Gowin ソフトウェア Tcl コマンド ユーザーガイド

SUG1220-2.0J, 2026-06-17

著作権について(2026)
著作権に関する全ての権利は、Guangdong Gowin Semiconductor Corporation に留保されています。

GOWIN高云、 、Gowin、Arora、LittleBee、及び GOWINSEMI は、当社により、中国、米国特許商標庁、及びその他の国において登録されています。商標又はサービスマークとして特定されたその他全ての文字やロゴは、それぞれの権利者に帰属しています。何れの団体及び個人も、当社の書面による許可を得ず、本文書の内容の一部もしくは全部を、いかなる視聴覚的、電子的、機械的、複写、録音等の手段によりもしくは形式により、伝搬又は複製をしてはなりません。

免責事項

当社は、GOWINSEMI Terms and Conditions of Sale(GOWINSEMI取引条件)に規定されている内容を除き、(明示的か又は黙示的かに拘わらず)いかなる保証もせず、また、知的財産権や材料の使用によりあなたのハードウェア、ソフトウェア、データ、又は財産が被った損害についても責任を負いません。当社は、事前の通知なく、いつでも本文書の内容を変更することができます。本文書を参照する何れの団体及び個人も、最新の文書やエラッタ(不具合情報)については、当社に問い合わせる必要があります。

バージョン履歴

日付	バージョン	説明
2025/08/29	1.0J	初版。
2025/12/31	1.1J	● 「3.1コマンドの分類」を更新。 ● set_optionにBitStream属性構成オプション-bsram_init_print_modeを追加。 ● set_optionにPlace & Route属性構成オプション-co-place_io_registersを追加。
2026/06/17	2.0J	● 「2 概要」を更新。 ● 第3章のタイトルと内容階層を更新。 ● 「4 TclによるIP生成フロー」を追加。 ● 設計ファイル暗号化コマンド「encrypt」を追加。 ● set_optionに合成属性構成オプション-opt_goal、-netlist_hierarchy、-dsp_style、-ram_style、-rom_style、-shift_reg、-map_optionを追加。

目次

図一覧

表一覧

1 本マニュアルについて

1.1 マニュアルの内容

本マニュアルは、主に Gowin ソフトウェアのコマンド・ライン・モードと関連する Tcl コマンドの使用法について説明します。

1.2 関連ドキュメント

GOWIN セミコンダクターのホームページ www.gowinsemi.com/ja から、以下の関連ドキュメントがダウンロード、参考できます：Gowin ソフトウェア ユーザーガイド(SUG100)。

1.3 用語、略語

本マニュアルで使用される用語、略語、及びその意味を表 1-1 に示します。

表 1-1 用語、略語

用語、略語	正式名称	意味
CRC	Cyclic Redundancy Check	巡回冗長検査
FPGA	Field Programmable Gate Array	フィールド・プログラマブル・ゲート・アレイ
GowinSynthesis	GowinSynthesis	GOWINセミコンダクターの合成ツール
GPA	Gowin Power Analyzer	Gowinパワーアナライザ
IP Core	Intellectual Property Core	設計資産コア
PnR	Place & Route	配置配線
SEU Handler	Single-Event Upsets Handler	シングル・イベント・アップセット・ハンドラ
Tcl	Tool Command Language	ツール・コマンド言語

1.4 テクニカル・サポートとフィードバック

GOWIN セミコンダクターは、包括的な技術サポートをご提供しています。使用に関するご質問、ご意見については、直接弊社までお問い合わせください。

ホームページ：www.gowinsemi.com/ja

E-mail：support@gowinsemi.com

2概要

2.1 Tcl の概要

ツールコマンド言語（Tool Command Language、Tcl）は、Gowin ソフトウェアに統合されたスクリプト言語です。ユーザーは Tcl コマンドを使用してプロジェクトを管理できます。

2.2 Tcl の実行

Gowin ソフトウェアは、コマンド・ライン・モードでの Tcl 実行に加え、IDE 下部の Tcl コンソールでの Tcl 実行もサポートしています。

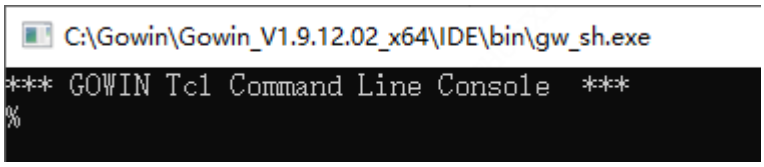
注記：

Windows システムは、/または¥¥の 2 種類のファイルパス区切り文字をサポートします。パスの区切り文字に「¥」を使用する場合は、エスケープ文字として「¥」をもう 1 つ追加する必要があります。

2.2.1 コマンド・ライン・モード

Windows を例にとると、インストールディレクトリ配下の ¥x.x¥IDE¥bin¥gw_sh.exe を起動することで、図 2-1 に示すようなコマンド・ライン・モードに入ることができます。

図 2-1 Tcl コマンド・ライン・モード



コマンド・ライン・モードでは、「単一の Tcl コマンドの入力」と「Tcl スクリプトの実行」の 2 つの方法をサポートしています。

単一の Tcl コマンドの入力

Tcl コマンドを入力した後に Enter キーを押すことで、該当するコマンドが実行されます（図 2-2）。

図 2-2 Tcl コマンド・ラインでの実行

Tcl スクリプトファイルの実行

方法 1： コマンド・ライン・モードにおいて、source コマンドで Tcl スクリプトファイルを指定し、Enter キーを押してスクリプトを実行します（図 2-3 参照）。

図 2-3 source コマンドによる Tcl スクリプトの実行

source コマンドの具体的な使用方法については、「3.3.22 source」を参照してください。

方法 2： 実行プログラム gw_sh.exe に対し、[script file]（スクリプトファイル）を指定して起動し、Enter キーを押して実行します（図 2-4）。

図 2-4 実行プログラムによる Tcl スクリプトの実行

2.2.2 GUI モード

Windows を例にとると、Gowin ソフトウェアを起動し、Gowin ソフトウェアの下部にある Tcl コンソールで Tcl を実行します（図 2-5）。

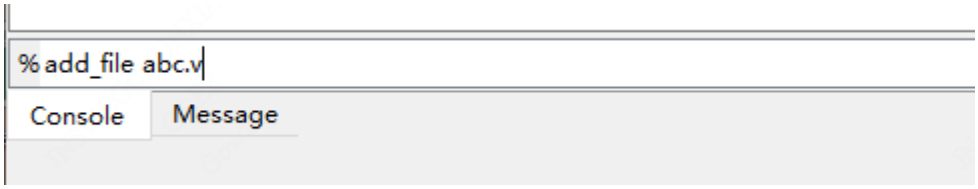
図 2-5 GUI の Tcl コンソール

GUI の Tcl コンソールでも同様に、「単一の Tcl コマンドの入力」と「Tcl スクリプトの実行」の 2 つの方法をサポートしています。

単一の Tcl コマンドの入力

このモードでの実行方法は、コマンド・ライン・モードでの実行方法と同じです（図 2-6）。

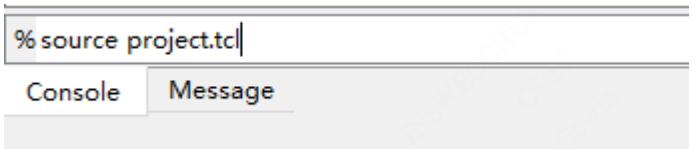
図 2-6 GUI の Tcl コンソールでの単一コマンドの実行



Tcl スクリプトファイルの実行

コマンド・ライン・モードと同様に、source コマンドを使用して Tcl スクリプトファイルを指定します（図 2-7）。

図 2-7 GUI の Tcl コンソールでの Tcl スクリプトの指定



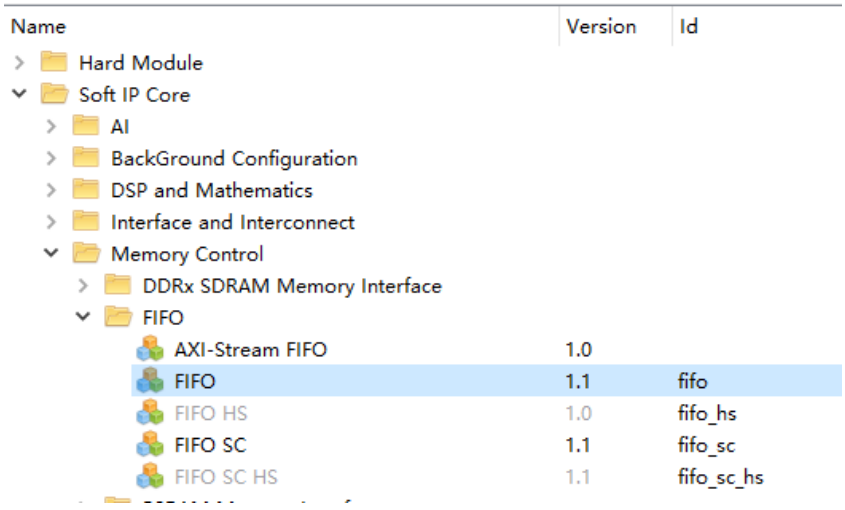
3 Tcl コマンド

Gowin ソフトウェアは、コマンド・ライン・モードでの実行をサポートします。以下の説明では、山括弧<>に含まれているコンテンツは必須コンテンツであり、角括弧[]に含まれているコンテンツはオプションのコンテンツです。「/」、「*」、「-」、スペースなどの特殊文字を含むファイル名はサポートされていません。数字で始まる IP module_name はサポートされていません。

3.1 コマンドの分類

コマンドには、IPCore 操作用の IPFlow カテゴリとプロジェクト操作用の Project カテゴリが含まれます。IPFlow カテゴリをサポートする IP の名称は、IPCore 画面上の ID を介して取得できます(図 3-1 ID の表示)。

図 3-1 ID の表示



サポートされている IP を表 3-1 に示します。

表 3-1 サポートされている IP およびその ID

IP名	Id
PLL_ADV	clock_plladv
MIPI_DPHY_RX	mipi_dphy_rx
MIPI_DPHY	mipi_dphy
MIPI_CPHY	mipi_cphy
ROM	rom
MIPI RX Advance	mipi_rx_advance
MIPI TX Advance	mipi_tx_advance
DDR3 Memory Interface	ddr3
FIFO	fifo
FIFO SC	fifo_sc
FIFO HS	fifo_hs
FIFO SC HS	fifo_sc_hs
DP	dp
SDP	sdp
SDP36KE	sdp36ke
SP	sp
Revised DP	revised_dp
Revised SP	revised_sp

IPFlow

Project

3.2 コマンド一覧

IPFlow：

create_ipc

generate_target

get_ips

list_property

read_ipc

report_property

set_property

source

write_ip_tcl

Project:

add_file

3.3 コマンドの説明

3.3.1 add_file

プロジェクトに追加される設計ファイルを指定します。

構文

add_file [-type] [-disable] [-h/--help] <file>

形式

名称	説明
[-type]	追加される設計ファイルのタイプ
[-disable]	無効な状態に設定します
[-h/--help]	ヘルプ情報を表示します
<file>	追加されるファイル

分類

Project

説明

設計ファイルを追加します。相対パスと絶対パスをサポートします。相対パスは、Gowin ソフトウェア GUI では現在プロジェクトのパスを基準にしており、コマンド・ライン・モードでは gw_sh.exe が起動された際のパスを基準にしています。

パラメータ

- <file>：追加したい設計ファイル。複数指定できます(スペースで区切ります)。
- [-type <type>]：add_file コマンドは、ファイルのサフィックスに基づいてファイルタイプを自動的に決定できますが、このオプションを使

用してファイルタイプを直接に指定することもできます。サポートされているファイルタイプ：verilog、vhdl、sv、vg、cst、sdc、gao、gpa、gsc など。

- -disable：追加したファイルを無効にします。無効な状態のファイルはファイルリストに追加されるだけで、実行されることはありません。関連コマンド：set_file_enable。
- [-h、--help]：ヘルプ情報を表示します。

例

add_file abc.v

add_file -type vhdl 1.vhd 2.vhdl 3.vhd

add_file D:/gowin_project/top.v

add_file D:¥¥gowin_project¥¥top.v

3.3.2 encrypt

プロジェクト管理エリア内の HDL 設計ファイルを暗号化します。

構文

encrypt [-verilog | -vhdl] <-i inputFile>[-o outputFile] [-l/list fileList] [-h/--help]

形式

名称	説明
[-verilog | -vhdl]	暗号化されるファイルのタイプを指定します。
<-i inputFile>	暗号化されるファイルを指定します。
[-o outputFile]	暗号化後の出力ファイルを指定します。
[-l/list fileList]	暗号化されるファイルリストを指定します。
[-h/--help]	ヘルプ情報を表示します

分類

Project

説明

プロジェクト管理エリア内の単一または複数の HDL 設計ファイルを暗号化し、暗号化されたファイルを出力します。本暗号化機能は、HDL 設計ファイルの内容を完全に暗号化するものですが、設計ファイルの module 階層構造およびその内容を変更することはありません。

パラメータ

- [-verilog | -vhdl] ：暗号化されるファイルのタイプです。指定しない場合は、ファイルの拡張子に基づいてファイルタイプが自動的に判別されます。
- <-i inputFile>：暗号化される入力ファイルです。相対パスと絶対パスの

両方をサポートしています。相対パスを使用する場合、プロジェクトファイルが存在するパスからの相対パスとなります。

- [-o outputFile]：暗号化後の出力ファイルです。この項目を指定しない場合、デフォルトで入力ファイルと同じパスに配置されます。
- [-l/list fileList]：暗号化されるファイルリストです。ファイルリスト内の複数のファイルを同時に暗号化できます。
- [-h、--help]：ヘルプ情報を表示します。

例

encrypt -verilog -i D:/gowin_project/top.v －o D:/gowin_project/top.vp

3.3.3 create_ipc

デフォルトの構成で IPC ファイルを生成します。

構文

create_ipc -name <ipName> -module_name <moduleName> [-language <arg>] [-file_name <fileName>] [-dir <path>] [-force]

形式

名称	説明
-name	IP name
-module_name	作成されるIPのモジュール名
[-language]	IPファイル、テンプレートファイル、シミュレーションファイルの言語
[-file_name]	IPファイルの名前
[-dir]	IP生成パス
[-force]	既存のファイルを上書きします

分類

IPFlow

説明

このコマンドは、IP Core Generator 内の IP の IPC ファイルを作成します。

パラメータ

- -name <ipName>：IP の名前を指定します。この名前は、IP Core Generator の画面から取得できます。
- -module_name <moduleName>：作成される IP のモジュール名を指定します。
- [-language <arg>]：生成される IP ファイル、テンプレートファイル、およびシミュレーションファイルの言語 (Verilog/VHDL)を指定します。指定しない場合は、デフォルトで Verilog が使用されます。

- [-file_name <fileName>]：生成される IP ファイルの名前を指定します。指定しない場合は、module_name で指定された名前と同じになります。
- [-dir] <path>：IP 生成パスを指定します。指定しない場合は、現在のプロジェクトの src フォルダーに生成されます。
- [-force]：既存のファイルを上書きします。

例

次の例では、現在のプロジェクトの src フォルダーに-name で指定された IP が作成され、モジュール名、言語、ファイル名が指定されます。

```
create_ipc -name fifo -module_name FIFO_Top -language Verilog -file_name fifo
```

関連項目

generate_target

3.3.4 create_project

プロジェクトを新規作成します。

構文

create_project [-name <prjName>] [-dir <path>] [-pn <pnName>] [-device_version <arg>] [-force] [-h/--help]

形式

名称	説明
[-name]	作成されるプロジェクトの名前
[-dir]	作成すされるプロジェクトのパス
[-pn]	作成されるプロジェクトの部品番号
[-device_version]	作成されるプロジェクトのデバイスバージョン
[-force]	既存のファイルを上書きします
[-h/--help]	ヘルプ情報を表示します

分類

Project

説明

プロジェクトを新規作成します。ファイルパスの形式については、add_file を参照してください。

パラメータ

- [-name <prjName>]：作成されるプロジェクトの名前を指定します。
- [-dir <path>]：作成されるプロジェクトのパスを指定します。指定したパスが存在しない場合は、新しいパスが作成されます。このオプションが指定されていない場合は、現在のプロジェクトのパスまたは tcl

スクリプトが配置されているパスが使用されます。

- [-pn <PnName>]：作成されるプロジェクトの Part Number(部品番号)を指定します。
- [-device_version <arg>]：Part Number の device version を指定します。初期バージョンのみを持つデバイスの場合、device version の値は NA です。
- [-force]：既存のプロジェクトを上書きします。
- [-h、--help]：ヘルプ情報を表示します。

例

create_project -name prj0 -dir D:/tclprj -pn GW1N-UV4LQ144C6/I5 -device_version B

create_project -name prjlest -pn GW1N-UV4LQ144C6/I5 -

device_version B -force

3.3.5 generate_target

指定された IP の IP 設計ファイル、テンプレートファイル、およびシミュレーション・ファイルを指定します。

構文

generate_target <objects> [-force]

形式

名称	説明
<objects>	IPファイルが生成されるIPを指定します。
[-force]	既存のファイルを上書きします

分類

IPFlow

説明

このコマンドは、指定された IP(get_ips)の IP 設計ファイル、テンプレートファイル、およびシミュレーション・ファイルを生成し、IP の設計ファイルを現在のプロジェクトに追加します。

パラメータ

- <objects>：設計ファイルが生成される 1 つまたは複数の IP を指定します。1 つの IP は[get_ips module_name]で指定されます。複数の IP は、[get_ips module_name0 module_name1 …] で指定されます。
- [-force]：既存のファイルを上書きします。

例

次の例では、指定された IP の関連ファイルルが生成され、現在のプ

ロジェクトに追加されます。

generate_target [get_ips FIFO_Top]

関連項目

- generate_target
- create_ipc
- read_ipc

3.3.6 get_ips

IP を指定します。

構文

get_ips <module_name>

形式

名称	説明
<module_name>	IP の module_name を指定します

分類

IPFlow

説明

現在のプロジェクトの IP を指定します。

パラメータ

<module_name>：IP の module_name(1 つまたは複数)を指定します。

例

次の例では、現在のプロジェクトに 1 つの IP が指定されます。

get_ips FIFO_Top

次の例では、現在のプロジェクトに複数の IP が指定されます。

get_ips FIFO_Top FIFO_Top_1

関連項目

- generate_target
- list_property
- report_property
- set_property

3.3.7 import_files

ファイルまたはディレクトリを現在のプロジェクトにコピーします。

構文

import_files [-file <file>] [-dir <path>] [-fileList <fileList>] [-force] [-h/--help]

形式

名称	説明
[-file]	コピーされるファイルを指定します
[-dir]	コピーされるディレクトリを指定します
[-fileList]	ファイルリストを指定します
[-force]	同じ名前の既存のファイルを上書きします
[-h/--help]	ヘルプ情報を表示します

分類

Project

説明

ファイルまたはディレクトリを現在のプロジェクトのパス/src にコピーします。-file、-dir、および-fileList は、相対パスと絶対パスをサポートします。相対パスは、Gowin ソフトウェア GUI では現在のプロジェクトのパスを基準にしており、コマンド・ライン・モードでは tcl スクリプトのパスを基準にしています。import_files の後にオプションがない場合は、add_file コマンドで指定されたすべてのファイルがプロジェクトのパス/src にコピーされます。

パラメータ

- [-file <file>]：1 つ以上のファイルをプロジェクトのパス/src に追加します。
- [-dir <path>]：パスの下にあるすべてのファイルとサブフォルダをプロジェクトのパス/src に追加します。
- [-fileList <fileList>]：リストファイルを指定します。ファイル内の各行は追加されるプロジェクトファイルです。その内容の例は次のとおりです：
  - D:/test1.v
  - D:/test2.v
  - このオプションにより、ファイル内の各行で指定されたファイルをプロジェクトのパス/src に追加できます。
- [-force]：プロジェクトのパス/src にある同名のファイルを上書きします。
- [-h/--help]：ヘルプ情報を表示します。

例

import_files -file D/test1 .v -force

import_files -file D:/test1 .v D:/test2.v -force

import_files -dir D:/sourceFile

import_files -fileList log。log ファイルの内容は次のとおりです：

D:/Test1.v

D:/Test2.v

3.3.8 list_property

IP の構成オプションをリストします。

構文

list_property <object>

形式

名称	説明
<object>	対象 IP を指定します。

分類

IPFlow

説明

指定された IP のすべてのオプションのリストを取得します。

パラメータ

<object>：IP オブジェクト。[get_ips module_name]で指定されます。

例

次の例では、指定された IP のすべての構成オプションがリストされます。

list_property [get_ips FIFO_Top]

関連項目

- report_property
- set_property

3.3.9 open_project

プロジェクトを開きます。

構文

open_project <file>] [-pn] [-device_version] [-h/--help]

形式

名称	説明
<file>	プロジェクトファイルを指定します

名称	説明
[-pn]	部品番号を指定します
[-device_version]	プロジェクトのデバイスバージョンを指定します
[-h/--help]	ヘルプ情報を表示します

分類

Project

説明

プロジェクトを開きます。開くプロジェクトのために新しい部品番号を指定することができます。ファイルパスの形式については、add_file を参照してください。

パラメータ

- <file>：開くプロジェクトファイルの名前を指定します。
- [-pn]：プロジェクトの部品番号を指定します。
- [-device_version] ：デバイスのバージョンを指定します。
- [-h/--help]：ヘルプ情報を表示します。

例

```
open_project D:¥test.gprj
```

3.3.10 read_ipc

IPC ファイルを読み出します。

構文

```
read_ipc <file>
```

形式

名称	説明
<file>	IPCファイル

分類

IPFlow

説明

指定された IPC ファイルを読み出します。ファイルパスの形式については、add_file を参照してください。

パラメータ

<file>：読み出される IPC ファイルを指定します。

例

次の例では、指定された IPC ファイルが読み出されいます。

```
read_ipc D:/gowin_project/src/fifo/fifio.ipc
```

関連項目

generate_target

3.3.11 report_property

IP のオプション名、オプションタイプ、およびオプション値を取得します。

構文

report_property <object>

形式

名称	説明
<object>	対象 IP を指定します。

分類

IPFlow

説明

指定された IP のオプション名、オプションタイプ、およびオプション値を取得します。

パラメータ

<object>：対象 IP。[get_ips module_name]で指定されます。

例

次の例では、指定された IP のすべてのオプションおよび現在のオプション値がリストされます。

report_property [get_ips FIFO_Top]

関連項目

- set_property
- list_property

3.3.12 rm_file

設計ファイルを削除します。

構文

rm_file [-h/--help] <files>

形式

名称	説明
<-files>	削除されるファイル
[-h/--help]	ヘルプ情報を表示します

分類

Project

説明

設計ファイルを削除します。ファイルパスの形式については、add_file を参照してください。

パラメータ

- <-files>：削除される設計ファイルを指定します。複数指定できます(スペースで区切ります)。
- [-h/--help]：ヘルプ情報を表示します。

例

rm_file a.v

rm_file a.v b.v c.v

rm_file D:/gowin_project/top.v

rm_file D:¥¥gowin_project¥¥top.v

3.3.13 run

プロセスを実行します。

構文

run [-h/--help] <syn/pnr/all>

形式

名称	説明
<syn/pnr/all>	実行されるプロセスを指定します
[-h/--help]	ヘルプ情報を表示します

分類

Project

説明

プロセスを実行します。

パラメータ

- <syn/pnr/all>：実行するプロセスの名前を指定します。実行可能なプロセスの名前は、syn および pnr であり、それぞれ合成および配置配線を表します。all を指定して、すべてのプロセスを実行することもできます。
- [-h/--help]：ヘルプ情報を表示します。

例

run pnr

run all

3.3.14 run close

プロジェクトを閉じます。

構文

run close

分類

Project

説明

現在のプロジェクトを閉じます。

例

run close

3.3.15 saveto

現在のプロジェクトのデータを Tcl スクリプトに保存します。

構文

saveto [-all_options] [-h/--help] <file>

形式

名称	説明
[-all_options]	すべてのオプション情報を保存します
[-h/--help]	ヘルプ情報を表示します
<file>	ファイルの名前

分類

Project

説明

現在のプロジェクトの設計データを Tcl スクリプトに保存します。ファイルパスの形式については、add_file を参照してください。

パラメータ

- [-all_options]： saveto コマンドは、デフォルトでは変更されたオプション情報、つまりデフォルト値とは異なるオプションのみを保存します。-all_options を使用して、すべてのオプション情報を保存することができます。
- [-h、--help]：ヘルプ情報を表示します。
- <file>： ファイルの名前。

例

saveto project.tcl

saveto -all_options project.tcl

saveto -all_options D:/gowin_project/project.tcl

saveto -all_options D:¥¥gowin_project¥¥project.tcl

3.3.16 set_device

デバイスの型番を設定します。

構文

set_device [-device_version <value>] [-h/--help] <part number>

形式

名称	説明
[-device_version <value>]	設定されるデバイスバージョン
[-h/--help]	ヘルプ情報を表示します
<part number>	設定される部品番号

分類

Project

説明

デバイスの型番を設定します。

パラメータ

- <part number>：ターゲットデバイスの部品番号(例えば、GW1N-UV4LQ144C6/I5)を指定します。
- [-device_version<value>]：デバイスのバージョンを指定します。サポートされるバージョンには、NA|B|C|D があります。
- [-h/--help]：ヘルプ情報を表示します。

例

set_device GW1N-LV1CS30C6/I5

set_device –device_version C GW1N-UV4LQ144C6/I5

3.3.17 set_file_enable

ファイルのイネーブル属性を設定します。

構文

set_file_enable <file> <true|false> [-h/--help]

形式

名称	説明
<file>	設定される設計ファイルを指定します。
<true|false>	ファイルを使用できるかを設定します
[-h/--help]	ヘルプ情報を表示します

分類

Project

説明

ファイルを使用できるかを設定します。ファイルパスの形式については、add_file を参照してください。

パラメータ

- <file>：設定されるファイルを指定します。
- <true | false> ：true はファイルを使用できることを意味し、false は使用できないことを意味します。
- [-h/--help]：ヘルプ情報を表示します。

例

set_file_enable top.v false

set_file_enable D:/gowin_project/top.v

set_file_enable D:¥¥gowin_project¥¥top.v

3.3.18 set_file_prop

ファイルの属性を設定します。

構文

set_file_prop <file> [-lib <name>] [-h/--help]

形式

名称	説明
<file>	設定される設計ファイルを指定します。
[-lib <name>]	ファイルのlibrary nameを設定します。
[-h/--help]	ヘルプ情報を表示します

分類

Project

説明

ファイルの属性を設定します。ファイルパスの形式については、add_file を参照してください。

パラメータ

- <file>：設定されるファイルを指定します。複数指定できます(スペースで区切ります)。
- [-lib <name>]：ファイルの library name を設定します。このオプションは、VHDL ファイルにのみ有効です。
- [-h/--help]：ヘルプ情報を表示します。

例

set_file_prop -lib work top .vhd

set_file_prop -lib work D:/gowin_project/top.vhd

set_file_prop -lib work D:¥¥gowin_project¥¥top.vhd

3.3.19 set_csr

csr ファイルを指定します。

構文

set_csr [-h/--help] <file>

形式

名称	説明
<file>	指定されるcsrファイル
[-h/--help]	ヘルプ情報を表示します

分類

Project

説明

csr ファイルを指定します。ファイルパスの形式については、add_file を参照してください。

パラメータ

- <files>：csr ファイルを指定します。
- [-h、--help]：ヘルプ情報を表示します。

例

set_csr a.csr

set_csr D:/gowin_project/a.csr

set_csr D:¥¥gowin_project¥¥a.csr

3.3.20 set_option

プロジェクトに関連する属性とプロセスのオプションを設定します。

構文

set_option [options] [-h/--help]

形式

名称	説明
[options]	属性とプロセスのオプションを設定します。
[-h/--help]	ヘルプ情報を表示します

分類

Project

説明

プロジェクトに関連する属性とプロセスのオプションを設定します。

パラメータ

- [options]：属性とプロセスのオプションを設定します。
- [-h/--help]：ヘルプ情報を表示します。

Global 属性の構成

-output_base_name

出力されるファイルの名前を指定します。

構文

-output_base_name <name>

形式

名称	説明
<name>	出力されるファイルの名前を指定します。

分類

Project

説明

出力されるファイルの名前を指定します。このオプションはファイルの base name のみを指定します。出力ファイルのタイプに応じて適切な拡張子が使用されます。例えば、-output_base_name abc の場合、gowinsynthesis によって生成されるネットリスト・ファイルの名前は abc.vg になります。

パラメータ

<name>：出力されるファイルの名前を指定します。

例

set_option -output_base_name abc

-global_freq

frequency の値を指定します。

構文

-global_freq <default|value>

形式

名称	説明
<default|value>	frequencyの値を指定します

分類

Project

説明

frequency の値を指定します。デフォルトでは、50MHz(LittleBee ファミリー)または 100MHz(Arora ファミリー)です。

パラメータ

<default|value>：frequency の値。

例

set_option -global_freq 80

合成属性の構成

-synthesis_tool

合成ツールを指定します。

構文

-synthesis_tool <tool>

形式

名称	説明
<tool>	合成ツールを指定します

分類

Project

説明

合成ツール(GowinSynthesis)を指定します。

パラメータ

<tool>：合成ツール(GowinSynthesis)を指定します。

例

set_option -synthesis_tool GowinSynthesis

-opt_goal

合成最適化の目標を指定します。

構文

-opt_goal <auto|area|timing>

形式

名称	説明
<auto|area|timing >	auto：自動最適化モード。 area：面積優先モード。 timing：タイミング優先モード。

分類

Project

説明

合成最適化の目標を指定します。デフォルトでは auto です。

パラメータ

<auto|area|timing>：合成最適化の目標。

例

set_option -opt_goal timing

-top_module

Top Module/Entity を指定します。

構文

-top_module <name>

形式

名称	説明
<name>	top moduleを指定します

分類

Project

説明

top module を指定します。

パラメータ

<name>：top module を指定します。

例

set_option -top_module test

-include_path

インクルードパスを指定します。

構文

-include_path <path or path list>

形式

名称	説明
<path or path list>	インクルードパスを指定します

分類

Project

説明

インクルードパスを指定します。複数のインクルードパスを指定する場合は、セミコロンを使用してパスを区切り、中かっこ{}ですべてのパス情報を含めます(例えば：-include_path {/path1;/path2;/path3})。相対パスと絶対パスをサポートします。

パラメータ

<path or path list>：インクルードパスを指定します。

例

set_option -include_path D:/project

-verilog_std

Verilog 言語のバージョンを指定します。

構文

-verilog_std<v1995|v2001|sysv2017>

形式

名称	説明
<v1995|v2001|sysv2017>	Verilog言語のバージョンを指定します

分類

Project

説明

Verilog 言語：Verilog 95/Verilog 2001/System Verilog 2017。デフォルトは Verilog 2001 です。

パラメータ

<v1995|v2001|sysv2017>：Verilog 言語のバージョンを指定します。

例

set_option -verilog_std v1995

-vhdl_std

VHDL 言語のバージョンを指定します。

構文

-vhdl_std <vhd1993|vhd2008|vhd2019>

形式

名称	説明
< vhd1993|vhd2008|vhd2019>	VHDL言語のバージョンを指定します

分類

Project

説明

VHDL 言語のバージョンを指定します：VHDL 1993/VHDL 2008/VHDL 2019。デフォルトは VHDL1993 です。

パラメータ

<vhd1993|vhd2008|vhd2019>：VHDL 言語のバージョンを指定します。

例

set_option -vhdl_std vhd2008

-print_all_synthesis_warning <0|1>

すべての合成警告情報を出力するかどうかを指定します。デフォルトは 0 です。

構文

-print_all_synthesis_warning <0|1>

形式

名称	説明
<0|1>	0：すべての警告情報を出力しません。 1：すべての警告情報を出力します。

分類

Project

説明

すべての合成警告情報を出力するかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：すべての合成警告情報を出力するかどうかを指定します。

例

set_option -print_all_synthesis_warning 1

-disable_io_insertion

I/O 挿入を有効または無効にします。

構文

-disable_io_insertion <0|1>

形式

名称	説明
<0|1>	0：I/O挿入を有効にします。 1：I/O挿入を無効にします。

分類

Project

説明

I/O 挿入を有効または無効にします。デフォルトは 0 です。

パラメータ

<0|1>：I/O 挿入を有効または無効にします。

例

set_option -disable_io_insertion 1

-netlist_hierarchy

生成されるネットリストが階層構造かフラット化構造かを指定します。

構文

-netlist_hierarchy <0|1>

形式

名称	説明
<0|1>	0：ネットリストはフラット構造です。 1：ネットリストは階層構造です。

分類

Project

説明

生成されるネットリストが階層構造かフラット化構造かを指定します。デフォルトは 1 です。

パラメータ

<0|1>：生成されるネットリストが階層構造かフラット化構造かを指定します。

例

set_option -netlist_hierarchy 1

-looplimit <value>

RTL 内のデフォルトのコンパイラのループ制限値。

構文

-looplimit <value>

形式

名称	説明
<value>	looplimitの値

分類

Project

説明

RTL のデフォルトのコンパイラのループ制限値で、デフォルト値は 2000 です。

パラメータ

<value>：RTL 内のデフォルトのコンパイラのループ制限値。

例

set_option -looplimit 1000

-max_fanout <value>

ファンアウト値を設定します。

構文

-max_fanout <value>

形式

名称	説明
<value>	max_fanoutの値

分類

Project

説明

入力ポート、net またはレジスタ出力ポートのファンアウト値を設定します。デフォルトは 1000 です。

パラメータ

<value>：入力ポート、net、またはレジスタの出力ポートのファンアウト値。

例

set_option -max_fanout　2000

-rw_check_on_ram

RAM の周囲にバイパスロジックを挿入します。

構文

-rw_check_on_ram <0|1>

形式

名称	説明
<0|1>	0：有効にしません。 1：有効にします。

分類

Project

説明

RAM に読み出しまたは書き込みの競合がある場合、このオプションを有効にすると、RAM の周りに、シミュレーションの不一致を防ぐためのバイパスロジックが挿入されます。デフォルトでは 0 です。

パラメータ

<0|1>：RAM の周囲にバイパスロジックを挿入することを有効または無効にします。

例

set_option -rw_check_on_ram 1

-dsp_style

DSP モジュールの実装方法を指定します。

構文

-dsp_style <auto|logic|dsp>

形式

名称	説明
<auto|logic|dsp>	auto：自動モード。 logic：ロジックで実装。 dsp：専用DSPモジュールで実装。

分類

Project

説明

DSP モジュールの実装方法を指定します。デフォルトは auto です。

パラメータ

<auto|logic|dsp>：DSP モジュールの実装方法を指定します。

例

set_option -dsp_style logic

-ram_style

RAM モジュールの実装方法を指定します。

構文

-ram_style <auto|block_ram|distributed_ram|registers>

形式

名称	説明
<auto|block_ram|distributed_ram|registers>	auto：自動モード。 block_ram：ブロックRAMで実装。 distributed_ram：分散RAMで実装。 registers：レジスタで実装。

分類

Project

説明

RAM モジュールの実装方法を指定します。デフォルトは auto です。

パラメータ

<auto|block_ram|distributed_ram|registers>：RAM モジュールの実装方法を指定します。

例

set_option -ram_style registers

-rom_style

ROM モジュールの実装方法を指定します。

構文

-rom_style <auto|block_rom|distributed_rom|logic>

形式

名称	説明
<auto|block_rom|distributed_rom|logic>	auto：自動モード。 block_rom：ブロックROMで実装。 distributed_rom：分散ROMで実装。 logic：組み合わせロジックで実装。

分類

Project

説明

ROM モジュールの実装方法を指定します。デフォルトは auto です。

パラメータ

<auto|block_rom|distributed_rom|logic>：ROM モジュールの実装方法を指定します。

例

set_option -rom_style logic

-shift_reg

シフトレジスタの実装方法を指定します。

構文

-shift_reg <auto|block_ram|distributed_ram|registers>

形式

名称	説明
<auto|block_ram|distributed_ram|registers>	auto：自動モード。 block_ram：ブロックRAMで実装。 distributed_ram：分散RAMで実装。 registers：レジスタで実装。

分類

Project

説明

シフトレジスタの実装方法を指定します。デフォルトは auto です。

パラメータ

<auto|block_ram|distributed_ram|registers>：シフトレジスタの実装方法を指定します。

例

set_option -shift_reg registers

-map_option

LUT マッピングの最適化モードを指定します。

構文

-map_option <1|2|3|4>

形式

名称	説明
<1|2|3|4>	1：デフォルトのマッピング最適化モード。 2：LUTリソースの消費を増やさない、LUT5指向のマッピング最適化モード。 3：タイミング向上のために、LUTリソースの消費増加を許容するLUT5指向のマッピング最適化モード。 4：最高のタイミング性能を得るために、LUTリソースの消費増加を許容するLUT5/LUT6指向のマッピング最適化モード。

分類

Project

説明

LUT マッピングの最適化モードを指定します。デフォルトは 1 です。

パラメータ

<1|2|3|4>：LUT マッピングの最適化モードを指定します。

例

set_option -map_option 2

Place & Route 属性の構成

-vccx

VCCX の値を指定します。

構文

-vccx <value>

形式

名称	説明
<value>	VCCXの値を指定します。

分類

Project

説明

VCCX の値を指定します。

パラメータ

<value>：VCCX の値を指定します。

例

set_option -vccx 3.3

-vcc

構文

-vcc <value>

形式

名称	説明
<value>	VCCの値を指定します。

分類

Project

説明

VCC の値を指定します。

パラメータ

<value>：VCC の値を指定します。

例

set_option -vcc 3.3

-gen_sdf

SDF ファイルを生成するかを指定します。

構文

-gen_sdf <0|1>

形式

名称	説明
<0|1>	0：SDFファイルを生成しません。 1：SDFファイルを生成します。

分類

Project

説明

Place&Route が SDF ファイルを生成するかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：SDF ファイルを生成するかを指定します。

例

set_option -gen_sdf 1

-gen_io_cst

*.io.cst という名前のポートの物理制約ファイルを生成するかどうかを指定します。

構文

-gen_io_cst <0|1>

形式

名称	説明
<0|1>	0：*.io.cstファイルを生成しません。 1：*.io.cstファイルを生成します。

分類

Project

説明

Place & Route が*.io.cst という名前のポートの物理制約ファイルを生

成するかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：ポートの物理制約ファイルを生成するかどうかを指定します。

例

set_option -gen_io_cst 1

-gen_ibis

*.ibs という名前の入出力バッファ情報指定ファイルを生成するかどうかを指定します。

構文

-gen_ibis <0|1>

形式

名称	説明
<0|1>	0：*.ibsファイルを生成しません。 1：*.ibsファイルを生成します。

分類

Project

説明

Place＆Route が*.ibs という名前の入出力バッファ情報指定ファイルを生成するかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：入出力バッファ情報指定ファイルを生成するかどうかを指定します。

例

set_option -gen_ibis 1

-gen_posp

配置ファイルを生成するかどうかを指定します。

構文

-gen_posp <0|1>

形式

名称	説明
<0|1>	0：*.pospファイルを生成しません。 1：*.pospファイルを生成します。

分類

Project

説明

Place＆Route が*.posp という名前の配置ファイルを生成するかどうかを指定します。このファイルには、BSRAM の配置情報のみが含まれます。デフォルトは 0 です。

パラメータ

<0|1>：配置ファイルを生成するかどうかを指定します。

例

set_option -gen_posp 1

-gen_text_timing_rpt

テキスト形式のタイミングレポートを生成します。

構文

-gen_text_timing_rpt <0|1>

形式

名称	説明
<0|1>	0：*.trファイルを生成しません。 1：*.trファイルを生成します。

分類

Project

説明

Place＆Route が*.tr という名前のテキスト形式のタイミングレポートを生成するかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：テキスト形式のタイミングレポートを生成するかどうかを指定します。

例

set_option -gen_text_timing_rpt 1

-gen_verilog_sim_netlist

Verilog タイミングシミュレーション・モデル・ファイルを生成するかどうかを指定します。

構文

-gen_verilog_sim_netlist <0|1>

形式

名称	説明
<0|1>	0：*.voファイルを生成しません。

	1：*.voファイルを生成します。

分類

Project

説明

Place&Route が*.vo という名前の Verilog タイミングシミュレーション・モデル・ファイルを生成するかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：Verilog タイミングシミュレーション・モデル・ファイルを生成するかどうかを指定します。

例

set_option -gen_verilog_sim_netlist 1

-gen_vhdl_sim_netlist

VHDL タイミングシミュレーション・モデル・ファイルを生成するかどうかを指定します。

構文

-gen_vhdl_sim_netlist <0|1>

形式

名称	説明
<0|1>	0：*.vhoファイルを生成しません。 1：*.vhoファイルを生成します。

分類

Project

説明

Place&Route が*.vho という名前の VHDL タイミングシミュレーション・モデル・ファイルを生成するかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：VHDL タイミングシミュレーション・モデル・ファイルを生成するかどうかを指定します。

例

set_option -gen_vhdl_sim_netlist 1

-show_init_in_vo

タイミングシシミュレーション・モデル・ファイルのインスタンスにデフォルトの初期値を追加します。

構文

-show_init_in_vo <0|1>

形式

名称	説明
<0|1>	0：タイミングシミュレーション・モデル・ファイルのインスタンスにデフォルトの初期値を追加しません。 1：タイミングシシミュレーション・モデル・ファイルのインスタンスにデフォルトの初期値を追加します。

分類

Project

説明

配置配線後のタイミングシミュレーション・モデル・ファイルのインスタンスにデフォルトの初期値を追加するかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：配置配線後のタイミングシミュレーション・モデル・ファイルのインスタンスにデフォルトの初期値を追加するかどうかを指定します。

例

set_option -show_init_in_vo 1

-show_all_warn

すべての警告情報を出力するかどうかを指定します。

構文

-show_all_warn<0|1>

形式

名称	説明
<0|1>	0：配置配線中のすべての警告情報を出力しません。 1：配置配線中のすべての警告情報を出力します。

分類

Project

説明

配置配線中にすべての警告情報を出力するかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：配置配線中にすべての警告情報を出力するかどうかを指定します。

例

set_option -show_all_warn 1

-timing_driven

タイミングドリブン配置配線を実行するかどうかを指定します。

構文

-timing_driven <0|1>

形式

名称	説明
<0|1>	0：タイミングドリブン配置配線を実行しません。 1：タイミングドリブン配置配線を実行します。

分類

Project

説明

タイミングドリブン配置配線を実行するかどうかを指定します。デフォルトは 1 です。

パラメータ

<0|1>：タイミングドリブン配置配線を実行するかどうかを指定します。

例

set_option -timing_driven 1

-cst_warn_to_error

物理制約の警告をエラーに昇格させます。

構文

-cst_warn_to_error <0|1>

形式

名称	説明
<0|1>	0：物理制約の警告をエラーに昇格させます。 1：物理制約の警告をエラーに昇格させません。

分類

Project

説明

配置配線中の物理制約の警告をエラーに昇格させるかどうかを指定し

ます。デフォルトは 1 です。

パラメータ

<0|1>：配置配線中の物理制約の警告をエラーに昇格させるかどうかを指定します。

例

set_option -cst_warn_to_error 1

-rpt_auto_place_io_info

自動配置の IO 位置情報を報告します。

構文

-rpt_auto_place_io_info <0|1>

形式

名称	説明
<0|1>	0：自動配置のIO位置情報を報告しません。 1：自動配置のIO位置情報を報告します。

分類

Project

説明

配置配線の実行中に自動配置の IO 位置情報を報告するかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：自動配置の IO 位置情報を報告するかどうかを指定します。

例

set_option -cst_warn_to_error 1

-place_option

配置アルゴリズムオプション。

構文

-place_option <0|1|2|3|4>

形式

名称	説明
<0|1|2|3|4>	0：デフォルトの配置アルゴリズムを使用します。 1：配置アルゴリズム 1 を使用します。 2：配置アルゴリズム 2 を使用します。 3：配置アルゴリズム 3 を使用します。 4：配置アルゴリズム 4 を使用します。

分類

Project

説明

配置アルゴリズムオプション。デフォルトは 0 です。

パラメータ

<0|1|2|3|4>：配置アルゴリズムオプション。

例

set_option -place_option 1

-route_option

配線アルゴリズムオプション。

構文

-route_option <0|1|2>

形式

名称	説明
<0|1|2>	0：デフォルトの配線アルゴリズムを使用します。 1：配線アルゴリズム1を使用します。 2：配線アルゴリズム2を使用します。

分類

Project

説明

配線アルゴリズムオプション。デフォルトは 0 です。

パラメータ

<0|1|2>：配線アルゴリズムオプション。

例

set_option -route_option 1

-ireg_in_iob

入力バッファに接続されるレジスタを IOB に配置します。

構文

-ireg_in_iob <0|1>

形式

名称	説明
<0|1>	0：入力バッファに接続されるレジスタをIOBに配置しません。 1：入力バッファに接続されるレジスタをIOBに配置します。

分類

Project

説明

このオプションを有効にすると、入力バッファに接続されるレジスタは IOB に配置されます。デフォルトは 1 です。

パラメータ

<0|1>：入力バッファに接続されるレジスタを IOB に配置するかどうかを指定します。

例

set_option -ireg_in_iob 1

-oreg_in_iob

出力/トライステートバッファに接続されるレジスタを IOB に配置します。

構文

-oreg_in_iob <0|1>

形式

名称	説明
<0|1>	0：出力/トライステートバッファに接続されるレジスタをIOBに配置しません。 1：出力/トライステートバッファに接続されるレジスタをIOBに配置します。

分類

Project

説明

このオプションを有効にすると、出力/トライステートバッファに接続されるレジスタは IOB に配置されます。デフォルトは 1 です。

パラメータ

<0|1>：出力/トライステートバッファに接続されるレジスタを IOB に

配置するかどうかを指定します。

例

set_option -oreg_in_iob 1

-ioreg_in_iob

双方向バッファに接続されるレジスタを IOB に配置します。

構文

-ioreg_in_iob <0|1>

形式

名称	説明
<0|1>	0：双方向バッファに接続されるレジスタをIOBに配置しません。 1：双方向バッファに接続されるレジスタをIOBに配置します。

分類

Project

説明

このオプションを有効にすると、双方向バッファに接続されるレジスタは IOB に配置されます。デフォルトは 1 です。

パラメータ

<0|1>：双方向バッファに接続されるレジスタを IOB に配置するかどうかを指定します。

例

set_option -ioreg_in_iob 1

-replicate_resources

高ファンアウトのリソースを複製してファンアウトを低減し、タイミングの結果を改善します。

構文

-replicate_resources <0|1>

形式

名称	説明
<0|1>	0：高ファンアウトのリソースを複製しません。 1：高ファンアウトのリソースを複製してファンアウトを低減します。

分類

Project

説明

このオプションを有効にすると、高ファンアウトのリソースが複製されてファンアウトが低減され、タイミングの結果が改善されます。デフォルトは 0 です。

パラメータ

<0|1>：高ファンアウトのリソースを複製してファンアウトを低減するかどうかを指定します。

例

set_option -replicate_resources 1

-serdesRetiming

配置の際にロジックリソースを SerDes node に近づけてタイミングを調整します。

構文

-serdesRetiming <0|1>

形式

名称	説明
<0|1>	0：配置の際にロジックリソースをSerDes nodeに近づけてタイミングを調整しません。 1：配置の際にロジックリソースをSerDes nodeに近づけてタイミングを調整します。

分類

Project

説明

このオプションを有効にすると、配置の際にロジックリソースが SerDes node に近づけられ、タイミングが調整されます。デフォルト値は 0 です。

パラメータ

<0|1>：配置の際にロジックリソースを SerDes node に近づけてタイミングを調整するかどうかを制御します。

例

set_option -serdesRetiming 1

-co-place_io_registers

Co-place IO registers(つまりレジスタを IO ブロックの近くに配置)のイネーブルの制御。

構文

-co-place_io_registers <0|1>

形式

名称	説明
<0|1>	0：使用しません。 1：使用します。

分類

Project

説明

このオプションを有効にすると、配置時に Co-place IO register のアルゴリズムが使用されます。デフォルト値は 0 です。

パラメータ

<0|1>：配置時に Co-place IO register を使用するかどうかを制御します。

例

set_option -co-place_io_registers 1

-clock_route_order

クロックプリミティブにより生成されたクロックライン以外のクロックラインの配線割り当て順を指定します。

構文

-clock_route_order <0|1>

形式

名称	説明
<0|1>	0：netのファンアウト数の多い順で割り当てます。 1：周波数の高い順で割り当てます。

分類

Project

説明

クロックプリミティブにより生成されたクロックライン以外のクロックラインの配線割り当て順を指定します。0 と 1 の 2 つのオプションがあり、デフォルトは 0 です。

パラメータ

<0|1>：クロックプリミティブにより生成されたクロックライン以外のクロックラインの配線割り当て順を指定します。

例

set_option -clock_route_order 1

-route_maxfan

配線のファンアウトの最大数を設定します。

構文

-route_maxfan <value>

形式

名称	説明
< value>	配線のファンアウトの最大数を設定します。

分類

Project

説明

配線のファンアウトの最大数を設定します。値は 0 より大きく 100 以下の整数である必要があります。デバイスが GW1NZ-1/GW1N-2/GW1NR-2/GW1NZ-2/GW1N-1P5 の場合、このオプションのデフォルト値は 10 で、他のデバイスの場合、デフォルト値は 23 です。

パラメータ

<value>：配線のファンアウトの最大数を設定します。

例

set_option -route_maxfan 60

-correct_hold_violation

配線によりタイミングの Hold 違反を自動修正します。

構文

-correct_hold_violation <0|1>

形式

名称	説明
<0|1>	0：配線によりタイミングのHold違反を自動修正しません。 1：配線によりタイミングのHold違反を自動修正します。

分類

Project

説明

このオプションを有効にすると、配線によりタイミングの Hold 違反が自動修正されます。デフォルト値は 1 です。

パラメータ

<0|1>：配線によりタイミングの Hold 違反を自動修正するかどうかを

指定します。

例

set_option -correct_hold_violation 1

-convert_sdp32_36_to_sdp16_18

32/36 ビット幅の SDPB/SDPX9B を 2 つの 16/18 ビット幅の SDPB/SDPX9B に変換します。このコマンドは Arora V デバイスでのみサポートされます。

構文

-convert_sdp32_36_to_sdp16_18 <0|1>

形式

名称	説明
<0|1>	0：32/36ビット幅のSDPB/SDPX9Bを2つの16/18ビット幅のSDPB/SDPX9Bに変換しません。 1：32/36ビット幅のSDPB/SDPX9Bを2つの16/18ビット幅のSDPB/SDPX9Bに変換します。

分類

Project

説明

このオプションを有効にすると、32/36 ビット幅の SDPB/SDPX9B が 2 つの 16/18 ビット幅の SDPB/SDPX9B に変換されます。デフォルト値は 0 です。

パラメータ

<0|1>：32/36 ビット幅の SDPB/SDPX9B を 2 つの 16/18 ビット幅の SDPB/SDPX9B に変換するかどうかを制御します。

例

set_option -convert_sdp32_36_to_sdp16_18 1

-inc_place <0|auto|file>

インクリメンタル配置。

構文

-inc_place <0|auto|file>

形式

名称	説明
<0|auto|file >	0：インクリメンタル配置をオフにします。 auto：自動的にインクリメンタル配置を実行します。 file：*.pファイルを指定してインクリメンタル配置を実行します。

分類

Project

説明

このオプションを有効にすると、インクリメンタル配置が使用されます。デフォルト値は 0 です。

パラメータ

<0|auto|file >：インクリメンタル配置を制御します。

例

set_option -inc_place auto

-inc_pnr <0|auto|file>

インクリメンタル配置配線。

構文

-inc_pnr <0|auto|file>

形式

名称	説明
<0|auto|file >	0：インクリメンタル配置配線をオフにします。 auto：自動的にインクリメンタル配置配線を実行します。 file：*.pファイルを指定してインクリメンタル配置配線を実行します。

分類

Project

説明

このオプションを有効にすると、インクリメンタル配置配線が使用されます。デフォルト値は 0 です。

パラメータ

<0|auto|file >：インクリメンタル配置配線を制御します。

例

set_option -inc_pnr auto

注記：

Place&Route のオプションの詳細については、『Gowin ソフトウェア ユーザーガイド(*SUG100*)』の Place & Route セクションを参照してください。

多重化ピンの属性の構成

-use_jtag_as_gpio

JTAG ピンを GPIO として多重化します。

構文

-use_jtag_as_gpio <0|1>

形式

名称	説明
<0|1>	0：JTAG 専用ピンとして使用します。 1：GPIO として多重化します

分類

Project

説明

JTAG ピンを GPIO として多重化します。デフォルトは 0 です。

パラメータ

<0|1>：JTAG ピンを GPIO として多重化するかどうかを指定します。

例

set_option -use_jtag_as_gpio 1

-use_sspi_as_gpio

SSPI ピンを GPIO として多重化します。

構文

-use_sspi_as_gpio <0|1>

形式

名称	説明
<0|1>	0：SSPI専用ピンとして使用します。 1：GPIOとして多重化します

分類

Project

説明

SSPI ピンを GPIO として多重化します。デフォルトは 0 です。

パラメータ

<0|1>：SSPI ピンを GPIO として多重化するかどうかを指定します。

例

set_option -use_sspi _as_gpio 1

-use_mspi_as_gpio

MSPI ピンを GPIO として多重化します。

構文

-use_mspi_as_gpio <0|1>

形式

名称	説明
<0|1>	0：MSPI専用ピンとして使用します。 1：GPIOとして多重化します

分類

Project

説明

MSPI ピンを GPIO として多重化します。デフォルトは 0 です。

パラメータ

<0|1>：MSPI ピンを GPIO として多重化するかどうかを指定します。

例

set_option -use_mspi _as_gpio 1

-use_ready_as_gpio

READY ピンを GPIO として多重化します。

構文

-use_ready_as_gpio <0|1>

形式

名称	説明
<0|1>	0：READY専用ピンとして使用します。 1：GPIOとして多重化します

分類

Project

説明

READY ピンを GPIO として多重化します。デフォルトは 0 です。

パラメータ

<0|1>：READY ピンを GPIO として多重化するかどうかを指定します。

例

set_option -use_ready_as_gpio 1

-use_done_as_gpio

DONE ピンを GPIO として多重化します。

構文

-use_done_as_gpio <0|1>

形式

名称	説明
<0|1>	0：DONE 専用ピンとして使用します。 1：GPIO として多重化します

分類

Project

説明

DONE ピンを GPIO として多重化します。デフォルトは 0 です。

パラメータ

<0|1>：DONE ピンを GPIO として多重化するかどうかを指定します。

例

set_option -use_ done_as_gpio 1

-use_reconfign_as_gpio

RECONFIG_N ピンを GPIO として多重化します。

構文

-use_ reconfign_as_gpio <0|1>

形式

名称	説明
<0|1>	0：RECONFIG_N専用ピンとして使用します。 1：GPIOとして多重化します

分類

Project

説明

RECONFIG_N ピンを GPIO として多重化します。デフォルトは 0 です。

パラメータ

<0|1>：RECONFIG_N ピンを GPIO として多重化するかどうかを指定します。

例

set_option -use_reconfign_as_gpio 1

-use_i2c_as_gpio

I2C ピンを GPIO として多重化します。

構文

-use_i2c_as_gpio <0|1>

形式

名称	説明
<0|1>	0：I2C 専用ピンとして使用します。 1：GPIO として多重化します

分類

Project

説明

I2C ピンを GPIO として多重化します。デフォルトは 0 です。

パラメータ

<0|1>：I2C ピンを GPIO として多重化するかどうかを指定します。

例

set_option -use_i2c_as_gpio 1

BitStream の属性の構成

-bit_format

生成されるビットストリーム・ファイルのコンテンツの形式を指定します。

構文

-bit_format <txt|bin>

形式

名称	説明
<txt|bin>	ビットストリーム・ファイルのコンテンツの形式。

分類

Project

説明

生成されるビットストリーム・ファイルのコンテンツの形式を指定します。デフォルトは bin です。

パラメータ

<txt|bin>：生成されるビットストリーム・ファイルのコンテンツの形式を指定します。

例

set_option -bit_format txt

-bit_crc_check

巡回冗長検査。

構文

-bit_crc_check <0|1>

形式

名称	説明
<0|1>	0：巡回冗長検査を有効にしません。 1：巡回冗長検査を有効にします。

分類

Project

説明

ビットストリーム・ファイルの巡回冗長検査を有効にするかどうかを指定します。デフォルトは 1 です。

パラメータ

<0|1>：ビットストリーム・ファイルの巡回冗長検査を有効にするかどうかを指定します。

例

set_option -bit_crc_check 1

-bit_compress

ビットストリーム・ファイルを圧縮します。

構文

-bit_compress <0|1>

形式

名称	説明
<0|1>	0：ビットストリーム・ファイルを圧縮しません。 1：ビットストリーム・ファイルを圧縮します。

分類

Project

説明

生成されるビットストリーム・ファイルを圧縮します。デフォルトは 1 です。

パラメータ

<0|1>：ビットストリーム・ファイルを圧縮するかどうかを指定します。

例

set_option -bit_compress 1

-bit_encrypt

ビットストリーム・ファイルを暗号化します。

構文

-bit_encrypt <0|1>

形式

名称	説明
<0|1>	0：暗号化しません。 1：暗号化します。

分類

Project

説明

ビットストリーム・ファイルを暗号化します(Arora ファミリーのみをサポート)。デフォルトは 0 です。

パラメータ

<0|1>：ビットストリーム・ファイルを暗号化するかどうかを指定します。

例

set_option -bit_encrypt 1

-bit_encrypt_key

暗号化キーをカスタマイズします。

構文

-bit_encrypt_key <key>

形式

名称	説明
<key>	暗号化キー。

分類

Project

説明

「-bit_encrypt」と併用することによりユーザーは暗号化キーをカスタマイズできます。デフォルトは全部 0 です。

パラメータ

<key>：暗号化キー。

例

set_option -bit_encrypt_key 00000000000000000000000000001101

-bit_security

セキュリティ・ビットを有効にするかどうかを指定します。

構文

-bit_security <0|1>

形式

名称	説明
<0|1>	0：セキュリティ・ビットを有効にしません。 1：セキュリティ・ビットを有効にします。

分類

Project

説明

セキュリティ・ビットを有効にするかどうかを指定します。デフォルトは 1 です。

パラメータ

<0|1>：セキュリティ・ビットを有効にするかどうかを指定します。

例

set_option -bit_security 1

-bit_incl_bsram_init

BSRAM の初期値をビットストリーム・ファイルに書き込みます。

構文

-bit_incl_bsram_init <0|1>

形式

名称	説明
<0|1>	0：BSRAMの初期値をビットストリーム・ファイルに書き込みません。 1：BSRAMの初期値をビットストリーム・ファイルに書き込みます。

分類

Project

説明

BSRAM の初期値をビットストリーム・ファイルに書き込みます。デフォルトは 1 です。GW1N(X)シリーズおよび GW2A(X)シリーズの場合、1 にセットすると、すべての位置の BSRAM の初期値がビットストリーム・ファイルに書き込まれます(占有されていない BSRAM 位置の初期値は 0 として取り扱われます)。GW5A(N)(S)(R)(T)デバイスの場合、1 にセットすると、占有された BSRAM の所在列にあるすべての BSRAM の初

期値がビットストリーム・ファイルに書き込まれます(この列の占有されていない BSRAM 位置の初期値は 0 として取り扱われます)。

パラメータ

<0|1>：BSRAM の初期値をビットストリーム・ファイルに書き込むかどうかを指定します。

例

set_option -bit_incl_bsram_init 1

-bsram_init_print_mode

BSRAM の初期値をビットストリームファイルに書き込むモードを制御します。このオプションは、GW5A(S)(T)-138/GW5AT-75 デバイスでのみサポートされています。

構文

-bsram_init_print_mode <default | partial>

形式

名称	説明
<default | partial>	default：すべての位置のBSRAMの初期値をビットストリーム・ファイルに書き込みます(占有されていないBSRAM位置の初期値は0として取り扱われます)。 partial：占有されたBSRAMの所在列にあるすべてのBSRAMの初期値をビットストリーム・ファイルに書き込ます(この列の占有されていないBSRAM位置の初期値は0として取り扱われます)。

分類

Project

説明

BSRAM の初期値をビットストリーム・ファイルに書き込むモードを制御します。デフォルトでは default です。このオプションは、GW5A(S)(T)-138/GW5AT-75 デバイスでのみサポートされています。このオプションの値が default の場合：すべての位置の BSRAM の初期値がビットストリーム・ファイルに書き込まれます(占有されていない BSRAM 位置の初期値は 0 として取り扱われます)。このオプションの値が partial の場合：占有された BSRAM の所在列にあるすべての BSRAM の初期値がビットストリーム・ファイルに書き込まれます(この列の占有されていない BSRAM 位置の初期値は 0 として取り扱われます)。partial を選択する場合は、まず multiboot_mode または mspijump_mode の値を quad に設定する必要があります。

パラメータ

<default | partial>：BSRAM の初期値をビットストリーム・ファイルに書き込むモードを制御します。

例

set_option -bsram_init_print_mode partial

-bg_programming

バックグラウンド・アップグレード機能。

構文

-bg_programming <off | jtag | i2c | goconfig | userlogic | i2c_jtag_sspi_qsspi | jtag_sspi_qsspi>

形式

名称	説明
<off|jtag|i2c|goconfig|userlogic| i2c_jtag_sspi_qsspi|jtag_sspi_qsspi>	off：バックグラウンド・アップグレード機能を有効にしません。 jtag：JTAGモードでバックグラウンド・アップグレードを実行します。 i2c：I2Cモードでバックグラウンド・アップグレードを実行します。 goconfig：goConfig IPでバックグラウンド・アップグレードを実行します。 userlogic：FPGAの内部論理でバックグラウンド・アップグレードを実行します。 i2c_jtag_sspi_qsspi：I2C/JTAG/SSPI/QSSPIモードでバックグラウンド・アップグレードを実行します。 jtag_sspi_qsspi：JTAG/SSPI/QSSPIモードでバックグラウンド・アップグレードを実行します。

分類

Project

説明

FPGA の動作を中断しないまま FPGA をプログラムするバックグラウンド・アップグレード機能です。デフォルトは off です。

パラメータ

< off | jtag | i2c | goconfig | userlogic | i2c_jtag_sspi_qsspi | jtag_sspi_qsspi>：バックグラウンド・アップグレードの方法。

例

set_option -bg_programming userlogic

-hotboot

ホットブートモード。

構文

-hotboot <0|1>

形式

名称	説明
<0|1>	0：ホットブートモードを使用しません。 1：ホットブートモードを使用します。

分類

Project

説明

ホットブートモードを使用するかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：ホットブートモードを使用するかどうかを指定します。

例

set_option -hotboot 1

-i2c_slave_addr

I2C デバイスのアドレスを設定します。

構文

-i2c_slave_addr <value>

形式

名称	説明
<value>	I2C デバイスのアドレスを設定します。

分類

Project

説明

I2C デバイスのアドレスを設定します。範囲は 00~7F。デフォルトは 00 です。

パラメータ

<value>：I2C デバイスのアドレス。

例

set_option -i2c_slave_addr 2F

-secure_mode

セキュアモードを有効にします。

構文

-secure_mode <0|1>

形式

名称	説明
<0|1>	0：セキュアモードを有効にしません。 1：セキュアモードを有効にします。

分類

Project

説明

セキュアモードを有効にすると、JTAG ピンは GPIO となり、デバイスは 1 回しかプログラムできなくなります。デフォルトは 0 です。

パラメータ

<0|1>：セキュアモードを有効にするかどうかを指定します。

例

set_option -secure_mode 1

-loading_rate

AutoBoot コンフィギュレーション・モードおよび MSPI コンフィギュレーション・モードでの、Flash から SRAM へのビットストリームデータの読み込み速度。

構文

-loading_rate <value>

形式

名称	説明
<value>	AutoBootコンフィギュレーション・モードおよびMSPIコンフィギュレーション・モードでの、FlashからSRAMへのビットストリームデータの読み込み速度。

分類

Project

説明

AutoBoot コンフィギュレーション・モードおよび MSPI コンフィギュレーション・モードでの、Flash から SRAM へのビットストリームデータの読み込み速度。デフォルトは 2.500MHz です。

パラメータ

<value>：Flash から SRAM へのビットストリームデータの読み込み速度。

例

set_option -loading_rate 21.000MHz

-seu_handler

シングル・イベント・アップセット・ハンドラを有効にするか選択します。

構文

-seu_handler <0|1>

形式

名称	説明
<0|1>	0：シングル・イベント・アップセット・ハンドラを有効にしません。 1：シングル・イベント・アップセット・ハンドラを有効にします。

分類

Project

説明

シングル・イベント・アップセット・ハンドラを有効にするか選択します。デフォルトは 0 です。

パラメータ

<0|1>：シングル・イベント・アップセット・ハンドラを有効にするか選択します。

例

set_option -seu_handler 1

-seu_handler_mode

SEU Handler を開始または停止するモードを選択します。

構文

-seu_handler_mode <auto|userlogic>

形式

名称	説明
<auto|userlogic>	auto：チップのウェイクアップ後、シングル・イベント・アップセット・ハンドラが自動的に有効にされます。 userlogic：ロジックを使用してシングル・イベント・アップセット・ハンドラを有効または無効にします。

分類

Project

説明

SEU Handler を開始または停止するモードを選択します。デフォルトは auto です。

パラメータ

<auto|userlogic>：SEU Handler を開始または停止するモードを選択します。

例

set_option -seu_handler_mode userlogic

-seu_handler_checksum

シングル・イベント・アップセット・ハンドリング、検出、計算、比較を有効にするか選択します。

構文

-seu_handler_checksum <0|1>

形式

名称	説明
<0|1>	0：シングル・イベント・アップセット・ハンドリング、検出、計算、比較を有効にしません。 1：シングル・イベント・アップセット・ハンドリング、検出、計算、比較を有効にします。

分類

Project

説明

シングル・イベント・アップセット・ハンドリング、検出、計算、比較を有効にするか選択します。デフォルトは 0 です。

パラメータ

<0|1>：シングル・イベント・アップセット・ハンドリング、検出、計算、比較を有効にするか選択します。

例

set_option -seu_handler_checksum 1

-error_detection

エラー検出のみを有効にします。

構文

-error_detection <0|1>

形式

名称	説明
<0|1>	0：エラー検出を有効にしません。

	1：エラー検出のみを有効にします。

分類

Project

説明

エラー検出のみを有効にします。デフォルトは 0 です。

パラメータ

<0|1>：エラー検出のみを有効にするかどうかを指定します。

例

set_option -error_detection 1

-error_detection_correction

エラーの検出と訂正を有効にします。

構文

-error_detection_correction <0|1>

形式

名称	説明
<0|1>	0：エラーの検出と訂正を有効にしません。 1：エラーの検出と訂正を有効にします。

分類

Project

説明

エラーの検出と訂正を有効にします。デフォルトは 0 です。

パラメータ

<0|1>：エラーの検出と訂正を有効にするかどうかを指定します。

例

set_option -error_detection_correction 1

-stop_seu_handler

SEU Handler を停止します。

構文

-stop_seu_handler <0|1>

形式

名称	説明
<0|1>	0：訂正不可能なECCエラーまたはCRCチェックサム・ミスマッチ・エラーが検出された場合、SEU Handlerを停止しません。 1：訂正不可能なECCエラーまたはCRCチェックサム・ミスマッチ・エラーが検出された場合、SEU Handlerを停止します。

分類

Project

説明

訂正不可能な ECC エラーまたは CRC チェックサム・ミスマッチ・エラーが検出された場合、SEU Handler を停止します。デフォルトは 0 です。

パラメータ

<0|1>：SEU Handler を停止するかどうかを指定します。

例

set_option -stop_seu_handler 1

-osc_div

拡張コントロール・レジスタの分周値を設定します。

構文

-osc_div <4|8|16|32>

形式

名称	説明
<4|8|16|32>	4：拡張コントロール・レジスタの分周値を4に設定します。 8：拡張コントロール・レジスタの分周値を8に設定します。 16：拡張コントロール・レジスタの分周値を16に設定します。 32：拡張コントロール・レジスタの分周値を32に設定します。

分類

Project

説明

拡張コントロール・レジスタの分周値を設定します。デフォルトは 8 です。

パラメータ

<4|8|16|32>：拡張コントロール・レジスタの分周値。

例

set_option -osc_div 8

-error_injection

エラー挿入を有効にします。

構文

-error_injection <0|1>

形式

名称	説明
<0|1>	0：エラー挿入を有効にしません。 1：エラー挿入を有効にします。

分類

Project

説明

エラー挿入を有効にします。デフォルトは 0 です。

パラメータ

<0|1>：エラー挿入を有効にするかどうかを指定します。

例

set_option -error_injection 1

-ext_cclk

外部マスター・コンフィギュレーション・クロックを有効にします。

構文

-ext_cclk <0|1>

形式

名称	説明
<0|1>	0：外部マスター・コンフィギュレーション・クロックを有効にしません。 1：外部マスター・コンフィギュレーション・クロックを有効にします。

分類

Project

説明

外部マスター・コンフィギュレーション・クロックを有効にします。

デフォルトは 0 です。

パラメータ

<0|1>：外部マスター・コンフィギュレーション・クロックを有効にするかどうかを指定します。

例

set_option -ext_cclk 1

-ext_cclk_div

分周器のパラメータを設定します。

構文

-ext_cclk_div <value>

形式

名称	説明
<value>	分周器のパラメータ。

分類

Project

説明

分周器のパラメータを設定します。

パラメータ

< value>：分周器のパラメータを設定します。

例

set_option -ext_cclk_div 4

-multi_boot

Multi Boot を有効にするかどうかを指定します。

構文

-multi_boot <0|1>

形式

名称	説明
<0|1>	0：Multi Bootを有効にしません。 1：Multi Bootを有効にします。

分類

Project

説明

Multi Boot を有効にするかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：Multi Boot を有効にするかどうかを指定します。

例

set_option -multi_boot 1

-multiboot_address_width

SPI Flash アドレスの幅を構成します。

構文

-multiboot_address_width<24|32>

形式

名称	説明
<24|32>	24：SPI Flashアドレスの幅を24に構成します。 32：SPI Flashアドレスの幅を32に構成します。

分類

Project

説明

SPI Flash アドレスの幅を構成します。デフォルトは 24 です。

パラメータ

<24|32>：SPI Flash アドレスの幅。

例

set_option -multiboot_address_width 32

-multiboot_spi_flash_address

SPI Flash アドレスを指定します。

構文

-multiboot_spi_flash_address <value>

形式

名称	説明
<value>	SPI Flash アドレス。

分類

Project

説明

SPI Flash アドレスを指定します。SPI Flash アドレスは、次の multiboot の際にビットストリーム・ファイルが読み込まれる開始アドレスで、デフォルトは 000000 です。

パラメータ

<value>：SPI Flash アドレス。

例

set_option -multiboot_spi_flash_address 000110

-multiboot_mode

SPI Flash のアクセスモードを構成します。

構文

-multiboot_mode <single | fast | dual | quad>

形式

名称	説明
< single|fast|dual|quad >	single：singleモードを使用します。 fast：fastモードを使用します。 dual：dualモードを使用します。 quad：quadモードを使用します。

分類

Project

説明

SPI Flash のアクセスモードを構成します。デフォルトは single です。

パラメータ

< single | fast | dual | quad >：SPI Flash のアクセスモード。

例

set_option -multiboot_mode single

-mspi_jump

MSPI JUMP を有効にするかどうかを指定します。

構文

-mspi_jump<0|1>

形式

名称	説明
<0|1>	0：MSPI JUMPを有効にしません。 1：MSPI JUMPを有効にします。

分類

Project

説明

MSPI JUMP を有効にするかどうかを指定します。デフォルトは 0 で

す。

パラメータ

<0|1>：MSPI JUMP を有効にするかどうかを指定します。

例

set_option -mspi_jump 1

-merge_jumpbit

MSPI JUMP ビットストリーム・ファイルを汎用ビットストリーム・ファイルにマージします。

構文

-merge_jumpbit <0|1>

形式

名称	説明
<0|1>	0：ビットストリーム・ファイルをマージしません。 1：ビットストリーム・ファイルをマージします。

分類

Project

説明

MSPI JUMP ビットストリーム・ファイルを汎用ビットストリーム・ファイルにマージするか選択します。デフォルトは 0 です。

パラメータ

<0|1>：MSPI JUMP ビットストリーム・ファイルを汎用ビットストリーム・ファイルにマージするかどうかを指定します。

例

set_option -merge_jumpbit 1

-mspijump_address_width

SPI Flash アドレスの幅を構成します。

構文

-mspijump_address_width <24|32>

形式

名称	説明
<24|32>	24：SPI Flashアドレスの幅を24に構成します。 32：SPI Flashアドレスの幅を32に構成します。

分類

Project

説明

SPI Flash アドレスの幅を構成します。デフォルトは 24 です。

パラメータ

<24|32>：SPI Flash アドレスの幅。

例

set_option - mspijump_address_width 32

-mspijump_spi_flash_address

SPI Flash アドレスを指定します。

構文

-mspijump_spi_flash_address <value>

形式

名称	説明
<value>	SPI Flashアドレス。

分類

Project

説明

SPI Flash アドレスを指定します。デフォルトは 000000 です。

パラメータ

<value>：SPI Flash アドレス。

例

set_option - mspijump_spi_flash_address 000110

-mspijump_mode

SPI Flash のアクセスモードを構成します。

構文

-mspijump_mode <single | fast | dual | quad>

形式

名称	説明
< single|fast|dual|quad >	single：singleモードを使用します。 fast：fastモードを使用します。 dual：dualモードを使用します。 quad：quadモードを使用します。

分類

Project

説明

SPI Flash のアクセスモードを構成します。デフォルトは single です。

パラメータ

< single | fast | dual | quad >：SPI Flash のアクセスモード。

例

set_option -mspijump_mode single

-program_done_bypass

新しいビットストリームデータを転送します。

構文

-program_done_bypass <0|1>

形式

名称	説明
<0|1>	0：この機能を有効にしません。 1：この機能を有効にします。

分類

Project

説明

Done Final 信号が有効になった場合、外部の Done 信号を Low のままにすることにより、ビットストリームがロードされた後に新しいビットストリームデータを転送できるようにします。デフォルトは 0 です。

パラメータ

<0|1>：新しいビットストリームデータを転送する機能を有効にするかどうかを指定します。

例

set_option -program_done_bypass 1

-power_on_reset_monitor

パワーオンリセット。

構文

-power_on_reset_monitor <0|1>

形式

名称	説明
<0|1>	0：パワーオンリセット機能を有効にしません。 1：パワーオンリセット機能を有効にします。

分類

Project

説明

パワーオンリセット機能を有効にするかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：パワーオンリセット機能を有効にするかどうかを指定します。

例

set_option -power_on_reset_monitor 1

-turn_off_bg

Bandgap 機能。

構文

-turn_off_bg <0|1>

形式

名称	説明
<0|1>	0：Bandgap機能を有効にします。 1：Bandgap機能を有効にしません。

分類

Project

説明

Bandgap 機能の制御。デフォルトは 0 です。

パラメータ

<0|1>：Bandgap 機能を有効にするかどうかを指定します。

例

set_option -turn_off_bg 1

-wakeup_mode

Wake Up Mode を有効にするかどうかを指定します。

構文

-wakeup_mode <0|1>

形式

名称	説明
<0|1>	0：Wake Up Modeを有効にしません。 1：Wake Up Modeを有効にします。

分類

Project

説明

Wake Up Mode を有効にするかどうかを指定します。デフォルトは 0 です。

パラメータ

<0|1>：Wake Up Mode を有効にするかどうかを指定します。

例

set_option -wakeup_mode 1

-user_code

User Code をカスタマイズします。

構文

-user_code <default|value>

形式

名称	説明
<default|value>	User Codeの値をカスタマイズします。

分類

Project

説明

User Code をカスタマイズできます。デフォルトは default(00000000)です。

パラメータ

<default|value>：User Code をカスタマイズします。

例

set_option -user_code 00000010

注記：

BitStream 関連オプションの詳細については、『Gowin ソフトウェア ユーザーガイド (*SUG100*)』の BitStream セクションを参照してください。

Unused Pin の属性の構成

-unused_pin

未使用 GPIO の IO 属性を設定します。

構文

-unused_pin <default|open_drain>

形式

名称	説明
<default|open_drain>	default：未使用のGPIOを、弱いプルアップを持つトライステート入力として構成します。 open_drain：未使用のGPIOを出力として構成します(OPEN DRAINはオン)。

分類

Project

説明

未使用 GPIO の IO 属性を設定します。

パラメータ

<default|open_drain>：未使用 GPIO の IO 属性を設定します。

例

set_option -unused_pin open_drain

注記：

Unused Pin 関連オプションの詳細については、『Gowin ソフトウェア ユーザーガイド(*SUG100*)』の Unused Pin セクションを参照してください。

3.3.21 set_property

IP のオプションを構成します。

構文

set_property [-dict <args>] <name> <value> <objects>

形式

名称	説明
[-dict]	設定される（name/value）ペアのオプションリスト。
<name>	設定されるオプションの名前。-dictを使用する場合は無効です
<value>	設定されるオプションの値。-dictを使用する場合は無効です
<objects>	オプションが設定されるIP

分類

IPFlow

説明

指定された IP のオプションとそれに対応するオプション値を設定します。

パラメータ

- [-dict]：複数のオプションとそれに対応するオプション値のペアを含む辞書を指定します。ペアは(<name> <value>)で指定され、複数のペ

アはスペースで区切られます。辞書は中括弧{}で囲む必要があります。

- <name>：設定されるオプションの名前を指定します。説明の形式は CONFIG.property です。ここで、property はオプション名を指します。
- <value>：設定されるオプションに対応するオプション値を指定します。属性タイプに応じて値を決定する必要があります。オプション値が文字列の場合、元の形式である必要があります。
- <objects>：設定されるオプションの 1 つまたは複数の IP を指定します。1 つの IP は[get_ips module_name]で指定されます。複数の IP は、[get_ips module_name0 module_name1 …] で指定されます。

例

-dict オプションを使用して現在のデザインで複数の IP の構成オプションを一度に指定します：

set_property -dict {CONFIG.Data_Width 16 CONFIG.Write_Depth 1024 CONFIG.Read_Depth 1024} [get_ips FIFO_Top]

（name, value, objects）を使用して現在のデザインで 1 つの IP の構成オプションを指定します。

set_property CONFIG.Data_Width {16} [get_ips FIFO_Top]

この例では、ダッシュ「-」またはスペースを含むオプション値を設定する方法を示します。

set_property {CONFIG.Almost_Full_Type} {Full-Single Threshold Constant Parameter} [get_ips FIFO_Top]

注記：
場合によっては、オプション値にダッシュ「-」やスペースなどの特殊文字が含まれることがあるため、オプション値が正しく解析されない可能性があります。この場合、オプション値を中括弧{}で囲む必要があります。

関連項目

- list_property
- report_property

3.3.22 source

Gowin ソフトウェアの Tcl コマンド編集ウィンドウで、またはコマンド・ライン・モードを開始した後、このコマンドを使用して Tcl スクリプトを実行します。ファイルパスの形式については、add_file を参照してください。

構文

source <file>

形式

名称	説明
<file>	実行されるTclスクリプト。

分類

IPFlow，Project

説明

未使用 GPIO の IO 属性を設定します。

パラメータ

<file>：実行される Tcl スクリプトファイル。

例

source project.tcl

source D:/gowin_project/project.tcl

source D:¥¥gowin_project¥¥project.tcl

3.3.23 write_ip_tcl

Tcl スクリプトをエクスポートします。このスクリプトは、指定された IP を再生成できます。

構文

write_ip_tcl [-ip_name <newModuleName>] [-multiple_files] [-force] [<tcl_filename>] <objects>

形式

名称	説明
[-ip_name]	IP のモジュール名
[-multiple_files]	IP ごとに.tcl ファイルを作成します
[-force]	既存のファイルを上書きします
[<tcl_filename>]	エクスポートされる tcl ファイル
<objects>	tcl ファイルがエクスポートされる IP を指定します。

分類

IPFlow

説明

このコマンドは、指定された IP の Tcl スクリプトファイルをエクスポートします。ファイルパスの形式については、add_file を参照してください。

パラメータ

- [<tcl_filename>]：生成される Tcl スクリプトファイルの名前。指定しない場合は、現在のプロジェクト名が使用されます。

- <objects>：Tcl スクリプトが生成される 1 つまたは複数の IP を指定します。1 つの IP は[get_ips module_name]で指定されます。複数の IP は、[get_ips module_name0 module_name1 …] で指定されます。
- [-ip_name <newModuleName>]：生成された Tcl スクリプト内の IP の module_name を変更します。1 つのオブジェクトのみを指定できます。
- [-multiple_files]：指定されたすべての IP に対して Tcl スクリプトファイルを生成します。スクリプト名は、対応する IP の module_name に基づいて命名されます。このオプションは、-ip_name および <tcl_filename>と相互に排他的です。
- [-force]：同じ名前の既存の Tcl ファイルを上書きします。

例

この例では、IP FIFO_Top のために Tcl ファイルがエクスポートされます。source コマンドを実行した時に作成される IP モジュールは FIFO_Top_new という名前になります。

```
write_ip_tcl -ip_name FIFO_Top_new [get_ips FIFO_Top]
```

この例では、指定された IP ごとに個別の Tcl ファイルが作成されます。

```
write_ip_tcl -multiple_files [get_ips FIFO_Top FIFO_Top_1]
```

この例では、指定された複数の IP が 1 つの Tcl ファイルに書き込まれます。

```
write_ip_tcl [get_ips FIFO_Top FIFO_Top_1] my_fifo.tcl
```

4 Tcl による IP 生成フロー

Tcl を使用して IP を生成することは、プロジェクト内で実行する必要があります。そのため、まず create_project コマンドでプロジェクトを新規作成するか、または open_project コマンドで既存のプロジェクトを開きます。その後のフローは以下の通りです。

1. create_ipc コマンドを使用して、指定した IP のデフォルト構成を持つ IPC ファイルを作成するか、または read_ipc コマンドを使用して既存の IPC ファイルを開きます。
2. set_property コマンドを使用して、IP の構成オプションを変更できます。IP にどのような構成オプションがあるかは、list_property または report_property コマンドで確認できます。また、各構成オプションの具体的な構成値については、該当する IP のユーザーガイドを参照してください。
3. 構成の完了後、generate_target コマンドを使用して IP の設計ファイルを生成し、それをプロジェクトに追加します。さらに、write_ip_tcl コマンドを使用して Tcl スクリプトファイルとしてエクスポートすることも可能です。これにより、次回以降、Tcl スクリプトを介して同じ IP を簡単に再生成できるようになります。

例

ROM IP を例に、Tcl による IP 生成のフローを紹介します。

```
set projectName "rom_tcl"
set dirPath "E:/tcl"
set partNumber "GW1N-LV9LQ144C6/I5"
set deviceVersion "C"
set initFile "E:/init_file/rom.mi"
#create project
create_project -name $projectName -dir $dirPath -pn $partNumber -device_version $deviceVersion
##########Generate ROM IP############
```

```
#create ROM IP
create_ipc -name rom -module_name Gowin_ROM -language Verilog
-file_name gowin_rom
#set the option value of specified options
set_property CONFIG.Data_Width 8 [get_ips Gowin_ROM]
set_property CONFIG.Memory_Initialization_File $initFile [get_ips
Gowin_ROM]
#generate IP file and add to project
generate_target [get_ips Gowin_ROM]
```


GOWIN高云
智慧逻辑 定制未来