサポートされていません。
<code>-quiet</code>	オプション		コマンド エラーを無視します。

カテゴリ

[XDC](#)

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] [-delay_type arg] [-setup]
[-hold] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-corner	オプション		変更するタイミング コーナーの名前を指定します。設定可能な値は、Slow、Fast です。
-delay_type	オプション		指定したタイミング コーナーを解析するパス遅延のタイプを指定します。有効な値は、none、max、min、min_max です。
-setup	オプション		セットアップ解析のタイミング コーナーをイネーブルにします (-delay_type max と同じ)。
-hold	オプション		ホールド解析のタイミング コーナーをイネーブルにします (-delay_type min と同じ)。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC

説明

シングル/マルチ コーナー タイミング解析でのスローおよびファースト タイミング コーナーを設定します。

注記：このコマンドを実行しても、その動作に関するメッセージや戻り値は返されません。

引数

-corner *value* (オプション)：設定するタイミング コーナーの名前を指定します。有効な値は、Slow または Fast です。

注記：この引数では大文字/小文字が区別されます。

-delay_type *value* (オプション)：指定したタイミング コーナーを解析するパス遅延のタイプを指定します。有効な値は max、min、および min_max です。

-setup (オプション)：指定のタイミング コーナーに対してセットアップ解析を指定します。これは **-delay_type max** と同じです。

-hold (オプション)：指定のタイミング コーナーに対してホールド解析を指定します。これは **-delay_type min** と同じです。

注記：**-setup** と **-hold** の両方を指定すると、**-delay_type min_max** と同じになります。

-quiet (オプション)：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、セットアップ解析およびホールド解析の両方でスロー タイミング コーナーを設定します。

```
config_timing_corners -corner Slow -setup -hold
config_timing_corners -corner Slow -delay_type min_max
```

注記：上記のどちらでも、同じ結果が得られます。

次の例では、最小遅延解析でファースト タイミング コーナーを設定します。

```
config_timing_corners -corner Fast -delay_type min
```

関連項目

- ・ [config_timing_analysis](#)
- ・ [config_timing_pessimism](#)

config_timing_pessimism

タイミング解析の共通ノードにおける不必要に悪い見積もり部分の削除を設定します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-enable	オプション		共通ノードにおける不必要に悪い見積もり部分の削除をイネーブルにします。
-disable	オプション		共通ノードにおける不必要に悪い見積もり部分の削除をディスエーブルにします。
-transition	オプション		指定した遷移から不必要に悪い見積もり部分を削除します。有効な値は、any_transition、same_transition です。
-common_node	オプション		タイミング ネットワークの共通ノードにおける不必要に悪い見積もり部分の削除を実行します。有効な値は、on、off です。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC

config_webtalk

ソフトウェア、IP、およびデバイスの使用統計をザイリンクスに送信する WebTalk をイネーブル/ディスエーブルにします。

構文

```
config_webtalk [-info] [-user arg] [-install arg] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-info	オプション		現在 WebTalk がイネーブルか ディスエーブルかを表示しま す。
-user	オプション	なし	現在のユーザーに対して WebTalk をイネーブル/ディス エーブルにします。イネーブル にする場合は on、ディスエー ブルにする場合は off に設定 します。
-install	オプション	なし	現在のインストールのすべての ユーザーに対して WebTalk を イネーブル/ディスエーブルに します。イネーブルにする場合 は on、ディスエーブルにする場 合は off に設定します。off に設 定した場合、個々のユーザー が -user オプションを使用して WebTalk をイネーブルにするこ とはできません。このオプション を使用するには、管理者権限 が必要な場合があります。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

ファイル入力および出力

connect_debug_port

ネットとピンをデバッグ ポート チャンネルに接続します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-channel_start_index</code>	オプション		チャンネル インデックスからネットを接続します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>port</code>	必須		デバッグ ポート名を指定します。
<code>nets</code>	必須		ネットまたはピンのリストを指定します。

カテゴリ

ChipScope

説明

ネットリスト デザインからの信号を ChipScope デバッグ コアのポートに接続します。信号は、ポートの特定のチャンネル インデックスに接続するか、ポートで使用可能なチャンネルに接続できます。

ポートに接続する信号が多すぎたり、接続をサポートするだけのチャンネルがない場合は、エラー メッセージが表示されます。

デバッグ コアにポートを追加するには `create_debug_port` コマンド、既存ポートで使用可能なチャンネルを増加するには `set_property port_width` コマンドを使用します。例を参照してください。

ポートから信号の接続を解除するには、`disconnect_debug_port` コマンドを使用します。

ChipScope デバッグ コアを定義して接続すると、デバッグ コアをブロックとしてインプリメントし、ネットリスト デザインに含めることができます。CORE Generator を使用してコアをインプリメントするには、`implement_debug_core` コマンドを使用します。

引数

-channel_start_index：接続に使用するチャネル インデックスを指定します。複数の信号を指定した場合、このオプションで指定したチャネル インデックス番号から接続が追加されます。チャネル インデックスの番号は 0 から開始します。

注記：このオプションを指定しない場合、最初に使用可能なチャネル インデックスに接続されます。

port：信号を接続するポートの名前を指定します。ポートは、core_name/port_name で指定する必要があります。

nets：指定したデバッグ ポートに接続する、ネットリスト デザインからのネット名のリストを指定します。

-quiet：コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、myCore デバッグ コアに新しい TRIG ポートを作成し、ポートの port_width を増加して、接続される信号数を受信できるようにし、信号を 3 番目のチャネル位置 (インデックス 2) からポートに接続します。

```
create_debug_port myCore TRIG
set_property port_width 8 [get_debug_ports myCore/TRIG0]
connect_debug_port myCore/TRIG0 [get_nets [list m0_ack_o m0_cyc_i m0_err_o m0_rty_o \
m0_stb_i m0_we_i ]] -channel_start_index 2
```

注記：ポートで使用可能なチャネルに接続するネット数が多すぎると、エラー メッセージが表示され、ポートは接続されません。

関連項目

- [create_debug_port](#)
- [disconnect_debug_port](#)
- [get_debug_ports](#)
- [get_nets](#)
- [implement_debug_core](#)
- [set_property](#)

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
```

戻り値

新しいデバッグ コア オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	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。
<i>type</i>	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。

カテゴリ

ChipScope

説明

現在のプロジェクトの開いているネットリスト デザインに追加する新しい ChipScope デバッグ コアを定義します。デバッグ コアでは、デバッグ目的でネットを接続するポートが定義されます。

デフォルトで作成されるコアには、CLK ポートとトリガー (TRIG) ポートが含まれます。CLK ポートでサポートされるクロック信号は 1 つだけなので、クロックドメインごとに別のデバッグ コアを作成する必要があります。

コアを作成したら、create_debug_port コマンドを使用してそのデバッグ コアに新しいポートを追加し、connect_debug_port コマンドを使用してそのポートに信号を接続できます。

注記：デバッグ コアは、PlanAhead で開いているネットリスト デザインにのみ追加できます。

引数

name : プロジェクトに追加する ChipScope デバッグ コアの名前を指定します。

type : 挿入する ChipScope デバッグ コアのタイプを指定します。現在 PlanAhead でサポートされるコアのタイプは、chipscope_ila_v1 のみです。ILA デバッグ コアは接続されたネットに別のロードを追加するだけで、その他の変更はありません。デバッグ コアのタイプおよび使用目的は、『ChipScope Pro ソフトウェアおよびコア ユーザー ガイド』(UG029) を参照してください。

注記：ILA をプロジェクトに追加すると、1 つまたは複数の ILA コアのコンテナーとして ICON コントローラー コアが 1 つ追加されます。ただし、ICON コアはプロジェクトに直接追加することはできません。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、ネットリスト デザインを開き、新しい ChipScope デバッグ コアを作成します。

```
open_netlist_design -name netlist_1  
create_debug_core myCore chipscope_ila_v1
```

注記：現在 PlanAhead でサポートされるコアのタイプは、chipscope_ila_v1 のみです。

次の例では、myCore という新しいデバッグ コアを作成し、そのコアのプロパティを返します。

```
report_property [create_debug_core myCore chipscope_ila_v1]
```

デバッグ コアのプロパティは、次の例のように set_property コマンドを使用するとカスタマイズできます。

```
set_property enable_storage_qualification false [get_debug_cores myCore]
```

関連項目

- [connect_debug_port](#)
- [delete_debug_core](#)
- [get_debug_cores](#)
- [implement_debug_core](#)
- [read_chipscope_cdc](#)
- [report_debug_core](#)
- [report_property](#)
- [set_property](#)
- [write_chipscope_cdc](#)

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
```

戻り値

新しいデバッグ ポート オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>name</code>	必須		デバッグ コア インスタンスの名前を指定します。
<code>type</code>	必須		デバッグ ポートのタイプを指定します。

カテゴリ

ChipScope

説明

既存の ChipScope デバッグ コアに追加する新しいポートを定義します。このポートは、デバッグ コアへの接続ポイントで、デバッグ目的でデザインのネットに接続するために使用されます。

`create_debug_core` コマンドを使用して新しいデバッグ コアを作成すると、デフォルトで CLK とトリガー (TRIG) ポートが含まれますが、DATA ポート、トリガー出力 (TRIG_OUT) ポート、追加の TRIG ポートを追加して、コアをデバッグすることもできます。

ポートに接続ポイントを複数設定し、デバッグ用に複数のネットをサポートできます。デフォルトでは、新しいポートの幅は 1 で定義されるので、接続できるネットは 1 つのみです。TRIG および DATA ポートの幅を変更して複数の信号をサポートできるようにするには、`set_property port_width` コマンドを使用します (例を参照)。

注記 : CLK および TRIG_OUT ポートの幅は 1 にしか設定できません。

`connect_debug_port` コマンドを使用すると信号をポートに接続でき、`disconnect_debug_port` コマンドを使用すると接続を解除できます。

引数

`name` : 新しいポートを接続する ChipScope デバッグ コアの名前を指定します。デバッグ コアは、`create_debug_port` で既に作成されているか、`read_chipscope_cdc` でインポートされ、既に存在している必要があります。

type: 挿入するデバッグ ポートのタイプを指定します。サポートされるポート タイプは、CLK、DATA、TRIG、および TRIG_OUT です。ポート タイプおよび使用目的は、『ChipScope Pro ソフトウェアおよびコア ユーザー ガイド』(UG029) を参照してください。

注記：各 ILA デバッグ コアには、CLK、DATA、TRIG_OUT ポートは 1 つずつしか含めることはできませんが、複数のトリガー (TRIG) ポートを作成することは可能です。

-quiet: コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、新しい ChipScope デバッグ コアを作成してから、そのコアに DATA ポートを追加します。

```
create_debug_core myCore chipscope_ila_v1
create_debug_port myCore DATA
```

次の例では、myCore デバッグ コアに新しいポートを作成し、そのポート幅を 8 に設定し、そのポートに信号を接続しています。

```
create_debug_port myCore TRIG
set_property PORT_WIDTH 8 [get_debug_ports myCore/TRIG0]
connect_debug_port -channel_start_index 1 myCore/TRIG0 {m1_cyc_i \
    m1_ack_o m1_err_o m1_rty_o}
```

注記：デバッグ コアは名前で、デバッグ ポートは `core_name/port_name` で参照されます。

関連項目

- [connect_debug_port](#)
- [create_debug_core](#)
- [disconnect_debug_port](#)
- [read_chipscope_cdc](#)
- [set_property](#)

create_fileset

新規ファイルセットを作成します。

構文

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

戻り値

新しいファイルセット オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-constrset</code>	オプション		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)。
<code>-simset</code>	オプション		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。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<i>name</i>	必須		作成するファイルセットの名前を指定します。

カテゴリ

プロジェクト

説明

PlanAhead プロジェクト内で新しいファイルセットを定義します。

ファイルセットは、プロジェクト内で特定のファンクションを持つファイルのコレクションです。1 つ以上の制約ファイルは制約ファイルセット (`-constrset`)、1 つ以上のシミュレーション テストベンチはシミュレーション ファイルセット (`-simset`) で指定できます。create_fileset コマンドを使用する際に指定できるファイルセット オプションは 1 つのみです。タイプを指定しない場合、デフォルトで制約ファイルセットが作成されます。

create_fileset コマンドを実行すると、新しく作成されたファイルセットの名前が返されるか、`-quiet` オプションが指定されていない場合はエラー メッセージが表示されます。

引数

-constrset : 1 つ以上の制約ファイルを含める制約ファイルセットを作成します。**-constrset** および **-simset** オプションのどちらも指定しない場合、デフォルトで制約ファイルセットが作成されます。

-simset : 1 つ以上のシミュレーション ソース ファイルを含めるシミュレーション ファイルセットを作成します。ファイルセットのタイプには、**-constrset** または **-simset** のいずれか 1 つしか設定できません。両方とも指定した場合は、エラー メッセージが表示されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

name : 作成するファイルセットの名前を指定します。

例

次の例では、constraints2 という名前の新しい制約ファイルセットを作成しています。

```
create_fileset -constrset -quiet constraints2
```

-quiet オプションが指定されているので、指定したファイルセットの作成中にエラーが発生しても、エラー メッセージは表示されません。

次の例では、sim_1 という名前の新しいシミュレーション ファイルセットを作成しています。

```
create_fileset -simset sim_1
```

ファイルは、**add_files** コマンドを使用して新しく作成したファイルセットに追加できます。

関連項目

[current_fileset](#)

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
```

戻り値

新しいインターフェイス オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
-parent	オプション		新しいインターフェイスを割り当てる親インターフェイスを指定します。
-quiet	オプション		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。

カテゴリ

[ピン配置](#)

create_ip

コンフィギュラブル 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]
[-vlnv arg] -module_name arg [-vendor arg] [-library arg]
[-name arg] [-version arg] [-quiet]
```

戻り値

追加されたファイル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-srcset	オプション		ソース セット名を指定します。
-vlnv	オプション		新しい IP の作成元となる IP カタログの VLNv 文字列を指定します。
-module_name	必須		プロジェクトに追加する新しい IP の名前を指定します。
-vendor	オプション		IP ベンダーの名前を指定します。
-library	オプション		IP ライブラリの名前を指定します。
-name	オプション		IP 名を指定します。
-version	オプション		IP のバージョン番号を指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

CORE Generator

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]
```

戻り値

新しい IP カタログ ファイルへのパス

使用法

名前	必須/ オプション	デフォルト	説明
-dir	必須		新しい IP カタログを保存するディレクトリを指定します。
-repositories	オプション		新しい IP カタログに含める IP レポジトリのリストを指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

[CORE Generator](#)

create_operating_conditions

ライブラリの新しい動作条件を作成します。

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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-name	必須		動作条件の名前を指定します。
-library	オプション		ライブラリ名を指定します。
-process	オプション	0.0	プロセス乗数を指定します。有効な値は 0 ～ 100 です。
-temperature	オプション	ファミリによって異なります。	周囲温度 (C) を指定します。有効な値は -55 ～ 125 です。
-voltage	オプション	0.0	電圧 (V) を指定します。有効な値は 0 ～ 1000 です。
-tree_type	オプション	balanced_tree	ツリー タイプを指定します。
-calc_mode	オプション	nominal	計算モードを指定します。有効な値は nominal または worst_case です。
-airflow	オプション	ファミリによって異なります。	エアフロー (LFM) を指定します。有効な値は 0 ～ 750 です。
-rail_voltages	オプション		レール電圧の名前と値の組み合わせリストを指定します。サポートされる値は vccint、vccaux、vcco33、vcco25、vcco18、vcco15、vcco12 です。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC

説明

このコマンドは廃止予定であり、今後のソフトウェア リリースで削除される予定です。代わりに set_operating_conditions を使用してください。

関連項目

[set_operating_conditions](#)

create_pblock

新規 Pblock を作成します。

構文

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

戻り値

新しい Pblock オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
-parent	オプション		新しい Pblock の親を指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	必須		新しい Pblock の名前を指定します。

カテゴリ

XDC、フロアプラン

説明

フロアプラン用にロジック インスタンスを追加する Pblock を定義します。Pblock を作成すると、制約ファイルに AREA_GROUP 制約が記述されます。

Pblock にロジック エlementを追加するには **add_cells_to_pblock** コマンド、Pblock を FPGA ファブリック上に配置するには **place_pblocks** コマンドを使用します。Pblock が自動的に配置された場合、**resize_pblocks** コマンドを使用して Pblock を手動で移動し、サイズを変更できます。

最初の例に示すように、**-parent** を使用して、階層フロアプラン用に 1 つの Pblock を別の Pblock にネストできます。また、2 番目の例に示すように、**set_property** コマンドを使用して PARENT プロパティを設定すると、既存の Pblock を別の Pblock の中にネストできます。

注記：ISE® インプリメンテーション ツールでは、この機能を多用することはできません。ネストされた Pblock が含まれていると、マップおよび配置エラーが発生することがあります。

引数

-parent arg：ネストされた Pblock を作成する親 Pblock の名前を指定します。親を指定しない場合、デフォルトで Root が親に指定され、Pblock がデザインの最上位に配置されます。**get_pblocks** コマンドを使用すると、現在定義されている Pblock で親として指定できるものをレポートできます。

注記：指定した親存在しない場合、エラー メッセージが表示されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

name : 作成する Pblock の名前を指定します。

例

次の例では、Vid_Array という Pblock の中に Video1 という Pblock を作成します。

```
create_pblock -parent Vid_Array Video1
```

次の例では、cpu1、cpu2、cpuEngine という 3 つの Pblock を作成した後、set_property コマンドを使用して cpuEngine の中に cpu1 および cpu2 をネストします。

```
create_pblock cpu1
create_pblock cpu2
create_pblock cpuEngine
set_property PARENT cpuEngine [get_pblocks {cpu1 cpu2}]
```

関連項目

- ・ [add_cells_to_pblock](#)
- ・ [get_pblocks](#)
- ・ [place_pblocks](#)
- ・ [resize_pblock](#)
- ・ [set_property](#)

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
```

戻り値

作成されたポート オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-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	オプション		新しいバスの開始インデックスを指定します。
-to	オプション		新しいバスの終了インデックスを指定します。
-diff_pair	オプション		ポートの差動ペアを作成します。
-interface	オプション		ポートを割り当てるインターフェイスを指定します。
-quiet	オプション		コマンド エラーを無視します。
name	必須		ポート名を指定します。

カテゴリ

ピン配置

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]
```

戻り値

新しいプロジェクト オブジェクト

使用法

名 前	必須/ オプション	デフォルト	説明
<code>-part</code>	オプション		ターゲット パーツを指定します。
<code>-force</code>	オプション		既存のプロジェクト ディレクトリを上書きします。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>name</code>	必須		プロジェクト名を指定します。
<code>dir</code>	オプション	.	プロジェクト ファイルを保存するディレクトリを指定します。

カテゴリ

プロジェクト

説明

指定したディレクトリに PlanAhead のプロジェクト ファイルを作成します。

引数

name : この引数はパラメーター名を必要としませんが、*dir* よりも前に記述する必要があります。パラメーターがないので、最初の引数は *name*、2 つ目の引数は *dir* として認識されます。プロジェクト ファイル *name.ppr* およびプロジェクト データ フォルダ *name.data* が作成され、両方とも指定したディレクトリ *dir* に書き込まれます。

PlanAhead で作成されるプロジェクト ファイルは、デフォルトでは RTL ソース ファイルです。set_property コマンドを使用して design_mode プロパティを設定し、RTL ソース プロジェクトから、たとえば I/O ピン配置などの別のプロジェクト タイプに変更する必要があります。

dir : 新しいプロジェクト ファイルを保存するディレクトリ名を指定します。指定したディレクトリが存在しない場合は、その名前の新しいディレクトリが作成されます。ディレクトリを完全なパスで指定すると、その指定したパス名が使用されます。*dir* をパスなしで指定すると、現在の作業ディレクトリまたは PlanAhead の起動ディレクトリでそのディレクトリが検索されるか、作成されます。

注記：プロジェクトを GUI モードで作成すると、ディレクトリ名 *dir* にファイル名 *name* を付けた *dir/name* という名前のプロジェクト ディレクトリが作成され、新しいプロジェクト ファイルとプロジェクト データ フォルダーが保存されます。

-part partName：プロジェクトで使用するザイリンクス パーツを指定します。これは、プロジェクトの作成後に変更できます。**-part** オプションを指定しない場合、デフォルトのパーツが使用されます。現在のところ、デフォルトのパーツは xc6vlx75tf484-1 です。

-force：既存のプロジェクトを上書きします。指定した *dir* に指定のプロジェクト名が既に存在する場合、**-force** オプションを使用して既存のプロジェクトを上書きする必要があります。

注記：既存プロジェクトを現在 PlanAhead で開いている場合、新しいプロジェクトでディスクの既存プロジェクトが上書きされますが、両方のプロジェクトが PlanAhead で開いたままになります。この場合、**create_project** を実行する前に **close_project** コマンドを実行しておくことをお勧めします。

-quiet：コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、dum というディレクトリに dee というプロジェクトを作成しています。

```
create_project dee dum
```

注記：*dir* にはフォルダー名のみが指定されているので、プロジェクトは現在の作業ディレクトリか PlanAhead の起動ディレクトリに作成されます。

次の例では、C:/Designs の dum というディレクトリに dee というプロジェクトを作成しています。指定したディレクトリに同名のプロジェクトが既に存在する場合は、それが上書きされます。2 行目と 3 行目では、**-force** の位置を変更しても問題ないことを示しています。

```
create_project dee C:/Designs/dum -force
-or-
create_project dee -force C:/Designs/dum
-or-
create_project -force dee C:/Designs/dum
```

注記：どの例でも、キーワードのない最初の引数は *name* 変数として、2 つ目の引数は *dir* 変数として認識されます。

次の例では、pin_project という新規プロジェクトを作成し、**design_mode** プロパティを I/O ピン配置プロジェクトに設定し、最後に I/O ピン配置デザインを開いています。

```
create_project pin_project C:/Designs/PinPlanning
set_property design_mode PinPlanning [current_fileset]
open_io_design -name io_1
```

関連項目

- [current_project](#)
- [set_property](#)
- [open_io_design](#)

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
```

戻り値

作成されたプロパティ、エラーが発生した場合は ""

使用法

名前	必須/ オプション	デフォルト	説明
-type	オプション	string	作成するプロパティのタイプを指定します。有効な値は string、int、double、bool です。
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	必須		作成するプロパティの名前を指定します。
<i>class</i>	必須		プロパティを作成するオブジェクトタイプを指定します。有効な値は、design、net、cell、pin、port、pblock です。

カテゴリ

プロパティおよびパラメーター

説明

指定したオブジェクトのクラスに対し、指定したタイプのプロパティを指定した名前で作成します。作成したプロパティは、**set_property** コマンドで指定したクラスのオブジェクトに割り当てることができますが、そのクラスのすべてのオブジェクトに自動的に関連付けられません。

プロパティをそのオブジェクトに割り当てないと、**report_property -all** コマンドを使用しても、指定したクラスのオブジェクトに対して新しく作成したプロパティはレポートされません。

引数

-type arg: 作成するプロパティのタイプを指定します。次のプロパティタイプを指定できます。

- ・ **string**: 新しいプロパティを文字列値で定義します。**-type** を使用しない場合、これがデフォルトです。
- ・ **int**: 新しいプロパティを長整数値で定義します。**int** プロパティタイプに 10 進数値を指定すると、エラー メッセージが表示されます。
- ・ **double**: 新しいプロパティを浮動小数点値で定義します。
- ・ **bool**: 新しいプロパティをブール値 (1 = True、0 = False) で定義します。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

name : 定義するプロパティの名前を指定します。大文字/小文字が区別されます。

class : 新しいプロパティを割り当てるオブジェクトのクラスを指定します。指定したクラスのオブジェクトはすべて、新しく定義したプロパティに割り当てられます。有効な値は、design、net、cell、pin、port、pblock です。

例

次の例では、セル オブジェクトに対して PURPOSE というプロパティを定義します。

```
create_property PURPOSE cell
```

注記 : **-type** は指定されていないので、値は文字列になります。

次の例では、整数値を指定する COUNT というピン プロパティを作成します。

```
create_property -type int COUNT pin
```

関連項目

- [get_property](#)
- [list_property](#)
- [list_property_value](#)
- [report_property](#)
- [reset_property](#)
- [set_property](#)

create_reconfig_module

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

構文

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

戻り値

新しく作成されたリコンフィギャラブル モジュール オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
-force	オプション		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	オプション		ブラック ボックス リコンフィギャラブル 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 リコンフィギャラブル モジュールには追加されない可能性があります。
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	必須		新しいリコンフィギャラブル モジュールの名前を指定します。
<i>cell</i>	必須		新しいリコンフィギャラブル モジュールを追加するセルを指定します。

カテゴリ

パーシャル リコンフィギュレーション

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 を作成します。

構文

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

戻り値

run オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-constrset</code>	オプション		使用する制約ファイルセットを指定します。
<code>-parent_run</code>	オプション		新しいインプリメンテーション run に関連付ける合成 run を指定します。
<code>-part</code>	オプション		ターゲット パーツを指定します。
<code>-flow</code>	必須		フロー名を指定します。
<code>-strategy</code>	オプション		run に適用するストラテジを指定します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>name</code>	必須		新規 run の名前を指定します。

カテゴリ

プロジェクト

説明

PlanAhead で使用する合成またはインプリメンテーション run を作成します。run は、config_run コマンドを使用して設定できます。

引数

`-srcset arg`: run で使用するソース ファイルセットを指定します。デフォルトは sources_1 です。

`-constrset arg`: 合成またはインプリメンテーション run で使用する制約ファイルセットを指定します。

`-parent_run arg`: RTL ソース プロジェクトの場合、インプリメンテーション run に parent_run を指定する必要があります。合成 run には必要ありません。parent_run は、インプリメンテーション run でどの合成 run をインプリメントするかを指定します。ネットリスト ベースのプロジェクトの場合、インプリメンテーション run を定義するのに parent_run 引数は必要ありません。

-part *partName* : run に使用するザイリンクス パーツを指定します。**-part** オプションを指定しない場合は、プロジェクトに定義されているデフォルト パーツが使用されます。

-flow *arg* : 合成ツール ({XST 13} または {RDS 13}) またはインプリメンテーション ツール ({ISE 13} または {RDI 13}) のツール フローおよびリリース バージョンを指定します。

-strategy *arg* - 合成またはインプリメンテーション run で使用するストラテジを指定します。PlanAhead には、ユーザー定義のカスタム ストラテジも含め、多くのストラテジが含まれます。使用可能な合成およびインプリメンテーション ストラテジは、『PlanAhead ユーザー ガイド』(UG632) を参照してください。ストラテジを指定しない場合、[Synthesis Defaults] または [Implementation Defaults] が使用されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

name : 設定する合成 run またはインプリメンテーション run の名前を指定します。

例

次の例では、XST 13 フローを使用する first_pass という名前の run を作成しています。

```
create_run -flow {XST 13} first_pass
```

この場合、合成 run でデフォルトの sources_1、constrs_1、プロジェクトのデフォルト パーツが使用されます。また、これは合成 run なので、parent_run 引数は必要ありません。

次の例では、ISE 13 ツール フローを使用するインプリメンテーション run を作成し、先ほどの例で作成した first_pass という合成 run に関連付けています。

```
create_run -flow {ISE 13} -parent_run first_pass imp_1
```

この場合、プロジェクトが RTL ソース ファイルであり、定義されるのがインプリメンテーション run なので、parent_run 引数を使用する必要があります。

関連項目

- [current_run](#)
- [launch_runs](#)

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] [-name arg] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-to	オプション		範囲の終わりのクロックを指定します。
-delay_type	オプション	max	パス遅延のタイプを指定します。有効な値は、max、min、min_max です。
-num_bins	オプション	10	ビンの最大数を指定します。
-slack_less_than	オプション	1e+30	この値よりも小さいスラックのパスを表示します。
-slack_greater_than	オプション	-1e+30	この値よりも大きいスラックのパスを表示します。
-group	オプション		指定のグループのパスのみをレポートします。
-report_unconstrained	オプション		制約の設定されていないエンドポイントをレポートします。
-significant_digits	オプション	3	表示する桁数を指定します。範囲は、0 ～ 13 です。
-scale	オプション	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 または logarithmic です。
-name	オプション		GUI パネルに表示する結果の名前を指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-run	オプション		FPGA Editor で開くインプリメント済み run を指定します。
-path	オプション		プリミティブのセルとネットを接続するパスを指定します。
-objects	オプション		クロスプローブするセルおよびネットのリストを指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

ツール起動

説明

PlanAhead から **launch_fpga_editor** コマンドで開いた FPGA Editor にクロスプローブします。クロスプローブするオブジェクトおよびタイミング パスの両方を選択できます。

引数

-run arg : クロスプローブに使用する run の名前を指定します。

-path args : FPGA Editor でクロスプローブする 1 つ以上のパスを指定します。

-objects args : FPGA Editor でクロスプローブする 1 つ以上のオブジェクトを指定します。**get_cells**、**get_ports** など、**get_*** コマンドのいずれかを使用して、FPGA Editor にクロスプローブするオブジェクトを選択できます。例を参照してください。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、**get_cells** コマンドを使用してプリミティブ セルを各階層で検索し、FPGA Editor にクロスプローブするオブジェクトのグループを定義しています。

```
crossprobe_fed -run impl_1 -objects [get_cells -hier -filter {IS_PRIMITIVE==1}]
```

次の例では、FPGA Editor にクロスプローブするパスを定義しています。

```
crossprobe_fed -path {wbClk i_2090 wbClk_IBUF \  
i_2089 n_0_2089 egressLoop[4].egressFifo/buffer_fifo/infer_fifo.empty_reg_reg}
```

関連項目

- [get_cells](#)
- [get_ports](#)
- [launch_fpga_editor](#)

current_design

現在のデザインを設定または取得します。

構文

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

戻り値

デザイン オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>design</code>	オプション		設定する現在のデザインの名前を指定します。

カテゴリ

SDC、XDC

説明

現在のデザインを定義するか、アクティブ プロジェクトの現在のデザインの名前を返します。

ほとんどの Tcl コマンド、PlanAhead で実行したデザインのおよび制約の変更では、現在のデザインおよび現在のインスタンスがターゲットとなります。現在のインスタンスを指定するには、**current_instance** コマンドを使用します。

get_designs コマンドを使用するとアクティブ プロジェクトで開いているデザインのリストを取得でき、**get_projects** コマンドを使用すると開いているプロジェクトのリストを取得できます。

注記：このコマンドを実行すると、現在のデザイン オブジェクトが返されます。

引数

-quiet (オプション)：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

design (オプション)：現在のデザインとして設定するデザインの名前を指定します。**design** を指定しない場合、アクティブ プロジェクトの現在のデザインが返されます。

例

次の例では、指定した RTL デザインが現在のデザインに設定されます。

```
current_design rtl_1
```

関連項目

- ・ [current_instance](#)
- ・ [get_designs](#)
- ・ [get_projects](#)

current_fileset

現在のファイルセットを取得します。

構文

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

戻り値

現在のファイルセット (デフォルトでは現在のソース ファイルセット)

使用法

名前	必須/ オプション	デフォルト	説明
-constrset	オプション		現在の制約ファイルセットを取得します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

説明

PlanAhead プロジェクト内でアクティブな制約ファイルセットの名前を取得します。

このコマンドを実行すると、現在アクティブなソース ファイルセットまたは制約ファイルセットの名前が返されます。

引数

-constrset : 現在アクティブな制約ファイルセットの名前を返します。このオプションを使用しない場合、デフォルトでアクティブなソース ファイルセットが返されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、アクティブな制約ファイルセット名が返されます。

```
current_fileset -constrset -quiet
```

-quiet オプションが指定されているので、コマンドの実行中にエラーが発生しても、エラー メッセージは表示されません。

current_instance

現在のインスタンスを設定または取得します。

構文

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

戻り値

インスタンス名

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
instance	オプション		インスタンス名を指定します。

カテゴリ

SDC、XDC

説明

デザイン階層の現在のインスタンスを指定のインスタンス セルまたは最上位セルに設定します。現在のインスタンスの名前が返されます。

PlanAhead で実行したほとんどのコマンドおよびデザインの変更では、現在のデザインおよび現在のインスタンスがターゲットとなります。現在のデザインを指定するには、**current_design** コマンドを使用します。

instance は現在定義されている現在のインスタンスを基準に指定し、階層区切り文字を使用してインスタンス パスを定義します。インスタンス パスの 1 つ上の階層に移動するには、「..」を使用します。

注記：このコマンドを実行すると、インスタンス オブジェクトが返されます。

引数

-quiet (オプション)：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

instance (オプション)：現在のデザインの現在のインスタンスとして設定するインスタンスの名前を指定します。*instance* を指定しない場合、現在のインスタンスがデザイン階層の最上位セルにリセットされます。*instance* に「.」を指定した場合、現在のインスタンスとして定義されているインスタンスの名前が返され、インスタンスは変更されません。

注記：階層インスタンス パスは、現在定義されている階層区切り文字を使用して指定する必要があります。現在定義されている階層区切り文字は、**get_hierarchy_separator** コマンドを使用して確認できます。

例

次の例では、現在のインスタンスを現在のデザインの最上位セルに設定します。

```
current_instance  
INFO: [PlanAhead-618] Current instance is the top level of design 'rtl_1'.
```

次の例では、まず階層区切り文字を設定し、その後現在定義されている現在のインスタンスを基準に現在のインスタンスを設定します。

```
set_hierarchy_separator |  
current_instance ..|cpu_iwb_dat_o|buffer_fifo
```

次の例では、現在定義されている現在のインスタンスの名前が返されます。

```
current_instance .  
cpuEngine|cpu_iwb_dat_o|buffer_fifo
```

関連項目

- ・ [current_design](#)
- ・ [get_hierarchy_separator](#)
- ・ [set_hierarchy_separator](#)

current_project

現在のプロジェクトを設定または取得します。

構文

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

戻り値

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

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
project	オプション		現在のプロジェクトとして設定するプロジェクトの名前を指定します。

カテゴリ

プロジェクト

説明

現在のプロジェクトを指定するか、プロジェクトを指定しない場合は現在のプロジェクトの名前を返します。

引数

project : 現在のプロジェクトとして設定する PlanAhead プロジェクトの名前を指定します。このコマンドを close_project コマンドの前に使用し、特定のプロジェクトをアクティブにしてから、プロジェクトを閉じます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、project_2 を現在のプロジェクトとして設定しています。

```
current_project project_2
```

このコマンドにより、現在のプロジェクトがすべての PlanAhead コマンドの対象となります。GUI モードでは、プロジェクト間の GUI を切り替えると current_project が自動的に定義されます。

次の例では、PlanAhead の現在のプロジェクト名が返されます。

```
current_project
```

戻り値は、プロジェクト ファイルの名前やパスではなく、PlanAhead のプロジェクト名です。

current_run

現在の run を設定または取得します。

構文

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

戻り値

run オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-synthesis</code>	オプション		現在の合成 run を設定または取得します。
<code>-implementation</code>	オプション		現在のインプリメンテーション run を設定または取得します。 -synthesis を指定しない場合、これがデフォルト設定です。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>run</code>	オプション		現在の run として設定する run を指定します。

カテゴリ

プロジェクト

説明

現在の合成またはインプリメンテーション run を定義するか、現在の run の名前を返します。現在の run とは、合成またはインプリメント コマンドを実行したときに自動的に選択される run のことです。

`get_runs` コマンドを使用すると、現在のデザインで定義されている run のリストを取得できます。

引数

`-synthesis`：現在の合成 run を設定または返します。

`-implementation`：現在のインプリメンテーション run を設定または返します。

`-quiet`：コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

`run`：現在の run として設定する合成 run またはインプリメンテーション run の名前を指定します。

例

次の例では、first_pass という run を現在の run として定義しています。

```
current_run first_pass
```

この場合、run の名前が指定されており、PlanAhead で特定の run 名が認識されるので、`-synthesis` および `-implementation` オプションは必要ありません。

次のコマンドでは、現在のインプリメンテーション run の名前が返されます。

```
current_run -implementation -quiet
```

関連項目

[get_runs](#)

data2mem

データをメモリに変換します。

構文

```
data2mem [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

ツール起動

delete_clocknetworks_results

メモリのクロック ネットワーク結果のセットを消去します。

構文

```
delete_clocknetworks_results [-quiet] name
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	必須		削除する結果のセット名を指定します。

カテゴリ

[レポート](#)

delete_debug_core

ChipScope デバッグ コアを削除します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>cores</code>	必須		削除する ChipScope デバッグ コアを指定します。

カテゴリ

ChipScope

説明

現在のプロジェクトから ChipScope デバッグ コアを削除します。デバッグ コアは、`create_debug_core` コマンドで追加されるか、`read_chipscope_cdc` コマンドでインポートされています。どちらの場合も、コアは現在のプロジェクトから削除されます。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

cores : 現在のプロジェクトから削除する ChipScope デバッグ コア名をリストします。

例

次の例では、現在のプロジェクトから `myCore` デバッグ コアを削除しています。

```
delete_debug_core myCore
```

次の例では、現在のプロジェクトからすべてのデバッグ コアを削除しています。

```
delete_debug_core [get_debug_cores]
```

注記 : `get_debug_cores` を実行すると、すべてのコアがデフォルトで返されます。

関連項目

- ・ [create_debug_core](#)
- ・ [get_debug_cores](#)
- ・ [read_chipscope_cdc](#)

delete_debug_port

ChipScope デバッグ ポートを削除します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>ports</code>	必須		削除する ChipScope デバッグ ポートを指定します。

カテゴリ

ChipScope

説明

現在のプロジェクトに含まれる ChipScope デバッグ コアからポートを削除します。デバッグ ポートからの信号は、`disconnect_debug_port` で削除するか、このコマンドでポートと共に削除できます。

引数

`-quiet` : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

`ports` : コアから削除するデバッグ ポートを `core_name/port_name` の形式で指定します。

例

次の例では、`myCore` デバッグ コアから DATA ポートを削除しています。

```
delete_debug_port myCore/DATA
```

ILA ポートには、最低 1 つの CLK ポートと 1 つの TRIG ポートが必要なので、削除できないポートもあります。

次の例では、`myCore` デバッグ コアからトリガー ポート (TRIG) が削除されます。

```
delete_debug_port [get_debug_ports myCore/TRIG*]
```

注記：この例の場合、ILA コアには少なくとも 1 つの TRIG ポートが必要なので、myCore からすべての TRIG ポートが削除されるわけではありません。このコマンドでは、最後のポートを除き、TRIG0 からすべての TRIG ポートが削除されます。

関連項目

- ・ [disconnect_debug_port](#)
- ・ [get_debug_ports](#)

delete_fileset

ファイルセットを削除します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>fileset</code>	必須		削除するファイルセットを指定します。

カテゴリ

プロジェクト

説明

指定したファイルセットを削除します。ただし、ファイルセットが削除できない場合でも、それを示すメッセージは表示されません。

引数

`-quiet` : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

`fileset` : 削除するファイルセットの名前を指定します。可能であればファイルセットは削除されますが、最後の制約ファイルセットまたはシミュレーション ファイルセットは削除されません。

例

次の例では、sim_2 というファイルセットを現在のプロジェクトから削除しています。

```
delete_fileset sim_2
```

ファイルセットとそのファイルすべてがプロジェクトから削除されます。ファイルは、ハードドライブからは削除されません。

関連項目

- [create_fileset](#)
- [current_fileset](#)

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 ...
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-all	オプション		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	オプション		コマンド エラーを無視します。
<i>interfaces</i>	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。

カテゴリ

[ピン配置](#)

delete_pblock

Pblock を削除します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-hier	オプション		Pblock の子ブロックすべても削除します。
-quiet	オプション		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。

カテゴリ

フロアプラン

説明

現在のプロジェクトから指定した Pblock を削除します。Pblock は、**create_pblock** コマンドを使用して作成します。

引数

-hier : 指定した Pblock にネストされている Pblock も削除します。**-hier** オプションを指定せずに親 Pblock を削除した場合、ネストされていた Pblock は 1 つ上のレベルに移動されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

pblocks : 削除する Pblock を 1 つ以上指定します。

例

次の例では、指定した Pblock と、その中にネストされている Pblock が削除されます。

```
delete_pblock -hier cpuEngine
```

関連項目

[create_pblock](#)

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 ...
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>ports</i>	必須		削除するポートを指定します。

カテゴリ

[ピン配置](#)

delete_power_results

指定した消費電力予測結果を削除します。

構文

```
delete_power_results -name arg [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-name	必須		削除する結果の名前を指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC

delete_reconfig_module

リコンフィギャラブル モジュールを削除します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<i>reconfig_module</i>	必須		削除するリコンフィギャラブル モジュールを指定します。

カテゴリ

パーシャル リコンフィギュレーション

delete_rpm

RPM を削除します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		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。

カテゴリ

フロアプラン

説明

デザインから指定した相対配置マクロ (RPM) を削除します。

RPM とは、ロジック エLEMENT (FFS、LUT、CY4、RAM など) を 1 つのセット (U_SET、H_SET、および HU_SET) にまとめたものです。セット内での各ELEMENTの配置は、相対ロケーション 制約 (RLOC) により同じセット内のほかのELEMENTに相対して設定されます。RLOC 制約の 設定されたロジック ELEMENTと共通セット名は、RPM 内で関連付けられます。これらの制約 の定義方法は、『制約ガイド』(UG625) を参照してください。

デザインから削除できるのは、ユーザー定義の RPM のみです。階層またはネットリストにより 定義された RPM をこのコマンドで削除することはできません。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コ マンドでエラーが発生した場合でもエラー メッセージは表示されません。

rpm : 削除する RPM を指定します。

例

次の例では、デザインから指定した RPM を削除しています。

```
delete_rpm cs_ila_0/U0
```

delete_run

既存の run を削除します。

構文

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

戻り値

なし

使用法

名 前	必須/ オプション	デフォルト	説明
<code>-noclean_dir</code>	オプション		すべての出力ファイルおよびディレクトリをディスクからは削除しません。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>run</code>	必須		削除する run を指定します。

カテゴリ

プロジェクト

説明

プロジェクトから run を削除し、指定しない場合は、ハードドライブのプロジェクト ディレクトリからも run の結果を削除します。

引数

-noclean_dir : run 結果をハードドライブから削除しないように指定します。run はプロジェクトからは削除されますが、run ファイルはプロジェクト ディレクトリに残ったままになります。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

run : プロジェクトから削除する合成 run またはインプリメンテーション run の名前を指定します。

例

次の例では、first_pass という run を プロジェクトから削除しています。

```
delete_run first_pass
```

この例の場合、run 結果もハードドライブのプロジェクト ディレクトリから削除されます。

次の例では、first_pass という run は削除されますが、run 結果はハードドライブに残ります。

```
delete_run -nocleandir first_pass
```

関連項目

- ・ [create_run](#)
- ・ [current_run](#)

delete_timing_results

メモリからタイミング結果のセットを消去します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-type	オプション		削除するタイミング結果のタイプを指定します。有効な値は、 <code>timing_path</code> 、 <code>slack_histogram</code> 、 <code>clock_interaction</code> です。
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	必須		削除する結果の名前を指定します。

カテゴリ

[レポート](#)、[XDC](#)

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、インポートに使用されないようにします。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-run	オプション		プロモートを解除する run を指定します。
-partition_names	オプション		プロモートを解除するパーティションのリストを指定します。
-promote_dir	オプション		プロモートを解除するディレクトリを指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

パーシャル リコンフィギュレーション、パーティション

disconnect_debug_port

デバッグ ポート チャンネルから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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-channel_index</code>	オプション		ネット接続を解除するチャンネルインデックスを指定します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>port</code>	必須		デバッグ ポート名を指定します。

カテゴリ

ChipScope

説明

ネットリスト デザインからの信号は、connect_debug_port コマンドを使用して ChipScope デバッグ コアのポートに接続されます。この disconnect_debug_port コマンドを使用すると、ポートからの信号の接続を解除できます。

ポートは単に接続を解除するだけでなく、delete_debug_port コマンドを使用してデバッグ コアから削除することもできます。

デバッグ コアのポート名を取得する必要がある場合は、get_debug_ports コマンドを使用するとコアのポートすべてをリストできます。プロジェクト内のコアと特定のパラメーターをすべてリストするには、report_debug_core コマンドを使用します。

引数

-channel_index *value* : 接続を解除するポートのチャンネル インデックスを指定します。

注記 : -channel_index を指定しない場合、ポート全体の接続が解除されます。

port : 接続を解除するデバッグ コアのポート名を指定します。ポート名は、core_name/port_name という形式で指定する必要があります。例を参照してください。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、myCore の TRIG0 ポートから指定したチャンネル インデックスのみの接続を解除しています。

```
disconnect_debug_port -channel_index 2 myCore/TRIG0
```

次の例のように -channel_index を指定しない場合、指定したポートのすべてのチャンネルの接続が解除されます。

```
disconnect_debug_port myCore/TRIG0
```

関連項目

- ・ [connect_debug_port](#)
- ・ [delete_debug_port](#)
- ・ [get_debug_ports](#)
- ・ [report_debug_core](#)

endgroup

グループ単位で実行を取り消し/やり直しできるコマンドシーケンスを終了します。

構文

```
endgroup [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

GUI 制御

説明

グループ単位で実行を取り消し/やり直しできるコマンドシーケンスを終了します。**startgroup** コマンドを使用してコマンドシーケンスを開始した後に、このコマンドを使用して終了します。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、まず **startgroup** を実行し、関連するコマンドのシーケンスを実行して、**endgroup** を実行します。このコマンドシーケンスは、グループ単位で実行を取り消すことができます。

```
startgroup
create_pblock pblock_wbArbEngine
create_pblock pblock_usbEng
add_cells_to_pblock pblock_wbArbEngine [get_cells [list wbArbEngine]] -clear_locs
add_cells_to_pblock pblock_usbEng [get_cells [list usbEngine1/usbEngineSRAM]] -clear_locs
endgroup
```

関連項目

- [redo](#)
- [startgroup](#)

filter

リストにフィルターを適用し、新しいリストを作成します。

構文

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

戻り値

新しいリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regex	オプション		=~ および !~ 演算子で正規表現を使用します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regex を指定した場合のみ有効)。
-quiet	オプション		コマンド エラーを無視します。
<i>objects</i>	オプション		フィルターを適用するオブジェクトのリストを指定します。
<i>filter</i>	オプション		式を使用してリストをフィルター処理します。

カテゴリ

オブジェクト、プロパティおよびパラメーター、XDC

説明

オブジェクトのリストを指定して、指定したフィルター検索パターンに一致するオブジェクトのリストを返します。デフォルトでは、開いているプロジェクトすべてのリストが返されます。

引数

-regex : フィルター パターンを正規表現で指定することを示します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、-regex を使用した場合にのみ適用されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

objects : フィルターを適用するオブジェクトのリストを指定します。オブジェクトのリストを指定するには、get_parts などの **get_*** コマンドのいずれかを使用できます。

filter: 指定した式でオブジェクトにフィルターを適用します。指定したパターンにより、オブジェクトのプロパティ値に基づいたオブジェクトのリストがフィルター処理されます。オブジェクトに設定されているプロパティは、`report_property` または `list_property` コマンドで確認できます。フィルターには、任意のプロパティと値の組み合わせを使用できます。part オブジェクトの場合、結果をフィルター処理できるプロパティには DEVICE、FAMILY、SPEED などがあります。

フィルター式に使用できる演算子は `==`、`!=`、`=~` で、フィルター式の間に `&&` および `||` も使用できます。

例

次の例では、指定したスピード グレードでパーツのリストがフィルター処理されます。

```
filter [get_parts] {speed == -3}
filter [get_parts] {speed == -3 || speed == -2}
```

2 つ目のコマンド例では、スピード グレード -3 または -2 のパーツが返されます。

関連項目

[get_parts](#)

find_top

供給されているファイル、ファイルセット、またはアクティブ ファイルセットからトップ モジュールの候補を検索します。ランク付けされた候補のリストを返します。

構文

```
find_top [-fileset arg] [-files args] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-fileset	オプション		トップ モジュールの候補を検索するファイルセットを指定します。
-files	オプション		トップ モジュールの候補を検索するファイルを指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

generate_ip

コンフィギャラブル IP を生成します。

構文

```
generate_ip [-srcset arg] [-output_products args] [-ips args]  
[-quiet]
```

戻り値

生成されたファイルのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-srcset	オプション		ソース セット名を指定します。
-output_products	オプション		生成するファイルを指定します。
-ips	オプション		生成する IP を指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

[CORE Generator](#)

get_cells

現在のデザインのセルのリストを取得します。

構文

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

戻り値

セル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-hsc	オプション	/	階層区切り文字を指定します。
-hierarchical	オプション		すべての階層レベルで検索します。
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-of_objects	オプション		セルを取得するピン、タイミング パス、またはネットを指定します。
-match_style	オプション	sdc	パターン一致のスタイルを指定します。有効な値は ucf、sdc です。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	セルを検索するパターンを指定します。

カテゴリ

SDC、XDC、オブジェクト

説明

現在のデザインに含まれるセルで、検索パターンに一致するものをリストします。デフォルトでは、デザインのすべてのセルのリストが返されます。

引数

-hsc *arg*: デフォルトの階層区切り文字は / です。それ以外の階層区切り文字を指定する場合は、このオプションを使用します。

-hierarchical: デザイン階層のすべてのレベルからセルを取得します。このオプションを指定しない場合、デザイン階層の最上位のセルのみが返されます。

-regexp: 検索パターンを正規表現で指定します。

-nocase: パターンの大文字/小文字を区別せずに検索します。このオプションは、**-regexp** を使用した場合にのみ適用されます。

-filter *args*: 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、`get_cells` で返されたオブジェクトのリストに、セルのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、`report_property` または `list_property` コマンドで確認できます。cell オブジェクトの場合、結果をフィルター処理できるプロパティには `IS_PARTITION`、`IS_PRIMITIVE`、`IS_LOC_FIXED` などがあります。

フィルター パターンに使用できる演算子は `==`、`!=`、`=~` で、フィルター式の間に `&&` および `||` も使用できます。

-of_objects *arg*: 指定したピンまたはネット オブジェクトに接続されているセルを取得します。

-match_style [*sdc* | *ucf*]: 検索パターンが UCF 制約または SDC 制約に一致することを示します。デフォルトは SDC です。

-quiet: コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns: 指定したパターンと一致するセルを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべてのセルが返されます。複数のパターンを指定して、異なる検索条件に基づいてセルを検索できます。

注記: 複数の検索パターンは {} で囲み、1 つのエレメントとして指定します。

例

次の例では、特定のセル オブジェクトに設定されているプロパティとその値がリストされます。

```
report_property [lindex [get_cells] 1]
```

注記: パターンに一致するセルがない場合は、警告メッセージが表示されます。

次の例では、デザインの階層の全レベルにインスタンス化されているライブラリ セルのリストが表示されます。リストは名前ごとに並べられるので、それぞれのセルが表示されるのは 1 度だけです。

```
foreach cell [lsort -unique [get_property LIB_CELL [get_cells -hier -filter \
{IS_PRIMITIVE==1}]]] [[SDLENTITYREF[nbsp]]] [[SDLENTITYREF[nbsp]]] \
{puts[[SDLENTITYREF[nbsp]]]$cell}
```

関連項目

- [get_lib_cells](#)
- [list_property](#)
- [report_property](#)

get_clocks

現在のデザインのクロックのリストを取得します。

構文

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

戻り値

クロックのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-of_objects	オプション		指定したピンまたはネットのクロックを取得します。
-match_style	オプション	sdc	パターン一致のスタイルを指定します。有効な値は ucf、sdc です。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	クロックを検索するパターンを指定します。

カテゴリ

SDC、XDC、オブジェクト

説明

現在のデザインに含まれるクロックで、検索パターンに一致するものをリストします。デフォルトでは、**all_clocks** コマンドと同様に、デザインのすべてのネットのリストが返されます。

クロックを作成するには、**create_clock** または **create_generated_clock** コマンドを使用します。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、**-regexp** を使用した場合にのみ適用されます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、**get_clocks** で返されるオブジェクトのリストに、クロックのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、**report_property** または **list_property** コマンドで確認できます。clock オブジェクトの場合、結果をフィルター処理できるプロパティには PERIOD、WAVEFORM、IS_GENERATED などがあります。

フィルター パターンに使用できる演算子は ==、!=、=~ で、フィルター式の間に && および || も使用できます。

-of_objects *args* : 指定したピンまたはネット オブジェクトに接続されているクロックを取得します。

-match_style [**sdc** | **ucf**] : 検索パターンが UCF 制約または SDC 制約に一致することを示します。デフォルトは SDC です。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するクロックを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべてのクロックが返されます。複数のパターンを指定して、異なる検索条件に基づいてクロックを検索できます。

注記 : 複数の検索パターンは {} で囲み、1 つのエレメントとして指定します。

例

次の例では、複数の検索パターンに一致するクロックのリストが返されます。

```
get_clocks {*clock *ck *Clk}
```

注記 : パターンに一致するクロックがない場合は、警告メッセージが表示されます。

次の例では、指定のクロックに設定されているプロパティとその値がリストされます。

```
report_property -all [get_clocks wbClk]
```

関連項目

- [all_clocks](#)
- [list_property](#)
- [report_property](#)

get_debug_cores

現在のデザインの ChipScope デバッグ コアのリストを取得します。

構文

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

戻り値

debug_core オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-filter	オプション		式を使用してリストをフィルター処理します。
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	デバッグ コアを検索するパターンを指定します。

カテゴリ

オブジェクト、ChipScope

説明

現在のプロジェクトの ChipScope デバッグ コアで、指定した検索パターンに一致するものをリストします。デフォルトでは、プロジェクトのデバッグ コアすべてのリストが返されます。

デバッグ コアは、create_debug_core または read_chipscope_cdc コマンドを使用するとプロジェクトに追加できます。ChipScope デバッグ コアをプロジェクトに追加すると、ICON コントローラー コア内に含まれ、CLK ポートとトリガー (TRIG) ポートがデフォルトで含まれます。create_debug_port コマンドを使用して、デバッグ コアにポートを追加することもできます。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、-regexp を使用した場合にのみ適用されます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、`get_debug_cores` で返されるオブジェクトのリストに、パーツのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、`report_property` または `list_property` コマンドで確認できます。

フィルター式に使用できる演算子は `==`、`!=`、`=~` で、フィルター式の間に `&&` および `||` も使用できます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するパーツを検索します。デフォルトのパターンはワイルドカード (*) で、すべてのパーツが返されます。複数のパターンを指定して、異なる検索条件に基づいてパーツを検索できます。

注記 : 複数の検索パターンは {} で囲み、1 つのエレメントとして指定します。

例

次の例では、現在のプロジェクトの ChipScope デバッグ コアのリストが返されます。

```
get_debug_cores
```

注記 : ICON コアがプロジェクトのデバッグ コアの 1 つとして返されます。このコアは直接作成できませんが、ILA コアをプロジェクトに追加すると自動的に追加されます。

次の例では、指定したデバッグ コアのプロパティが返されます。

```
report_property [get_debug_cores myCore]
```

返されるプロパティの値は、コアのコンフィギュレーション方法によって異なります。特定コアのプロパティをコンフィギュレーションするには、次の例のように `set_property` コマンドを使用します。

```
set_property enable_storage_qualification false [get_debug_cores myCore]
```

関連項目

- [create_debug_core](#)
- [create_debug_port](#)
- [get_debug_cores](#)
- [read_chipscope_cdc](#)
- [report_property](#)
- [set_property](#)

get_debug_ports

現在のデザインの ChipScope デバッグ ポートのリストを取得します。

構文

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

戻り値

デバッグ ポート オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-filter	オプション		式を使用してリストをフィルター処理します。
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	デバッグ ポートを検索するパターンを指定します。

カテゴリ

オブジェクト、ChipScope

説明

現在のプロジェクトの ChipScope デバッグ コアで定義されているポートで、指定した検索パターンに一致するものをリストします。デフォルトでは、プロジェクトのデバッグ ポートすべてのリストが返されます。

デバッグ ポートは、ChipScope デバッグ コアを read_chipscope_cdc コマンドまたは create_debug_core コマンドで作成すると定義されます。また、create_debug_port コマンドでポートを既存のデバッグ コアに追加することもできます。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、-regexp を使用した場合にのみ適用されます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、`get_debug_ports` で返されるオブジェクトのリストに、ポートのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、`report_property` または `list_property` コマンドで確認できます。フィルターには、任意のプロパティと値の組み合わせを使用できます。`debug_port` オブジェクトの場合、結果をフィルター処理するのに使用できるプロパティには `PORT_WIDTH`、`MATCH_TYPE` などがあります。

フィルター式に使用できる演算子は `==`、`!=`、`=~` で、フィルター式の間に `&&` および `||` も使用できます。例を参照してください。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するパーツを検索します。デフォルトのパターンはワイルドカード (*) で、すべてのパーツが返されます。複数のパターンを指定して、異なる検索条件に基づいてパーツを検索できます。

注記 : 複数の検索パターンは {} で囲み、1 つのエレメントとして指定します。

例

次の例では、現在のプロジェクトの ChipScope デバッグ コアのポートで、`PORT_WIDTH` プロパティが 8 のものをリストします。

```
get_debug_ports -filter {PORT_WIDTH==8}
```

次の例では、指定したデバッグ ポートに設定されているプロパティが返されます。

```
report_property [get_debug_ports myCore/TRIG0]
```

注記 : デバッグ ポートは、`core_name/port_name` の形式で指定します。

関連項目

- [create_debug_core](#)
- [create_debug_port](#)
- [read_chipscope_cdc](#)
- [report_property](#)

get_designs

現在のデザインに含まれるデザインのリストを取得します。

構文

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

戻り値

デザイン オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	デザインを検索するパターンを指定します。

カテゴリ

XDC、オブジェクト

説明

現在のプロジェクトで開いているデザインで、指定した検索パターンに一致するものをリストします。デフォルトでは、プロジェクトで開いているデザインすべてのリストが返されます。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、-regexp を使用した場合にのみ適用されます。

-filter args : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、get_designs で返されるオブジェクトのリストに、デザインのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、report_property コマンドで確認できます。design オブジェクトの場合、結果をフィルター処理するのに使用できるプロパティには CONSTRSET、PART などがあります。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するデザインを検索します。デフォルトのパターンはワイルドカード (*) で、すべてのデザインが返されます。複数のパターンを指定して、異なる検索条件に基づいてデザインを検索できます。

例

次の例では、現在のプロジェクトで開いているデザインすべてのリストが返されます。

```
get_designs
```

次の例では、検索パターンに一致するデザインに設定されているプロパティが返されます。

```
report_property [get_designs r*]
```

注記 : パターンに一致するデザインがない場合は、警告メッセージが表示されます。

関連項目

[report_property](#)

get_files

ソース ファイルのリストを取得します。

構文

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

戻り値

ファイル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-regexp</code>	オプション		検索パターンを正規表現で指定します。
<code>-nocase</code>	オプション		パターンの大文字/小文字を区別せずに検索します。
<code>-filter</code>	オプション		式を使用してリストをフィルター処理します。
<code>-of_objects</code>	オプション		指定したファイルセットのファイルを取得します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	ファイルを検索するパターンを指定します。

カテゴリ

オブジェクト、プロジェクト

説明

現在のプロジェクトに含まれるファイルで、検索パターンに一致するものをリストします。デフォルトでは、プロジェクトのファイルすべてのリストが返されます。

引数

`-regexp`：検索パターンを正規表現で指定します。

`-nocase`：パターンの大文字/小文字を区別せずに検索します。このオプションは、`-regexp` を使用した場合にのみ適用されます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、`get_files` で返されるオブジェクトのリストに、ファイルのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、`report_property` コマンドで確認できます。フィルターには、任意のプロパティと値の組み合わせを使用できます。file オブジェクトの場合、結果をフィルター処理するのに使用できるプロパティには `FILE_TYPE`、`IS_ENABLED` などがあります。

-of_objects *args* : ファイルを検索するファイルセットを 1 つまたは複数指定します。デフォルトでは、すべてのファイルセットが検索されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するファイルを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべてのファイルが返されます。複数のパターンを指定して、異なる検索条件に基づいてファイルを検索できます。

例

次の例では、現在のプロジェクトのファイルすべてのリストが返されます。

```
get_files
```

次の例では、`constrs_1` および `sim_1` ファイルセットで検出された Verilog ファイル (*.v) のリストが返されます。

```
get_files -of_objects {constrs_1 sim_1} *.v
```

パターンに一致するファイルがない場合は、警告メッセージが表示されます。

関連項目

[report_property](#)

get_filesets

現在のプロジェクトのファイルセットのリストを取得します。

構文

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

戻り値

ファイルセット オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します。
-filter	オプション		式を使用してリストをフィルター処理します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	ファイルセットを検索するパターンを指定します。

カテゴリ

オブジェクト、プロジェクト

説明

現在のプロジェクトに含まれるファイルセットで、検索パターンに一致するものをリストします。デフォルトでは、プロジェクトのファイルセットすべてのリストが返されます。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、**-regexp** を使用した場合にのみ適用されます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、`get_filesets` で返されるオブジェクトのリストに、ファイルセットのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、`report_property` コマンドで確認できます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するファイルセットを検索します。デフォルトのパターンはワイルドカード (*) で、すべてのファイルセットが返されます。複数のパターンを指定して、異なる検索条件に基づいてファイルセットを検索できます。

例

次の例では、現在のプロジェクトのファイルセットすべてのリストが返されます。

```
get_filesets
```

次の例では、`project_2` をアクティブ プロジェクトに指定した後、`s` または `r` で始まるファイルセットのリストを取得します。

```
current_project project_2  
get_filesets s* r* -quiet
```

パターンに一致するファイルセットがない場合、通常は警告メッセージが表示されますが、上記の例では `-quiet` オプションが指定されているので、警告メッセージは表示されません。

次の例では、大文字/小文字を区別せずに、`C` で始まるファイルセットを検索しています。

```
get_filesets C.* -regexp -nocase
```

この例では、`constrs_1` および `constrs_2` 制約セットが現在のプロジェクトで定義されていれば返されます。

関連項目

[report_property](#)

get_generated_clocks

現在のデザインの生成済みクロックのリストを取得します。

構文

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

戻り値

クロックのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	生成したクロックを検索するパターンを指定します。

カテゴリ

XDC、オブジェクト

説明

現在のプロジェクトに含まれる生成済みクロックで、指定した検索パターンに一致するものをリストします。デフォルトでは、プロジェクトの生成済みクロックすべてのリストが返されます。

生成済みクロックは、create_generated_clock を使用するとデザインに追加できます。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、-regexp を使用した場合にのみ適用されます。

-filter args : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、get_generated_clocks で返されるオブジェクトのリストに、クロックのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、report_property コマンドで確認できます。generated_clock オブジェクトの場合、結果をフィルター処理するのに使用できるプロパティには DUTY_CYCLE、MASTER_CLOCK などがあります。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致する生成済みクロックを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべての生成済みクロックが返されます。

例

次の例では、現在のプロジェクトの生成済みクロックすべてのリストが返されます。

```
get_generated_clocks
```

関連項目

[report_property](#)

get_hierarchy_separator

階層区切り文字を取得します。

構文

```
get_hierarchy_separator [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

SDC、XDC

説明

デザインで現在設定されている階層区切り文字を返します。階層区切り文字を設定するには、**set_hierarchy_separator** コマンドを使用します。

引数

-quiet (オプション) : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、現在定義されている階層区切り文字が返されます。

```
get_hierarchy_separator
```

関連項目

[set_hierarchy_separator](#)

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取得します。

構文

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

戻り値

インターフェイス オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-regexp</code>	オプション		検索パターンを正規表現で指定します。
<code>-nocase</code>	オプション		パターンの大文字/小文字を区別せずに検索します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>patterns</code>	オプション		I/O ポート インターフェイスを検索するパターンを指定します。

カテゴリ

XDC、オブジェクト

説明

現在のプロジェクトに含まれる I/O インターフェイスで、指定した検索パターンに一致するものをリストします。デフォルトでは、プロジェクトの I/O インターフェイスすべてのリストが返されます。

引数

`-regexp` : 検索パターンを正規表現で指定します。

`-nocase` : パターンの大文字/小文字を区別せずに検索します。このオプションは、`-regexp` を使用した場合にのみ適用されます。

`-quiet` : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

`patterns` : 指定したパターンと一致するインターフェイスを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべてのインターフェイスが返されます。

例

次の例では、プロジェクトのインターフェイスすべてのリストが返されます。

```
get_interfaces
```

関連項目

- ・ [create_interface](#)
- ・ [delete_interface](#)

get_iobanks

I/O バンクのリストを取得します。

構文

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

戻り値

I/O バンクのリスト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-regexp</code>	オプション		検索パターンを正規表現で指定します。
<code>-nocase</code>	オプション		パターンの大文字/小文字を区別せずに検索します。
<code>-filter</code>	オプション		式を使用してリストをフィルター処理します。
<code>-of_objects</code>	オプション		指定したパッケージピンの I/O バンクを取得します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	I/O バンクを検索するパターンを指定します。

カテゴリ

XDC、オブジェクト

説明

現在のプロジェクトのターゲット デバイスの I/O バンクで、指定した検索パターンに一致するものをリストします。デフォルトでは、ターゲット デバイスの I/O バンクすべてのリストが返されます。

引数

`-regexp` : 検索パターンを正規表現で指定します。

`-nocase` : パターンの大文字/小文字を区別せずに検索します。このオプションは、`-regexp` を使用した場合にのみ適用されます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、`get_iobanks` で返されるオブジェクトのリストに、I/O バンクのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、`report_property` コマンドで確認できます。`iobank` オブジェクトの場合、結果をフィルター処理するのに使用できるプロパティには `DCI_CASCADE`、`INTERNAL_VREF` などがあります。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致する I/O バンクを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべての I/O バンクが返されます。

例

次の例では、ターゲット デバイスの I/O バンクすべてのリストが返されます。

```
get_iobanks
```

関連項目

[report_property](#)

get_ipdefs

現在の IP カタログから IP のリストを取得します。

構文

```
get_ipdefs [-vlnv] [-regexp] [-nocase] [-quiet] [patterns ...]
```

戻り値

カタログ IP オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-vlnv	オプション		パターンを IP の VLNV 文字列と比較して検索します。このオプションを設定しない場合は、IP 表示名が検索されます。
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	カタログ IP を検索するパターンを指定します。

カテゴリ

オブジェクト、プロジェクト

get_ips

現在のデザインの IP のリストを取得します。

構文

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

戻り値

IP オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	IP を検索するパターンを指定します。

カテゴリ

オブジェクト、プロジェクト

get_lib_cells

ライブラリ セルのリストを取得します。デフォルトでは、現在のデザインのパーツに関連するすべてのライブラリ セルが返されます。

構文

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

戻り値

ライブラリ セルのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-of_objects	オプション		指定のインスタンスまたはライブラリ ピンのライブラリ セルを取得します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	ライブラリ セルを検索するパターンを指定します。パターンは、libname /* のようにライブラリ名を指定する必要があります。ライブラリ名は、get_libs コマンドで検索できます。

カテゴリ

SDC、XDC、オブジェクト

説明

現在のデザインのターゲット パーツのライブラリに含まれるセルのリストを取得します。このコマンドを使用すると、特定のライブラリ セルやセル タイプを検索して、そのセルのプロパティを取得できます。

注記：このコマンドを実行するには、ライブラリ名とセル名 (lib_name/cell_name) を含む階層名が必要です。

引数

-regexp：検索パターンを正規表現で指定します。

-nocase：パターンの大文字/小文字を区別せずに検索します。このオプションは、-regexp を使用した場合にのみ適用されます。

-filter *args*：結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、get_cells で返されるオブジェクトのリストに、セルのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、report_property または list_property コマンドで確認できます。

フィルター式に使用できる演算子は ==、!=、=~ で、フィルター式の間に && および || も使用できます。

-of_object *arg*：指定のインスタンス (cells/insts) またはライブラリ ピン (get_lib_pins) のライブラリ セルを取得します。

-quiet：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns：指定したパターンと一致するライブラリ セルを検索します。パターンには、ライブラリ名とセル名の両方を指定する必要があります。

例

次の例では、現在のデザインのターゲット パーツのライブラリのセル数が返され、そのライブラリの AND タイプのセル数が返されます。

```
llength [get_lib_cells [get_libs]/*]  
795  
llength [get_lib_cells [get_libs]/AND*]  
18
```

次の例では、指定したセル オブジェクトのライブラリ セルが返されます。

```
get_lib_cells -of_objects [lindex [get_cells] 1]
```

関連項目

- [get_cells](#)
- [get_libs](#)
- [get_lib_pins](#)
- [list_property](#)
- [report_property](#)

get_lib_pins

ライブラリ セル ピンのリストを作成します。

構文

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

戻り値

ライブラリ セル ピンのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-of_objects	オプション		指定したピンまたはライブラリセルのライブラリセルピンを取得します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	ライブラリセルを検索するパターンを指定します。

カテゴリ

SDC、XDC、オブジェクト

説明

現在のデザインのターゲット パーツのセル ライブラリから、指定したセルのピンをリストします。

注記：このコマンドを実行するには、ライブラリ名、セル名、ピン (lib_name/cell_name/pins) を含む階層名が必要です。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、-regexp を使用した場合にのみ適用されます。

-filter *args*: 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、`get_lib_pins` で返されるオブジェクトのリストに、ピンのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、`report_property` または `list_property` コマンドで確認できます。

フィルター式に使用できる演算子は `==`、`!=`、`=^` で、フィルター式の間に `&&` および `||` も使用できます。

-of_object *arg*: 指定したピン オブジェクトまたはライブラリ セル (`get_lib_cells`) のライブラリ セル ピンを取得します。

-quiet: コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns: 指定したパターンと一致するピンを検索します。パターンには、ライブラリ名、セル名、およびピンを指定する必要があります。

例

次の例では、すべてのライブラリ セル ピンが返されます。

```
get_lib_pins xt_virtex6/AND2/*
```

次の例では、ターゲット デバイスのセル ライブラリのすべてのセルのすべてのピンが返されます。

```
get_lib_pins [get_libs]/*/*
```

関連項目

- [get_libs](#)
- [get_lib_cells](#)
- [list_property](#)
- [report_property](#)

get_libs

ライブラリのリストを作成します。

構文

```
get_libs [-regex] [-nocase] [-filter arg] [-of_objects args]
[-quiet] [patterns]
```

戻り値

ライブラリのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regex	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regex を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-of_objects	オプション		指定したライブラリ セルのライブラリを取得します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	ライブラリを検索するパターンを指定します。

カテゴリ

SDC、XDC、オブジェクト

説明

現在のデザインのターゲット デバイスのセル ライブラリを返します。デバイス ファミリによって使用できるプリミティブは異なるので、デバイス ファミリごとに 1 つのライブラリがあります。

引数

-regex : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、-regex を使用した場合にのみ適用されます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、get_libs で返されるオブジェクトのリストに、ライブラリのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、report_property または list_property コマンドで確認できます。

-of_object *arg* : 指定したオブジェクトのライブラリを取得します。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するライブラリを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべてのライブラリが返されます。

例

次の例では、ターゲット パーツのセル ライブラリが返されます。

```
get_libs
```

関連項目

- [get_lib_cells](#)
- [get_lib_pins](#)
- [list_property](#)
- [report_property](#)

get_msg_count

メッセージ数を取得します。

構文

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

戻り値

メッセージ数

使用法

名前	必須/ オプション	デフォルト	説明
-severity	オプション	ALL	「ERROR」または「CRITICAL WARNING」など、メッセージの重要度を指定します。-id と共に使用することはできません。
-id	オプション		「Common-99」など、メッセージの ID を指定します。-severity と共に使用することはできません。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

説明

PlanAhead を起動してから表示されたメッセージ数を返します。これにより、既に表示されているメッセージ数がメッセージの制限数にどれくらい近いかを確認できます。現在のメッセージの制限数は、**get_msg_limit** コマンドで確認できます。メッセージの制限数は、**set_msg_limit** コマンドで変更できます。

デフォルトでは、すべてのメッセージの合計数が返されます。指定した重要度のメッセージや、指定した ID のメッセージ数のみを取得することもできます。

引数

-id value : PlanAhead ツールのメッセージ ビューやほかのレポートに表示される ID を指定します。特定のメッセージ ID を指定します。

-severity value：メッセージの重要度を指定します。次の 5 種類の重要度があります。

- ・ **ERROR**：デザインの結果が使用不可となったり、ユーザーの操作がないと回避できないような問題が発生していることを示すエラー メッセージです。
- ・ **{CRITICAL WARNING}**：入力や制約に適用されないものがあったり、FPGA ファミリには適していないものがあることを示すクリティカル警告メッセージです。修正することが強く推奨されます。注記：これは 2 単語の値なので、**{ }** で囲む必要があります。
- ・ **WARNING**：制約または仕様が指定どおりに適用されず、デザインが最適な結果にならない可能性を示す警告メッセージです。修正するかどうかは、ユーザーが判断します。
- ・ **INFO**：STATUS メッセージと同じですが、重要度とメッセージ ID タグが含まれる点が異なります。INFO メッセージには、必要に応じてアンサー データベースで確認できるようにメッセージ ID が含まれます。
- ・ **STATUS**：デザインの処理に関するプロセスおよびフィードバックのステータスを示すステータス メッセージです。STATUS メッセージには、メッセージ ID は含まれません。

-quiet：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、メッセージの総数が返されます。

```
get_msg_count -severity ALL
get_msg_count
```

注記：-severity または -id を指定しない場合、デフォルトですべてのメッセージの数が返されるので、上記のどちらの行も同じ結果になります。

次の例では、指定した ID のメッセージ数が返されます。

```
get_msg_count -id Netlist-1129
```

関連項目

- ・ [get_msg_limit](#)
- ・ [set_msg_limit](#)

get_msg_limit

メッセージ数の制限を取得します。

構文

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

戻り値

メッセージの制限数

使用法

名前	必須/ オプション	デフォルト	説明
-severity	オプション	ALL	「ERROR」または「CRITICAL WARNING」など、メッセージの重要度を指定します。-id と共に使用することはできません。
-id	オプション		「Common-99」など、メッセージの ID を指定します。-severity と共に使用することはできません。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

説明

PlanAhead の起動中に表示されるメッセージ数を返します。定義した制限数に到達すると、メッセージは表示されなくなります。デフォルト値は 4,294,967,295 です。このデフォルト値は、**set_msg_limit** コマンドで変更できます。

デフォルトでは、すべてのメッセージに対する制限数が返されます。指定した重要度のメッセージや、指定した ID のメッセージの制限数のみを取得することもできます。次は PlanAhead で表示されるメッセージの例です。

INFO: [common-99] This is an example INFO message

CRITICAL WARNING: [Netlist-1129] This message is a CRITICAL WARNING and requires user attention

注記：特定のメッセージ ID の重要度を変更するには、**set_msg_severity** コマンドを使用します。

引数

-id value : PlanAhead ツールのメッセージ ビューやほかのレポートに表示される ID を指定します。特定のメッセージ ID を指定します。上記の例の場合、メッセージ ID は common-99 および Netlist-1129 になります。

-severity value：メッセージの重要度を指定します。次の 5 種類の重要度があります。

- ・ **ERROR**：デザインの結果が使用不可となったり、ユーザーの操作がないと回避できないような問題が発生していることを示すエラー メッセージです。
- ・ **{CRITICAL WARNING}**：入力や制約に適用されないものがあったり、FPGA ファミリには適していないものがあることを示すクリティカル警告メッセージです。修正することが強く推奨されます。注記：これは 2 単語の値なので、{} で囲む必要があります。
- ・ **WARNING**：制約または仕様が指定どおりに適用されず、デザインが最適な結果にならない可能性を示す警告メッセージです。修正するかどうかは、ユーザーが判断します。
- ・ **INFO**：STATUS メッセージと同じですが、重要度とメッセージ ID タグが含まれる点が異なります。INFO メッセージには、必要に応じてアンサー データベースで確認できるようにメッセージ ID が含まれます。
- ・ **STATUS**：デザインの処理に関するプロセスおよびフィードバックのステータスを示すステータス メッセージです。STATUS メッセージには、メッセージ ID は含まれません。

-quiet：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、CRITICAL WARNING メッセージの制限数が返されます。

```
get_msg_limit -severity {CRITICAL WARNING}
```

-severity または **-id** を使用しない場合、デフォルトですべてのメッセージの制限数が返されます。

次の例では、指定したメッセージ ID の制限数が返されます。

```
get_msg_limit -id Netlist-1129
```

関連項目

- ・ [set_msg_limit](#)
- ・ [set_msg_severity](#)

get_nets

現在のデザインのネットのリストを取得します。

構文

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

戻り値

ネット オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-hsc	オプション	/	階層区切り文字を指定します。
-hierarchical	オプション		すべての階層レベルで検索します。
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-of_objects	オプション		指定したピン/ポート、セル、タイミングパス、またはクロックのネットを取得します。
-match_style	オプション	sdc	パターン一致のスタイルを指定します。有効な値は ucf、sdc です。
-top_net_of_hierarchical_group	オプション		階層ネットの上位に属するネットセグメントを返します。
-segments	オプション		階層全体のネットのセグメントをすべて返します。
-boundary_type	オプション	upper	ピンと同じレベル (upper)、下のレベル (lower)、またはその両方 (both) にある階層ピンに接続されているネットセグメントを返します。有効な値は、upper、lower、both です。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	ネットを検索するパターンを指定します。

カテゴリ

SDC、XDC、オブジェクト

説明

現在のデザインに含まれるネットで、検索パターンに一致するものをリストします。デフォルトでは、デザインのすべてのネットのリストが返されます。

引数

-hsc *arg* : デフォルトの階層区切り文字は / です。それ以外の階層区切り文字を指定する場合は、このオプションを使用します。

-hierarchical : デザイン階層のすべてのレベルからのネットを取得します。このオプションを指定しない場合、デザイン階層の最上位のネットのみが返されます。

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、**-regexp** を使用した場合にのみ適用されます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、`get_nets` で返されるオブジェクトのリストに、ネットのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、`report_property` または `list_property` コマンドで確認できます。`nets` オブジェクトの場合、結果をフィルター処理するのに使用できるプロパティには PARENT、TYPE、MARK_DEBUG などがあります。

フィルター パターンに使用できる演算子は ==、!=、=~ で、フィルター式の間に && および || も使用できます。

-of_objects *arg* : 指定したセル、ピン、ポート、またはクロックに接続されるネットを取得します。

-match_style [**sdc** | **ucf**] : 検索パターンが UCF 制約または SDC 制約に一致することを示します。デフォルトは SDC です。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するネットを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべてのネットが返されます。複数のパターンを指定して、異なる検索条件に基づいてネットを検索できます。

注記：複数の検索パターンは {} で囲み、1 つのエレメントとして指定します。

例

次の例では、指定したセルに接続されているネットのリストが返されます。

```
get_nets -of_objects [lindex [get_cells] 1]
```

注記：パターンに一致するネットがない場合は、警告メッセージが表示されます。

次の例では、`connect_debug_port` コマンドでデバッグ用にマークされたネットのリストが返されます。

```
get_nets -hier -filter {MARK_DEBUG==1}
```

関連項目

- ・ [connect_debug_port](#)
- ・ [list_property](#)
- ・ [report_property](#)

get_package_pins

パッケージ ピンのリストを取得します。

構文

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

戻り値

パッケージ ピン オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-regexp</code>	オプション		検索パターンを正規表現で指定します。
<code>-nocase</code>	オプション		パターンの大文字/小文字を区別せずに検索します。
<code>-filter</code>	オプション		式を使用してリストをフィルター処理します。
<code>-of_objects</code>	オプション		指定したサイト I/O バンクの パッケージ ピン オブジェクトの リストを取得します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>patterns</code>	オプション	*	パッケージ ピン オブジェクトを 検索するパターンを指定しま す。

カテゴリ

XDC、オブジェクト

説明

ターゲット デバイスの選択したパッケージのピンをリストします。デフォルトでは、パッケージのすべてのピンのリストが返されます。

引数

`-regexp` : 検索パターンを正規表現で指定します。

`-nocase` : パターンの大文字/小文字を区別せずに検索します。このオプションは、`-regexp` を使用した場合にのみ適用されます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、`get_package_pins` で返されるオブジェクトのリストに、ピンのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、`report_property` コマンドで確認できます。pin オブジェクトの場合、結果をフィルターできるプロパティには `IS_CLK_CAPABLE`、`IS_VREF`、`IS_GLOBAL_CLK` などがあります。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するピンを検索します。デフォルトのパターンはワイルドカード (*) で、パッケージのすべてのピンが返されます。複数のパターンを指定して、異なる検索条件に基づいてピンを検索できます。

例

次の例では、ターゲット デバイスのピンすべてのリストが返されます。

```
get_package_pins
```

次の例では、パッケージのクロック兼用 (CC) ピンの数が返されます。

```
llength [get_package_pins -filter {IS_CLK_CAPABLE==1}]
```

注記 : パターンに一致するピンがない場合は、警告メッセージが表示されます。

get_param

パラメーター値を取得します。

構文

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

戻り値

パラメーター値

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
name	必須		パラメーター名を指定します。

カテゴリ

プロパティおよびパラメーター

説明

指定した PlanAhead パラメーターに現在定義されている値を返します。これらのパラメーターは、ツールのさまざまな動作を制御するためのユーザー定義可能なコンフィギュレーション設定です。各パラメーターが何をコンフィギュレーションまたは制御するかは、**report_param** を参照してください。

引数

name : 値を取得するパラメーターの名前を指定します。ユーザー定義可能なパラメーターのリストは、**list_param** を実行すると確認できます。このコマンドでは、パラメーターの完全な名前を指定する必要があります。パターン一致は実行されず、1 つのパラメーターのみ指定可能です。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、指定したパラメーターの現在の値が返されます。

```
get_param tcl.statsThreshold
```

関連項目

- ・ [list_param](#)
- ・ [report_param](#)
- ・ [reset_param](#)
- ・ [set_param](#)

get_parts

ソフトウェアで使用可能なパーツのリストを取得します。

構文

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

戻り値

パーツ オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します。
-filter	オプション		式を使用してリストをフィルター処理します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	パーツを検索するパターンを指定します。

カテゴリ

オブジェクト、プロジェクト

説明

現在のプロジェクトのパーツで、検索パターンに一致するものをリストします。デフォルトでは、プロジェクトのパーツすべてのリストが返されます。

引数

-regexp：検索パターンを正規表現で指定します。

-nocase：パターンの大文字/小文字を区別せずに検索します。このオプションは、-regexp を使用した場合にのみ適用されます。

-filter *args*：結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、get_parts で返されるオブジェクトのリストに、パーツのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、report_property または list_property コマンドで確認できます。フィルターには、任意のプロパティと値の組み合わせを使用できます。part オブジェクトの場合、結果をフィルター処理できるプロパティには DEVICE、FAMILY、SPEED などがあります。

フィルター式に使用できる演算子は ==、!=、=~ で、フィルター式の間に && および || も使用できます。例を参照してください。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するパーツを検索します。デフォルトのパターンはワイルドカード (*) で、すべてのパーツが返されます。複数のパターンを指定して、異なる検索条件のパーツを複数検索することができます。

注記 : 複数の検索パターンは {} で囲み、1 つのエレメントとして指定します。

例

次の例では、7vx485t パーツのスピード グレード -1 のリストが返されます。

```
get_parts -filter {DEVICE == 7vx485t && speed == -1}
```

次の例では、7 シリーズおよび Virtex-6 の数が返されます。

```
llength [get_parts -regexp {xc7v.* xc6V.*} -nocase]
```

注記 : パターンに一致するパーツがない場合は、警告メッセージが表示されます。

関連項目

- ・ [list_property](#)
- ・ [report_property](#)

get_path_groups

現在のデザインのパス グループのリストを取得します。

構文

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

戻り値

パス グループのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	パス グループ名と一致するパターンを指定します。

カテゴリ

XDC、オブジェクト

説明

現在のプロジェクトのタイミング パス グループで、指定した検索パターンに一致するものをリストします。デフォルトでは、デザインのすべてのパス グループのリストが返されます。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、-regexp を使用した場合にのみ適用されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するパス グループを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべてのパス グループが返されます。

例

次の例では、デザインのすべてのタイミング パス グループのリストが返されます。

```
get_path_groups
```

次の場合、名前に Clk という文字列を含むタイミング パス グループがすべて返されます。

```
get_path_groups *Clk*
```

注記 : パターンに一致するパス グループがない場合は、警告メッセージが表示されます。

get_pblocks

現在のデザインの Pblock のリストを取得します。

構文

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

戻り値

Pblock オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	Pblock を検索するパターンを指定します。

カテゴリ

オブジェクト、フロアプラン

説明

現在のプロジェクトで定義されている Pblock で、指定した検索パターンに一致するものをリストします。デフォルトでは、プロジェクトの Pblock すべてのリストが返されます。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、-regexp を使用した場合にのみ適用されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致する Pblock を検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべての Pblock が返されます。

例

次の例では、現在のプロジェクトの Pblock すべてのリストが返されます。

```
get_pblocks
```

関連項目

[create_pblock](#)

get_pins

現在のデザインのピンのリストを取得します。

構文

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

戻り値

ピン オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-hsc	オプション	/	階層区切り文字を指定します。
-hierarchical	オプション		すべての階層レベルで検索します。
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-leaf	オプション		-of_objects を使用した場合、ネットのリーフ/グローバル ピンを取得します。
-filter	オプション		式を使用してリストをフィルター処理します。
-of_objects	オプション		指定したセル、ネット、タイミングパス、またはクロックのピンを取得します。
-match_style	オプション	sdc	パターン一致のスタイルを指定します。有効な値は ucf、sdc です。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	ピンを検索するパターンを指定します。

カテゴリ

SDC、XDC、オブジェクト

説明

現在のデザインに含まれるピン オブジェクトで、検索パターンに一致するものをリストします。デフォルトでは、デザインのすべてのピンのリストが返されます。

引数

-hsc *arg* : デフォルトの階層区切り文字は / です。それ以外の階層区切り文字を指定する場合は、このオプションを使用します。

-hierarchical : デザイン階層のすべてのレベルのピンを取得します。このオプションを指定しない場合、デザイン階層の最上位のピンのみが返されます。

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、**-regexp** を使用した場合にのみ適用されます。

-leaf : **-of_object** オプションで指定したパラメーターのリーフ ピンを含めます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、**get_pins** で返されるオブジェクトのリストに、ピンのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、**report_property** または **list_property** コマンドで確認できます。**pins** オブジェクトの場合、結果をフィルター処理するのに使用できるプロパティには PARENT、TYPE などがあります。

フィルター パターンに使用できる演算子は ==、!=、=~ で、フィルター式の間に && および || も使用できます。

-of_objects *arg* : 指定したセル、ピン、ポート、またはクロックに接続されるピンを取得します。

-match_style [**sdc** | **ucf**] : 検索パターンが UCF 制約または SDC 制約に一致することを示します。デフォルトは SDC です。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するピンを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべてのピンが返されます。複数のパターンを指定して、異なる検索条件に基づいてピンを検索できます。

注記 : 複数の検索パターンは {} で囲み、1 つのエレメントとして指定します。

例

次の例では、指定したセルに接続されているピンのリストが返されます。

```
get_pins -of_objects [lindex [get_cells] 1]
```

注記 : パターンに一致するピンがない場合は、警告メッセージが表示されます。

関連項目

- [list_property](#)
- [report_property](#)

get_ports

現在のデザインのポートのリストを取得します。

構文

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

戻り値

ポート オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-of_objects	オプション		指定したネット、タイミング パス、クロックのポートを取得します。
-match_style	オプション	sdc	パターン一致のスタイルを指定します。有効な値は ucf、sdc です。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	ポートを検索するパターンを指定します。

カテゴリ

SDC、XDC、オブジェクト

説明

現在のデザインに含まれるポート オブジェクトで、検索パターンに一致するものをリストします。デフォルトでは、デザインのすべてのポートのリストが返されます。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、-regexp を使用した場合にのみ適用されます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、`get_ports` で返されるオブジェクトのリストに、ポートのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、`report_property` または `list_property` コマンドで確認できます。ports オブジェクトの場合、結果をフィルター処理するのに使用できるプロパティには PARENT、TYPE などがあります。

フィルター パターンに使用できる演算子は `==`、`!=`、`=~` で、フィルター式の間に `&&` および `||` も使用できます。

-of_objects *arg* : 指定したセル、ピン、ポート、またはクロックに接続されているポートを取得します。

-match_style [*sd*c | *uc*f] : 検索パターンが UCF 制約または SDC 制約に一致することを示します。デフォルトは SDC です。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するポートを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべてのポートが返されます。複数のパターンを指定して、異なる検索条件に基づいてポートを検索できます。

注記 : 複数の検索パターンは {} で囲み、1 つのエレメントとして指定します。

例

次の例では、指定したセルに接続されているピンのリストが返されます。

```
get_ports -of_objects [lindex [get_cells] 1]
```

注記 : パターンに一致するポートがない場合は、警告メッセージが表示されます。

関連項目

- [list_property](#)
- [report_property](#)

get_projects

プロジェクトのリストを取得します。

構文

```
get_projects [-regexp] [-nocase] [-filter arg] [-quiet]
[patterns]
```

戻り値

プロジェクト オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	プロジェクトを検索するパターンを指定します。

カテゴリ

オブジェクト、プロジェクト

説明

現在開いているプロジェクトで、検索パターンに一致するものをリストします。デフォルトでは、開いているプロジェクトすべてのリストが返されます。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、-regexp を使用した場合にのみ適用されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するプロジェクトを返します。デフォルトのパターンはワイルドカード (*) で、すべてのパーツが返されます。複数のパターンを指定して、異なる検索条件に基づいてプロジェクトを検索できます。

例

次の例では、開いているプロジェクトすべてのリストが返されます。

```
get_projects
```

次の例では、project_found という変数が get_projects コマンドで返されるプロジェクトリストの長さに設定され、そのプロジェクトが検出されたか、検出されなかったかが表示されます。

```
set project_found [llength [get_projects ISC*] ]  
if {$project_found > 0} {puts "Project Found."} else {puts "No Projects Found."}
```

注記：パターンに一致するプロジェクトがない場合は、警告メッセージが表示されます。

get_property

オブジェクトのプロパティを取得します。

構文

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

戻り値

プロパティ値

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>property_name</i>	必須		値を取得するプロパティの名前を指定します。
<i>object</i>	必須		プロパティを取得するオブジェクトを指定します。

カテゴリ

XDC、オブジェクト、プロパティおよびパラメーター

説明

指定したオブジェクトの指定したプロパティの値を返します。指定のオブジェクトに指定のプロパティが設定されていない場合、または値が設定されていない場合は、何も返されません。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

property_name : 値を取得するプロパティの名前を指定します。大文字/小文字は区別されません。

object : オブジェクトを指定します。

例

次の例では、指定したセルの NAME プロパティが返されます。

```
get_property NAME [lindex [get_cells] 3]
```

関連項目

- ・ [create_property](#)
- ・ [get_cells](#)
- ・ [list_property](#)
- ・ [list_property_value](#)
- ・ [report_property](#)
- ・ [reset_property](#)
- ・ [set_property](#)

get_reconfig_modules

現在のプロジェクトのリコンフィギャラブル モジュールのリストを取得します。

構文

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

戻り値

リコンフィギャラブル モジュール オブジェクトのリスト

使用法

名 前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-of_objects	オプション		指定したのセルのリコンフィギャラブル モジュールを取得します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	リコンフィギャラブル モジュールを検索するパターンを指定します。

カテゴリ

オブジェクト、パーシャル リコンフィギュレーション

説明

現在のプロジェクトに含まれるリコンフィギャラブル モジュールで、指定した検索パターンに一致するものをリストします。デフォルトでは、プロジェクトのリコンフィギャラブル モジュールすべてのリストが返されます。

引数

-filter *args*: 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、get_reconfig_modules で返されるオブジェクトのリストに、モジュールのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、report_property または list_property コマンドで確認できます。

フィルター式に使用できる演算子は ==、!=、~ で、フィルター式の間に && および || も使用できます。

-of_object *arg*: 指定したオブジェクトのリコンフィギャラブル モジュールを取得します。

-quiet: コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns: 指定したパターンと一致するリコンフィギャラブル モジュールを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべてのモジュールが返されます。

例

次の例では、プロジェクトのリコンフィギャラブル モジュールすべてのリストが返されます。

```
get_reconfig_modules
```

関連項目

- ・ [create_reconfig_module](#)
- ・ [list_property](#)
- ・ [report_property](#)

get_runs

run (実行パターン) のリストを取得します。

構文

```
get_runs [-regexp] [-nocase] [-filter arg] [-quiet] [patterns]
```

戻り値

run オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-nocase	オプション		パターンの大文字/小文字を区別せずに検索します (-regexp を指定した場合のみ有効)。
-filter	オプション		式を使用してリストをフィルター処理します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	run を検索するパターンを指定します。

カテゴリ

オブジェクト、プロジェクト

説明

現在のプロジェクトの合成 run およびインプリメンテーション run で、指定した検索パターンに一致するものをリストします。デフォルトでは、プロジェクトで定義された run すべてのリストが返されます。

引数

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、get_runs で返されるオブジェクトのリストに、run のプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、report_property コマンドで確認できます。フィルターには、任意のプロパティと値の組み合わせを使用できます。runs オブジェクトの場合、結果をフィルター処理するのに使用できるプロパティには CONSTRSET、IS_IMPLEMENTATION、IS_SYNTHESIS、FLOW などがあります。

注記：このオプションは、式を使用してプロパティ値ではなく結果をフィルター処理する filter コマンドとは異なります。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致する run 名を検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべての定義済み run が返されます。複数のパターンを指定して、異なる検索条件に基づいて run を検索できます。

例

次の例では、現在のプロジェクトで定義された run すべてのリストが返されます。

```
get_runs -filter {IS_IMPLEMENTATION==1}
```

次の例では、パターンに一致する run のリストが返されます。

```
get_runs imp*
```

パターンに一致する run がない場合は、警告メッセージが表示されます。

関連項目

[report_property](#)

get_selected_objects

選択したオブジェクトを取得します。

構文

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

戻り値

選択したオブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-primary	オプション		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	オプション		コマンド エラーを無視します。

カテゴリ

オブジェクト、GUI 制御

説明

select_objects コマンドで現在選択されているオブジェクトを返します。プライマリ選択オブジェクトとセカンダリ選択オブジェクトの両方を返すことができます。

プライマリ オブジェクトとは直接選択されたオブジェクトで、セカンダリ オブジェクトとは PlanAhead の [Tools] → [Options] → [Selection Rules] で現在定義されている選択規則にしたがって選択されたオブジェクトを指します。選択規則の設定方法は、『PlanAhead ユーザー ガイド』(UG632) を参照してください。

引数

-primary : プライマリ選択オブジェクトのみを返します。セカンダリ オブジェクトは返しません。デフォルトでは、現在選択されているオブジェクトがすべて返されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、現在選択されているすべてのオブジェクト (プライマリとセカンダリの両方) のプロパティがレポートされます。

```
report_property [get_selected_objects]
```

関連項目

[select_objects](#)

get_sites

サイトのリストを取得します。

構文

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

戻り値

サイト オブジェクトのリスト

使用法

名 前	必須/ オプション	デフォルト	説明
-regexp	オプション		検索パターンを正規表現で指定します。
-filter	オプション		式を使用してリストをフィルター処理します。
-range	オプション		サイトを検索する範囲を指定します。範囲は、2 つのサイト名で定義します。
-of_objects	オプション		指定したオブジェクトのサイトを取得します。
-quiet	オプション		コマンド エラーを無視します。
<i>patterns</i>	オプション	*	サイトを検索するパターンを指定します。ボンディングされたサイトも、パッケージ ピン名に一致します。

カテゴリ

XDC、オブジェクト

説明

ターゲット デバイスのサイトで、指定した検索パターンに一致するものをリストします。デフォルトでは、ターゲット デバイスのサイトすべてのリストが返されます。

引数

-regexp : 検索パターンを正規表現で指定します。

-nocase : パターンの大文字/小文字を区別せずに検索します。このオプションは、**-regexp** を使用した場合にのみ適用されます。

-filter *args* : 結果のリストに指定した式のフィルターを適用します。このオプションを使用すると、`get_sites` で返されるオブジェクトのリストに、サイトのプロパティ値に基づいてフィルターを適用できます。オブジェクトに設定されているプロパティは、`report_property` コマンドで確認できます。site オブジェクトの場合、結果をフィルター処理できるプロパティには `IS_OCCUPIED`、`NUM_INPUTS`、`NUM_OUTPUTS` などがあります。

-range *arg* : 指定した範囲内にあるサイト名を検索します。

-of_object *arg* : 指定したオブジェクトのサイトを取得します。サポートされるオブジェクトは、タイル、BEL、ピン、パッケージ ピン、I/O バンクのみです。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

patterns : 指定したパターンと一致するサイトを検索します。デフォルトのパターンはワイルドカード (*) で、プロジェクトのすべてのサイトが返されます。

例

次の例では、ターゲット デバイスのサイトすべてのリストが返されます。

```
get_sites
```

次の例では、ターゲット デバイスで使用されていないサイト数が返されます。

```
llength [get_sites -filter {IS_OCCUPIED==0}]
```

注記 : パターンに一致するサイトがない場合は、警告メッセージが表示されます。

get_timing_arcs

タイミング アークのリストを取得します。

構文

```
get_timing_arcs [-from args] [-to args] [-filter arg]
[-of_objects args] [-quiet]
```

戻り値

タイミング アーク オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-from	オプション		ピンまたはポートのリストを指定します。
-to	オプション		ピンまたはポートのリストを指定します。
-filter	オプション		式を使用してリストをフィルター処理します。
-of_objects	オプション		指定したのセルのタイミングアークを取得します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

SDC、XDC、オブジェクト

get_timing_paths

タイミング パスを取得します。

構文

```
get_timing_paths [-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] [-setup] [-hold] [-max_paths arg]
[-nworst arg] [-path_type arg] [-input_pins] [-nets]
[-slack_lesser_than arg] [-slack_greater_than arg] [-group args]
[-no_report_unconstrained] [-filter arg] [-regexp] [-nocase]
[-match_style arg] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-from	オプション		タイミング パスの開始点 (ピン、ポート、セル、またはクロック) を指定します。
-rise_from	オプション		ピン、ポート、セル、またはクロックの立ち上がりエッジを開始点として指定します。
-fall_from	オプション		ピン、ポート、セル、またはクロックの立ち下がりエッジを開始点として指定します。
-to	オプション		タイミング パスの終点 (ピン、ポート、セル、またはクロック) を指定します。
-rise_to	オプション		ピン、ポート、セル、またはクロックの立ち上がりエッジを終点として指定します。
-fall_to	オプション		ピン、ポート、セル、またはクロックの立ち下がりエッジを終点として指定します。
-through	オプション		タイミング パスの通過点 (ピン、ポート、セル、またはネット) を指定します。
-rise_through	オプション		ピン、ポート、セル、またはネットの立ち上がりエッジを通過点として指定します。
-fall_through	オプション		ピン、ポート、セル、またはネットの立ち下がりエッジを通過点として指定します。

名前	必須/ オプション	デフォルト	説明
<code>-delay_type</code>	オプション	max	パス遅延のタイプを指定します。指定可能な値は、max、min、min_max、max_rise、max_fall、min_rise、min_fall です。
<code>-setup</code>	オプション		最大遅延タイミング パスを取得します (<code>-delay_type max</code> と同じ)。
<code>-hold</code>	オプション		最小遅延タイミング パスを取得します (<code>-delay_type min</code> と同じ)。
<code>-max_paths</code>	オプション	1	出力するパスの最大数を指定します。
<code>-nworst</code>	オプション	1	エンドポイントまでのワーストパスを N 個リストします。
<code>-path_type</code>	オプション	full_clock_expanded	パス タイプのフォーマットを指定します。設定可能な値は、end、summary、short、full、full_clock、full_clock_expanded です。
<code>-input_pins</code>	オプション		パスに入力ピンを含めます。
<code>-nets</code>	オプション		パスにネットを含めます。
<code>-slack_lesser_than</code>	オプション	1e+30	この値よりも小さいスラックのパスを含めます。
<code>-slack_greater_than</code>	オプション	-1e+30	この値よりも大きいスラックのパスを含めます。
<code>-group</code>	オプション		指定のグループのパスのみを返します。
<code>-no_report_unconstrained</code>	オプション		制約が適用されていないパスは取得しません。
<code>-filter</code>	オプション		式を使用してリストをフィルター処理します。
<code>-regexp</code>	オプション		検索パターンを正規表現で指定します。
<code>-nocase</code>	オプション		パターンの大文字/小文字を区別せずに検索します (<code>-regexp</code> を指定した場合のみ有効)。
<code>-match_style</code>	オプション	ucf	パターン一致のスタイルを指定します。有効な値は ucf、sdc です。
<code>-quiet</code>	オプション		コマンド エラーを無視します。

カテゴリ

SDC、XDC、オブジェクト

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 [-category arg] [-short] [-quiet] [pattern]
```

戻り値

なし

使用法

名 前	必須/ オプション	デフォルト	説明
<code>-category</code>	オプション		指定したカテゴリでトピックを検索します。
<code>-short</code>	オプション		コマンドの簡単な説明を表示します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>pattern</code>	オプション	*	指定したパターンに一致するトピックのヘルプを表示します。

カテゴリ

プロジェクト

説明

PlanAhead の Tcl コマンドのカテゴリのリスト、指定したパターンに一致するコマンドのリスト、または特定コマンドの詳細なヘルプを表示します。

引数

help : Tcl コマンドのカテゴリのリストを返します。コマンドのカテゴリとは、たとえば [File I/O] のような特定の機能を実行するコマンドのグループです。

help -category category : 指定したカテゴリのコマンドのリストを返します。

help pattern : 指定した検索パターンに一致するコマンドのリストを返します。この構文を使用すると、コマンド グループから特定のコマンドをすばやく検索できます。

help command : 指定したコマンドに関する詳細情報を表示します。

help -short command : 指定したコマンドの簡単な説明を表示します。

例

次の例では、files を含むコマンドすべてのリストが返されます。

```
help *files*
```

このリストは、remove_files や delete_files などの特定の機能のコマンドを検索するのに使用できます。

次の例では、remove_files コマンドとその引数の詳細な説明が返されます。

```
help remove_files
```

-short オプションを使用すると、そのコマンドの簡単な説明を表示できます。

highlight_objects

オブジェクトを指定の色でハイライトします。

構文

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

戻り値

なし

使用法

名 前	必須/ オプション	デフォルト	説明
-color_index	オプション		色を色インデックスで指定します。
-rgb	オプション		色を RGB で指定します。
-color	オプション		色の名前を指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>objects</i>	必須		ハイライトするオブジェクトを指定します。

カテゴリ

GUI 制御

説明

PlanAhead の GUI でオブジェクトをハイライトします。色オプションのいずれかで指定した色で指定のオブジェクトをハイライトします。オブジェクトのハイライトを解除するには、**unhighlight_objects** コマンドを使用します。

注記：ハイライト色を指定する色オプションは、1 つのみ指定します。複数の色オプションを指定すると、-rgb、-color_index、-color の優先順位で色が決定されます。

引数

-rgb args：オブジェクトのハイライト色を、RGB コードを使用して {R G B} の形式で指定します。たとえば、{255 255 0} は黄色を指定します。

-color_index arg：オブジェクトのハイライト色を色インデックスで指定します。色インデックスは、[Tools] → [Options] → [Themes] の [Highlight] の下に定義されています。テーマの設定については、『PlanAhead ユーザー ガイド』(UG632) を参照してください。

-color arg：オブジェクトのハイライト色を色の名前で指定します。指定可能な値は、red、green、blue、magenta、yellow、cyan、および orange です。

注記：白は **select_objects** コマンドで指定したオブジェクトを表示するのに使用されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

objects : ハイライトするオブジェクトを指定します。

例

次の例では、現在選択されているオブジェクトを赤でハイライトします。

```
highlight_objects -color red [get_selected_objects]
```

関連項目

- ・ [get_selected_objects](#)
- ・ [unhighlight_objects](#)

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 ...]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
コア	オプション		インプリメントする ChipScope デバッグ コアを指定します。

カテゴリ

[ChipScope](#)

説明

CORE Generator を使用して ChipScope デバッグ コアをインプリメントします。CORE Generator は、指定した ILA デバッグ コアに対して 1 回実行され、すべてのコアが指定されると、ICON コントローラー コアでもう 1 回実行されます。現在 create_debug_core コマンドでサポートされるコアタイプは、この ILA コアのみです。プロジェクトに ILA コアを含めてコンフィギュレーションするため、PlanAhead により ICON コントローラー コアが自動的に追加されます。

ChipScope の ICON および ILA コアは、最初 PlanAhead でブラックボックスとして作成されます。これらのコアは、配置配線を実行する前にインプリメントしておく必要があります。create_debug_core でコアを作成した後、**create_debug_port** および **connect_debug_port** でポートを追加および接続しますが、デバッグ コアの内容はデザインでは定義されていません。

ChipScope デバッグ コアは、**launch_runs** コマンドを使用してインプリメンテーション run を起動すると自動的にインプリメントされますが、**implement_debug_core** コマンドを使用すると、デザイン全体をインプリメントせずに、CORE Generator で 1 つ以上のコアをインプリメントできます。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

cores : インプリメントするデバッグ コアを 1 つ以上指定します。

例

次の例では、現在のプロジェクトに含まれるすべての ChipScope デバッグ コアをインプリメントします。

```
implement_debug_core [get_debug_cores]
```

この結果、ICON と ILA タイプの両方のコアが CORE Gnerator でインプリメントされ、ネットリスト デザインに含まれます。

関連項目

- ・ [connect_debug_port](#)
- ・ [create_debug_core](#)
- ・ [create_debug_port](#)
- ・ [get_debug_cores](#)
- ・ [launch_runs](#)

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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-run</code>	オプション		結果をインポートする run を指定します。
<code>-twx</code>	オプション		インポートする TWX ファイルを指定します。
<code>-pcf</code>	オプション		インポートする PCF ファイルを指定します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>ncd</code>	必須		インポートする配線済み NCD ファイルを指定します。

カテゴリ

プロジェクト

説明

NCD ファイルとオプションの TWX ファイルを現在のプロジェクトのインプリメンテーション run にインポートします。このコマンドは、配置配線済みデザインを ISE から PlanAhead にインポートする手順の 1 つです。

引数

`-run arg`：結果をインポートするインプリメンテーション run の名前を指定します。

このコマンドでは run は作成されるのではなく、必須の NCD ファイルと指定した場合は TWX ファイルを、指定した run にインポートして、そのステートを implemented に設定します。

`-twx arg`：NCD ファイルの配置配線データと共にインポートするオプションの TRACE タイミング ファイル (.TWX) のパスとファイル名を指定します。

`-pcf arg`：読み込む物理制約ファイルを指定します。NGD ファイルの論理制約は MAP で読み込まれます。MAP では、一部の制約を使用してデザインがマップされ、論理制約が物理制約に変換されて、物理制約ファイル (PCF) に書き込まれます。

`-quiet`：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

ncd: 指定したインプリメンテーション run にインポートする NCD ファイルのパスおよび名前を指定します。

例

次の例では、新しい RTL ソース プロジェクトを作成して `design_mode` プロパティを `GateLvl` (ネットリスト入力プロジェクト) に変更し、デザインの最上位となる EDIF ファイルを指定して、最後に NCD ファイルをプロジェクトと共に作成したインプリメンテーション run にインポートしています。

```
create_project myProject C:/Data/myProject
set_property design_mode GateLvl [current_filesset]
set_property edif_top_file C:/Data/ise/drp_des/drp_demo.ngc [current_filesset]
import_as_run -run impl_1 C:/Data/ise/drp_des/drp_demo.ncd
```

インプリメンテーション run は、プロジェクトが最初に作成されたときに合成 run と共に作成されます。合成 run はまだ完了していないので、**import_as_run** コマンドをインプリメンテーション run に使用することはできませんが、`design_mode` プロパティが `GateLvl` に設定されたために合成 run は不要となって削除され、インプリメンテーション run のみが残ります。

次の例では、まずインプリメント済みの run をリセットし、その後 **import_as_run** コマンドを使用して NCD ファイル、TWX ファイル、PCF ファイルを ISE の配置配線済みデザインからインポートしています。

```
reset_run impl_1
import_as_run -run impl_1 -twx C:/Data/ise/drp_des/drp_demo.twx-pcf \
C:/Data/ise/drp_des/drp_demo.pcf C:/Data/ise/drp_des/drp_demo.ncd
```

impl_1 に対して **reset_run** コマンドを実行しない場合、次のようなエラー メッセージが表示されます。

ERROR: Run needs to be reset before importing into it.

関連項目

- [create_project](#)
- [reset_run](#)
- [set_property](#)

import_files

ファイルまたは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 ...]
```

戻り値

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

名前	必須/ オプション	デフォルト	説明
-fileset	オプション		ファイルセット名を指定します。
-force	オプション		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	オプション		下位ディレクトリの検索をディセーブルにします。
-flat	オプション		ファイルをフラットなディレクトリ構造でインポートします。
-relative_to	オプション		指定したディレクトリに相対するパスでファイルをインポートします。
-quiet	オプション		コマンド エラーを無視します。
<i>files</i>	オプション		ファイルセットにインポートするファイルの名前を指定します。

カテゴリ

プロジェクト

説明

1 つ以上のファイルまたは 1 つ以上のディレクトリ内のソース ファイルを指定したファイルセットにインポートします。

このコマンドは、指定したファイルセットを参照することによりファイルを追加する add_files コマンドとは異なり、project.srcs\<fileset>\imports のローカル プロジェクト フォルダーにファイルをインポートしてから、そのファイルを指定したファイルセットに追加します。

引数

-fileset name: 指定したソース ファイルを追加するファイルセットを指定します。指定したファイルセットが存在しない場合は、エラー メッセージが表示されます。ファイルセットを指定しない場合は、デフォルトでソース ファイルセットに追加されます。

-force : ローカル プロジェクト ディレクトリおよびファイルセットの同じ名前のファイルを上書きします。

-norecurse : 指定したディレクトリの下位ディレクトリでコマンドを実行しないよう指定します。このオプションを指定しない場合、下位ディレクトリでもプロジェクトに追加可能なソース ファイルが検索されます。

-flat : デフォルトでは、ファイルをデザインにインポートする際にディレクトリ構造が保持されます。このオプションを指定すると、相対パスが保持されずに、すべてのファイルがインポート フォルダーにインポートされます。

-relative_to arg : 指定したディレクトリに相対するパスでファイルをインポートします。これにより、ローカル プロジェクトのディレクトリ構造でインポート ファイルへのパスを保持できます。ファイルは、指定したディレクトリに相対的なパスを使用してインポート フォルダーにインポートされます。

注記 : **-relative_to** オプションは、**-flat** オプションを指定すると無視されます。**-flat** オプションを使用すると、インポート ファイルのディレクトリ構造が削除されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

files : 指定したファイルセットに追加するファイルまたはディレクトリ名のリストを指定します。ディレクトリ名を指定した場合は、そのディレクトリと下位ディレクトリに含まれる有効なソース ファイルすべてが追加されます。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリで指定ファイルが検索されます。

例

次の例では、top.ucf ファイルを constrs_1 制約ファイルセットにインポートしています。

```
import_files -fileset constrs_1 top.ucf
```

次の例では、**-fileset** オプションが指定されていないので、有効なソース ファイルがソース ファイルセット (sources_1) にデフォルトでインポートされます。また、**-norecurse** オプションが指定されているので、指定した \level1 ディレクトリのみが検索され、下位ディレクトリは検索されません。**-flat** オプションは指定されていないので、すべての有効なソース ファイルがプロジェクトの \imports フォルダーにインポートされます。

```
import_files C:/Data/FPGA_Design/level1 -norecurse -flat
```

注記 : **-flat** オプションが指定されていないので、プロジェクトの \imports フォルダー内に \level1 ディレクトリが作成されます。

次の例では、**-fileset** オプションが指定されていないので、ファイルがソース ファイルセット (sources_1) にインポートされます。有効なソース ファイルは \level1 ディレクトリおよびすべての下位ディレクトリからインポートされ、**-relative_to** オプションが使用されているので、プロジェクトの \imports フォルダーに \Data ディレクトリから書き込まれます。

```
import_files C:/Data/FPGA_Design/level1 -relative_to C:/Data
```

関連項目

[add_files](#)

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]
```

戻り値

追加されたファイル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-srcset	オプション		ソース セット名を指定します。
-file	必須		インポートする IP ファイルの名前を指定します。
-name	必須		作成される新しい IP の名前を指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト、CORE Generator

説明

既存の XCO ファイルを IP として現在のプロジェクトにインポートします。

引数

-file *arg*: 現在のプロジェクトにインポートする IP ファイルを指定します。既存の XCO ファイルを指定する必要があります。XCO ファイルは、IP コアを生成する際に使用されるカスタマイズ パラメーターとプロジェクト オプションがすべて記述された CORE Generator ファイルで、PlanAhead の現在のプロジェクトでコアを作成し直すために使用されます。

-name *arg*: 現在のソース ファイルセットに追加する IP オブジェクトに割り当てる名前を指定します。

-quiet: コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、10 ギガビット イーサネット コアを現在のプロジェクトに追加し、IP_block1 という名前を割り当てています。

```
import_ip C:/Data/FPGA_Design/10gig_eth.xco -name IP_block1
```

IP ブロックが現在のプロジェクトの現在のソース ファイルセットにインポートされます。

import_synplify

指定した Synplify プロジェクト ファイルをインポートします。

構文

```
import_synplify [-copy_sources] [-quiet] file
```

戻り値

Synplify 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

名前	必須/ オプション	デフォルト	説明
<code>-copy_sources</code>	オプション		作成したプロジェクトに Synplify プロジェクト ファイルのソースをすべてコピーします。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<i>file</i>	必須		インポートする Synplify プロジェクト ファイルの名前を指定します。

カテゴリ

プロジェクト

import_xise

作成したプロジェクトに XISE 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 [-copy_sources] [-quiet] file
```

戻り値

true

使用法

名前	必須/ オプション	デフォルト	説明
<code>-copy_sources</code>	オプション		作成したプロジェクトに ISE ソースすべてをコピーします。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<i>file</i>	必須		インポートする XISE プロジェクトファイルの名前を指定します。

カテゴリ

プロジェクト

説明

ISE プロジェクト ファイル (XISE) を PlanAhead にインポートします。これにより、ISE プロジェクトを PlanAhead にすばやく移行して、合成、シミュレーション、インプリメンテーションを実行できます。すべてのプロジェクトソース ファイル、制約ファイル、シミュレーション ファイル、および run 設定が ISE プロジェクトからインポートされ、現在の PlanAhead プロジェクトに再現されます。

このコマンドは、新しい空のプロジェクトで実行する必要があります。ソース ファイル、制約、および run 設定は ISE プロジェクトからインポートされるので、既存のソース ファイルや制約ファイルがある場合はすべて上書きされます。

引数

-copy_sources : ISE プロジェクトのソース ファイルを、現在のディレクトリから参照するのではなく、ローカル プロジェクト ディレクトリ構造にコピーするよう指定します。デフォルトでは、現在のディレクトリからソース ファイルが参照されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

file : 現在のプロジェクトにインポートする ISE プロジェクト ファイル (.xise) の名前を指定します。

例

次の例では、importISE という新規プロジェクトを作成し、ISE プロジェクト ファイル (first_use.xise) をインポートしています。

```
create_project importISE C:/Data/importISE import_xise \  
C:/Data/FPGA_design/ise_designs/drp_des/first_use.xise
```

この例では **-copy_sources** オプションが指定されていないので、ISE プロジェクトのすべてのソース ファイルは現在の場所から参照されてプロジェクトに追加されます。

関連項目

[create_project](#)

import_xst

指定した XST プロジェクト ファイルをインポートします。

構文

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

戻り値

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

名前	必須/ オプション	デフォルト	説明
<code>-copy_sources</code>	オプション		作成したプロジェクトに XST プロジェクト ファイルのソースをすべてコピーします。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<i>file</i>	必須		インポートする XST プロジェクト ファイルの名前を指定します。

カテゴリ

プロジェクト

説明

XST 合成プロジェクト ファイル (XST の実行に使用されるさまざまなソース ファイル) を PlanAhead の現在のプロジェクトに追加します。

引数

`-quiet` : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

file : ソース ファイルのインポート元である XST プロジェクト ファイルの名前を指定します。

例

次の例では、`xst_test` という新規プロジェクトを作成し、`drp_des.xst` ファイルをインポートしています。

```
create_project xst_test C:/Data/FPGA_Design/xst_test import_xst \
C:/Data/ise_designs/drp_des.xst
```

指定した XST プロジェクト ファイルで指定されているソース ファイルが `xst_test` プロジェクトにインポートされます。

関連項目

[create_project](#)

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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-run	オプション		ChipScope Analyzer で開くインプリメント済み run を指定します。
-csproject	オプション		ChipScope プロジェクトを指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

ツール起動、ChipScope

説明

アクティブな run または指定したインプリメント済みデザイン run に対して ChipScope Pro Analyzer ツールを起動します。インプリメンテーションの前に、ネットリスト デザインを ChipScope で使用するよう設定するには、**create_debug_core**、**create_debug_port**、および **connect_debug_port** コマンドを使用します。

インプリメント済みデザインに対して **launch_chipscope_analyzer** を実行するには、BitGen でビットストリーム ファイルを生成しておく必要があります。BitGen が実行されていない場合、エラー メッセージが表示されます。

注記：ビットストリーム ファイルを作成するには、**write_bitstream** コマンドを使用するだけでは不十分です。2 番目の例に示されている手順に従う必要があります。

引数

-run arg : ChipScope Pro Analyzer を起動するのに使用する run の名前を指定します。run は、インプリメントが完了しており、ビットストリーム ファイル (.bit) が生成されていることが必要です。ChipScope では、指定の run のビットストリーム ファイルと debug_nets.cdc ファイルが使用されます。

-csproject arg : ChipScope Pro Analyzer で開くプロジェクトの名前を指定します。プロジェクト名を指定しない場合、デフォルトのプロジェクト名 csdefaultproj.cpj が使用されます。プロジェクト名を指定する場合は、拡張子 .cpj も含める必要があります。

注記：プロジェクトは `project/project.data/sources_1/cs` フォルダに作成されます。

-quiet：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、使用するインプリメンテーション run を指定し、作成する ChipScope プロジェクトの名前を指定して、ChipScope Pro Analyzer を起動します。

```
launch_chipscope_analyzer -run impl_3 -csproject impl_3_cs_project
```

次の例では、impl4 という run に **add_step Bitgen** プロパティを設定し、impl4 run を起動して、その run に対して ChipScope Pro Analyzer を起動します。

```
set_property add_step Bitgen [get_runs impl_4]  
launch_runs impl_4 -jobs 2  
launch_chipscope_analyzer -run impl_4
```

この例では、ChipScope プロジェクトの名前は `csdefaultproj.cpj` となります。

関連項目

- ・ [connect_debug_port](#)
- ・ [create_debug_core](#)
- ・ [create_debug_port](#)
- ・ [launch_runs](#)
- ・ [set_property](#)
- ・ [write_bitstream](#)

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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-run	オプション		FPGA Editor で開くインプリメント済み run を指定します。
-more_options	オプション		追加のコマンドライン オプションを指定します。
-mapped_ncd	オプション		マップ済み NCD ファイルを開きます。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

ツール起動

説明

アクティブな run または指定の run に対して FPGA Editor ツールを起動します。FPGA Editor をデザインで開くと、クリティカルなエレメントを配置したり、クリティカル タイミング パスを調べたりできます。また、**crossprobe_fed** コマンドを使用して PlanAhead と FPGA Editor と間でのクロスプローブが可能であり、2 つのツール間でデザインを簡単に見つけることができます。

引数

-run arg : FPGA Editor を起動するのに使用する run の名前を指定します。FPGA Editor を起動するには、マップ済み NCD または配線済み NCD ファイルが必要です。

-more_options arg : FPGA Editor を起動するのに使用する追加のオプションを指定します。詳細は、FPGA Editor ヘルプを参照してください。

-mapped_ncd : FPGA Editor の入力としてマップ済みの NCD ファイルを使用します。これにより、PAR で配置配線を実行する前の、MAP で生成された NCD ファイルを表示できます。このオプションを使用すると、必要に応じて一部の配置を実行したり、クリティカル コンポーネントを配線したりできます。

注記 : このオプションを指定しない場合、デフォルトで *design_routed.ncd* ファイルが読み込まれます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、使用するインプリメンテーション run を指定し、FPGA Editor を起動してマップ済み NCD ファイルを開きます。

```
launch_fpga_editor -run impl_4 -mapped_ncd
```

関連項目

[crossprobe_fed](#)

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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-run	オプション		iMPACT で開くインプリメント済み run を指定します。
-ipf	オプション		iMPACT のプロジェクトを指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

ツール起動

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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-simset	オプション		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	オプション	behavioral	シミュレーション モードを指定し ます。有効な値は behavioral、 timing です。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

ツール 起動

launch_runs

run のセットを起動します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-jobs	オプション	1	並列実行するジョブ数を指定します。
-scripts_only	オプション		スクリプトのみを生成します。
-all_placement	オプション		固定配置と未固定配置すべてを ISE にエクスポートします。デフォルトでは、固定配置のみがエクスポートされます。
-dir	オプション		起動ディレクトリを指定します。
-to_step	オプション		実行する最後のステップを指定します。複数の run を起動する際には無視されます。ISE フローでのみサポートされます。-next_step と共に使用することはできません。
-next_step	オプション		次のステップを実行します。複数の run を起動する際には無視されます。ISE フローでのみサポートされます。-to_step と共に使用することはできません。
-host	オプション		指定したリモート ホストで指定したジョブ数を起動します。 例 : -host {machine1 2} -host {machine2 4}
-remote_cmd	オプション	ssh -q -o BatchMode=yes	リモート ホストにログインするコマンドを指定します。
-email_to	オプション		ジョブが完了したとき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	オプション		各ジョブの完了時に電子メールを送信します。
-pre_launch_script	オプション		各ジョブの起動前に実行するスクリプトを指定します。

名前	必須/ オプション	デフォルト	説明
<code>-post_launch_script</code>	オプション		各ジョブの完了後に実行するスクリプトを指定します。
<code>-force</code>	オプション		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)。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>runs</code>	必須		起動する run を指定します。

カテゴリ

プロジェクト

説明

PlanAhead で合成 run およびインプリメンテーション run を起動します。run は、**config_runs** コマンドを使用して設定しておく必要があります。同じ **launch_runs** コマンドで合成 run とインプリメンテーション run の両方を指定できますが、インプリメンテーション run を起動するには、親の合成 run が完了している必要があります。

引数

-scripts_only : 指定した run ごとに runme.bat というスクリプトを生成し、後で実行できるようにします。

-jobs arg : ローカル ホストで並列実行するジョブ数を指定します。リモート ホストのジョブ数は、**-host** オプションの一部として指定されるので、**-jobs** と **-host** の両方を指定する必要はありません。

-host args : Linux でのみサポートされます。指定したリモート ホストで指定したジョブ数を実行します。オプションは、次の例のような `{hostname jobs}` という形式で使います。

```
-host {machine1 2}
```

-host オプションを指定しない場合、run はローカル ホストから起動されます。

-pre_launch_script arg : 各ジョブを起動する前に実行するスクリプトを指定します。

-remote_cmd arg : ジョブを起動するため、リモート ホストにログインするために使用するコマンドを指定します。デフォルトのリモート コマンドは「ssh -q -o BatchMode=yes」です。

-post_launch_script arg : すべてのジョブの完了後に実行するスクリプトを指定します。

-all_placement : ユーザーが割り当てた (固定) 配置と自動的に割り当てられた (未固定) 配置を、インプリメンテーション用に ISE の MAP と PAR ツールにエクスポートします。デフォルトでは、ユーザーが割り当てた (固定) 配置のみがインプリメンテーション用にエクスポートされます。

-dir arg : run 結果を書き込むディレクトリを指定します。指定したディレクトリに、各 run ごとに個別のフォルダーが作成されます。デフォルトでは、各 run の結果が `project.runs` ディレクトリの下に個別のフォルダーで書き込まれます。

-email_to *args* : run が完了したことを示す通知を送付する電子メール アドレスを指定します。

-email_all : 各 run が完了するごとに電子メールが送付されます。

-force : このオプションは、パーシャル リコンフィギュレーション プロジェクトでのみ使用できます。パーシャル リコンフィギュレーション デザインの制約の変更が未決定かどうかにかかわらず、run を起動します。未決定の制約の変更は、指定した run では失われます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

runs : 起動する合成 run および インプリメンテーション run を指定します。

例

次の例では、並列実行するジョブ数を 2 に設定し、3 つの異なる合成 run を実行します。

```
launch_runs synth_1 synth_2 synth_4 -jobs 2
```

各 run の結果は、*project.runs* ディレクトリの synth_1、synth_2、および synth_4 フォルダーにそれぞれ書き込まれます。

次の例では、run 結果を書き込むディレクトリが作成されます。この場合、指定したディレクトリに impl_3、impl_4、および synth_3 が書き込まれます。また、**-scripts_only** オプションにより、各フォルダーに runme.bat スクリプトが書き込まれますが、この段階では run は起動されません。

```
launch_runs impl_3 impl_4 synth_3 -dir C:/Data/FPGA_Design/results -scripts_only
```


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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-run	オプション		XPower Analyzer で開くインプリメント済み run を指定します。
-more_options	オプション		追加のコマンド ライン オプションを指定します。
-mapped_ncd	オプション		マップ済み NCD ファイルを開きます。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

ツール 起動

link_design

ネットリスト デザインを開きます。

構文

```
link_design [-name arg] [-part arg] [-constrset arg] [-top arg]
[-block] [-quiet]
```

戻り値

デザイン オブジェクト

使用法

名 前	必須/ オプション	デフォルト	説明
-name	オプション		デザイン名を指定します。
-part	オプション		ターゲット パーツを指定します。
-constrset	オプション		使用する制約ファイルセットを指定します。
-top	オプション		構造ネットリストが Verilog の場合、トップ モジュール名を指定します。
-block	オプション		ブロック レベルのデザインを開きます。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

list_param

すべてのパラメーター名を取得します。

構文

```
list_param [-quiet]
```

戻り値

リスト

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロパティおよびパラメーター

説明

PlanAhead でユーザー定義可能なコンフィギュレーション パラメーターのリストを返します。これらのパラメーターでは、さまざまなツールの設定と動作が指定されます。特定のパラメーターの詳細を確認するには、**report_param** コマンドを実行し、パラメーターの説明とそのパラメーターの現在の値を返します。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、ユーザー定義可能なパラメーターすべてのリストが返されます。

```
list_param
```

関連項目

- [get_param](#)
- [report_param](#)
- [reset_param](#)
- [set_param](#)

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 [-class arg] [-quiet] [object]
```

戻り値

プロパティ名のリスト

使用法

名前	必須/ オプション	デフォルト	説明
-class	オプション		プロパティを取得するオブジェクトタイプを指定します。オブジェクトを指定した場合は無視されます。
-quiet	オプション		コマンド エラーを無視します。
<i>object</i>	オプション		プロパティを取得するオブジェクトを指定します。

カテゴリ

オブジェクト、プロパティおよびパラメーター

説明

指定したオブジェクトのプロパティのリストを返します。

注記：report_property でもオブジェクトのプロパティリストが返されますが、プロパティタイプと値も含まれます。

引数

object : プロパティを取得するオブジェクトを 1 つ指定します。

注記：複数のオブジェクトを指定すると、エラー メッセージが表示されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、指定したオブジェクトのプロパティすべてが返されます。

```
list_property [get_cells cpuEngine]
```

関連項目

- ・ [create_property](#)
- ・ [get_cells](#)
- ・ [get_property](#)
- ・ [list_property_value](#)
- ・ [report_property](#)
- ・ [reset_property](#)
- ・ [set_property](#)

list_property_value

オブジェクトの有効なプロパティ値をリストします。

構文

```
list_property_value [-class arg] [-quiet] name [object]
```

戻り値

プロパティ値のリスト

使用法

名 前	必須/ オプション	デフォルト	説明
<code>-class</code>	オプション		有効なプロパティ値を取得するオブジェクト タイプを指定します。オブジェクトを指定した場合は無視されます。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>name</code>	必須		有効な値を取得するプロパティの名前を指定します。
<code>object</code>	オプション		有効なプロパティ値を取得するオブジェクトを指定します。

カテゴリ

オブジェクト、プロパティおよびパラメーター

説明

指定のオブジェクトのクラスまたは指定のオブジェクトの列挙型プロパティに有効な値をリストします。

注記：このコマンドでは、列挙型プロパティ以外のプロパティの有効な値はリストされません。report_property コマンドを実行すると、プロパティ タイプが返されるので、列挙型プロパティを特定できます。

引数

`-class arg`：プロパティ値を取得するオブジェクトのクラスを指定します。オブジェクトのクラスは、実際のオブジェクトを指定する代わりに使用できます。

`name`：値取得示するプロパティの名前を指定します。このコマンドでは、列挙型の値、または定義済みの値のセットを持つプロパティのみを指定できます。指定したプロパティの有効な値すべてが返されます。

`object`：オブジェクトを指定します。`-class` オプションでオブジェクトのタイプを指定する代わりに、実際のオブジェクトを指定できます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、セル オブジェクトから KEEP_HIERARCHY プロパティの有効な値のリストが返されます。

```
list_property_value KEEP_HIERARCHY -class cell
```

次の例でも同じリストが返されますが、セル クラスに代わりに実際のセル オブジェクトが指定されています。

```
list_property_value KEEP_HIERARCHY [get_cells cpuEngine]
```

関連項目

- [create_property](#)
- [get_cells](#)
- [get_property](#)
- [list_property](#)
- [report_property](#)
- [reset_property](#)
- [set_property](#)

load_reconfig_modules

指定の run から特定のリコンフィギュラブル 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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-force</code>	オプション		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。
<code>-run</code>	オプション		リコンフィギュラブル モジュール を読み込む run を指定します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>reconfig_modules</code>	オプション		読み込むリコンフィギュラブル モジュールを指定します。

カテゴリ

パーシャル リコンフィギュレーション

make_diff_pair_ports

2 つのポートから差動ペアを作成します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>ports</code>	必須		差動ペアにするポートを指定します。

カテゴリ

ピン配置

mark_objects

GUI でオブジェクトをマークします。

構文

```
mark_objects [-rgb args] [-color arg] [-quiet] objects
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-rgb	オプション		色を RGB で指定します。
-color	オプション		色の名前を指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>objects</i>	必須		マークするオブジェクト

カテゴリ

GUI 制御

説明

PlanAhead の GUI で指定のオブジェクトをマークします。指定したオブジェクトの位置を見つけやすくするため、アイコン マークが配置されます。マークは、色オプションのいずれかで指定した色で表示されます。

オブジェクトのマークを解除するには、**unmark_objects** コマンドを使用します。

注記：マークの色を指定する色オプションは、1 つのみ指定します。両方の色オプションを指定すると、**-rgb** の方が **-color** より優先されます。

引数

-rgb args：オブジェクトのハイライト色を、RGB コードを使用して {R G B} の形式で指定します。たとえば、{255 255 0} は黄色を指定し、{0 255 0} は緑を指定します。

-color arg：指定のオブジェクトをマークする色を色の名前で指定します。指定可能な値は、red、green、blue、magenta、yellow、cyan、および orange です。

-quiet：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

objects：マークするオブジェクトを指定します。

例

次の例では、現在選択されているオブジェクトをマークする赤いアイコンを追加します。

```
mark_objects -color red [get_selected_objects]
```

関連項目

- ・ [get_selected_objects](#)
- ・ [unmark_objects](#)

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]
```

戻り値

デザイン オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>run</code>	オプション		開く run を指定します。

カテゴリ

プロジェクト

説明

インプリメント済みデザインを開きます。

引数

`-quiet` : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

`run` : 開くインプリメント済みデザインの run 名を指定します。指定した run は、インプリメント済みデザインを開く前にまずインプリメントしておく必要があります。インプリメントされていない run を開こうとすると、エラー メッセージが表示されます。

例

次は、impl_1 というインプリメント済みデザインを開きます。

```
open_impl_design impl_1
```

関連項目

[launch_runs](#)

open_io_design

I/O ピン配置デザインを開きます。

構文

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

戻り値

デザイン オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
-name	オプション		デザイン名を指定します。
-part	オプション		ターゲット パーツを指定します。
-constrset	オプション		使用する制約ファイルセットを指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

説明

新規または既存の I/O ピン配置デザインを開きます。

注記：I/O ピン配置デザインを開くには、現在のソース ファイルセットの `design_mode` プロパティを `PinPlanning` に設定しておく必要があります。PinPlanning に設定していないと、次のようなエラー メッセージが表示されます。

```
ERROR: The design mode of 'sources_1' must be PinPlanning
```

引数

-name arg：新規または既存の I/O ピン配置デザインの名前を指定します。

-part arg：新規デザインを作成する際に使用するザイリンクス デバイスを指定します。**-part** オプションを指定しない場合、デフォルトのパーツが使用されます。

-constrset arg：I/O ピン配置デザインを開く際に使用する制約ファイルセットの名前を指定します。

注記：**-constrset** オプションには、既存の制約ファイルセットを指定する必要があります。新しいファイルセットを作成することはできません。新しいファイルセットを作成する場合は、`create_fileset` を使用します。

-quiet：コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、myIO という I/O ピン配置デザインが新規に作成されます。

```
open_io_design -name myIO
```

この場合、デフォルトのソース セット、制約セット、パーツが使用されます。

次の例では、myIO という既存の I/O デザインを開いて、使用する制約セットを指定しています。

```
open_io_design -name myIO -constrset topCon
```

この場合、デフォルトのソース セットとパーツが使用されます。

関連項目

[create_project](#)

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] [-run arg] [-part arg]
[-constrset arg] [-top arg] [-block] [-quiet]
```

戻り値

デザイン オブジェクト

使用法

名 前	必須/ オプション	デフォルト	説明
-name	オプション		デザイン名を指定します。
-run	オプション		ネットリスト デザインで開く run を指定します。
-part	オプション		ターゲット パーツを指定します。
-constrset	オプション		使用する制約ファイルセットを指定します。
-top	オプション		構造ネットリストが Verilog の場合、トップ モジュール名を指定します。
-block	オプション		ブロック レベルのデザインを開きます。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

説明

新規または既存のネットリスト デザインを開きます。

ネットリスト デザインを開くには、現在のソース ファイルセットの design_mode を GateLvl に設定しておく必要があります。GateLvl に設定していないと、次のようなエラー メッセージが表示されます。

```
ERROR: The design mode of 'sources_1' must be GateLvl
```

引数

-name arg: 新規または既存のネットリスト デザインの名前を指定します。

-run arg: ネットリスト デザインで開くインプリメンテーション run を指定します。

-part arg: 新規デザインを作成する際に使用するザイリンクス デバイスを指定します。-part オプションを指定しない場合、デフォルトのパーツが使用されます。

-top *arg*: ネットリスト デザインが Verilog の場合、デザイン階層のトップ モジュールを指定します。

-constrset *arg*: デザインを開く際に使用する制約ファイルセットの名前を指定します。

注記: **-constrset** オプションには、既存の制約ファイルセットを指定する必要があります。新しいファイルセットを作成することはできません。新しいファイルセットを作成する場合は、**create_fileset** を使用します。

-quiet: コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、Net1 というネットリスト デザインが新規に作成されます。

```
open_netlist_design -name Net1
```

この場合、デフォルトのソース セット、制約セット、パーツが使用されます。

次の例では、Net1 という既存のネットリスト デザインを開いて、使用する制約セットを指定しています。

```
open_netlist_design -name Net1 -constrset con1
```

この場合、デフォルトのソース セットとパーツが使用されます。

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
```

戻り値

開いたプロジェクト オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-read_only</code>	オプション		プロジェクトを読み出し専用モードで開きます。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>file</code>	必須		開くプロジェクト ファイルを指定します。

カテゴリ

プロジェクト

説明

PlanAhead プロジェクト ファイル (.ppr) を開きます。

このコマンドでプロジェクト ファイルを開いて、デザイン ソース ファイルおよび階層を編集し、I/O ピン配置とフロアプランを実行し、デバイスを合成およびインプリメントします。

引数

`-read_only` : プロジェクトを読み出し専用モードで開きます。変更は保存できませんが、`save_project_as` を使用すると、読み出し専用モードのプロジェクトでも新しいプロジェクトとして保存できるので、変更を加えることができます。

`-quiet` : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

`file` : 開く PlanAhead プロジェクト ファイルを指定します。ファイルを指定する際は、ファイルへのパスと .ppr ファイル拡張子を含める必要があります。

例

次の例では、Designs ディレクトリの my_project1 という名前のプロジェクトを開きます。

```
open_project C:/Designs/my_project1.ppr
```

プロジェクトに拡張子 `.ppr` を付けて指定しないと、PlanAhead でファイルがプロジェクト ファイルとして認識されません。プロジェクト ファイル名にファイルへのパスも含めておかないと、指定したファイルが見つからないことを示すエラー メッセージが表示されます。

関連項目

- ・ [create_project](#)
- ・ [current_project](#)

open_run

ネットリストまたはインプリメンテーション デザインで run を開きます。

構文

```
open_run [-name arg] [-block] [-quiet] run
```

戻り値

デザイン オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
-name	オプション		デザイン名を指定します。
-block	オプション		ブロックレベルのデザインを開きます。
-quiet	オプション		コマンド エラーを無視します。
run	必須		デザインで開く run を指定します。

カテゴリ

プロジェクト

place_cell

1 つまたは複数のインスタンスを新しい場所に移動または配置します。サイトおよびセルは正しい順序でリストし、サイトの数とセルの数は同じである必要があります。

構文

```
place_cell [-quiet] cell_site_list ...
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>cell_site_list</i>	必須		セルおよびサイトを交互にリストします。

カテゴリ

[フロアプラン](#)

place_pblocks

Pblock 配置ツールを実行します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-effort	オプション	HIGH	Pblock ごとの配置ツールのエフォートレベルを指定します。有効な値は LOW、MEDIUM、HIGH です。
-utilization	オプション		Pblock ごとの使用率を指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>Pblock</i>	必須		配置する Pblock のリストを指定します。

カテゴリ

フロアプラン

説明

Pblock を FPGA のファブリックに配置します。Pblock は create_pblock コマンドを使用して作成し、add_cells_to_pblock コマンドを使用してロジックを割り当てておく必要があります。

注記：空の Pblock も指定どおり配置されますが、1 つの CLB タイル (2 つのスライス) が使用されます。

引数

-effort arg: 各 Pblock をファブリックに配置するのに Pblock 配置ツールで使用するエフォートレベルを指定します。有効な値は LOW、MEDIUM、HIGH で、このオプションを指定しない場合のデフォルトは HIGH です。

-utilization arg: Pblock が FPGA に配置されたときに、Pblock に割り当てられたロジックエレメントで消費されるデバイスリソースの割合を指定します。たとえば 50% に指定した場合、半分のリソースを Pblock のロジックに割り当て、残りの半分のリソースはその他のデザインエレメントを組み込むために残すように指定されます。使用率を大きくすると Pblock は小さくなり配置が困難になりますが、使用率を小さくすると Pblock が大きくなります。

注記 : Pblock の使用率は合成後の予測です。実際の結果は異なる可能性があり、**resize_pblock** コマンドを使用して Pblock のサイズ調整が必要な場合があります。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

pblocks : FPGA のファブリックに配置する Pblock を 1 つ以上指定します。

例

次の例では、指定した Pblock を使用率を 75% に設定して配置しています。

```
place_pblocks -effort LOW -utilization 75 block1 block2 block3 block4 block5
```

関連項目

- [add_cells_to_pblock](#)
- [create_pblock](#)
- [resize_pblock](#)

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 ...]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-skip_unconnected_ports	オプション		接続のないポートは配置しません。
-check_only	オプション		I/O クロック配置 DRC のみをチェックします。
-quiet	オプション		コマンド エラーを無視します。
ports	オプション		配置するポートを指定します。 指定しない場合、すべてのポートが配置されます。

カテゴリ

ピン配置

power_opt_design

高度なクロック ゲーティングを使用して、ダイナミック消費電力を最適化します。

構文

`power_opt_design [-quiet]`

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。

カテゴリ

電力

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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-partition_names	オプション		プロモートするパーティションのリストを指定します。
-promote_dir	オプション		パスをプロモートします。
-description	オプション		説明をプロモートします。
-no_state_update	オプション		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ないよう指定します。
-quiet	オプション		コマンド エラーを無視します。
run	必須		プロモートするインプリメント済み run を指定します。

カテゴリ

パーシャル リコンフィギュレーション、パーティション

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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>file</i>	必須		ChipScope CDC ファイル名を指定します。

カテゴリ

ファイル入力および出力、ChipScope

説明

CDC (ChipScope Definition and Connection) ファイルを読み込み、現在のプロジェクトのネットリスト デザインに関連付けます。このファイルは、コアのパラメーターおよび ChipScope ILA デバッグ コアの設定に関する情報を含み、信号名をインポートするために ChipScope Pro Analyzer への入力として使用されます。

Chipscope CDC ファイルは、ISE プロジェクトまたは write_chipscope_cdc コマンドを使用して別の PlanAhead プロジェクトで作成されます。

現在のプロジェクトで使用できない CDC ファイルのパラメーターは、インポートされません。たとえば、ポートへの接続に指定した信号が現在のネットリストに存在しない場合、それらの信号は無視されます。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

file : PlanAhead に読み込む CDC ファイルのパスおよびファイル名を指定します。CDC ファイルでは、挿入するデバッグ コア、プローブする信号、それらの信号のクロックドメインが定義されます。

例

次の例では、指定した CDC ファイルを読み込んでいます。

```
read_chipscope_cdc C:/Data/FPGA_Design/bft.cdc
```

関連項目

- ・ [create_debug_core](#)
- ・ [get_debug_cores](#)
- ・ [write_chipscope_cdc](#)

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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>file</code>	必須		ピン配置 CSV ファイルを指定します。

カテゴリ

ファイル入力および出力

説明

CSV (Comma Separated Value) ファイルからパッケージ ピンおよびポート配置情報をインポートします。

CSV ファイルは I/O ピン配置プロジェクトにのみインポートできます。その他のプロジェクトの場合、ピン定義はソース デザイン データからインポートされます。

CSV ファイルの形式および要件は、『PlanAhead ユーザー ガイド』(UG632) を参照してください。

引数

`file` : インポートする CSV ファイルの名前を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリで指定ファイルが検索されます。

`-quiet` : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、開いているプロジェクトに CSV ファイルをインポートしています。

```
read_csv C/Data/pinList.csv
```

まず、新しい I/O ピン配置プロジェクトを設定してから、指定した CSV ファイルをそのプロジェクトに読み込みます。

```
create_project myPinPlan C:/Data/myPinPlan -part xc7v285tffg1157-1
set_property design_mode PinPlanning [current_fileset]
open_io_design -name io_1
read_csv C:/Data/import.csv
```

注記：プロジェクトの特性は、ソース ファイルセットの design_mode プロパティで指定します。

関連項目

- ・ [create_project](#)
- ・ [open_io_design](#)
- ・ [set_property](#)
- ・ [write_csv](#)

read_edif

1 つまたは複数の EDIF ファイルを読み込みます。

構文

```
read_edif [-quiet] files
```

戻り値

追加されたファイル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>files</i>	必須		EDIF ファイル名を指定します。

カテゴリ

ファイル入力および出力

説明

EDIF ネットリスト ファイルを現在のプロジェクトのデザイン ソース ファイルセットにインポートします。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

files : インポートする EDIF ファイルの名前を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリで指定ファイルが検索されます。

例

次の例では、開いているプロジェクトに EDIF ファイルをインポートしています。

```
read_edif C/Data/bft_top.edf
```

関連項目

[write_edif](#)

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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>file</i>	必須		PXML ファイル名を指定します。

カテゴリ

ファイル入力および出力

説明

パーティション XML ファイルを現在のデザインにインポートします。PXML ファイルには、階層デザインに関するパーティション情報が含まれます。PXML ファイルは、合成またはインプリメンテーション中に PlanAhead で作成されます。このファイルは、手動で作成するか、または PlanAhead のインストール ディレクトリにある XML テンプレートを使用して作成することもできます。デザインのパーティションおよび PXML ファイルの作成については、『階層デザイン手法ガイド』(UG748) を参照してください。

合成およびインプリメンテーションでパーティションが正しく処理されるようにするため、パーティションをインプリメントするよう定義するか、または RTL デザインにインポートするよう定義しておく必要があります。このため、開いた RTL デザインに read_pxml を使用する必要があります。

引数

file : PXML ファイルの名前を指定します。PXML ファイルの名前は xpartition.pxml にする必要があります。異なる名前を使用すると、ファイルを読み込む際に PlanAhead でエラー メッセージが表示されます。

注記：パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリで指定ファイルが検索されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、デザイン階層に関するパーティション情報を含む指定した PXML ファイルを読み込んでいます。

```
read_pxml C:/Data/FPGA_Design/pxmlTest.pxml
```

関連項目

[config_partition](#)

read_twx

TRACE スタティック タイミング解析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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-cell	オプション		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	オプション		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	オプション		コマンド エラーを無視します。
<i>name</i>	必須		結果のセット名を指定します。
<i>file</i>	必須		TRACE インポート ファイルの名前を指定します。

カテゴリ

ファイル入力および出力

説明

ザイリンクス TRACE (Timing Reporter And Circuit Evaluator) ツールで生成された TWX 形式のタイミング レポートをインポートします。TWX は最上位レベルにインポートするか (デフォルト)、特定のセル レベルまたは指定の Pblock に対してインポートできます。

TWX ファイルをインポートすると、タイミング結果が GUI の [Timing Results] ビューに表示されます。

引数

-cell *arg*: TWX ファイルをインポートする現在のデザインの階層セルの名前を指定します。タイミング パスは、指定したセルに適用されます。

-pblock *arg*: 現在のデザインの Pblock の名前を指定します。タイミング パスは、指定したブロックに対してインポートされます。

name: TWX ファイルのタイミング パスをインポートする際に作成する [Timing Results] ビューの名前を指定します。

注記: *name* と *file* の両方を指定し、*name* を先に指定する必要があります。

file : インポートする TWX ファイルの名前を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリで指定ファイルが検索されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、指定した TWX ファイルをデザインの最上位に読み込んでいます。

```
read_twx C:/Data/timing_files/bft.twx
```

関連項目

[report_timing](#)

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
```

戻り値

追加したファイルのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-cell	オプション		制約をインポートするセルを指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>file</i>	必須		ファイル名を指定します。

カテゴリ

ファイル入力および出力

説明

ユーザー制約ファイル (UCF) から物理制約をインポートします。UCF は最上位にインポートするか (デフォルト)、特定のセル レベルにインポートできます。最上位にインポートすると、指定した UCF ファイルがアクティブな制約ファイルセットに追加されます。

このコマンドは add_files コマンドと似ており、UCF はローカルのプロジェクト ディレクトリにインポートされるのではなく、参照されて追加されます。

注記：UCF からの制約は、同じ名前の現在の制約を上書きします。このため、UCF を読み込む際には重要な制約が上書きされないよう注意が必要です。

引数

file：インポートする UCF ファイルの名前を指定します。

注記：パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリで指定ファイルが検索されます。

-cell *arg*：UCF ファイルをインポートする現在のデザインの階層セルの名前を指定します。制約は指定したセルに適用され、インポートした UCF ファイルはアクティブな制約ファイルセットには追加されません。

注記：-cell オプションを指定する場合は、デザインを開いておく必要があります。

-quiet：コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、指定した UCF ファイルをデザインの最上位に読み込んでいます。

```
read_ucf C:/Data/FPGA_Design/top1.ucf
```

関連項目

- ・ [add_files](#)
- ・ [write_ucf](#)
- ・ [save_design](#)

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] [-sv] [-quiet] files ...
```

戻り値

追加されたファイル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-library	オプション	work	ライブラリ名を指定します。
-sv	オプション		SystemVerilog コンパイルをイネーブルにします。
-quiet	オプション		コマンド エラーを無視します。
<i>files</i>	必須		Verilog ファイル名を指定します。

カテゴリ

ファイル入力および出力

説明

Verilog または SystemVerilog ソース ファイルを読み込みます。このコマンドは `add_files` コマンドと似ています。Verilog ファイルを読み込むと、ソース ファイルセットに追加されます。**-library** オプションを指定すると、ファイルはライブラリ プロパティを設定して追加されます。

SystemVerilog には Verilog が含まれるので、**read_verilog** コマンドで両方のファイル タイプを読み込むことができますが、SystemVerilog ファイルでは SystemVerilog モードのコンパイルをイネーブルにするため **-sv** オプションを指定する必要があります。このモードでは、PlanAhead で SystemVerilog のキーワードおよび構文が認識されます。

1 つの PlanAhead プロジェクトに Verilog ファイル (.v) と SystemVerilog ファイル (.sv) の両方を含めることができ、また VHDL ファイル (**read_vhdl** を使用) を混合することも可能です。PlanAhead では、合成用にこれらのファイルをコンパイルする際に、各ファイル タイプにそれぞれコンパイル ユニットが作成され、同じタイプのファイルは一緒にコンパイルされます。

引数

-library *arg*: Verilog ファイルが参照するライブラリを指定します。デフォルトの Verilog ライブラリは `work` です。

-sv: ファイルを SystemVerilog コンパイル グループとして読み込む場合に指定します。

注記 : SystemVerilog には Verilog が含まれるので、Verilog ソース ファイルにユーザー定義名として SystemVerilog の予約キーワードが使用されていないければ、`-sv` オプションを使用して Verilog ファイルを読み込んでも SystemVerilog コンパイル モードがイネーブルになります。ただし、SystemVerilog ファイルを `-sv` を使用せずに Verilog コンパイル ユニットとして追加することはできません。

files : 読み込む Verilog ファイルの名前を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリで指定ファイルが検索されます。

`-quiet` : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例は、指定した Verilog ファイルを読み込んで、ソース ファイルセットに追加しています。

```
read_verilog C:/Data/FPGA_Design/new_module.v
```

次の例では、SystemVerilog ファイルと Verilog ファイルに 1 つずつコンパイル ユニットを作成しています。

```
read_verilog -sv { file1.sv file2.sv file3.sv }  
read_verilog { file1.v file2.v file3.v }
```

関連項目

- [add_files](#)
- [read_vhdl](#)

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
```

戻り値

追加されたファイル オブジェクトのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-library	オプション	work	VHDL ライブラリを指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>files</i>	必須		VHDL ファイル名を指定します。

カテゴリ

ファイル入力および出力

説明

VHDL ソース ファイルを読み込みます。このコマンドは add_files コマンドと似ています。VHDL ファイルを読み込むと、ソース ファイルセットに追加されます。-library オプションを指定すると、ファイルはライブラリ プロパティを設定して追加されます。

引数

-library *arg*: VHDL ファイルが参照するライブラリを指定します。デフォルトの VHDL ライブラリは work です。

file: 読み込む VHDL ファイルの名前を指定します。

注記: パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリで指定ファイルが検索されます。

-quiet: コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、指定した VHDL ファイルを読み込んで、ソース ファイルセットに追加しています。

```
read_vhdl C:/Data/FPGA_Design/new_module.vhdl
```

関連項目

[add_files](#)

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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-pblock	オプション		配置をインポートする Pblock を指定します。
-cell	オプション		配置をインポートするセルを指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>file</i>	必須		配置ファイル名を指定します。

カテゴリ

ファイル入力および出力

説明

XDL 形式のデータを使用して ISE の配置結果を PlanAhead にインポートできます。XDL データは、インプリメンテーション run を実行すると自動的に作成されます。また、NCD ファイルから XDL コマンドライン ツールを使用して XDL 形式のファイルを作成することも可能です。

デザイン全体、個々のモジュール、または特定の Pblock に対して XDL ファイルを作成し、配置をインポートできます。

引数

-cell *arg* : XDL ファイルをインポートする現在のデザインの階層セルの名前を指定します。XDL ファイルのデータは、指定したセルに対してインポートされます。

-pblock *arg* : 現在のデザインの Pblock の名前を指定します。XDL ファイルのデータは、指定したブロックに対してインポートされます。

file : インポートする XDL ファイルの名前を指定します。

注記：パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリで指定ファイルが検索されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、指定した XDL ファイルをデザインの最上位に読み込んでいます。

```
read_xdl C:/Data/FPGA_Designs/bft.xdl
```

redo

取り消されたコマンドをやり直します。

構文

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

戻り値

やり直し可能なタスクのリスト (-list を使用した場合)

使用法

名前	必須/ オプション	デフォルト	説明
-list	オプション		やり直し可能なタスクのリストを返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

GUI 制御

説明

取り消されたコマンドをやり直します。このコマンドを繰り返し使用し、一連のコマンドをやり直すことができます。

startgroup および **endgroup** コマンドを使用してコマンド グループを作成した場合、redo コマンドでコマンド グループがシーケンスとしてやり直されます。

引数

-list : やり直すことができるコマンドのリストを返します。**undo** コマンドを使用すると、コマンドのリストを順にさかのぼってコマンドが取り消されます。**redo** コマンドを使用すると、これらのコマンドをやり直すことができます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、やり直し可能なコマンドのリストが返されます。

```
redo -list
```

関連項目

- [endgroup](#)
- [startgroup](#)
- [undo](#)

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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-part	オプション		ターゲット パーツを指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

説明

ハードドライブのプロジェクト データから現在のデザインを読み込み直します。これにより、メモリのデザインが上書きされ、保存されていない変更は失われます。

引数

-part arg : 指定したパーツを新しいターゲット パーツとして読み込み直します。これにより、ハードドライブのプロジェクト データで指定されている制約ファイルのパーツが上書きされます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、ハード ディスクのプロジェクト データから現在のデザインを読み込み直します。これにより、メモリにある未保存の変更が上書きされます。

```
refresh_design
```

デザインに対する一連の変更を取り消して、デザインを保存されている状態に戻します。

次の例では、指定した Virtex-6 パーツをターゲット デバイスとして使用して現在のデザインを更新しています。2 つ目のコマンドは、選択したパーツをアクティブなインプリメンテーション run のターゲット デバイスにするために必要です。

```
refresh_design -part xc6vcx75tff784-1
set_property part xc6vcx75tff784-1 [get_runs impl_6]
```

注記：ターゲット パーツを変更しない場合は、2 つ目のコマンドは必要ありません。

関連項目

[set_property](#)

reimport_files

最新でないファイルをインポートし直します。

構文

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

戻り値

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

名前	必須/ オプション	デフォルト	説明
-force	オプション		ローカル ファイルの方が新しい場合でもインポートし直します。
-quiet	オプション		コマンド エラーを無視します。
<i>files</i>	オプション		インポートし直すファイルのリストを指定します。ファイルを指定しない場合は、プロジェクトの最新でないファイルがすべてインポートし直されます。

カテゴリ

プロジェクト

説明

プロジェクト ファイルを再インポートします。これにより、参照先のソース ファイルからローカルのプロジェクト ファイルがアップデートされます。

引数

-force : ローカルのプロジェクト ファイルの方が参照先のソース ファイルよりも新しい場合でも、ファイルを再インポートします。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

files : 再インポートするファイルのリストを指定します。ファイルを指定しない場合は、プロジェクトの最新でないファイルがすべてインポートし直されます。**-force** を使用してファイルを指定しない場合は、プロジェクトのすべてのファイルがインポートし直されます。

例

次の例では、ファイルが最新でないか、ローカル ファイルが参照先ソース ファイルよりも新しいかどうかにかかわらず、すべてのプロジェクト ファイルが再インポートされます。

```
reimport_files -force
```

注記：上書きされるローカル ファイルの方が新しくても、警告メッセージは表示されません。

次の例では、指定したファイルがプロジェクトに再インポートされます。

```
reimport_files C:/Data/FPGA_Design/source1.v C:/Data/FPGA_Design/source2.vhdl
```

元のソース ファイルがローカル プロジェクトよりも新しい場合にのみ、2 つの指定したファイルが再インポートされます。

関連項目

- ・ [add_files](#)
- ・ [import_files](#)

remove_cells_from_pblock

Pblock からセルを削除します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>pblock</code>	必須		セルを削除する Pblock を指定します。
<code>cells</code>	必須		削除するセルを指定します。

カテゴリ

フロアプラン

説明

指定したロジック インスタンスを Pblock から削除します。Pblock にセルを追加するには、`add_cells_to_pblock` コマンドを使用します。

Pblock からセルを削除しても、配置が解除されるわけではありません。LOC 制約はそのまま残ります。

引数

`-quiet` : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

`pblock` : 指定のインスタンスを削除する Pblock の名前を指定します。

`cells` : 指定の Pblock から削除する 1 つまたは複数のセル オブジェクトを指定します。

例

次の例では、`pb_cpuEngine` Pblock から指定したセルを削除しています。

```
remove_cells_from_pblock pb_cpuEngine [get_cells cpuEngine/cpu_dwb_dat_o/*]
```

関連項目

[add_cells_to_pblock](#)

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 ...]
```

戻り値

削除されたファイルのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-fileset	オプション		ファイルセット名を指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>files</i>	オプション		削除するファイルおよびディレクトリ名を指定します。

カテゴリ

プロジェクト

説明

指定したファイルセットからファイルを削除します。ファイルは、そのファイルへの完全パスで指定する必要があります。

引数

-fileset *arg* : ファイルを削除するファイルセットを指定します。**-fileset** を指定しない場合は、**source_1** ファイルセットから指定したファイルが削除されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

files : ファイルセットから削除するファイルを 1 つまたは複数指定します。ファイルは、そのファイルへの完全パスで指定する必要があります。

注記： パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリで指定ファイルが検索されます。

例

次の例では、**C:/Design/top.ucf** というファイルを制約セット **constrs_1** から削除しています。

```
remove_files -fileset constrs_1 C:/Design/top.ucf
```

複数のファイルを削除する場合は、次のように指定します。

```
remove_files -fileset sim_1 top_tb1.vhdl top_tb2.vhdl
```

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 ...
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-fileset	オプション		ファイルの順序を並べ替えるファイルセットを指定します。
-before	オプション		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	オプション		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	オプション		リストされたファイルを一番前に移動します (デフォルト)。
-back	オプション		リストされたファイルを一番後に移動します。
-auto	オプション		指定したファイルセットを自動的に並べ替えます。
-disable_unused	オプション		トップ 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	オプション		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。

カテゴリ

プロジェクト

説明

指定したファイルセットのソース ファイルを並べ替えます。指定したファイルを、ファイルセット内で一番頭または一番後に移動したり、ほかのファイルの前後に移動したりできます。このコマンドには自動並べ替え機能もあり、現在のトップ モジュールの要件に基づいてファイルを並べ替えることもできます。

引数

-fileset *arg* : 並べ替えるファイルを含むファイルセットを指定します。デフォルトは sources_1 ソース ファイルセットです。

-before *arg* : 指定したファイルをファイルセットのこのファイルの前に配置します。ファイルはファイルセットの完全パス名で指定しないと、PlanAhead でファイルが検出されません。

-after *arg* : 指定したファイルをファイルセットのこのファイルの後に配置します。ファイルはファイルセットの完全パス名で指定しないと、PlanAhead でファイルが検出されません。

-front : 指定したファイルをファイルセットのファイル リストの一番前に配置します。

-back : 指定したファイルをファイルセットのファイル リストの一番後に配置します。

-auto : プロジェクトの現在のトップ モジュールの階層要件に基づいて、ファイルを自動的に並べ替えます。このオプションは、トップ モジュールを **set_property top** コマンドで変更した後によく使用されます。

-disable_unused : トップ モジュールに基づく階層で現在使用されないファイルをすべてディスエーブルにします。このオプションは、トップ モジュールを **set_property top** コマンドで変更した後によく使用されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

files : ファイルセットで位置を変更するファイルを 1 つまたは複数指定します。これらのファイルは、ファイルセットの完全パス名で指定する必要があり、指定した順に並べ替えられます。

例

次の例では、指定したファイルをソース ファイルセットの一番前に移動しています。

```
reorder_files -front {C:/Data/FPGA/file1.vhdl C:/Data/FPGA/file2.vhdl}
```

-fileset オプションが使用されていないので、デフォルトのソース ファイルセットが使用されます。

次の例では、デザインでトップ モジュールを設定し、このトップ モジュールの階層に基づいてファイルを自動的に並べ替え、未使用のファイルをディスエーブルにしています。

```
set_property top block1 [current_fileset]
reorder_files -auto -disable_unused
```

report_clock_gating

クロック ゲーティングをレポートします。

構文

```
report_clock_gating [-power_opt] [-file arg] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-power_opt	オプション		消費電力で最適化するデザインで実行されるクロック ゲーティングをレポートします。
-file	オプション		出力ファイルを指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

電力

report_clock_interaction

クロックの相互関係をレポートします。

構文

```
report_clock_interaction [-delay_type arg] [-setup] [-hold]
[-significant_digits arg] [-file arg] [-append] [-name arg]
[-return_string] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-delay_type	オプション	max	パス遅延のタイプを指定します。有効な値は、max、min、min_max です。
-setup	オプション		最大遅延タイミング パスを考慮します (-delay_type max と同じ)。
-hold	オプション		最小遅延タイミング パスを考慮します (-delay_type min と同じ)。
-significant_digits	オプション	2	有効桁数を指定します。有効な値は、0 ~ 13 です。
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-append	オプション		結果をファイルに追加します。上書きはしません。
-name	オプション		GUI パネルに表示する結果の名前を指定します。
-return_string	オプション		レポートを文字列として返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

説明

クロックの相互関係と複数のクロックドメインを通過する信号をレポートし、メタステーブル状態、データ損失、非干渉性などの問題を特定します。

このコマンドを実行するには、デザインを開いている必要があります。

引数

-delay_type *value*: クロックの相互関係レポートを実行する際に解析に使用する遅延のタイプを指定します。有効な値は min、max、および min_max です。デフォルトは max です。

-setup: セットアップ違反がないかどうかをチェックします。これは **-delay_type max** を指定するのと同じです。

-hold: ホールド違反がないかどうかをチェックします。これは **-delay_type min** を指定するのと同じです。

注記: **-setup** と **-hold** の両方を指定すると、**-delay_type min_max** を指定するのと同じになります。

-significant_digits *arg*: 出力結果の有効桁数を指定します。有効な値は 0 ～ 13 です。デフォルト値は 2 です。

-file *arg*: クロックの相互関係レポートを出力するファイルの名前を指定します。パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-append : report_clock_interaction コマンドの結果を指定したファイルに上書きするのではなく追加します。

注記: **-append** コマンドは、**-file** オプションを使用している場合にのみ使用可能です。

-name *arg*: PlanAhead ツールの GUI モードで表示する [Clock Interaction] ビューの名前を指定します。指定した名前が開いているレポートビューで既に使用されている場合は、そのビューが閉じられ、新しいレポートが開きます。

-return_string: 出力を Tcl 文字列として返します。Tcl 文字列は、変数定義でキャプチャしたり、解析またはその他の処理が可能です。

-quiet: コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、インターコネクト遅延モデルを設定し、デバイスのスピード グレードを選択して、**report_clock_interaction** を実行しています。

```
set_delay_model -interconnect none
set_speed_grade -3
report_clock_interaction -delay_type min_max -significant_digits 3 -name "results_1"
```

次の例では、クロックの相互関係レポートを GUI および指定のファイルに出力し、返された文字列を指定の変数に割り当てています。

```
set clk_int [report_clock_interaction -file clk_int.txt -name clk_int1 \
-return_string]
```

関連項目

- [report_clocks](#)
- [set_delay_model](#)
- [set_speed_grade](#)

report_clock_utilization

デザインのクロック ネットに関する情報をレポートします。

構文

```
report_clock_utilization [-file arg] [-write_xdc arg]
[-return_string] [-quiet]
```

戻り値

レポート

使用法

名 前	必須/ オプション	デフォルト	説明
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-write_xdc	オプション		クロック制約を出力するファイルの名前を指定します。指定しない場合、クロック制約はクロックレポートに追加されます。
-return_string	オプション		レポートを文字列として返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

説明

デザインのクロック ネットに関する情報と、ターゲット デバイスでのクロック リソースの使用率を返します。

デフォルトでは出力は標準出力に表示されますが、その後の処理用にファイルまたは文字列に出力することもできます。

引数

-file arg: クロック使用率レポートを出力するファイルの名前を指定します。パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-write_xdc arg: さまざまなクロック リソースの XDC ロケーション制約を出力するファイルの名前を指定します。パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

注記：クロック使用率レポートの XDC 制約は、「Location of...」で開始します。**-write_xdc** オプションを指定すると、それらの行が指定のファイルに出力されます。

-return_string：出力を Tcl 文字列として返します。Tcl 文字列は、変数定義でキャプチャしたり、解析またはその他の処理が可能です。

-quiet：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、デザインのクロック ネットに関する情報とターゲット デバイスでのクロック リソースの使用率が返され、指定のファイルに保存されます。

```
report_clock_utilization -file C:/Data/FPGA_Design/clock_util.txt
```

次の例では、クロック ネットとクロック リソースの使用率を標準出力に表示し、XDC ロケーション制約は指定のファイルに記述しています。

```
report_clock_utilization -write_xdc clock_util_xdc.txt
```

XDC ファイル名の一部としてパスが指定されていないので、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

report_clocknetworks

クロック ネットワークをレポートします。

構文

```
report_clocknetworks [-file arg] [-append] [-name arg]
[-return_string] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-append	オプション		結果をファイルに追加します。上書きはしません。
-name	オプション		GUI パネルに表示する結果の名前を指定します。
-return_string	オプション		レポートを文字列として返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

report_clocks

クロックをレポートします。

構文

```
report_clocks [-file arg] [-append] [-return_string] [-quiet]
[clocks]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-append	オプション		結果をファイルに追加します。上書きはしません。
-return_string	オプション		レポートを文字列として返します。
-quiet	オプション		コマンド エラーを無視します。
<i>clocks</i>	オプション	*	クロックのリストを指定します。

カテゴリ

レポート

説明

デザインに含まれるすべてのクロックを含む表を返します。返されるクロックには、伝搬されたクロック、生成されたクロック、自動生成されたクロック、仮想クロック、および反転クロックが含まれます。各クロック ネットに関するより詳細な情報は、**report_clock_utilization** コマンドを使用して取得できます。

デフォルトでは出力は標準出力に表示されますが、その後の処理用にファイルまたは文字列に出力することもできます。

引数

-file *arg*: クロック レポートを出力するファイルの名前を指定します。パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-append: report_clocks コマンドの結果を指定したファイルに上書きするのではなく追加します。

注記 : **-append** コマンドは、**-file** オプションを使用している場合にのみ使用可能です。

-return_string : 出力を Tcl 文字列として返します。Tcl 文字列は、変数定義でキャプチャしたり、解析またはその他の処理が可能です。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

clocks : 指定したパターンと一致するクロックを検索します。デフォルトのパターンはワイルドカード (*) で、デザインのすべてのクロックが返されます。複数のパターンを指定して、異なる検索条件に基づいてクロックを検索できます。

例

次の例では、現在のデザインに含まれるクロックの名前、周期、波形、およびソースが返されます。

```
report_clocks -file C:/Data/FPGA_Design/clock_out.txt
```

次の例では、デザインに含まれるクロックで、名前に「Clock」が含まれるものが返されます。

```
report_clocks *Clock*
```

関連項目

[report_clock_utilization](#)

report_config_timing

タイミング解析に影響する設定をレポートします。

構文

```
report_config_timing [-file arg] [-append] [-name arg]  
[-return_string] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		結果をファイルに出力します。
-append	オプション		結果をファイルに追加します。 上書きはしません。
-name	オプション		GUI パネルに表示する結果の 名前を指定します。
-return_string	オプション		レポートを文字列として返しま す。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

report_constraints

デザインの制約に関連する情報を表示します。

構文

```
report_constraints [-file arg] [-append] [-return_string]
[-all_violators] [-verbose] [-path_type arg] [-max_delay]
[-min_delay] [-recovery] [-removal] [-clock_gating_setup]
[-clock_gating_hold] [-min_pulse_width] [-min_period]
[-significant_digits arg] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-append	オプション		結果をファイルに追加します。上書きはしません。
-return_string	オプション		レポートを文字列として返します。
-all_violators	オプション		すべての制約違反を表示します。
-verbose	オプション		詳細情報を表示します。
-path_type	オプション	slack_only	パスレポートのフォーマットを指定します。有効な値は、end または slack_only です。
-max_delay	オプション		最大遅延とセットアップのみを表示します。
-min_delay	オプション		最小遅延とホールドのみを表示します。
-recovery	オプション		非同期リカバリののみを表示します。
-removal	オプション		非同期削除ののみを表示します。
-clock_gating_setup	オプション		クロック ゲーティングのセットアップのみを表示します。
-clock_gating_hold	オプション		クロック ゲーティングのホールドのみを表示します。
-min_pulse_width	オプション		最小パルス幅のみを表示します。
-min_period	オプション		最小周期のみを表示します。

名 前	必須/ オプション	デフォルト	説明
-significant_digits	オプション	2	有効桁数を指定します。有効な値は、0 ～ 13 です。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

[レポート](#)

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 [-file arg] [-verbose] [-sort_by args]
[-cells args] [-return_string] [-quiet]
```

戻り値

レポート

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-verbose	オプション		詳細情報を表示します。
-sort_by	オプション		並べ替えの基準を指定します。 -verbose オプションと共に使用します。
-cells	オプション		制御セットをレポートするセル/BEL インスタンスを指定します。
-return_string	オプション		レポートを文字列として返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

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 [-file arg] [-return_string] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-return_string	オプション		レポートを文字列として返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート、ChipScope

説明

現在のプロジェクトのさまざまな ChipScope デバッグ コアおよびこれらのコアのパラメーターをレポートします。デバッグ コアは、create_debug_core または read_chipscope_cdc コマンドを使用するとプロジェクトに追加できます。

レポートは、デフォルトでは Tcl コンソールまたは STD 出力に表示されますが、必要に応じてファイルに書き込むこともできます。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

-file arg : デバッグ コアのレポートを保存するファイル名を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

例

次の例では、デバッグ コアのレポートが指定したディレクトリの指定したファイルに書き込まれます。

```
report_debug_core C:/Data/FPGA_Design/project_1_cores.txt
```

関連項目

- ・ [create_debug_core](#)
- ・ [read_chipscope_cdc](#)

report_default_switching_activity

指定したデフォルト タイプのスウィッチング アクティビティを取得します。

構文

```
report_default_switching_activity [-static_probability]
[-toggle_rate] -type args [-file arg] [-return_string] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-static_probability</code>	オプション		スタティック確率をレポートします。
<code>-toggle_rate</code>	オプション		トグル レートをレポートします。
<code>-type</code>	必須		ベクターなしの伝搬エンジンで指定したタイプのデフォルトのシード値をレポートします。有効なデフォルト タイプの値は、input、input_set、input_reset、input_enable、register、dsp、bram_read_enable、bram_write_enable、output_enable、clock、all です。
<code>-file</code>	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
<code>-return_string</code>	オプション		レポートを文字列として返します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。

カテゴリ

XDC

report_drc

DRC を実行します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-name	オプション		GUI パネルに表示する結果の 名前を指定します。
-list_rules	オプション		-rules オプションに渡すことが できるルールをリストします。
-rules	オプション		DRC ルールを指定します。指 定可能なルールは、-list_rules を参照してください。
-file	オプション		DRC レポート ファイルを指定 します。
-return_string	オプション		レポートを文字列として返しま す。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

report_io

デバイスのすべての I/O サイトに関する情報を表示します。

構文

```
report_io [-file arg] [-return_string] [-quiet]
```

戻り値

レポート

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-return_string	オプション		レポートを文字列として返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

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 [-file arg] [-append] [-return_string]
[-path_type arg] [-significant_digits arg] [-input_pins]
[-verbose] [-quiet] [objects]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-append	オプション		結果をファイルに追加します。上書きはしません。
-return_string	オプション		レポートを文字列として返します。
-path_type	オプション	summary	パス レポートのフォーマットを指定します。有効な値は、summary、short、full_clock、および full_clock_expanded です。
-significant_digits	オプション	2	有効桁数を指定します。有効な値は、0 ~ 13 です。
-input_pins	オプション		パスの入力ピンを表示します。
-verbose	オプション		詳細情報を表示します。
-quiet	オプション		コマンド エラーを無視します。
objects	オプション		最小パルス幅をチェックするオブジェクトのリストを指定します。

カテゴリ

レポート

report_operating_conditions

消費電力予測の動作条件値を取得します。

構文

```
report_operating_conditions [-voltage args] [-grade] [-process]
[-junction_temp] [-ambient_temp] [-thetaja] [-thetasa] [-airflow]
[-heatsink] [-thetajb] [-board] [-board_temp] [-board_layers]
[-all] [-file arg] [-return_string] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-voltage	オプション		電圧値を取得します。サポートされる電圧は、ファミリによって異なります。
-grade	オプション		温度グレードを取得します。
-process	オプション		プロセスを取得します。
-junction_temp	オプション		ジャンクション温度を取得します。
-ambient_temp	オプション		周囲温度を取得します。
-thetaja	オプション		ThetaJA を取得します。
-thetasa	オプション		ThetaSA を取得します。
-airflow	オプション		エアフローを取得します。
-heatsink	オプション		ヒートシンクのサイズを取得します。
-thetajb	オプション		ThetaJB を取得します。
-board	オプション		ボード タイプを取得します。
-board_temp	オプション		ボード温度を取得します。
-board_layers	オプション		ボード レイヤーを取得します。
-all	オプション		このコマンドラインにリストされるすべての動作条件を取得します。
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-return_string	オプション		レポートを文字列として返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

SDC、XDC

report_param

すべてのパラメーターに関する情報を取得します。

構文

```
report_param [-quiet] [pattern]
```

戻り値

パラメーター レポート

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>pattern</code>	オプション	*	パターンに一致するパラメーターを取得します。

カテゴリ

プロパティおよびパラメーター、レポート

説明

PlanAhead でユーザー定義可能なパラメーターのリスト、現在の値、および説明をリストします。

引数

patterns : 指定したパターンに一致するパラメーターを取得します。デフォルトのパターンはワイルドカード (*) で、すべてのユーザー定義可能なパラメーターが返されます。複数のパターンを指定して、異なる検索条件に基づいてパラメーターを検索できます。

`-quiet` : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、すべてのユーザー定義可能なパラメーターの名前、値、説明が返されます。

```
report_param
```

次の例では、指定した検索パターンに一致するユーザー定義可能なパラメーターの名前、値、および説明が返されます。

```
report_param *coll*
```

関連項目

- ・ [get_param](#)
- ・ [list_param](#)
- ・ [reset_param](#)
- ・ [set_param](#)

report_power

消費電力予測を実行し、レポートを表示します。

構文

```
report_power [-no_propagation] [-file arg] [-results arg]  
[-xpe arg] [-return_string] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-no_propagation	オプション		ネットのスイッチング アクティビティを予測する伝搬エンジンをデイスエーブルにします。
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-results	オプション		結果を保存する名前を指定します。
-xpe	オプション		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。
-return_string	オプション		レポートを文字列として返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート、電力

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] [-return_string] [-quiet] object
```

戻り値

プロパティ レポート

使用法

名前	必須/ オプション	デフォルト	説明
-all	オプション		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。
-return_string	オプション		Tcl インタープリターの result 変数に report_property の実行結果を設定します。
-quiet	オプション		コマンド エラーを無視します。
<i>object</i>	必須		プロパティを取得するオブジェクトを指定します。

カテゴリ

オブジェクト、プロパティおよびパラメーター、レポート

説明

指定したオブジェクトのすべてのプロパティに関する情報を返します。この情報には、プロパティ名、プロパティタイプ、プロパティ値が含まれます。

注記：list_property でもオブジェクトのプロパティリストが返されますが、プロパティタイプと値は含まれません。

引数

-all：プロパティ値が定義されていなくても、オブジェクトのプロパティすべてを返します。

object：プロパティを取得するオブジェクトを 1 つ指定します。

注記：複数のオブジェクトを指定すると、エラー メッセージが表示されます。

-quiet：コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、指定したオブジェクトのプロパティすべてが返されます。

```
report_property -all [get_cells cpuEngine]
```

関連項目

- ・ [create_property](#)
- ・ [get_cells](#)
- ・ [get_property](#)
- ・ [list_property](#)
- ・ [list_property_value](#)
- ・ [reset_property](#)
- ・ [set_property](#)

report_resources

リソース予測を実行し、レポートを表示します。

構文

```
report_resources [-verbose] [-hierarchical] [-levels arg]
[-file arg] [-return_string] [-format arg] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-verbose	オプション		詳細情報を表示します。
-hierarchical	オプション		階層ごとに予測をレポートします。
-levels	オプション	1	レポートする階層レベル数を指定します。
-file	オプション		結果を出力するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-return_string	オプション		レポートを文字列として返します。
-format	オプション	table	リソース予測レポートのフォーマットを指定します。有効な値は table および xml です。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

report_route_status

配線のステータスをレポートします。デフォルトでは、物理ネットの総数、配線済みネット数、未配線のネット数、部分的に配線済みのネット数がレポートされます。配線オブジェクトを指定すると、各ノードの名前と接続を含むその配線オブジェクトの接続がレポートされます。

構文

```
report_route_status [-of_objects args] [-return_string]
[-file arg] [-append] [-verbose] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-of_objects	オプション		指定した配線オブジェクトの詳細な配線情報をレポートします。
-return_string	オプション		Tcl インタープリターの result 変数にレポートの実行結果を設定します。
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-append	オプション		結果をファイルに追加します。上書きはしません。
-verbose	オプション		デザインの各ネットに対し、詳細な配線情報を表示します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

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

戻り値

SSN レポート

使用法

名前	必須/ オプション	デフォルト	説明
-name	オプション		GUI パネルに表示する結果の名前を指定します。
-return_string	オプション		レポートを文字列として返します。
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

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 -name arg [-return_string] [-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]
```

戻り値

SSO レポート

使用法

名 前	必須/ オプション	デフォルト	説明
-name	必須		GUI パネルに表示する結果の名前を指定します。
-return_string	オプション		レポートを文字列として返します。
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-board_thickness	オプション		PCB の厚さをミル単位で指定します。
-via_diameter	オプション		ビアの直径をミル単位で指定します。
-pad_to_via_breakout_length	オプション		パッドからビア ブレイクアウトまでの長さをミル単位で指定します。
-breakout_width	オプション		ブレイクアウト幅をミル単位で指定します。
-other_pcb_inductance	オプション		ほかの PCB 寄生インダクタンスをナノヘンリー単位で指定します。
-socket_inductance	オプション		ソケット インダクタンスをナノヘンリー単位で指定します。
-ground_bounce	オプション		最大グラウンド バウンスをミリボルト単位で指定します。
-output_cap	オプション		各出力ドライバーのキャパシタンスをピコファラッド単位で指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

[レポート](#)

report_stats

統計をレポートします。

構文

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

戻り値

レポート

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-cell	オプション		指定したセルの統計を記述します。
-pblock	オプション		指定した Pblock の統計を記述します。
-clock_region	オプション		指定したクロック領域の統計を記述します。
-format	オプション	TABLE	レポートフォーマットを指定します。有効な値は、TABLE、CSV、XML です。
-level	オプション	1	レポートレベルを指定します (-cell または -pblock と共に使用)。
-all	オプション		すべてのレベルをレポートします (-cell または -pblock と共に使用)。
-tables	オプション		表のタイプを指定します。有効な値は、rtlMacro、rtlPrimitive、rtlHierarchy、rtlMemory、rtlPower、rtlPower2、primitive、netBoundary、carryChain、physicalResource、io、RPM、clock、PRModule、PRModule、pblockOverlap、ioBank です。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

report_switching_activity

指定したオブジェクトのスウィッチング アクティビティを取得します。

構文

```
report_switching_activity [-static_probability] [-signal_rate]  
[-file arg] [-return_string] [-quiet] object_list ...
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-static_probability</code>	オプション		スタティック確率をレポートします。
<code>-signal_rate</code>	オプション		信号レートをレポートします。
<code>-file</code>	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
<code>-return_string</code>	オプション		レポートを文字列として返します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>object_list</code>	必須		オブジェクトを指定します。

カテゴリ

[XDC](#)

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]
[-setup] [-hold] [-max_paths arg] [-nworst arg]
[-path_type arg] [-input_pins] [-nets] [-slack_lesser_than arg]
[-slack_greater_than arg] [-group args] [-sort_by arg]
[-no_report_unconstrained] [-match_style arg] [-of_objects args]
[-significant_digits arg] [-file arg] [-append] [-name arg]
[-return_string] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-from	オプション		タイミング パスの開始点 (ピン、ポート、セル、またはクロック) を指定します。
-rise_from	オプション		ピン、ポート、セル、またはクロックの立ち上がりエッジを開始点として指定します。
-fall_from	オプション		ピン、ポート、セル、またはクロックの立ち下がりエッジを開始点として指定します。
-to	オプション		タイミング パスの終点 (ピン、ポート、セル、またはクロック) を指定します。
-rise_to	オプション		ピン、ポート、セル、またはクロックの立ち上がりエッジを終点として指定します。
-fall_to	オプション		ピン、ポート、セル、またはクロックの立ち下がりエッジを終点として指定します。
-through	オプション		タイミング パスの通過点 (ピン、ポート、セル、またはネット) を指定します。
-rise_through	オプション		ピン、ポート、セル、またはネットの立ち上がりエッジを通過点として指定します。
-fall_through	オプション		ピン、ポート、セル、またはネットの立ち下がりエッジを通過点として指定します。

名前	必須/ オプション	デフォルト	説明
<code>-delay_type</code>	オプション	max	パス遅延のタイプを指定します。指定可能な値は、max、min、min_max、max_rise、max_fall、min_rise、min_fall です。
<code>-setup</code>	オプション		最大遅延タイミング パスをレポートします (<code>-delay_type max</code> と同じ)。
<code>-hold</code>	オプション		最小遅延タイミング パスをレポートします (<code>-delay_type min</code> と同じ)。
<code>-max_paths</code>	オプション	1	スラック順に並べた場合に出力するパスの最大数、またはグループごとに並べた場合のパスグループごとに出力するパスの最大数を指定します。
<code>-nworst</code>	オプション	1	エンドポイントまでのワーストパスを N 個リストします。
<code>-path_type</code>	オプション	full_clock_expanded	パス レポートのフォーマットを指定します。設定可能な値は、end、summary、short、full、full_clock、full_clock_expanded です。
<code>-input_pins</code>	オプション		パスの入力ピンを表示します。
<code>-nets</code>	オプション		ネット名をリストします。
<code>-slack_lesser_than</code>	オプション	1e+30	この値よりも小さいスラックのパスを表示します。
<code>-slack_greater_than</code>	オプション	-1e+30	この値よりも大きいスラックのパスを表示します。
<code>-group</code>	オプション		指定のグループのパスのみをレポートします。
<code>-sort_by</code>	オプション	slack	パスの並べ替え順を指定します。有効な値は、group、slack です。
<code>-no_report_unconstrained</code>	オプション		制約が適用されていないパスはレポートしません。
<code>-match_style</code>	オプション	ucf	パターン一致のスタイルを指定します。有効な値は ucf、sdc です。
<code>-of_objects</code>	オプション		タイミングをレポートするパスを指定します。
<code>-significant_digits</code>	オプション	3	有効桁数を指定します。有効な値は、0 ~ 13 です。
<code>-file</code>	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
<code>-append</code>	オプション		結果をファイルに追加します。上書きはしません。

名 前	必須/ オプション	デフォルト	説明
<code>-name</code>	オプション		GUI パネルに表示する結果の名前を指定します。
<code>-return_string</code>	オプション		レポートを文字列として返します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。

カテゴリ

レポート

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 [-file arg] [-return_string]  
[-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		出力ファイルを指定します。
-return_string	オプション		レポートを文字列として返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

report_utilization

デバイス使用率を算出し、レポートを表示します。

構文

```
report_utilization [-file arg] [-append] [-pblocks args]  
[-cells args] [-return_string] [-quiet]
```

戻り値

レポート

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		結果を保存するファイルの名前を指定します。このオプションを使用しない場合、出力はコンソールに表示されます。
-append	オプション		結果をファイルに追加します。上書きはしません。
-pblocks	オプション		指定した Pblock のリストの使用率をレポートします。
-cells	オプション		指定したセルのリストの使用率をレポートします。
-return_string	オプション		レポートを文字列として返します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

reset_default_switching_activity

デフォルト タイプのスイッチング アクティビティをリセットします。

構文

```
reset_default_switching_activity [-static_probability]
[-toggle_rate] -type args [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-static_probability	オプション		スタティック確率をリセットします。
-toggle_rate	オプション		トグル レートをリセットします。
-type	必須		ベクターなしの伝搬エンジンで指定したタイプのデフォルト シード値を、ツールのデフォルトにリセットします。有効なデフォルト タイプの値は、input、input_set、input_reset、input_enable、register、dsp、bram_read_enable、bram_write_enable、output_enable、clock、all です。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC

reset_drc

DRC の結果を削除します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	オプション		削除する DRC 結果の名前を指定します。

カテゴリ

[レポート](#)

reset_ip

コンフィギャラブル IP をリセットします。

構文

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

戻り値

リセットされたファイルのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-srcset	オプション		ソース セット名を指定します。
-ips	オプション		リセットする IP を指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

CORE Generator

reset_msg_limit

メッセージ数制限をリセットします。

構文

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

戻り値

新しいメッセージ数制限

使用法

名前	必須/ オプション	デフォルト	説明
-severity	オプション	ALL	数制限をリセットするメッセージの重要度 (「ERROR」または「CRITICAL WARNING」など) を指定します。-id と共に使用することはできません。
-id	オプション		数制限をリセットするメッセージの ID (「Common-99」など) を指定します。-severity と共に使用することはできません。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

レポート

説明

PlanAhead ツールで表示されるメッセージ数の制限をリセットします。特定のメッセージ ID、特定の重要度、またはすべてのメッセージのデフォルトの数制限を元に戻すことができます。表示されるすべてのメッセージの現在のデフォルト制限数は、4,294,967,295 です。

デフォルトでは、すべてのメッセージに対する制限数がリセットされます。指定した重要度のメッセージや、指定した ID のメッセージの制限数のみをリセットすることもできます。次は PlanAhead で表示されるメッセージの例です。

INFO: [common-99] This is an example INFO message

CRITICAL WARNING: [Netlist-1129] This message is a CRITICAL WARNING and requires user attention

注記：特定のメッセージ ID の重要度を変更するには、**set_msg_severity** コマンドを使用します。

引数

-id value : PlanAhead のメッセージビューやほかのレポートに表示されたメッセージで、指定した ID のものの数をリセットします。特定のメッセージ ID を指定します。上記の例の場合、メッセージ ID は common-99 および Netlist-1129 になります。

-severity value : 指定した重要度のメッセージの数をリセットします。次の 5 種類の重要度があります。

- ・ **ERROR** : デザインの結果が使用不可となったり、ユーザーの操作がないと回避できないような問題が発生していることを示すエラー メッセージです。
- ・ **{CRITICAL WARNING}** : 入力や制約に適用されないものがあったり、FPGA ファミリには適していないものがあることを示すクリティカル警告メッセージです。修正することが強く推奨されます。注記 : これは 2 単語の値なので、{} で囲む必要があります。
- ・ **WARNING** : 制約または仕様が指定どおりに適用されず、デザインが最適な結果にならない可能性を示す警告メッセージです。修正するかどうかは、ユーザーが判断します。
- ・ **INFO** : STATUS メッセージと同じですが、重要度とメッセージ ID タグが含まれる点が異なります。INFO メッセージには、必要に応じてアンサー データベースで確認できるようにメッセージ ID が含まれます。
- ・ **STATUS** : デザインの処理に関するプロセスおよびフィードバックのステータスを示すステータス メッセージです。STATUS メッセージには、メッセージ ID は含まれません。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、すべてのメッセージのデフォルト制限数をリセットします。

```
reset_msg_limit
```

注記 : デフォルト制限値は 4,294,967,295 です。

次の例では、指定したメッセージ ID のメッセージ数制限がリセットされます。

```
reset_msg_limit -id Netlist-1129
```

関連項目

- ・ [get_msg_limit](#)
- ・ [set_msg_limit](#)
- ・ [set_msg_severity](#)

reset_msg_severity

指定した ID のメッセージの重要度をリセットします。

構文

```
reset_msg_severity [-quiet] id
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>id</code>	必須		重要度をリセットするメッセージ ID (「common-99」など) を指定します。

カテゴリ

プロジェクト

説明

指定したメッセージ ID の重要度を PlanAhead ツールのデフォルト設定に戻します。
set_msg_severity の後にこのコマンドを使用すると、特定のメッセージ ID の重要度を元のレベルに戻すことができます。

引数

`-id`: PlanAhead ツールのメッセージ ビューやほかのレポートに表示される ID を指定します。特定メッセージのメッセージ ID を指定して、その重要度を元に戻します。次は PlanAhead で表示されるメッセージの例です。

INFO: [common-99] This is an example INFO message

CRITICAL WARNING: [Netlist-1129] This message is a CRITICAL WARNING and requires user attention

上記の例の場合、メッセージ ID は common-99 および Netlist-1129 になります。

-quiet: コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、common-99 というメッセージ ID の重要度が元に戻されます。

```
reset_msg_severity common-99
```

次の例では、Netlist-1129 というメッセージ ID の重要度がリセットされます。

```
reset_msg_severity Netlist-1129
```

関連項目

[set_msg_severity](#)

reset_operating_conditions

消費電力予測の動作条件をツールのデフォルトにリセットします。

構文

```
reset_operating_conditions [-voltage args] [-grade] [-process]
[-junction_temp] [-ambient_temp] [-thetaja] [-thetasa] [-airflow]
[-heatsink] [-thetajb] [-board] [-board_temp] [-board_layers]
[-all] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-voltage	オプション		電圧値をリセットします。サポートされる電圧は、ファミリによって異なります。
-grade	オプション		温度グレードをリセットします。
-process	オプション		プロセスをリセットします。
-junction_temp	オプション		ジャンクション温度をリセットします。
-ambient_temp	オプション		周囲温度をリセットします。
-thetaja	オプション		ThetaJA をリセットします。
-thetasa	オプション		ThetaSA をリセットします。
-airflow	オプション		エアフローをリセットします。
-heatsink	オプション		ヒートシンクのサイズをリセットします。
-thetajb	オプション		ThetaJB をリセットします。
-board	オプション		ボード タイプをリセットします。
-board_temp	オプション		ボード温度をリセットします。
-board_layers	オプション		ボード レイヤーをリセットします。
-all	オプション		このコマンドラインにリストされるすべての動作条件をリセットします。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

SDC、XDC

reset_param

パラメーターをリセットします。

構文

```
reset_param [-quiet] name
```

戻り値

元のパラメーター値

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
name	必須		パラメーター名を指定します。

カテゴリ

[プロパティおよびパラメーター](#)

説明

PlanAhead ツール内で **set_param** コマンドを使用して変更したユーザー定義可能なコンフィギュレーション パラメーターをリセットします。指定したパラメーターの値がデフォルト値に戻ります。

report_param コマンドを使用すると、現在定義されているパラメーターを表示できます。

引数

name : 値をデフォルト設定にリセットするパラメーターの名前を指定します。このコマンドでリセットできるのは、一度に 1 つのパラメーターのみです。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、指定したパラメーターの値がデフォルト値にリセットされます。

```
reset_param tcl.statsThreshold
```

関連項目

- [get_param](#)
- [list_param](#)
- [report_param](#)
- [set_param](#)

reset_property

オブジェクトのプロパティをリセットします。

構文

```
reset_property [-quiet] property_name objects ...
```

戻り値

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

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>property_name</i>	必須		リセットするプロパティの名前を指定します。
<i>objects</i>	必須		プロパティをリセットするオブジェクトの名前を指定します。

カテゴリ

XDC、オブジェクト、パーシャル リコンフィギュレーション、プロパティおよびパラメーター、パーティション

説明

指定のオブジェクトの指定のプロパティをデフォルト値にリセットします。プロパティにデフォルト値が定義されていない場合は、指定のオブジェクトにそのプロパティは割り当てられません。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

property_name : リセットするプロパティの名前を指定します。

objects : プロパティを割り当てる 1 つまたは複数のオブジェクトを指定します。

例

次の例では、すべてのセルの ALL_PROPS プロパティをリセットします。

```
reset_property ALL_PROPS [get_cells]
```

関連項目

- ・ [create_property](#)
- ・ [get_cells](#)
- ・ [get_property](#)
- ・ [list_property](#)
- ・ [list_property_value](#)
- ・ [report_property](#)
- ・ [set_property](#)

reset_run

既存の run をリセットします。

構文

```
reset_run [-prev_step] [-from_step arg] [-noclean_dir]
[-quiet] run
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-prev_step</code>	オプション		最後の run ステップをリセットします。
<code>-from_step</code>	オプション		リセットする最初のステップを指定します。
<code>-noclean_dir</code>	オプション		すべての出力ファイルおよびディレクトリをディスクから削除しません。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>run</code>	必須		リセットする run を指定します。

カテゴリ

プロジェクト

説明

指定した run をインプリメント前または合成前の状態にリセットします。このコマンドは、run をリセットして再実行できるようにするために使用します。

引数

-noclean_dir : run ディレクトリのファイルを削除しません。run をリセットすると、その run を再インプリメントする際に新たに開始するため、デフォルトでは run ディレクトリとそのファイルがすべて削除されます。このオプションを使用すると、run をリセットしたときにその run のディレクトリとファイルは削除されなくなります。この場合、run を再インプリメントすると、プロジェクトの run ディレクトリに新しい run ディレクトリが作成されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

run : リセットする run を指定します。

例

次の例では、インプリメンテーション run がリセットされています。

```
reset_run impl_1
```

-noclean_dir オプションが指定されていないので、run ディレクトリとそのファイルがハード ディスクから削除されます。

次の例では、合成 run をリセットしていますが、run ディレクトリは削除されません。

```
reset_run -noclean_dir synth_1
```

この場合、synth_1 run ディレクトリは削除されないため、run を再実行すると、synth_1_2 という新しい run ディレクトリが作成されます。

reset_ssn

メモリから SSN 結果を消去します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	必須		結果のセット名を指定します。

カテゴリ

[レポート](#)

reset_sso

メモリから WASSO 結果を消去します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	必須		結果のセット名を指定します。

カテゴリ

[レポート](#)

reset_switching_activity

指定したオブジェクトのスイッチング アクティビティをリセットします。

構文

```
reset_switching_activity [-static_probability] [-signal_rate]
[-hier] -object_list args [-verbose] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-static_probability	オプション		スタティック確率をリセットします。
-signal_rate	オプション		信号レートをリセットします。
-hier	オプション		-object_list オプションで指定した階層インスタンスのスイッチング アクティビティを階層的にリセットします。このオプションは、-object_list オプションと共に使用する必要があります。
-object_list	必須		objects
-verbose	オプション		詳細モードに設定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC、電力

reset_ucf

ファイルから読み込んだフロアプラン制約を消去します。

構文

`reset_ucf [-quiet] file`

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>file</code>	必須		ファイル名を指定します。

カテゴリ

フロアプラン

resize_pblock

Pblock 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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-add	オプション		サイト範囲を追加します。
-remove	オプション		サイト範囲を削除します。
-from	オプション		移動するサイト範囲を指定します。
-to	オプション		サイト範囲の移動先を指定します。
-replace	オプション		既存の範囲をすべて削除します。
-locs	オプション	keep_all	LOC の処理方法を指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>pblock</i>	必須		サイズを変更する Pblock を指定します。

カテゴリ

フロアプラン

説明

指定した Pblock をサイズ変更、移動、削除します。Pblock は、**create_pblock** コマンドを使用して作成されており、**place_pblocks** コマンドを使用して FPGA のファブリック上に配置されている必要があります。

Pblock は、独立したまたはオーバーラップした 1 つまたは複数の矩形で定義されたスライスで構成されます。次に定義されているさまざまなオプションを使用して、矩形にスライスを追加したり、矩形からスライスを削除したり、既存の Pblock に関連付ける新しい矩形を定義したりできます。

引数

-add arg : Pblock に指定の範囲のスライスを追加します。スライスの範囲は、矩形の 1 つの角からその対角線上の角までで指定します。たとえば、「SLICE_X0Y0:SLICE_X20Y12」のように指定します。

注記 : 1 つのスライスは、Pblock に指定可能な最小エリアです。スライスより小さいエリアの Pblock を定義することはできません。

-remove arg : Pblock から指定の範囲のスライスを削除します。Pblock からスライスを削除すると、Pblock を 1 つまたは複数の矩形で定義するという要件に従うため、Pblock が複数の矩形に分割される場合があります。

-from args : **-from** と **-to** オプションはペアとして指定し、1 つの場所から別の場所に移動するスライスまたはスライス範囲を指定します。

-to args : スライスまたはスライス範囲の移動先を指定します。このオプションは、**-from** と共に使用する必要があります。

注記 : **-from** と **-to** で指定する範囲のサイズは一致している必要があります。

-locs args : Pblock を移動またはサイズ変更したときに、Pblock の配置されているロジックをどのように処理するかを指定します。有効な値は、次のとおりです。

- ・ **keep_all** : すべての配置を現在設定されているとおり保持します。**-locs** を指定しない場合、これがデフォルトです。Pblock の外に配置されているロジックは、Pblock に割り当てられなくなります。
- ・ **keep_only_fixed** : ユーザーが配置したロジック (固定) のみを保持します。自動的に配置された未固定のロジックは配置が解除されます。
- ・ **keep_none** : すべてのロジックの配置を解除します。
- ・ **move** : すべての配置を Pblock の座標に相対して移動します。
- ・ **move_unfixed** : 未固定の要素のみを移動するよう指定します。ユーザーが配置した (固定) ロジックは移動しません。
- ・ **trim** : 新しい Pblock の境界外となるロジックの配置を解除します。配置されているロジックで Pblock の境界内になるものは、配置が保持されます。
- ・ **trim_unfixed** : 新しい Pblock の境界外となる未固定の要素のみの配置を解除します。

-replace : Pblock の配置を FPGA から解除し、**place_pblocks** コマンドを使用して再配置できるようにします。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

pblock : サイズ変更、移動、または削除する Pblock の名前を指定します。

例

次の例では、指定した Pblock にスライス範囲を追加し、既存のスライス範囲を削除して移動し、現在の配置をすべて保持します。

```
resize_pblock block3 -add SLICE_X6Y67:SLICE_X11Y71 -remove SLICE_X6Y71:SLICE_X7Y71 \  
-locs keep_all
```

次の例では、指定した Pblock にスライス範囲を追加し、既存のスライス範囲を削除して移動し、新しい Pblock の境界外となるロジックの配置を解除します。その後、新しいスライス範囲を別の矩形として追加します。

```
resize_pblock block3 -add SLICE_X3Y8:SLICE_X10Y3 -remove SLICE_X6Y67:SLICE_X11Y71 \  
-locs trim  
resize_pblock block3 -add SLICE_X6Y67:SLICE_X11Y71
```

関連項目

- ・ [add_cells_to_pblock](#)
- ・ [create_pblock](#)
- ・ [place_pblocks](#)

save_design

現在のデザインを保存します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-force	オプション		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	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

説明

アクティブな制約セットの制約ファイルへの変更を保存します。このコマンドにより、制約ファイルへのすべての変更がハードドライブのプロジェクト データに書き込まれ、作業中の変更がすべて保存されます。

引数

-force : 変更されているかどうかにかかわらず、アクティブな制約ファイルを保存します。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、ファイルが変更されているかどうかにかかわらず、アクティブな制約セットの制約ファイルが保存されます。

```
save_design -force
```

関連項目

[save_design_as](#)

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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-dir	オプション		デザインを保存するディレクトリを指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	必須		新しい制約ファイルセットの名前を指定します。

カテゴリ

プロジェクト

説明

アクティブな制約セットとその制約セットに含まれる制約ファイルのローカル コピーを、新しい制約セットとして保存します。デザイン名はこのコマンドでは変更されません。

このコマンドは、現在の制約ファイルに影響を与えずに、デザインの制約への変更を保存するために使用します。これにより、さまざまな条件のデザイン制約を試すことができます。

注記：save_design_as コマンドで作成された新しい制約セットは、参照されますが、アクティブにはなりません。この制約セットをアクティブにするには、特定の run で constrset プロパティを新しい制約セットに設定する必要があります。例を参照してください。

引数

-dir arg：制約ファイルを保存するディレクトリを指定します。アクティブな制約セットからの制約ファイルは、指定したディレクトリにコピーされます。この際、新しいコピーで現在の制約ファイルからの相対パスが保持されます。

-quiet：コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

name：新しく作成する制約セットの名前を指定します。

例

次の例では、アクティブな制約セットが newCon という新しい制約セットに保存され、この制約セットの制約ファイルすべてが指定したディレクトリにコピーされます。

```
save_design_as -dir C:/Data/con1 newCon
```

次の例では、アクティブな制約セットが newCon2 という新しい制約セットに保存され、制約ファイルすべてがプロジェクト ソースの下の newCon2 制約ディレクトリにコピーされます。この後、指定した合成 run およびインプリメンテーション run の constrset プロパティが、新しい制約セットに設定されています。

```
save_design_as newCon2
set_property constrset newCon2 [get_runs synth_1]
set_property constrset newCon2 [get_runs impl_1]
```

注記：制約セットは、現在の run に対してアクティブに設定されない限り、アクティブにはなりません。

関連項目

[save_design](#)

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]
```

戻り値

保存されたプロジェクト オブジェクト

使用法

名前	必須/ オプション	デフォルト	説明
<code>-force</code>	オプション		既存のプロジェクト ディレクトリを上書きします。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>name</code>	必須		保存するプロジェクトの新しい名前を指定します。
<code>dir</code>	オプション	.	プロジェクト ファイルを保存するディレクトリを指定します。

カテゴリ

プロジェクト

説明

現在開いている PlanAhead プロジェクト ファイルを指定したディレクトリに新しい名前で保存します。

引数

`name`：この引数はパラメーター名を必要としませんが、`dir` よりも前に記述する必要があります。パラメーターがないので、最初の引数は `name`、2 つ目の引数は `dir` として認識されます。プロジェクト ファイルは、`name.ppr` という名前で保存され、指定したディレクトリ `dir` に書き込まれます。

`dir`：新しいプロジェクト ファイルを保存するディレクトリ名を指定します。指定したディレクトリが存在しない場合は、その名前の新しいディレクトリが作成されます。ディレクトリを完全なパスで指定すると、その指定したパス名が使用されます。`dir` をパスなしで指定すると、現在の作業ディレクトリまたは PlanAhead の起動ディレクトリでそのディレクトリが検索されるか、作成されます。

注記：プロジェクトを GUI モードで作成すると、ディレクトリ名 `dir` にファイル名 `name` が付いた `dir/name` という名前のプロジェクト ディレクトリが作成され、新しいプロジェクト ファイルとプロジェクト データ フォルダーが保存されます。

-force : 既存のプロジェクトを上書きします。指定した *dir* に指定のプロジェクト名が既に存在する場合、**-force** オプションを使用して既存のプロジェクトを上書きする必要があります。

注記 : 既存プロジェクトを現在 PlanAhead で開いている場合、新しいプロジェクトでディスクの既存プロジェクトが上書きされますが、両方のプロジェクトが PlanAhead で開いたままになります。この場合、**create_project** を実行する前に **close_project** コマンドを実行しておくことをお勧めします。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、アクティブ プロジェクトが myProjectDir というディレクトリに myProject という新しいプロジェクト名で保存されます。

```
save_project_as myProject myProjectDir
```

注記 : *dir* にはフォルダー名のみが指定されているので、プロジェクトは現在の作業ディレクトリか PlanAhead の起動ディレクトリに作成されます。

次の例では、現在のプロジェクトが C:/Designs/myProjectDir というディレクトリに myProject という新しいプロジェクト名で保存されます。**-force** オプションを使用すると、指定したディレクトリに同名のプロジェクトがある場合はそれが上書きされます。

```
save_project_as myProject C:/Designs/myProjectDir -force
```

select_objects

GUI でオブジェクトを選択します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>objects</i>	必須		選択するオブジェクト

カテゴリ

GUI 制御

説明

PlanAhead ツールの GUI で開いている適切なビューで、指定のオブジェクトを選択します。このコマンドは、表示目的のためにのみ使用されます。ほかのコマンドで使用するために選択されているオブジェクトを返すには、**get_selected_objects** コマンドを使用します。

select_objects コマンドでは、指定されたプライマリ オブジェクトに加えてセカンダリ オブジェクトが選択される場合があります。セカンダリ オブジェクトの選択は、[Tools] → [Options] → [Selection Rules] で定義します。選択規則の設定方法は、『PlanAhead ユーザー ガイド』(UG632) を参照してください。

オブジェクトの選択を解除するには、**unselect_objects** コマンドを使用します。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

objects : 選択するオブジェクトを指定します。

例

次の例では、デバイス上の指定したサイトが選択されます。

```
select_objects [get_sites SLICE_X56Y214]
```

関連項目

- ・ [get_selected_objects](#)
- ・ [unselect_objects](#)

set_clock_gating

消費電力で最適化するデザインのクロック ゲーティング オプションを設定します。

構文

```
set_clock_gating [-include_cells args] [-exclude_cells args]
[-include_clocks args] [-exclude_clocks args]
[-dont_touch_nets args] [-include_types args]
[-exclude_types args] [-minimum_groupsize arg] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-include_cells	オプション	なし	クロック ゲーティングに含めるインスタンスを指定します。デフォルトはトップ インスタンスです。
-exclude_cells	オプション	なし	クロック ゲーティングから除外するインスタンスを指定します。デフォルトはなしです。
-include_clocks	オプション	なし	指定したクロック ネットが供給されるインスタンスのみにクロック ゲーティングを適用します。デフォルトでは、すべてのクロックが指定されます。
-exclude_clocks	オプション	なし	指定したクロック ネットが供給されるインスタンスをクロック ゲーティングから除外します。デフォルトはなしです。
-dont_touch_nets	オプション	なし	指定したネットをクロック イネーブル ロジックで使用しないようにします。デフォルトはなしです。
-include_types	オプション	なし	クロック ゲーティングを適用するタイプを指定します。all、bram、reg、srl のいずれかを指定します。デフォルトは all です。
-exclude_types	オプション	なし	クロック ゲーティングを適用しないタイプを指定します。all、bram、reg、srl のいずれかを指定します。デフォルトはなしです。
-minimum_groupsize	オプション	8	クロックでイネーブルになるフリップフロップのグループの最小サイズを指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

電力、XDC

set_default_switching_activity

指定したタイプのデフォルトのスイッチング アクティビティを設定します。

構文

```
set_default_switching_activity [-toggle_rate arg]
[-static_probability arg] -type args [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-toggle_rate	オプション	-9999.0	トグル レート値を指定します。 0% 有効な値は 0 ~ 200% です。値は、クロックの % です。
-static_probability	オプション	-9999.0	スタティック確率値を指定します。 有効な値は 0 ~ 1 です。
-type	必須		ベクターなしの伝搬エンジン で指定したタイプのデフォルト のシード値を設定します。有効な デフォルト タイプの値は、 input、input_set、input_reset、 input_enable、register、 dsp、bram_read_enable、 bram_write_enable、 output_enable、clock、all です。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC

set_delay_model

タイミング解析用のインターコネクト遅延モデルを設定します。

構文

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

戻り値

なし

使用法

名 前	必須/ オプション	デフォルト	説明
-interconnect	オプション		タイミング解析に使用するインターコネクト遅延モデルを指定します。有効な値は estimated、none です。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

XDC

説明

タイミング解析用のインターコネクト遅延モデルを設定します。インターコネクト遅延モデルには、estimated および none の 2 つの設定があります。

estimated に設定すると、インプリメンテーション前のデバイス上のデザインの配置および接続に基づくインターコネクト遅延の予測がタイミング解析に含まれます。none に設定すると、タイミング遅延にインターコネクト遅延は含まれません。

注記：このコマンドを実行しても、その動作に関するメッセージや戻り値は返されません。

引数

-interconnect *value* (オプション)：使用する遅延モデルを指定します。有効な値は、estimated または none です。

-quiet (オプション)：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、予測値であるタイミング遅延モデルが使用されます。

```
set_delay_model -interconnect estimated
```


set_hierarchy_separator

階層区切り文字を設定します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>separator</code>	オプション	/	階層区切り文字を指定します。

カテゴリ

SDC、XDC

説明

デザインで使用する階層区切り文字を設定します。

注記：このコマンドを実行しても、その動作に関するメッセージや戻り値は返されません。

引数

`separator` (必須)：階層区切り文字として使用する文字を指定します。デフォルトの階層区切り文字は / です。有効な文字は、/、@、^、#、.、および | です。

`-quiet` (オプション)：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、階層区切り文字を | に変更しています。

```
set_hierarchy_separator |
```

関連項目

[get_hierarchy_separator](#)

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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-rise	オプション		立ち上がりキャパシタンス値を指定します (ポートのみ)。
-fall	オプション		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	オプション		最大キャパシタンス値を指定します。
-min	オプション		最小キャパシタンス値を指定します。
-subtract_pin_load	オプション		値からピン キャパシタンスを減算します (ネットのみ)。
-pin_load	オプション		ピン キャパシタンスを指定します (ポートのみ)。
-wire_load	オプション		ワイヤ キャパシタンスを指定します (ポートのみ)。
-quiet	オプション		コマンド エラーを無視します。
<i>capacitance</i>	必須		キャパシタンス値を指定します。
<i>objects</i>	必須		ポートまたはネットのリストを指定します。

カテゴリ

SDC、XDC

説明

指定したポートの負荷を指定した値に設定します。

注記：このコマンドを実行しても、その動作に関するメッセージや戻り値は返されません。

引数

-rise (オプション)：指定したポートに設定する立ち上がりキャパシタンス値を指定します。

- fall** (オプション)：指定したポートに設定する立ち下がりキャパシタンス値を指定します。
- max** (オプション)：最大キャパシタンス値を指定します。
- min** (オプション)：最小キャパシタンス値を指定します。
- subtract_pin_load** (オプション)：値からピン キャパシタンスを減算します (ネットのみ)。
- pin_load** (オプション)：指定したポートに設定するピン キャパシタンス値を指定します。
- wire_load** (オプション)：指定したポートに設定するワイヤ キャパシタンス値を指定します。
- quiet** (オプション)：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。
- <**capacitance**> (必須)：リセットする run を指定します。
- <**objects**> (必須)：キャパシタンスを設定する現在のデザインのポートのリストを指定します。

set_msg_limit

メッセージ数制限を設定します。

構文

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

戻り値

新しいメッセージ数制限

使用法

名前	必須/ オプション	デフォルト	説明
-severity	オプション	ALL	数制限を設定するメッセージの重要度 (「ERROR」または「CRITICAL WARNING」など) を指定します。-id と共に使用することはできません。
-id	オプション		数制限を設定するメッセージの ID (「Common-99」など) を指定します。-severity と共に使用することはできません。
-quiet	オプション		コマンド エラーを無視します。
<i>count</i>	必須		新しいメッセージ数制限を指定します。

カテゴリ

レポート

説明

デザイン セッション中または 1 回の起動中に PlanAhead ツールで表示されるメッセージ数を定義します。メッセージ タイプまたは ID 別の現在の制限数は、**get_msg_limit** コマンドを使用して取得できます。

特定のメッセージ ID または特定の重要度のメッセージの制限数を設定するか、すべてのメッセージの制限数を定義できます。次は PlanAhead で表示されるメッセージの例です。

INFO: [common-99] This is an example INFO message

CRITICAL WARNING: [Netlist-1129] This message is a CRITICAL WARNING and requires user attention

メッセージ ID またはメッセージの重要度を指定しない場合は、PlanAhead ツールで表示されるすべてのメッセージに対する制限数が設定されます。

注記 : 特定のメッセージ ID の重要度を変更するには、**set_msg_severity** コマンドを使用します。元のメッセージ制限数に戻すには、**reset_msg_limit** コマンドを使用します。

引数

-id value : PlanAhead ツールのメッセージ ビューやほかのレポートに表示される ID を指定します。特定のメッセージ ID を指定します。上記の例の場合、メッセージ ID は common-99 および Netlist-1129 になります。

-severity value : メッセージの重要度を指定します。次の 5 種類の重要度があります。

- ・ **ERROR** : デザインの結果が使用不可となったり、ユーザーの操作がないと回避できないような問題が発生していることを示すエラー メッセージです。
- ・ **{CRITICAL WARNING}** : 入力や制約に適用されないものがあったり、FPGA ファミリには適していないものがあることを示すクリティカル警告メッセージです。修正することが強く推奨されます。注記：これは 2 単語の値なので、{ } で囲む必要があります。
- ・ **WARNING** : 制約または仕様が指定どおりに適用されず、デザインが最適な結果にならない可能性を示す警告メッセージです。修正するかどうかは、ユーザーが判断します。
- ・ **INFO** : STATUS メッセージと同じですが、重要度とメッセージ ID タグが含まれる点が異なります。INFO メッセージには、必要に応じてアンサー データベースで確認できるようにメッセージ ID が含まれます。
- ・ **STATUS** : デザインの処理に関するプロセスおよびフィードバックのステータスを示すステータス メッセージです。STATUS メッセージには、メッセージ ID は含まれません。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

count : 特定の ID または重要度のメッセージの制限数を指定します。この引数は必須です。最大数に設定するには、-1 を使用します。

例

次の例では、すべての重要度のメッセージの制限数が 50 に設定されます。

```
set_msg_limit 50
```

注記 : この制限は特定のメッセージ ID では問題ないかもしれませんが、すべてのメッセージに対する制限数としては小さ過ぎます。

次の例では、指定したメッセージ ID のメッセージ数制限が設定されます。

```
set_msg_limit -id Netlist-1129 100
```

次の例では、すべてのメッセージの精減数が最大値に設定されます。

```
set_msg_limit -1  
reset_msg_limit
```

注記 : どちらの行もデフォルトの最大メッセージ数に戻しています。

関連項目

- ・ [get_msg_limit](#)
- ・ [reset_msg_limit](#)
- ・ [set_msg_severity](#)

set_msg_severity

指定した ID のメッセージの重要度を設定します。

構文

```
set_msg_severity [-quiet] id severity
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>id</code>	必須		重要度を設定するメッセージ ID (「common-99」など) を指定します。
<code>severity</code>	必須		メッセージの重要度 (「ERROR」または「CRITICAL WARNING」など) を指定します。

カテゴリ

プロジェクト

説明

指定したメッセージ ID の重要度を変更します (「WARNING」から「ERROR」など)。

このコマンドを使用すると、PlanAhead ツールで表示されるメッセージの重要度を特定のレベルに変更できます。次は PlanAhead で表示されるメッセージの例です。

- ・ INFO: [common-99] This is an example INFO message
- ・ CRITICAL WARNING: [Netlist-1129] This message is a CRITICAL WARNING and requires user attention

上記の例の場合、メッセージ ID は common-99 および Netlist-1129 になります。これらのメッセージ ID の重要度は、必要に応じて INFO から WARNING に、または CRITICAL WARNING から ERROR に変更できます。

注記：特定の ID のメッセージの重要度は、**reset_message_severity** コマンドを使用して元の設定に戻すことができます。

引数

`id`: PlanAhead ツールのメッセージビューやほかのレポートに表示される ID を指定します。特定のメッセージ ID を指定します。

severity：メッセージの重要度を指定します。次の 5 種類の重要度があります。

- ・ **ERROR**：デザインの結果が使用不可となったり、ユーザーの操作がないと回避できないような問題が発生していることを示すエラー メッセージです。
- ・ **{CRITICAL WARNING}**：入力や制約に適用されないものがあったり、FPGA ファミリには適していないものがあることを示すクリティカル警告メッセージです。修正することが強く推奨されます。注記：これは 2 単語の値なので、**{ }** で囲む必要があります。
- ・ **WARNING**：制約または仕様が指定どおりに適用されず、デザインが最適な結果にならない可能性を示す警告メッセージです。修正するかどうかは、ユーザーが判断します。
- ・ **INFO**：STATUS メッセージと同じですが、重要度とメッセージ ID タグが含まれる点が異なります。INFO メッセージには、必要に応じてアンサー データベースで確認できるようにメッセージ ID が含まれます。
- ・ **STATUS**：デザインの処理に関するプロセスおよびフィードバックのステータスを示すステータス メッセージです。STATUS メッセージには、メッセージ ID は含まれません。注記：STATUS メッセージにはメッセージ ID がないので、重要度を変更することはできません。

-quiet：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、メッセージ ID **common-99** の重要度を INFO から WARNING に変更しています。

```
set_msg_severity common-99 WARNING
```

これにより、標準的な INFO メッセージが WARNING メッセージになるので、該当する問題が発生したときに注意しやすくなります。

次の例では、メッセージ ID **Netlist-1129** の重要度を CRITICAL WARNING から WARNING に変更しています。

```
set_msg_severity Netlist-1129 WARNING
```

これにより、CRITICAL WARNING が単純な WARNING レベルに下げられるので、該当する問題が発生したときに注意度が下がる可能性があります。

関連項目

- ・ [get_msg_limit](#)
- ・ [reset_msg_severity](#)
- ・ [set_msg_limit](#)

set_operating_conditions

消費電力予測の動作条件を設定します。

構文

```
set_operating_conditions [-voltage args] [-grade arg]
[-process arg] [-junction_temp arg] [-ambient_temp arg]
[-thetaja arg] [-thetasa arg] [-airflow arg] [-heatsink arg]
[-thetajb arg] [-board arg] [-board_temp arg] [-board_layers arg]
[-quiet]
```

戻り値

なし

使用法

名 前	必須/ オプション	デフォルト	説明
-voltage	オプション		電圧ペアのリストを指定します (例: {name value})。サポートされる電圧は、ファミリによって異なります。
-grade	オプション	commercial	電圧グレードを指定します。サポートされる値は、ファミリによって異なります。
-process	オプション	typical	プロセス データを指定します。有効な値は typical または maximum です。
-junction_temp	オプション	auto	ジャンクション温度 (C) を指定します。有効な値は auto または実際の温度 (C) です。
-ambient_temp	オプション	default	周囲温度 (C) を指定します。有効な値は default または実際の温度 (C) です。
-thetaja	オプション	auto	ThetaJA (C/W) を指定します。有効な値は auto または実際の値 (C/W) です。
-thetasa	オプション	auto	ThetaSA (C/W) を指定します。有効な値は auto または実際の値 (C/W) です。
-airflow	オプション	ファミリによって異なる	エアフロー (LFM) を指定します。有効な値は 0 ~ 750 です。
-heatsink	オプション	medium	ヒートシンクのサイズを指定します。有効な値は none、low、medium、high、custom です。
-thetajb	オプション	auto	ThetaJB (C/W) を指定します。有効な値は auto または実際の値 (C/W) です。

名前	必須/ オプション	デフォルト	説明
-board	オプション	medium	ボード タイプを指定します。 有効な値は jedec、small、 medium、large、custom です。
-board_temp	オプション		ボード温度 (C) を指定します。
-board_layers	オプション	8to11	ボードの層数を指定します。有 効な値は 4to7、8to11、12to15、 16+ です。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

SDC、XDC

説明

デザイン パフォーマンスを解析する際に使用する実際の動作条件を設定します。

注記：このコマンドを実行しても、その動作に関するメッセージや戻り値は返されません。

引数

-voltage *arg* (オプション)：電圧ペアのリストを指定します。サポートされる電圧は、ファミリによって異なります。

-grade *arg* (オプション)：ターゲット デバイスの温度グレードを指定します。サポートされる値は、ファミリによって異なります。デフォルト値は commercial です。

-process *arg* (オプション)：製造プロセス特性を指定します。有効な値は、typical または maximum です。デフォルト値は typical です。

-junction_temp *arg* (オプション)：モデリングに使用するデバイスのジャンクション温度を指定します。有効な値は auto または実際の温度 (摂氏) で、デフォルト値は auto です。

-ambient_temp *arg* (オプション)：環境周囲温度を摂氏で指定します。デフォルト値は default です。

-thetaja *arg* (オプション)：モデリングに使用する Theta-JA 熱抵抗を C/W で指定します。デフォルト値は auto です。

-thetasa *arg* (オプション)：モデリングに使用する Theta-SA 熱抵抗を C/W で指定します。デフォルト値は auto です。

-airflow [*0:750*] (オプション)：モデリングに使用するエアフローを LFM (リニア フィート/分) で指定します。デフォルト値はデバイス ファミリによって異なります。

-heatsink *arg* (オプション)：モデリングに使用するヒートシンクのプロファイルを指定します。有効な値は none、low、medium、high、custom で、デフォルト値は medium です。

-thetajb *arg* (オプション)：モデリングに使用する Theta-JB 熱抵抗を C/W で指定します。デフォルト値は auto です。

-board *arg* (オプション)：モデリングに使用するボードのサイズを指定します。有効な値は jedec、small、medium、large、custom で、デフォルト値は medium です。

-board_temp *arg* (オプション) : モデリングに使用するボードの温度を摂氏で指定します。

-board_layers *arg* (オプション) : モデリングに使用するボードの層数を指定します。有効な値は、4 ～ 7 層のボードでは 4to7、8 ～ 11 層のボードでは 8to11、12 ～ 15 層のボードでは 12to15、16 層以上のボードでは 16+ です。デフォルト値は 8to11 です。

-quiet (オプション) : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、インダストリアル温度グレードのデバイスの周囲温度を 75°C に設定しています。

```
set_operating_conditions -grade industrial -junction_temp 75
```

set_package_pin_val

1 つまたは複数のパッケージ ピンのユーザー列を設定します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-package_pins	必須		パッケージ ピン名を指定します。
-column	必須		ユーザー列名を指定します。
-value	必須		設定する値を指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

[ピン配置](#)

set_param

パラメーター値を設定します。

構文

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

戻り値

新しく設定されたパラメータ値

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	必須		パラメーター名を指定します。
<i>value</i>	必須		パラメータ値を指定します。

カテゴリ

プロパティおよびパラメーター

説明

PlanAhead ツール内でユーザーが定義可能なコンフィギュレーション パラメーターの値を定義します。これらのパラメーターを使用すると、PlanAhead のさまざまな動作をコンフィギュレーションおよび制御できます。現在定義されているパラメーターの説明は、**report_parameter** を参照してください。

reset_param コマンドを使用すると、変更されたパラメーターの値をデフォルト設定に戻すことができます。

注記：指定したパラメーターの値を -1 に設定すると、PlanAhead ツールでその機能をオフにできます。

引数

name：値を設定するパラメーターの名前を指定します。このコマンドで値を設定できるのは、一度に 1 つのパラメーターのみです。

value：指定したパラメーターを設定する値を指定します。

-quiet：コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、指定したパラメーターの値を設定しています。

```
set_param tcl.statsThreshold 15
```

関連項目

- ・ [get_param](#)
- ・ [list_param](#)
- ・ [report_param](#)
- ・ [reset_param](#)

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 ...
```

戻り値

設定された値、エラーが発生した場合は ""

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>property_name</i>	必須		設定するプロパティの名前を指定します。
<i>property_value</i>	必須		設定するプロパティの値を指定します。
<i>objects</i>	必須		プロパティを設定するオブジェクトの名前を指定します。

カテゴリ

XDC、オブジェクト、パーシャルリコンフィギュレーション、プロパティおよびパラメーター、パーティション

説明

指定のオブジェクトにプロパティと値を設定します。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

property_name : オブジェクトに設定するプロパティの名前を指定します。

注記 : *property_name* では大文字/小文字が区別されるので、適切に指定してください。

property_value : 指定のオブジェクトの *property_name* プロパティに設定する値を指定します。プロパティタイプに対して値が有効であるかどうかチェックされます。値が指定のプロパティに有効でない場合は、エラーが返されます。

objects : プロパティを割り当てる 1 つまたは複数のオブジェクトを指定します。

例

次の例では、指定したセルに TRUTH というプロパティを設定しています。

```
set_property TRUTH false [lindex [get_cells] 1]
```

次の例では、ALL_PROPS という新しいセル プロパティを作成し、デザイン内のすべてのセルに割り当て、指定のセルに設定されているすべてのプロパティをレポートします。

```
create_property ALL_PROPS cell
set_property ALL_PROPS is_here [get_cells]
report_property -all [lindex [get_cells] 0]
```

関連項目

- ・ [create_property](#)
- ・ [get_cells](#)
- ・ [get_property](#)
- ・ [list_property](#)
- ・ [list_property_value](#)
- ・ [report_property](#)
- ・ [reset_property](#)

set_speed_grade

タイミング解析に使用するスピード グレードを指定します。

構文

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

戻り値

結果文字列

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>value</code>	必須		タイミング解析に使用するスピード グレードを指定します。

カテゴリ

プロジェクト

説明

現在のデザインのターゲット デバイスのスピード グレードを設定します。このコマンドは、ターゲット デバイスのスピード グレードをタイミング解析用に変更するために使用し、開いている合成済みデザインまたはインプリメント済みデザインで実行する必要があります。通常は、`report_timing` コマンドまたはほかのタイミング コマンドの前に実行して、解析用のスピード グレードを変更します。

引数

`-quiet` : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

`value` : ターゲット デバイスのスピード グレードを指定します。有効な値は、-1、-2、-3 です。

例

次の例では、現在のデザインのデバイスのスピード グレードを -1 に設定しています。

```
set_speed_grade -1
```


set_switching_activity

指定したオブジェクトまたはデフォルトのタイプのスイッチング アクティビティを設定します。

構文

```
set_switching_activity [-toggle_rate arg] [-type args]
[-static_probability arg] [-signal_rate arg] [-hier] [-verbose]
[-quiet] [object_list ...]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-toggle_rate	オプション	-9999.0	トグル レート値を指定します。有効な値は 0 ～ 100% です。RTL 消費電力予測で使用します。
-type	オプション		RTL 消費電力予測で使用します。有効なタイプ値は、registers、inputs、outputs、inouts、ports、outputEnable、three_states、dsps、brams、bramWrite、bramEnable、clockEnable です。
-static_probability	オプション	-9999.0	スタティック確率値を指定します。0 < 値 < 1 です。
-signal_rate	オプション	-9999.0	信号レートを指定します。
-hier	オプション		-object_list オプションで指定された階層インスタンスにスイッチング アクティビティを設定します。このオプションは、-object_list オプションと共に使用する必要があります。
-verbose	オプション		詳細モードに設定します。
-quiet	オプション		コマンド エラーを無視します。
object_list	オプション		オブジェクトのリストを指定します。

カテゴリ

XDC、電力

説明

RTL 消費電力予測を実行する際に使用するトグル レートまたはスタティック確率を指定します。

注記：このコマンドを実行しても、その動作に関するメッセージや戻り値は返されません。

引数

-toggle_rate *rate* (オプション)：RTL 消費電力予測に使用するトグル レートを指定します。有効な値は 0 ～ 100 です。デフォルト値は -9999.0 です。

-type *value* (オプション)：RTL 消費電力予測で使用するスイッチング アクティビティを定義するロジック エンティティのタイプを指定します。有効な値は registers、inputs、outputs、inouts、ports、outputEnable、three_states、dsps、brams、bramWrite、bramEnable、clockEnable です。

-static_probability *value* (オプション)：解析で使用するスタティック確率を指定します。有効な値は $0 < \text{value} < 1$ で、デフォルト値は -9999.0 です。

-signal_rate *value* (オプション)：解析で使用する信号レートを指定します。デフォルト値は -9999.0 です。

-hier (オプション)：解析を階層インスタンスで実行するよう指定します。このオプションは、**-object_list** オプションと共に使用する必要があります。

-object_list *args* (オプション)：スイッチング アクティビティ設定を適用するオブジェクトのリストを指定します。

-verbose (オプション)：解析を詳細モードで実行します。

-quiet (オプション)：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、すべての DSP ブロックのトグル レートを 85% に設定しています。

```
set_switching_activity -toggle_rate 85 -type dsps -object_list [all_dsps]
```

関連項目

- [get_nets](#)
- [get_clocks](#)
- [get_cells](#)
- [get_ports](#)

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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-capacitance	オプション	pF	キャパシタンスの単位 (ファラッド) を指定します。有効な値は、kF ~ fF です。
-time	オプション	ns	時間の単位 (秒) を指定します。有効な値は、ks ~ fs です。
-current	オプション	mA	電流の単位 (アンペア) を指定します。有効な値は、kA ~ fA です。
-voltage	オプション	V	電圧の単位 (ボルト) を指定します。有効な値は、kV ~ fV です。
-power	オプション	mW	電力の単位 (ワット) を指定します。有効な値は、kW ~ fW です。
-resistance	オプション	ohm	抵抗の単位 (オーム) を指定します。有効な値は、kOhm ~ fOhm です。
-suffix	オプション		単位の接尾辞を指定します。
-digits	オプション	1	桁数を指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

SDC、XDC

説明

デザインの解析で使用するデフォルトの単位を指定します。このコマンドで単位を設定すると、その後に実行するすべての解析で指定の単位が使用されます。

注記：このコマンドを実行しても、その動作に関するメッセージや戻り値は返されません。

引数

-capacitance *value* (オプション) : キャパシタンスの単位 (ファラド) を指定します。有効な値の範囲はキロファラド (kF) からフェムトファラド (fF) までで、デフォルトのキャパシタンスの単位はピコファラド (pF) です。

-time *value* (オプション) : デフォルトの時間の単位 (秒) を指定します。有効な値の範囲はキロ秒 (ks) からフェムト秒 (fs) までで、デフォルトの時間の単位はナノ秒 (ns) です。

-current *value* (オプション) : デフォルトの電流の単位 (アンペア) を指定します。有効な値の範囲はキロアンペア (kA) からフェムトアンペア (fA) までで、デフォルトの電流の単位はミリアンペア (mA) です。

-voltage *value* (オプション) : デフォルトの電圧の単位 (ボルト) を指定します。有効な値の範囲はキロボルト (kV) からフェムトボルト (fV) までで、デフォルトの電圧の単位はボルト (V) です。

-power *value* (オプション) : デフォルトの電力の単位 (ワット) を指定します。有効な値の範囲はキロワット (kW) からフェムトワット (fW) までで、デフォルトの電力の単位はミリワット (mW) です。

-resistance *value* (オプション) : デフォルトの抵抗の単位 (オーム) を指定します。有効な値の範囲はキロオーム (kOhm) からフェムトオーム (fOhm) までで、デフォルトの抵抗の単位はオーム (Ohm) です。

-suffix *value* (オプション) : 指定の単位に使用する接尾辞を指定します。

-digits *value* (オプション) : 単位を表示する桁数を指定します。デフォルト値は 1 です。

-quiet (オプション) : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、電圧の単位をミリボルトに設定し、すべての値に 3 桁を使用するように設定しています。

```
set_units -voltage mV -digits 3
```

次の例では、上記のコマンドを実行した後、電流のデフォルト単位をアンペアに設定しています。

```
set_units -current A
```

2 つ目の `set_units` コマンドを実行した際、最初のコマンドで設定された値が無効になったり変更されたりすることはありません。

関連項目

- ・ [set_operating_conditions](#)
- ・ [set_property](#)
- ・ [set_hierarchy_separator](#)

split_diff_pair_ports

2 ポート間の差動ペアの関係を削除します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>ports</code>	必須		分割するポートを指定します。

カテゴリ

[ピン配置](#)

start_gui

PlanAhead の GUI を起動します。

構文

```
start_gui
```

戻り値

なし

カテゴリ

[GUI 制御](#)

説明

PlanAhead ツールをバッチ モードで実行しているときに、GUI モードを起動します。GUI は、現在のプロジェクト、デザイン、run の情報を読み込んで起動します。

例

次の例では、PlanAhead ツールをバッチ モードで実行しているときに、GUI モードを起動します。

```
PlanAhead% start_gui
```

関連項目

[stop_gui](#)

startgroup

グループ単位で実行を取り消し/やり直しできるコマンドシーケンスを開始します。

構文

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

戻り値

int

使用法

名前	必須/ オプション	デフォルト	説明
-try	オプション		既にグループが開始されている場合は、開始しません。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

GUI 制御

説明

グループ単位で実行を取り消し/やり直しできるコマンドシーケンスを開始します。このコマンドを endgroup コマンドと共に使用して、コマンドシーケンスを作成します。

注記：PlanAhead ツールでは、**undo** または **redo** を実行できるコマンドグループを複数作成できますが、ネストされたコマンドグループはサポートされません。**startgroup** を使用して新しいコマンドシーケンスを作成する前に、**endgroup** を使用してコマンドシーケンスを終了する必要があります。

startgroup コマンドを実行したとき、グループが既に開始している場合は整数値 0 が返され、新しいグループが開始された場合は整数値 1 が返されます。

引数

-try：新しいグループが開始された場合に 1 を返します。グループが既に開始している場合は 0 が返され、新しいグループは開始できません。

-quiet：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラーメッセージは表示されません。

例

次の例では、まず startgroup を実行し、関連するコマンドのシーケンスを実行して、endgroup を実行します。このコマンド シーケンスは、グループ単位で実行を取り消したりやり直したりすることができます。

```
startgroup
create_pblock pblock_wbArbEngine
create_pblock pblock_usbEngnSRM
add_cells_to_pblock pblock_wbArbEngine [get_cells [list wbArbEngine]] -clear_locs
add_cells_to_pblock pblock_usbEngnSRM [get_cells [list usbEngine1/usbEngineSRAM]] \
-clear_locs
endgroup
```

関連項目

- [endgroup](#)
- [redo](#)
- [undo](#)

stop_gui

PlanAhead の GUI を閉じます。

構文

```
stop_gui
```

戻り値

なし

カテゴリ

GUI 制御

説明

PlanAhead ツールの GUI モードを終了し、バッチ モードにします。バッチ モードでは、すべてのコマンドは Tcl コマンドまたは Tcl スクリプトで入力する必要があります。

例

次の例では、PlanAhead ツールの GUI モードを終了し、バッチ モードにしています。

```
stop_gui
```

関連項目

[start_gui](#)

swap_locs

2 つの位置を入れ替えます。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>aloc</code>	必須		1 つ目の位置 (ポート、セル、サイト) を指定します。bloc と同じタイプである必要があります。
<code>bloc</code>	必須		2 つ目の位置 (ポート、セル、サイト) を指定します。aloc と同じタイプである必要があります。

カテゴリ

フロアプラン

説明

2 つの類似したロジック エLEMENT に設定されている LOC 制約を入れ替えます。ロジック エLEMENT とは、FPGA のデバイス リソースに配置可能なELEMENT です。

swap_locs コマンドを実行すると、選択された 2 つのELEMENT を新しいロケーションに配置できるかどうかを確認するため、一部の DRC チェックが実行されます。いずれかのELEMENT のロケーションが何らかの理由で無効である場合、**swap_locs** コマンドでエラーが返されます。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

aloc : 入れ替える 1 つ目のロジックのロケーションを指定します。ポート、セル、またはデバイス サイトとして指定できます。

bloc : 入れ替える 2 つ目のロジックのロケーションを指定します。ポート、セル、またはデバイス サイトとして指定できます。**aloc** で指定したタイプと一致している必要があります。

例

次の例では、指定した 2 つのデバイス サイトに割り当てられているインスタンスを入れ替えています。

```
swap_locs [get_sites {OLOGIC_X2Y1}] [get_sites {OLOGIC_X2Y0}]
```

関連項目

- [get_cells](#)
- [get_ports](#)
- [get_sites](#)

undo

前のコマンドの実行を取り消します。

構文

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

戻り値

取り消すことができるタスクのリスト (-list を使用した場合)

使用法

名前	必須/ オプション	デフォルト	説明
-list	オプション		取り消すことができるタスクのリストを表示します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

GUI 制御

説明

前に実行したコマンドを取り消します。このコマンドを繰り返して使用し、一連のコマンドを取り消すことができます。

startgroup および **endgroup** コマンドを使用してコマンド グループを作成した場合、undo コマンドでコマンド グループがシーケンスとして取り消されます。undo コマンドは **endgroup** コマンドから開始し、**startgroup** コマンドに到達するまで繰り返されます。

あるコマンドに対して **undo** を実行し、その後そのコマンドを実行することにした場合は、**redo** コマンドを使用できます。

引数

-list : 取り消すことができるコマンドのリストを返します。undo コマンドを使用すると、コマンドのリストを順にさかのぼってコマンドが取り消されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、取り消すことが可能なコマンドのリストが返されます。

```
undo -list
```

関連項目

- ・ [endgroup](#)
- ・ [redo](#)
- ・ [startgroup](#)

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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-color_index	オプション		色を色インデックスで指定します。
-rgb	オプション		色を RGB で指定します。
-color	オプション		色の名前を指定します。
-quiet	オプション		コマンド エラーを無視します。
objects	オプション		ハイライトを解除するオブジェクトを指定します。

カテゴリ

GUI 制御

説明

PlanAhead ツールの GUI で使用し、**highlight_objects** コマンドでハイライトしたオブジェクトのハイライトを解除します。

このコマンドでは、下に説明する色オプションがサポートされますが、これらのオプションは指定のオブジェクトのハイライトを解除するには必要ありません。これらの色オプションは、指定の色でハイライトされているオブジェクトすべてのハイライトを解除するために使用できます。例を参照してください。

引数

-rgb args : ハイライトを解除する色を、RGB コードを使用して {R G B} の形式で指定します。たとえば、{255 255 0} は黄色を指定します。

-color_index arg : ハイライトを解除する色を色インデックスで指定します。色インデックスは、[Tools] → [Options] → [Themes] の [Highlight] の下に定義されています。テーマの設定については、『PlanAhead ユーザー ガイド』(UG632) を参照してください。

-color arg : ハイライトを解除する色を色の名前で指定します。指定可能な値は、red、green、blue、magenta、yellow、cyan、および orange です。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

objects : ハイライトを解除するオブジェクトを指定します。オブジェクトを指定しない場合は、指定した色でハイライトされているオブジェクトすべてのハイライトが解除されます。色を指定しない場合は、すべてのオブジェクトのハイライトが解除されます。

例

次の例では、選択したオブジェクトのハイライトが解除されます。

```
unhighlight_objects [get_selected_objects]
```

次の例では、現在黄色でハイライトされているオブジェクトのハイライトが解除されます。

```
unhighlight_objects -color yellow
```

関連項目

- [get_selected_objects](#)
- [highlight_objects](#)

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

戻り値

なし

使用法

名 前	必須/ オプション	デフォルト	説明
-rgb	オプション		色を RGB で指定します。
-color	オプション		色の名前を指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>objects</i>	オプション		マークを解除するオブジェクトを指定します。

カテゴリ

GUI 制御

説明

PlanAhead ツールの GUI で使用し、**mark_objects** コマンドでマークしたオブジェクトのマークを解除します。

このコマンドでは、下に説明する色オプションがサポートされますが、これらのオプションは指定のオブジェクトのマークを解除するには必要ありません。これらの色オプションは、指定の色でマークされているオブジェクトすべてのマークを解除するために使用できます。例を参照してください。

引数

-rgb args : マークを解除する色を、RGB コードを使用して {R G B} の形式で指定します。たとえば、{255 255 0} は黄色を指定します。

-color arg : マークを解除する色を色の名前で指定します。指定可能な値は、red、green、blue、magenta、yellow、cyan、および orange です。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

objects : マークを解除するオブジェクトを指定します。オブジェクトを指定しない場合は、指定した色でマークされているオブジェクトすべてのマークが解除されます。色を指定しない場合は、すべてのオブジェクトのマークが解除されます。

例

次の例では、選択したオブジェクトのマークが解除されます。

```
unmark_objects [get_selected_objects]
```

次の例では、現在黄色でマークされているオブジェクトのマークが解除されます。

```
unmark_objects -color yellow
```

関連項目

- ・ [get_selected_objects](#)
- ・ [mark_objects](#)

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]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。
<i>objects</i>	オプション		選択を解除するオブジェクトを指定します。

カテゴリ

GUI 制御

説明

select_objects コマンドで選択したオブジェクトの選択を解除します。

プライマリ オブジェクトとセカンダリ オブジェクト両方の選択が解除されます。セカンダリ オブジェクトの選択は、[Tools] → [Options] → [Selection Rules] で定義します。選択規則の設定方法は、『PlanAhead ユーザー ガイド』(UG632) を参照してください。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

objects : 選択を解除するオブジェクトを指定します。オブジェクトを指定しない場合は、すべてのオブジェクトの選択が解除されます。

例

次の例では、デバイス上の指定したサイトの選択が解除されます。

```
unselect_objects [get_sites SLICE_X56Y214]
```

次の例では、現在選択されているオブジェクトすべての選択が解除されます。

```
unselect_objects
```

関連項目

- ・ [get_selected_objects](#)
- ・ [select_objects](#)

update_design

現在のデザインのネットリストをアップデートします。

構文

```
update_design -cells args [-strict]
[-from_file arg] -from_design arg [-from_cell arg] [-quiet]
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-cells	必須		新しいサブネットリストでアップデートするセルのリストを指定します。
-strict	オプション		セルを置き換えるのに完全に一致したポートを必要とします。このオプションを設定しない場合、余分なポートが許可されます。
-from_file	オプション		新しいサブネットリストを含むファイルの名前を指定します。
-from_design	必須		新しいサブネットリストを含む、開いているネットリスト デザインの名前を指定します。
-from_cell	オプション		新しいサブネットリストを定義する from_design のセルの名前を指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

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]
```

戻り値

アップデートされたファイル

使用法

名前	必須/ オプション	デフォルト	説明
-file	必須		アップデートするインポート ファイルを指定します。
-remote_file	必須		インポートするリモート ファイルを指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

説明

指定したリモート ファイルの内容で 1 つのファイルをアップデートします。このコマンドは、元のリモート ファイルの内容でローカル ファイルをアップデートしたり、別のリモート ファイルの内容に置き換えるために使用できます。

引数

-file arg : アップデートするローカル プロジェクト ファイルを指定します。

-remote arg : ローカル ファイルを置換するリモート ファイルのパスと名前を指定します。このリモート ファイルは、指定したファイルと置換され、ローカル プロジェクト ディレクトリ構造にコピーされ、プロジェクトに追加されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、指定したファイルをアップデートしています。

```
update_file C:/Data/design1.v C:/Source/design1.v
```

注記：上書きされるローカル ファイルの方が新しくても、警告メッセージは表示されません。

関連項目

[reimport_files](#)

upgrade_ip

コンフィギャラブル IP を最新バージョンにアップグレードします。

構文

```
upgrade_ip [-srcset arg] [-latest arg] [-tonative arg]  
[-ips args] [-quiet]
```

戻り値

アップグレードされたファイルのリスト

使用法

名前	必須/ オプション	デフォルト	説明
-srcset	オプション		ソース セット名を指定します。
-latest	オプション		IP を最新バージョンにアップグレードします。
-tonative	オプション		IP をネイティブにアップグレードします。
-ips	オプション		アップグレードする IP を指定します。
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

プロジェクト

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従っているかどうかを確認します。

構文

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

戻り値

パーシャル リコンフィギュレーション レポート

使用法

名前	必須/ オプション	デフォルト	説明
-file	オプション		出力レポートファイル名を指定します。
-verbose	オプション		ログ ファイルに詳細情報を出力します。
-quiet	オプション		コマンド エラーを無視します。
runs	必須		検証するインプリメント済み run のリストを指定します。

カテゴリ

パーシャル リコンフィギュレーション

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 [-quiet]
```

戻り値

PlanAhead のバージョン

使用法

名前	必須/ オプション	デフォルト	説明
-quiet	オプション		コマンド エラーを無視します。

カテゴリ

[レポート](#)

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
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
<code>-timeout</code>	オプション	-1	run が完了するまで待機する最大時間 (分) を指定します。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>run</code>	必須		待機する run を指定します。

カテゴリ

プロジェクト

説明

指定した run が終了するまで、または指定した時間が経過するまで、Tcl コマンドの実行を停止します。

注記：このコマンドは、バッチ モードまたは Tcl スクリプトで PlanAhead ツールを実行する場合に使用します。GUI からインタラクティブに実行した場合は、無視されます。

引数

-timeout *arg* : run が終了するまで **wait_on_run** コマンドで待機する時間 (分) を指定します。これにより、指定した run が終了していなくても、この時間を越えると PlanAhead が Tcl の実行を再開します。タイムアウトを指定しない場合 (run が終了するまでの PlanAhead ツールの待機時間を制限しない場合)、デフォルト値の -1 が使用されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

run : PlanAhead ツールが終了まで待機する run の名前を指定します。**wait_on_run** コマンドを実行したときに指定した run がアクティブでない場合、エラーが返されます。

例

次の例では、impl_1 という run が起動され、指定した run が終了するまでか、1 時間経過するか、どちらか早く発生する方まで PlanAhead での Tcl の実行が停止されます。

```
launch_runs impl_1  
wait_on_run -timeout 60 impl_1
```

関連項目

[launch_runs](#)

write_bitstream

現在のデザインのビットストリームを生成します。

構文

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

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-bitgen_options	オプション		bitgen のコマンドライン オプションを指定します。
-quiet	オプション		コマンド エラーを無視します。
<i>file</i>	オプション		生成する BIT ファイルの名前を指定します。

カテゴリ

プロジェクト、ファイル入力および出力

説明

PlanAhead ツールから BitGen を呼び出して、現在のプロジェクトのビットストリーム ファイルを生成します。このコマンドは、インプリメント済みデザインで実行する必要があります。ビットストリームは、開いているインプリメント済みデザインに基づいて出力されます。

引数

-bitgen_options arg : BitGen のコマンドライン オプションを 1 つまたは複数指定します。有効な引数は、『コマンドライン ツール ユーザー ガイド』(UG628) を参照してください。

注記 : PlanAhead ツールでは、BitGen を呼び出す際に **-intstyle pa** オプションが使用されます。このオプションを **-bitgen_options** で指定すると、エラーが発生します。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

file : 生成するビットストリーム ファイルの名前を指定します。ビットストリーム ファイルのデフォルトの拡張子は **.bit** ですが、**write_bitstream** コマンドを使用する場合は、このファイルの拡張子を指定する必要があります。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

例

次の例では、指定した名前のビットストリーム ファイルが生成されます。

```
write_bitstream design1.bit
```

次の例では、ビットストリーム ファイルを指定したフォルダーに書き出し、BitGen の `-d` および `-l` コマンドライン オプションを指定しています。

```
write_bitstream -bitgen_options {-d -l -g compress -g crc:disable} C:/Data/design.bit
```

注記： `-d` コマンドライン オプションは、ビットストリーム ファイルを生成する前に DRC を実行しないよう指定し、`-l` オプションは ASCII 形式のロジック アロケーション ファイル (*file.ll*) を作成するよう指定します。また、2 つの `-g` オプションを使用して 2 つのコンフィギュレーション オプション (Compress および CRC) を指定しています。

関連項目

- ・ [launch_runs](#)
- ・ [open_impl_design](#)

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
```

戻り値

出力ファイル名

使用法

名前	必須/ オプション	デフォルト	説明
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<code>file</code>	必須		ChipScope CDC ファイル名を指定します。

カテゴリ

ファイル入力および出力、ChipScope

説明

現在のプロジェクトで定義されたデバッグ コア、ポート、信号を含む CDC (ChipScope Definition and Connection) ファイルを生成します。

ChipScope デバッグ コアは、`create_debug_core` コマンドを使用してプロジェクトに追加します。CDC ファイルには、ChipScope Pro Analyzer のソース ファイル、デスティネーション ファイル、コア パラメーター、コア設定などの情報が含まれます。

この CDC ファイルを ChipScope Analyzer にインポートし、ILA コア データおよびトリガー ポートのネット名を自動的に設定できます。生成した CDC ファイルは、`read_chipscope_cdc` コマンドを使用して、PlanAhead の別のプロジェクトの入力としても使用できます。

引数

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

file : 生成するファイル名を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

例

次の例では、`bft.cdc` という CDC ファイルが生成されます。

```
write_chipscope_cdc bft.cdc
```

この CDC ファイルには、ChipScope でデバッグされる信号と、信号のクロックドメイン、ChipScope で使用されるその他の設定が含まれます。

関連項目

- ・ [create_debug_core](#)
- ・ [read_chipscope_cdc](#)

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] [-force] [-quiet] file
```

戻り値

出力ファイル名

使用法

名前	必須/ オプション	デフォルト	説明
-mode	オプション	port	出力モードを指定します。
-force	オプション		既存のファイルを上書きします。
-quiet	オプション		コマンド エラーを無視します。
<i>file</i>	必須		ピン配置 CSV ファイルを指定します。

カテゴリ

ファイル入力および出力

説明

パッケージ ピンおよびポート配置情報を CSV (Comma Separated Value) ファイルにエクスポートします。

CSV ファイルの形式および要件は、『PlanAhead ユーザー ガイド』(UG632) を参照してください。

引数

-mode *arg* : CSV ファイルを書き出す際に使用するモードを指定します。このオプションを使用しない場合は、デフォルト モードの port が使用されます。

file : エクスポートする CSV ファイルの名前を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、現在のプロジェクトから CSV ファイルがエクスポートされます。

```
write_csv C:/Data/pinList.csv
```


関連項目

[read_csv](#)

write_edif

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_edif [-pblocks args] [-cell arg] [-force] [-quiet] file
```

戻り値

出力ファイルまたはディレクトリ名

使用法

名前	必須/ オプション	デフォルト	説明
-pblocks	オプション		指定した Pblock のネットリストをエクスポートします。-cell と共に使用することはできません。
-cell	オプション		指定したセルのネットリストをエクスポートします。-pblocks と共に使用することはできません。
-force	オプション		既存のファイルを上書きします。
-quiet	オプション		コマンド エラーを無視します。
<i>file</i>	必須		出力ファイルを指定します。 -pblocks、-cell を指定している場合は、ディレクトリを指定します。

カテゴリ

ファイル入力および出力

説明

現在のネットリストを EDIF ファイルとしてエクスポートするか、指定の Pblock または階層セルの内容を EDIF ネットリスト ファイルとして出力します。

引数

-pblocks *arg*: 指定した Pblock の内容を EDIF ネットリスト ファイルとして出力します。各 Pblock の内容が個別の EDIF ファイルに書き込まれます。

-cell *arg*: 指定した階層セルの内容を EDIF ネットリスト ファイルとして出力します。出力するように指定できるセルは 1 つのみです。

注記: -pblock と -cell オプションを同時に使用することはできません。これらはオプションですが、1 回に指定できるのはどちらか 1 つのみです。

file: 書き込む EDIF ファイルの名前を指定します。EDIF ネットリストのデフォルトのファイル拡張子は、.edn です。

注記：パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-pblock または -cell オプションのいずれかを使用する場合、この引数で各 Pblock またはセルの EDIF ネットリスト ファイルを保存するディレクトリ名を指定します。EDIF ネットリスト ファイルの名前は、その Pblock またはセルと同じになります。指定したディレクトリが存在しない場合は、エラーが返されます。

-quiet：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、デザイン全体の EDIF ネットリスト ファイルを指定したファイル名で書き出します。

```
write_edif C:/Data/edifOut.edn
```

次の例では、デザインのすべての Pblock の EDIF ネットリストが出力されます。ファイルは、指定したディレクトリに書き込まれます。

```
write_edif -pblocks [get_pblocks] C:/Data/FPGA_Design/
```

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] [-nopin] [-truncate arg] [-ibs arg]
[-pkg arg] [-quiet] file
```

戻り値

出力ファイル名

使用法

名 前	必須/ オプション	デフォルト	説明
-allmodels	オプション		アーキテクチャで使用可能なバッファ モデルをすべて含みます。デフォルトでは、フロアプランで使用されるバッファ モデルのみが含まれます。

カテゴリ

ファイル入力および出力

説明

現在のデザインのターゲット デバイスの IBIS モデルをエクスポートします。PlanAhead ツールでは、デザインからのネットリストおよびインプリメンテーションの詳細とピンごとの寄生パッケージ情報がまとめられ、デザイン専用のカスタム IBIS ファイルが作成されます。

write_ibis コマンドではデザイン情報が IBIS モデルにまとめられるので、このコマンドを実行するには RTL、ネットリスト、またはインプリメント済みデザインが開いている必要があります。

引数

-allmodels : ターゲット デバイスのバッファ モデルをすべてエクスポートします。デフォルトでは、デザインで使用されるバッファ モデルのみが記述されます。

-nopin : ダイ パッドからパッケージ ピンへのパスのピンごとのコード記述をオフにします。IBIS モデルには、[Package] セクションのすべてのピンが 1 本の RLC 伝送ラインで記述されます。デフォルトでは、データが存在する場合、パッケージのピンごとのモデリングが RLC メトリックとして [Define Package Model] セクションに含められます。

-truncate arg : 出力ファイルの信号名の最大長を指定します。最大長を超えるとその部分は表示されません。有効な値は、20、40、0 (制限なし) です。デフォルトでは、IBIS バージョン 4.2 の仕様に準拠するよう信号名は 40 文字に切り詰められます。

-ibs arg：アップデートされた汎用 IBIS モデル ファイルを指定します。これは、PlanAhead ツールのインストールに含まれる parts ディレクトリの IBIS モデルを上書きするために使用します。このオプションは、インストール ディレクトリに汎用モデルのないパーツすべてで必要となります。

-pkg arg：アップデートされたピンごとの寄生パッケージ データ ファイルを指定します。これは、PlanAhead ツールのインストールに含まれる parts ディレクトリの寄生パッケージ ファイルを上書きするために使用します。このオプションは、インストール ディレクトリに汎用モデルのないパーツすべてで必要となります。

file：エクスポートする IBIS ファイルの名前を指定します。

注記：パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-quiet：コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、ターゲット デバイスのバッファ モデルすべてがエクスポートされ、信号名が切り詰められないよう設定され、書き込むファイル名およびパスが指定されています。

```
write_ibis -allmodels -truncate 0 C:/Data/FPGA_Design/ibisOut.txt
```

write_ncd

配置を NCD ファイルにエクスポートします。

構文

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

戻り値

出力ファイル名

使用法

名 前	必須/ オプション	デフォルト	説明
-force	オプション		既存のファイルを上書きします。
-quiet	オプション		コマンド エラーを無視します。
<i>file</i>	必須		出力ファイルを指定します。

カテゴリ

ファイル入力および出力

説明

インプリメント済みデザインからザイリンクス NCD (Native Circuit Description) ファイルをエクスポートします。NCD は、インプリメンテーション中に MAP および PAR で作成され、論理的なネットリスト デザインがターゲットのザイリンクス デバイスにインプリメントできる物理デザインに変換されたファイルです。

引数

file : エクスポートする NCD ファイルの名前を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、現在のインプリメント済みデザインから NCD ファイルがエクスポートされます。

```
write_ncd C:/Data/FPGA_Design/designOut.ncd
```

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 [-force] [-quiet] file
```

戻り値

出力ファイル名

使用法

名前	必須/ オプション	デフォルト	説明
-force	オプション		既存のファイルを上書きします。
-quiet	オプション		コマンド エラーを無視します。
<i>file</i>	必須		出力ファイルを指定します。

カテゴリ

ファイル入力および出力

説明

デザインの現在の制約から PCF ファイルを作成します。PCF ファイルは、MAP で定義された物理制約とユーザーの定義した物理制約を含む ASCII ファイルです。MAP セクションは、インプリメンテーションのたびに書き直されます。ユーザー制約が最後に読み込まれ、MAP 制約が上書きされるようになっています。

PCF ファイルは、PAR、FPGA Editor、TRACE、NetGen、BitGen のオプションの入力ファイルです。

引数

file : エクスポートする PCF ファイルの名前を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、designOut.pcf という PCF ファイルが作成されます。

```
write_pcf designOut.pcf
```

関連項目

- ・ [read_ucf](#)
- ・ [write_ucf](#)

write_sdf

イベント シミュレーション用のフラットな SDF 遅延ファイルを生成します。

構文

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

戻り値

なし

使用法

名 前	必須/ オプション	デフォルト	説明
-process_corner	オプション	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。有効な値は slow および fast です。
-cell	オプション	デザイン全体	書き込むデザインのルートを指定します (例: des.subblk.cpu)。
-rename_top_module	オプション	新しいトップ モジュール名	トップ モジュール名を netlist などのカスタム名に置き換えます。
-quiet	オプション		コマンド エラーを無視します。
<i>file</i>	必須		ファイル名を指定します。

カテゴリ

ファイル入力および出力

説明

デザインのセルのタイミング遅延を標準遅延フォーマット (SDF) ファイルに記述します。

出力される SDF ファイルは、write_verilog コマンドで使用して、スタティック タイミング解析およびタイミング シミュレーション用の Verilog ネットリストを作成できます。

引数

-process_corner [fast | slow] : 指定したプロセス コーナーの遅延を記述します。遅延は fast プロセス コーナーよりも slow プロセス コーナーの方が大きくなります。有効な値は、slow または fast です。デフォルトでは、SDF ファイルは slow プロセス コーナー用に記述されます。

-cell *arg* : デザイン階層の指定したセルから SDL ファイルを記述します。デフォルトでは、デザイン全体の SDF ファイルが作成されます。

-rename_top_module *arg* : 出力 SDF ファイルのトップ モジュールの名前を変更します。

file : エクスポートする SDF ファイルの名前を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、指定したディレクトリに SDF ファイルが生成されます。

```
write_sdf C:/Data/FPGA_Design/designOut.sdf
```

関連項目

[write_verilog](#)

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 [-force] [-quiet] name file
```

戻り値

なし

使用法

名前	必須/ オプション	デフォルト	説明
-force	オプション		既存のファイルを上書きします。
-quiet	オプション		コマンド エラーを無視します。
<i>name</i>	必須		結果のセット名を指定します。
<i>file</i>	必須		結果を書き込むファイルの名前を指定します。

カテゴリ

ファイル入力および出力

説明

タイミング解析の結果を指定したファイルに書き込みます。このコマンドでは、タイミング解析は実行されず、**report_timing** コマンドで作成されたタイミング解析の結果を記述します。

write_timing では、古いタイミング パス レポートが生成されます。このファイルのフォーマットは、**report_timing -file** コマンドの出力とは異なります。

引数

name : **report_timing** コマンドで作成された結果セットの名前を指定します。指定したファイルに出力されるのは、これらのタイミング結果です。

file : 書き込むタイミング ファイルの名前を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンド ライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、タイミング解析を実行して time_1 というタイミング結果を作成し、そのタイミング結果を指定のファイルに記述しています。

```
report_timing -name time_1  
write_timing time_1 C:/Data/FPGA_Design/bft_time_1.txt
```

関連項目

[report_timing](#)

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 [-no_fixed_only] [-constraints arg] [-pblocks args]
[-cell arg] [-quiet] file
```

戻り値

出力ファイルまたはディレクトリ名

使用法

名前	必須/ オプション	デフォルト	説明
<code>-no_fixed_only</code>	オプション		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。
<code>-constraints</code>	オプション	valid	無効と示されている制約を含めます。有効な値は valid、invalid、または all です。
<code>-pblocks</code>	オプション		指定した Pblock の配置をエクスポートします。 <code>-cell</code> と共に使用することはできません。
<code>-cell</code>	オプション		指定したセルの配置をエクスポートします。 <code>-pblocks</code> と共に使用することはできません。
<code>-quiet</code>	オプション		コマンド エラーを無視します。
<i>file</i>	必須		出力ファイルを指定します。 <code>-pblocks</code> 、 <code>-cell</code> を指定している場合は、ディレクトリを指定します。

カテゴリ

ファイル入力および出力

説明

ユーザー制約ファイル (UCF) に物理制約をエクスポートします。UCF は、最上位からエクスポートするか (デフォルト)、指定の Pblock または階層セルからエクスポートできます。

このコマンドでは、アクティブな制約ファイルセットに含まれるすべての UCF ファイルの制約がエクスポートされます。複数ファイルからの制約が指定した UCF ファイルに含まれます。

引数

-no_fixed_only : 配置が固定されているかどうかに関わらず、すべての LOC を制約ファイルにエクスポートします。デフォルトでは、固定された LOC 制約のみが UCF ファイルに書き込まれます。固定された LOC はユーザーの割り当てた配置で、固定されていない LOC はツールで自動的に割り当てられた配置です。

-constraints *arg* : 有効 (valid)、無効 (invalid) と指定した制約、またはすべての制約 (valid と invalid の両方) をエクスポートします。デフォルトでは、有効な制約のみが UCF ファイルにエクスポートされますが、-constraints で VALID、INVALID、または ALL を指定できます。

-pblocks *arg* : 制約をエクスポートする Pblock を 1 つまたは複数指定します。

-cell *arg* : 制約をエクスポートする現在のデザインの階層セルの名前を指定します。制約は、指定したセルに相対的に指定した UCF ファイルに書き込まれます。

注記 : -cell オプションを指定する場合は、デザインを開いておく必要があります。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

file : 作成する UCF ファイルの名前を指定します。

注記 : パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-pblock または -cell オプションのいずれかを使用する場合、この引数で各 Pblock またはセルの UCF ファイルを保存するディレクトリ名を指定します。UCF ファイルの名前は、その Pblock またはセルと同じになります。指定したディレクトリが存在しない場合は、エラーが返されます。

例

次の例では、デザインのすべての Pblock の固定されている LOC と固定されていない LOC を含む、有効な制約と無効な制約が書き込まれます。

```
write_ucf -no_fixed_only -constraints all -pblocks [get_pblocks] C:/Data/FPGA_Design
```

関連項目

- [read_ucf](#)
- [save_design](#)
- [save_design_as](#)

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] [-lib]
[-write_all_overrides] [-rename_top arg] [-sdf_anno arg]
[-sdf_file arg] [-force] [-quiet] file
```

戻り値

出力ファイルまたはディレクトリ名

使用法

名 前	必須/ オプション	デフォルト	説明
-cell	オプション	デザイン全体	書き込むデザインのルートを指定します (例: des.subblk.cpu)。
-mode	オプション	design	有効な値は design、port、sta、sim です。
-lib	オプション		各ライブラリを個別のファイルに記述します。
-write_all_overrides	オプション		パラメーター値がプリミティブのデフォルト値と同じであっても、デザインのすべてのパラメーター値を記述します。
-rename_top	オプション	新しいトップ モジュール名	トップ モジュール名を netlist などのカスタム名に置き換えます。
-sdf_anno	オプション		sdf_annotate システム タスク文を生成するかどうかを指定します。
-sdf_file	オプション	file.sdf	SDF ファイルの完全なパスを指定します。
-force	オプション		既存のファイルを上書きします。
-quiet	オプション		コマンド エラーを無視します。
file	必須		記述するファイルを指定します。

カテゴリ

ファイル入力および出力

説明

現在のデザインまたは指定したセルの Verilog ネットリストを指定したファイルまたはディレクトリに記述します。出力は IEEE 1364-2001 準拠の Verilog HDL ファイルで、入力デザインファイルからのネットリスト情報が含まれます。

デザインまたは指定のセルの完全なネットリスト、デザインのポート リスト、シミュレーションまたはスタティック タイミング解析用の Verilog ネットリストを出力できます。

引数

-cell *arg*: 指定したセルまたはデザイン階層のブロックレベルから Verilog ネットリストを記述します。出力される Verilog ファイルには、指定したセルまたはモジュール内の情報だけが含まれます。

-mode *arg*: Verilog ファイルを記述する際に使用するモードを指定します。デフォルトでは、デザイン全体の Verilog ネットリストが記述されます。有効なモードは、次のとおりです。

- ・ **design**: デザイン全体の Verilog ネットリストを出力します。これは、すべての配置、インプリメンテーション、配線情報を含むデザインのスナップショットとして使用されます。
- ・ **port**: デザインの最上位の I/O ポートのみを出力します。
- ・ **sta**: スタティック タイミング解析 (STA) に使用する Verilog ネットリストを出力します。
- ・ **sim**: シミュレーション モデルとして使用する Verilog ネットリストを出力します。この出力ネットリストは、合成には使用できません。

-lib: デザインで使用される各ライブラリに個別の Verilog ファイルを出力します。

注記: このオプションは、以前のリリースの **-nolib** オプションに置き換わるもので、動作は反対になります。以前のリリースでは、**write_verilog** を実行したときに、**-nolib** を使用しなければ、デザインで使用される各ライブラリにデフォルトで個別の Verilog ファイルが出力されていました。今リリースでは、各ライブラリに個別の Verilog ファイルを生成するには **-lib** オプションを指定する必要があります。

-write_all_overrides [*true* | *false*]: 有効な値は **true** または **false** です。**true** に設定すると、パラメーターの値が定義されたプリミティブのデフォルト値と同じであっても、デザインのすべてのパラメーター値が Verilog ファイルに記述されます。**false** に設定すると、パラメーター値がプリミティブのデフォルト値と同じである場合は Verilog ファイルに記述されません。このオプションを **true** に設定しても結果は変わりませんが、出力される Verilog ファイルがより詳細なものとなります。

-rename_top_module *arg*: Verilog ネットリストを最上位シミュレーション テストベンチに組み込むことができるようにします。**-mode sim** を指定した場合にのみ有効です。Verilog ファイルに出力されるトップ モジュールの名前が、指定したトップ モジュール名に変更されます。

-sdf_anno [*true* | *false*]: このオプションは、**-mode sim** を指定している場合にのみ有効で、デフォルトでは *false* に設定されています。*true* に設定すると、Verilog ネットリストに **\$sdf_annotate** 文が追加されます。有効な値は、*true* (または 1) および *false* (または 0) です。

-sdf_file *arg*: SDF ファイルのパスとファイル名を指定します。指定しない場合、SDF ファイルのパスと名前は *file* で指定した Verilog ファイルと同じになり、ファイル拡張子は **.sdf** になります。

file: エクスポートする Verilog ファイルの名前を指定します。ファイル拡張子 **.v** または **.ver** を指定しない場合、指定した名前はディレクトリであると判断され、Verilog ファイルの名前はトップ モジュールと同じになり、指定したディレクトリに出力されます。

注記: パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-quiet: コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、デザイン全体の Verilog シミュレーション ネットリスト ファイルを指定したファイル名とパスで記述します。

```
write_verilog C:/Data/my_verilog.v
```

次の例では、ファイル名に拡張子 `.v` または `.ver` が指定されていないので、ディレクトリ名と判断されます。デザインのトップ モジュールと同じ名前の Verilog ファイルが指定したディレクトリに作成されます。`-mode sim` と `-sdf_anno` オプションが指定されているので、Verilog ネットリストに `$sdf_annotate` 文が追加されます。`-sdf_file` は指定されていないので、SDF ファイルの名前は Verilog ファイルと同じになり、ファイル拡張子は `.sdf` になります。

```
write_verilog C:/Data/my_verilog.net -mode sim
```

注記：この例の Verilog ファイル名は `top.v` となり、`my_verilog.net` ディレクトリに記述されます。SDF ファイルは同じディレクトリに記述され、`top.sdf` という名前になります。

関連項目

- [write_sdf](#)
- [write_vhdl](#)

write_vhdl

現在のネットリストを VHDL 形式でエクスポートします。

構文

```
write_vhdl [-cell arg] [-mode arg] [-lib] [-write_all_overrides]
[-rename_top arg] [-arch_only] [-force] [-quiet] file
```

戻り値

出力ファイルまたはディレクトリ名

使用法

名前	必須/ オプション	デフォルト	説明
-cell	オプション	デザイン全体	書き込むデザインのルートを指定します (例: des.subblk.cpu)。
-mode	オプション	sim	出力モードを指定します。有効な値は sim、port です。
-lib	オプション		各ライブラリを個別のファイルに記述します。
-write_all_overrides	オプション		パラメーター値がプリミティブのデフォルト値と同じであっても、デザインのすべてのパラメーター値を記述します。
-rename_top	オプション	新しいトップ モジュール名	トップ モジュール名を netlist などのカスタム名に置き換えます。
-arch_only	オプション		アーキテクチャのみを記述し、トップ セルにエンティティ宣言を記述しません。
-force	オプション		既存のファイルを上書きします。
-quiet	オプション		コマンド エラーを無視します。
file	必須		記述するファイルを指定します。

カテゴリ

ファイル入力および出力

説明

現在のデザインまたは指定したセルの VHDL ネットリストを指定したファイルまたはディレクトリに記述します。

出力は VHDL IEEE 1076.4 VITAL-2000 準拠の VHDL ファイルで、入力デザイン ファイルからのネットリスト情報が含まれます。デザインまたは指定のセルの完全なネットリスト、またはデザインのポートリストを出力できます。

引数

-cell arg：指定したセルまたはデザイン階層のブロックレベルから VHDL ネットリストを記述します。出力される VHDL ファイルには、指定したセルまたはモジュール内の情報だけが含まれます。

-mode arg：VHDL ファイルを記述する際に使用するモードを指定します。デフォルトでは、デザイン全体のシミュレーション ネットリストが記述されます。有効なモードは、次のとおりです。

- ・ **sim**：シミュレーション モデルとして使用する VHDL ネットリストを出力します。この出力 ネットリストは、合成には使用できません。これがデフォルト設定です。
- ・ **port**：最上位モジュールのエンティティ宣言に含まれる I/O ポートのみを出力します。

-lib：デザインで使用される各ライブラリに個別の VHDL ファイルを出力します。

注記：このオプションは、以前のリリースの **-nolib** オプションに置き換わるもので、動作は反対になります。以前のリリースでは、**write_vhdl** を実行したときに、**-nolib** を使用しなければ、デザインで使用される各ライブラリにデフォルトで個別の VHDL ファイルが出力されていました。今リリースでは、各ライブラリに個別の VHDL ファイルを生成するには **-lib** オプションを指定する必要があります。

-write_all_overrides [true | false]：有効な値は **true** または **false** です。**true** に設定すると、パラメーターの値が定義されたプリミティブのデフォルト値と同じであっても、デザインのすべてのパラメーター値が VHDL ファイルに記述されます。**false** に設定すると、パラメーター値がプリミティブのデフォルト値と同じである場合は VHDL ファイルに記述されません。このオプションを **true** に設定しても結果は変わりませんが、出力される VHDL ファイルがより詳細なものとなります。

-arch_only：最上位モジュールのエンティティ定義を含めず、アーキテクチャのみを出力します。これにより、出力 VHDL ネットリストを別のテストベンチと使用する場合に、出力 VHDL ネットリストの使用が簡略化されます。

file：エクスポートする VHDL ファイルの名前を指定します。ファイル拡張子 **.vhd** または **.vhdl** を指定しない場合、指定した名前はディレクトリであると判断され、VHDL ファイルの名前はトップ モジュールと同じになり、指定したディレクトリに出力されます。

注記：パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-quiet：コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

例

次の例では、デザイン全体の VHDL シミュレーション ネットリスト ファイルを指定したファイル名とパスで記述します。

```
write_vhdl C:/Data/bft_top.vhd
```

次の例では、最上位モジュールのエンティティ定義は VHDL ネットリストに含まれません。

```
write_vhdl C:/Data/vhdl_arch_only.vhd -arch_only
```

関連項目

[write_verilog](#)

write_xdc

作成する XDC ファイルの名前を指定します。XDC ファイルのデフォルトのファイル拡張子は .xdc です。このコマンドを -pblocks または -cell を使用して呼び出した場合、file 引数は出力ディレクトリを指定します。

構文

```
write_xdc [-no_fixed_only] [-constraints arg] [-pblocks args]
[-cell arg] [-sdc] [-no_tool_comments] [-force] [-quiet] file
```

戻り値

出力ファイルまたはディレクトリ名

使用法

名前	必須/ オプション	デフォルト	説明
-no_fixed_only	オプション		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。
-constraints	オプション	valid	無効と示されている制約を含めます。有効な値は valid、invalid、all です。
-pblocks	オプション		指定した Pblock の配置をエクスポートします。-cell と共に使用することはできません。
-cell	オプション		指定したセルの配置をエクスポートします。-pblocks と共に使用することはできません。
-sdc	オプション		すべてのタイミング制約をエクスポートします。
-no_tool_comments	オプション		UCF から変換する際にツールで生成された詳細コメントを XDC に書き込みません。
-force	オプション		既存のファイルを上書きします。
-quiet	オプション		コマンド エラーを無視します。
file	必須		出力ファイルを指定します。-pblocks、-cell を指定している場合は、ディレクトリを指定します。

カテゴリ

ファイル入力および出力

説明

物理制約をザイリンクス デザイン制約 (XDC) にエクスポートします。XDC は、最上位からエクスポートするか (デフォルト)、指定の Pblock または階層セルからエクスポートできます。

このコマンドは、UCF ファイルを含むデザインから XDC ファイルを作成するために使用できます。アクティブな制約ファイルセットからの制約はすべて、それらが複数ファイルからの制約であっても、XDC にエクスポートされます。

引数

-no_fixed_only : 配置が固定されているかどうかに関わらず、すべての LOC を制約ファイルにエクスポートします。デフォルトでは、固定された LOC 制約のみが XDC ファイルに書き込まれます。固定された LOC はユーザーの割り当てた配置で、固定されていない LOC はツールで自動的に割り当てられた配置です。

-constraints arg : 有効 (valid)、無効 (invalid) と指定した制約、またはすべての制約 (valid と invalid の両方) をエクスポートします。デフォルトでは、有効な制約のみが XDC ファイルにエクスポートされますが、-constraints で VALID、INVALID、または ALL を指定できます。

-pblocks arg : 制約をエクスポートする Pblock を 1 つまたは複数指定します。

-cell arg : 制約をエクスポートする現在のデザインの階層セルの名前を指定します。制約は、指定したセルに相対的に指定した XDC ファイルに書き込まれます。

注記： -cell オプションを指定する場合は、デザインを開いておく必要があります。

-sdc : 現在のデザインからタイミング制約のみを SDC フォーマットでエクスポートします。その他の定義された制約は、エクスポートされません。

-no_tool_comments : XDC ファイルにツール コメントが書き込まれないようにします。このオプションを指定しない場合は、XDC ファイルの作成に関するコメントおよび警告が記述されます。

-quiet : コマンドをメッセージを表示せずに実行します。コマンドライン エラーは無視され、コマンドでエラーが発生した場合でもエラー メッセージは表示されません。

file : 作成する XDC ファイルの名前を指定します。XDC ファイルのデフォルトのファイル拡張子は、.xdc です。

注記： パスをファイル名の一部として指定しない場合は、現在の作業ディレクトリまたは PlanAhead を起動したディレクトリにファイルが作成されます。

-pblock または -cell オプションのいずれかを使用する場合、この引数で各 Pblock またはセルの XDC ファイルを保存するディレクトリ名を指定します。XDC ファイルの名前は、その Pblock またはセルと同じになります。指定したディレクトリが存在しない場合は、エラーが返されます。

例

次の例では、デザインのすべての Pblock の固定されている LOC と固定されていない LOC を含む、有効な制約と無効な制約が書き込まれます。

```
write_xdc -no_fixed_only -constraints all -pblocks [get_pblocks] C:/Data/FPGA_Design
```


その他のリソース

PlanAhead ツールの使用方法の詳細は、『[PlanAhead ユーザー ガイド](#)』(UG632) に記載されています。『階層デザイン手法ガイド』(UG748) には、ザイリンクス FPGA デバイス用の階層デザインに関する情報が記載されています。

ザイリンクスのウェブサイトから、次のリソースを参照できます。

- ・ ザイリンクス用語集：
http://japan.xilinx.com/support/documentation/sw_manuals/glossary.pdf
- ・ ザイリンクス マニュアル：
<http://japan.xilinx.com/support/documentation>
- ・ ザイリンクス サポート：
<http://japan.xilinx.com/support>
- ・ 『階層デザイン手法ガイド』(UG748)

Tcl Developer Xchange

Tcl リファレンス資料は、インターネットから入手できます。ザイリンクスでは、Tcl のオープンソース ベースを管理する Tcl Developer Xchange サイトをお勧めしています。

<http://www.tcl.tk>

入門用チュートリアルは、次から入手できます。

<http://www.tcl.tk/man/tcl/tutorial/tcltutorial.html>

SDC について

SDC (Synopsys Design Constraints) は、特にタイミング解析において設計の要件をツールに渡すための業界標準です。SDC 仕様のリファレンス コピーは、次の Synopsys 社のサイトから登録をすると入手できます。

<http://www.synopsys.com/Community/Interoperability/Pages/TapinSDC.aspx>